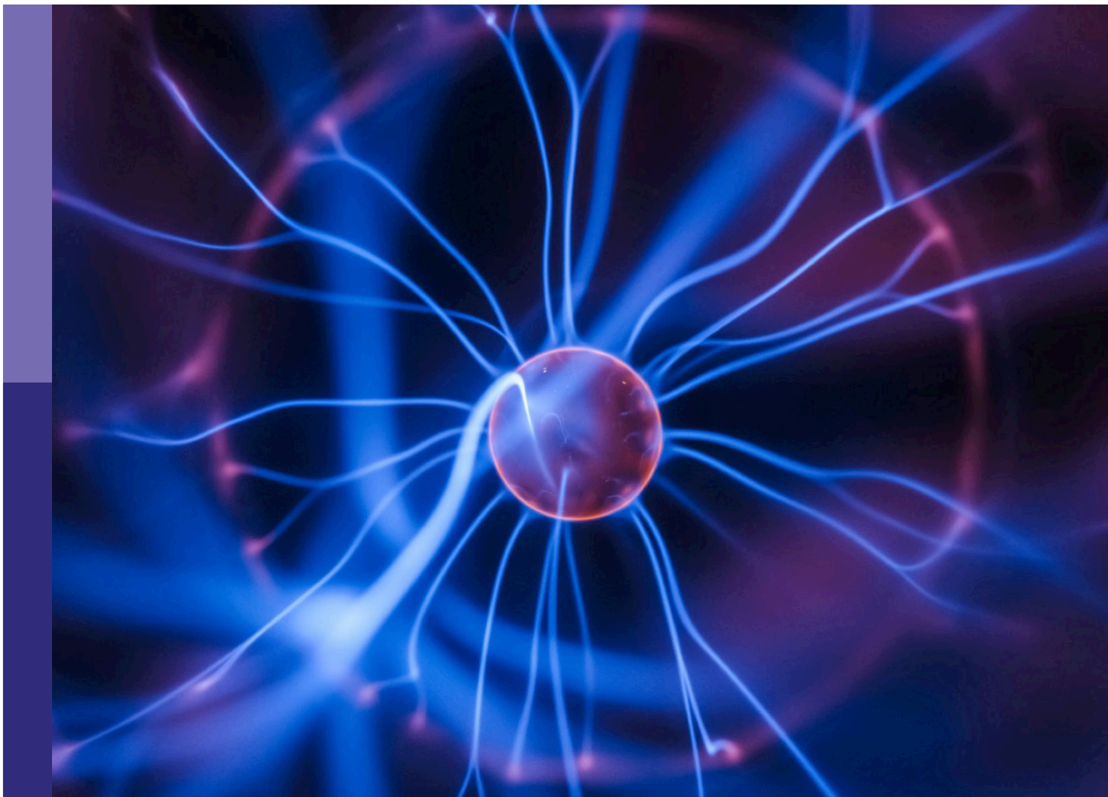
frontiers | Research Topics

Security, governance, and challenges of the new generation of cyber-physical-social systems

Edited by
Yuanyuan Huang, Qifei Wang, Jianping Gou,
Amin Ul Haq and Xin Lu

Coordinated by
Jiazhong Lu

Published in
Frontiers in Physics



ISSN 1664-8714
ISBN 978-2-8325-5644-3
DOI 10.3389/978-2-8325-5644-3

About Frontiers

Frontiers is more than just an open access publisher of scholarly articles: it is a pioneering approach to the world of academia, radically improving the way scholarly research is managed. The grand vision of Frontiers is a world where all people have an equal opportunity to seek, share and generate knowledge. Frontiers provides immediate and permanent online open access to all its publications, but this alone is not enough to realize our grand goals.

Frontiers journal series

The Frontiers journal series is a multi-tier and interdisciplinary set of open-access, online journals, promising a paradigm shift from the current review, selection and dissemination processes in academic publishing. All Frontiers journals are driven by researchers for researchers; therefore, they constitute a service to the scholarly community. At the same time, the *Frontiers journal series* operates on a revolutionary invention, the tiered publishing system, initially addressing specific communities of scholars, and gradually climbing up to broader public understanding, thus serving the interests of the lay society, too.

Dedication to quality

Each Frontiers article is a landmark of the highest quality, thanks to genuinely collaborative interactions between authors and review editors, who include some of the world's best academicians. Research must be certified by peers before entering a stream of knowledge that may eventually reach the public - and shape society; therefore, Frontiers only applies the most rigorous and unbiased reviews. Frontiers revolutionizes research publishing by freely delivering the most outstanding research, evaluated with no bias from both the academic and social point of view. By applying the most advanced information technologies, Frontiers is catapulting scholarly publishing into a new generation.

What are Frontiers Research Topics?

Frontiers Research Topics are very popular trademarks of the *Frontiers journals series*: they are collections of at least ten articles, all centered on a particular subject. With their unique mix of varied contributions from Original Research to Review Articles, Frontiers Research Topics unify the most influential researchers, the latest key findings and historical advances in a hot research area.

Find out more on how to host your own Frontiers Research Topic or contribute to one as an author by contacting the Frontiers editorial office: frontiersin.org/about/contact

Security, governance, and challenges of the new generation of cyber-physical-social systems

Topic editors

Yuanyuan Huang — Chengdu University of Information Technology, China
Qifei Wang — University of California, Berkeley, United States
Jianping Gou — Southwest University, China
Amin Ul Haq — University of Electronic Science and Technology of China, China
Xin Lu — De Montfort University, United Kingdom

Topic coordinator

Jiazhong Lu — Chengdu University of Information Technology, China

Citation

Huang, Y., Wang, Q., Gou, J., Ul Haq, A., Lu, X., Lu, J., eds. (2024). *Security, governance, and challenges of the new generation of cyber-physical-social systems*. Lausanne: Frontiers Media SA. doi: 10.3389/978-2-8325-5644-3

Table of contents

frontiers | Frontiers in Physics

TYPE Editorial
PUBLISHED 15 August 2024
DOI 10.3389/fphy.2024.1464919

Check for updates

OPEN ACCESS

EDITED AND REVIEWED BY
Matjaž Perc,
University of Maribor, Slovenia

*CORRESPONDENCE
Yuanyuan Huang,
iyyhuang@hotmail.com

RECEIVED 15 July 2024
ACCEPTED 17 July 2024
PUBLISHED 15 August 2024

CITATION
Huang Y and Lu X (2024), Editorial: Security, governance, and challenges of the new generation of cyber-physical-social systems.
Front. Phys. 12:1464919.
doi: 10.3389/fphy.2024.1464919

Editorial: Security, governance, and challenges of the new generation of cyber-physical-social systems

Yuanyuan Huang[1]* and Xin Lu[2]

[1]Department of Network Engineering, Chengdu University of Information Technology, Chengdu, China,
[2]School of Computer Science and Informatics, De Montfort University, Leicester, United Kingdom

KEYWORDS

cyber-physical-social systems (CPSS), internet of things-IoT, artificial intelligence-AI, big data, cyber-physical systems (CPS), advanced attacks, system security

Editorial on the Research Topic
Security, governance, and challenges of the new generation of cyber-physical-social systems

In recent years, the transformation of devices and systems into intelligent, interconnected entities has given rise to the concepts widely recognized as the Internet of Things (IoT) and cyber-physical systems (CPSs). The integration of social networks with CPSs leads to an innovative paradigm known as cyber-physical-social systems (CPSSs). CPSS, harmonizing the cyber, physical, and social spaces, constitutes the next evolution of intelligent systems. It is founded on the integration of embedded systems, computer networks, control theory, and sensor networks. A typical CPSS is comprised of sensors, controllers, actuators, and communication networks. Its salience lies in the seamless connection of physical devices to the Internet and social networks, thereby imbuing these devices with capabilities such as computation, communication, precise control, remote coordination, and autonomy. The applicability of CPSS spans diverse fields, including intelligent transportation systems, telemedicine, smart grid technology, aerospace, smart home appliances, environmental monitoring, intelligent buildings, defense systems, and weaponry. Thus, CPSS stands as a vital component of a nation’s essential infrastructure.

CPSS exhibits a range of distinctive features, including the amalgamation of human and computer intelligence, the integration across various spatial domains, inherent network heterogeneity, and the incorporation of multi-source information. In the context of CPSS, data serves as a vital link, seamlessly connecting the three principal components: cyber systems, physical mechanisms, and social constructs. Information from physical and social systems is conveyed to the corresponding information system via network channels. Simultaneously, this information system reciprocates by supplying feedback to the physical and social domains through meticulous computation and informed decision-making processes. Nevertheless, the heterogeneous nature of CPSS, coupled with their reliance on confidential and sensitive data, and expansive deployment, makes them susceptible to an array of security threats. These threats span across the cyber, physical, and social realms, presenting significant challenges related to privacy and trust.

This Research Topic is dedicated to presenting original research and insightful reviews, emphasizing innovations and enhancements in the domains of advanced attacks, system security, privacy, and trust technology in CPSS. Submissions focusing on detection and defense strategies employing artificial intelligence and big data are particularly encouraged. This includes a wide range of topics, including but not limited to, theoretical foundations, design methodologies, modeling techniques, configuration approaches, representational frameworks, data processing mechanisms, analytical methods, and their relevant applications within the context of CPSS.

Author contributions

YH: Writing–original draft, Writing–review and editing. XL: Writing–review and editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher’s note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 14 February 2024
DOI 10.3389/fphy.2024.1354482

A Cyber-physical-social systems approach to the semantic segmentation of pulmonary embolism

Siyu Zhan[1,2], Xin Lei[3], Lu Guo[4], Mingxiu Xiong[5], Tingyu Liu[6]*, Shuang Liu[7] and Hao Yu[3]

[1]Laboratory of Intelligent Collaborative Computing, University of Electronic Science and Technology of China, Chengdu, Sichuan, China, [2]Trusted Cloud Computing and Big Data Key Laboratory of Sichuan Province, Chengdu, Sichuan, China, [3]School of Optoelectronic Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China, [4]Department of Pulmonary and Critical Care Medicine, Sichuan Provincial People's Hospital, University of Electronic Science and Technology of China, Chengdu, China, [5]School of Medicine, University of Electronic Science and Technology of China, Chengdu, China, [6]School of Network and Communication Engineering, Chengdu Technological University, Chengdu, China, [7]Yingcai Experimental College, University of Electronic Science and Technology of China, Chengdu, China

Check for updates

OPEN ACCESS

EDITED BY
Yuanyuan Huang,
Chengdu University of Information Technology, China

REVIEWED BY
Tianxi Huang,
Chengdu Textile College, China
Dekun Hu,
Chengdu University, China

*CORRESPONDENCE
Tingyu Liu,
✉ liutingyu23@163.com

RECEIVED 12 December 2023
ACCEPTED 26 January 2024
PUBLISHED 14 February 2024

CITATION
Zhan S, Lei X, Guo L, Xiong M, Liu T, Liu S and Yu H (2024), A Cyber-physical-social systems approach to the semantic segmentation of pulmonary embolism.
Front. Phys. 12:1354482.
doi: 10.3389/fphy.2024.1354482

Cyber-Physical-Social Systems (CPSS) epitomize the modern era's intelligent connectivity. They integrate physical devices, computer networks, and social networks, forming an innovative paradigm for intelligent systems. Utilizing CPSS to enhance intelligence, automation, and remote services in healthcare represents a primary research focus. Pulmonary embolism, a prevalent condition resulting from the blockage of the pulmonary artery and its branches by emboli, leads to a spectrum of clinical syndromes marked by impaired pulmonary circulation and right heart dysfunction, contributing to sudden and unpredictable fatalities. Nevertheless, the diagnosis of pulmonary embolism remains challenging due to non-specific clinical presentations, constrained diagnostic capabilities, delayed diagnoses, insufficient physician knowledge, and suboptimal diagnostic techniques. Consequently, we introduce the innovative LSCU-Net architecture within the CPSS framework, designed to develop an automated segmentation and intelligent assessment system for pulmonary embolism, facilitating its automated and intelligent detection. The experimental findings demonstrate that the model accurately segments pulmonary embolism, evidenced by a Jaccard index of 0.6958, a Dice coefficient of 0.8193, a Mean Pixel Accuracy (mPA) of 0.8519, and an accuracy of 0.9993. Empirical studies reveal that our proposed model substantially surpasses existing models in performance. Consequently, this model can aid physicians in the diagnosis of pulmonary embolism during clinical practice. The established pulmonary embolism automatic segmentation and assessment system also showcases the application successes of CPSS in intelligent remote healthcare. The system's development and deployment not only streamline physicians' diagnostic processes but also elevate public health standards and advance CPSS research within the medical domain.

KEYWORDS

Cyber-physical-social systems, U-Net, pulmonary embolism, CBAM, Bi-LSTM

1 Introduction

In recent years, the paradigm of Cyber-Physical Social Systems (CPSS [1]) has emerged. As an interdisciplinary field, CPSS integrates theories and technologies from computer science, physics, and social sciences, with a focus on exploring the interactions among sensors, physical systems, and social networks, and is dedicated to intelligent decision-making and optimization. Particularly in the field of medical image processing, the adoption of CPSS offers new opportunities to surmount the challenges of medical image data processing. Investigating the utilization of CPSS to achieve remote, automated, and intelligent medical treatment is a key area of interest. The medical treatment field is rapidly evolving from a traditional centralized treatment approach by hospitals and experts to a patient-centric distributed treatment model.

CPSS powers smart personal healthcare. In CPSS, the integration of physical devices, social networks, and networked systems enables the real-time monitoring of patient data (including blood pressure, vital signs, and activity monitoring) and facilitates the conversion of these data from the physical to the digital realm. This process not only endows devices with the ability to compute, coordinate remotely, and operate autonomously, but also advances medical treatment towards greater remoteness, automation, and intelligence. In this context, H et al. [2] introduced a doctor recommendation algorithm based on the doctor's diagnostic and treatment efficacy and the patient's personal preferences in 2014. In 2018, Q et al. [3] focused on e-medical services leveraging social networks and proposed an e-medical system model that utilizes the green CPSS framework to detect and predict disease transmission.

This study aims to conduct an in-depth analysis of CPSS applications within the medical field, with a particular focus on the efficacy of CPSS in supporting pulmonary embolism patients through remote, real-time, and intelligent treatment modalities.

Pulmonary embolism (PE), a prevalent medical condition, is characterized by a spectrum of diseases or clinical syndromes wherein endogenous or exogenous emboli obstruct the pulmonary artery and its branches, leading to impaired pulmonary circulation and right ventricular dysfunction [4]. Pulmonary embolism represents a critical health threat and is a primary cause of unexpected sudden death [5, 6]. Extensive data indicate that pulmonary embolism has a high global incidence. Annually, the United States sees between 650,000 and 700,000 new cases of pulmonary embolism, with a mortality rate surpassed only by cancer and coronary heart disease, making it the third leading cause of death [7]. In France, the incidence of pulmonary embolism rivals that of myocardial infarction fatalities, with over 100,000 new cases reported annually [6]. The resulting disability and morbidity from PE are significant drivers of medical expenditures and bear socioeconomic implications [8–14].

The diagnosis of pulmonary embolism remains challenging owing to the non-specific clinical manifestations of the condition, constrained diagnostic resources, delays in seeking medical attention, and the lack of physician awareness coupled with inappropriate diagnostic methods. Pulmonary embolism clinically presents as respiratory dysfunction. Numerous conditions can cause this symptom, making it challenging for physicians to directly associate this clinical feature with pulmonary embolism. Consequently, the majority of patients with pulmonary embolism do not receive an accurate diagnosis in their lifetime, with misdiagnosis rates reaching as high as 70%. An American study examining the correlation between the timing of pulmonary embolism diagnosis and mortality revealed that each hour of diagnostic advancement reduced the patient's mortality risk by 5% [15].

Prior to the extensive adoption of artificial intelligence (AI) technology, especially deep learning, in medical image analysis, medical image segmentation was heavily dependent on traditional image processing techniques. Traditional image segmentation methods have played a pivotal role in analyzing medical images, including X-rays, MRIs, CT scans, and ultrasound images. These methods typically entail manual or semi-automatic processes and rely on a variety of mathematical and algorithmic approaches to identify and delineate the boundaries of distinct structures within the images. Traditional medical image segmentation methods can be broadly classified into several categories: 1) Threshold-based methods [16], which classify pixels within the image as target or background by predefining a characteristic feature. 2) Region-based methods [17], which extract the target area by delineating a sub-region manually and subsequently merging adjacent pixels with similar attributes. 3) Edge-based segmentation methods [18], which achieve segmentation by identifying pixels where edges undergo significant changes at the juncture between the target and background areas, effectively delineating the boundaries. 4) Atlas-based segmentation methods [19], through the incorporation of shape information and the utilization of prior knowledge, these methods can yield improved segmentation outcomes even for medical images with indistinct boundaries and substantial noise.

In practical applications, the unique shape and contour of pulmonary embolism lesions in CTPA images make it so that traditional image segmentation methods often fail to achieve optimal outcomes in segmenting pulmonary embolism.

To address the above challenges, researchers are working to develop methods that utilize Cyber-Physical-Social Systems (CPSS) to enhance medical diagnosis. In 2014, researchers such as Long [20] introduced fully convolutional neural network (FCN) by enhancing the traditional convolutional neural network architecture. FCN abandons the fully connected layer in traditional convolutional neural networks and instead uses deconvolution layers. This change brings significant advantages - the network can handle image inputs of any size and is no longer limited to a fixed input size. However, this approach is not without drawbacks. The network needs to use basic upsampling technology at the back end of the processing process to match the size of the original image. This step often leads to the loss of a large amount of spatial information, thus limiting the accuracy of the network in image segmentation tasks, resulting in segmentation results. Accuracy is compromised.

Then in 2015, Ronneberger et al. [21] proposed the U-Net architecture based on FCN, aiming to more effectively utilize the rich contextual information in images. U-Net further optimizes the performance of fully convolutional networks through carefully designed downsampling and upsampling operations, and introduces four key components: encoder, decoder, bottleneck layer, and skip connection. The proposal of U-Net is widely

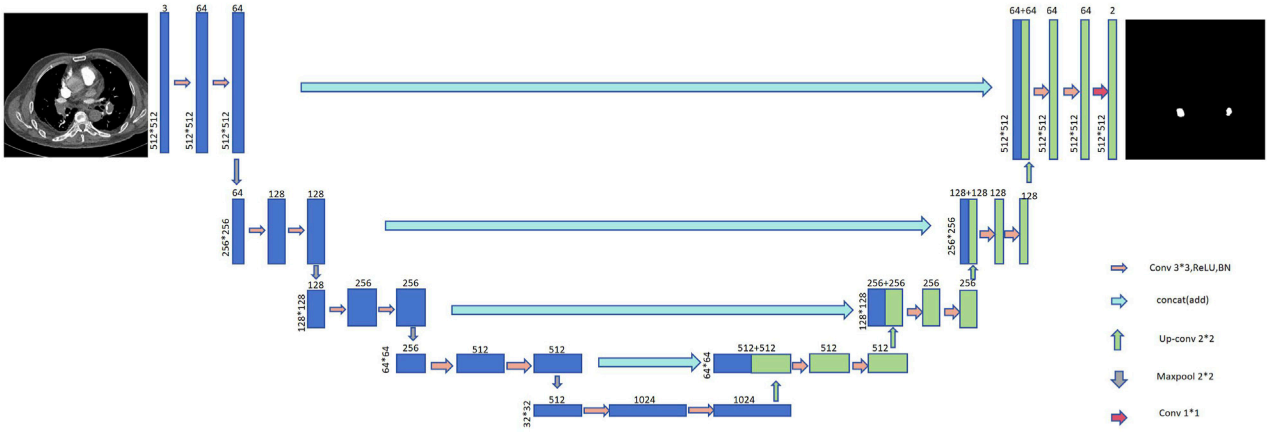


FIGURE 1
Schematic diagram of U-net model.

regarded as a milestone in the field of encoder-decoder network structure, especially in the field of medical image segmentation. U-Net not only demonstrates excellent performance, but also promotes rapid progress in this field.

Owing to U-Net's superior performance in medical image segmentation, we aimed to enhance U-Net accordingly to better suit the pulmonary embolism segmentation task. In comparison to traditional medical image segmentation methods, the automatic segmentation of pulmonary embolism via deep learning presents multiple challenges. The pulmonary embolism region is small in comparison to the lung area, and the marked imbalance between positive and negative samples can impair the model's predictive accuracy, potentially leading to a complete loss of predictive capability [22]. Therefore, we have incorporated an attention mechanism into the U-Net architecture, thereby directing the model's focus more towards the pulmonary embolism region. Furthermore, considering the challenge of accessing and the limited information in medical datasets, we have endeavored to integrate the Bi-LSTM architecture into the U-Net framework to capture the inter-slice information pertinent to pulmonary embolism. Additionally, to tackle the substantial imbalance of positive and negative samples in the pulmonary embolism dataset, we employ a hybrid loss function combining Cross-Entropy Loss (CELoss) and Tversky Loss (TLoss) during the neural network's training.

This article's principal contributions are as follows:

(1) An enhanced LSCU-Net, derived from the U-Net architecture, was developed. This framework integrates contextual information to automatically and accurately identify and segment pulmonary embolism, thereby aiding medical professionals and reducing the rate of misdiagnosis, and actualizing the application of CPSS in medical image segmentation.
(2) During model training, a hybrid loss function combining CELoss and FTLoss addresses the challenges of small pulmonary embsolism targets and disproportionate scales between pulmonary embolism and background in segmentation tasks.
(3) Experimental results demonstrate that our proposed method achieves the following metrics on the test set: a Jaccard index (JAC) of 0.6958, a Dice similarity coefficient (DSC) of 0.8206, a Mean Pixel Accuracy (mPA) of 0.8519, and an accuracy of 0.9993, thereby substantiating the method's feasibility.

2 Methods

2.1 LSCU-net

The U-Net architecture is extensively employed in medical image segmentation tasks due to its advantageous compact model parameterization, minimal data requirements, and rapid training capabilities. As depicted in Figure 1, the U-Net model is an evolution of the Fully Convolutional Network (FCN). The network's structure resembles the letter 'U', which is the origin of its nomenclature. The U-Net framework primarily consists of four key components: the encoder, decoder, skip connections, and a bottleneck layer. The encoder module comprises convolutional and pooling layers, serving primarily to downsample and extract features. The decoder module, consisting of convolutional and upsampling layers, is chiefly responsible for localizing the target and reconstructing the image dimensions. Skip connections merge the encoder and decoder information along the channel dimension, facilitating the integration of contextual data.

The enhanced architecture of our neural network model, termed Long Short-Term Memory and Convolutional Block Attention Module U-Net (LSCU-Net), is depicted in Figure 2. This model is a modification of the benchmark U-Net architecture. In comparison to the standard U-Net model, the principal enhancements of the LSCU-Net include:

(1) Integration of the Convolutional Block Attention Module (CBAM) into the encoder module to refine the neural network's learning strategy, thereby focusing more acutely on pertinent areas within the channel and spatial dimensions.

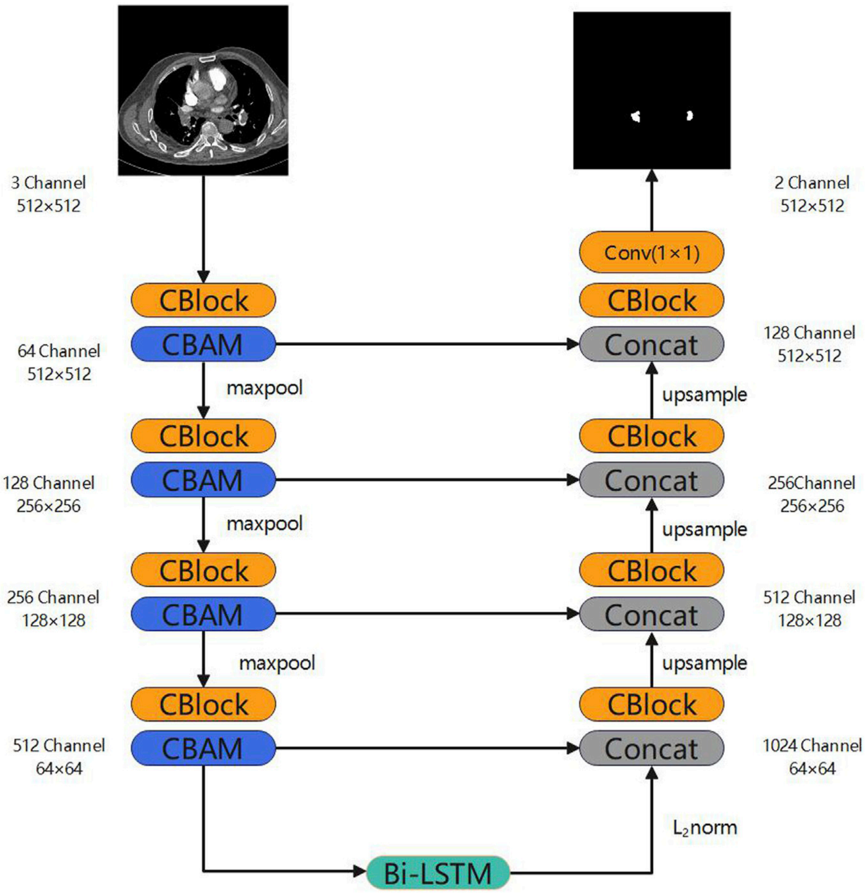


FIGURE 2
Schematic diagram of LSCU-net model.

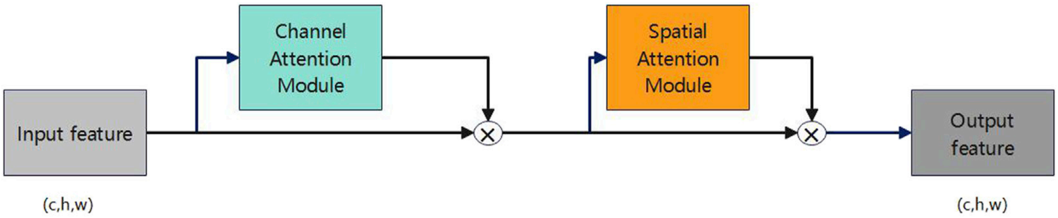


FIGURE 3
CBAM.

The attention mechanism [23] constitutes a significant concept within the domain of deep learning. It emulates the human cognitive process of attention and concentration during information processing. This mechanism enables the neural network model to focus more intensively on pertinent information throughout the training phase, thereby enhancing the model's capacity for representation learning and feature selection with respect to input data.

The Convolutional Block Attention Module (CBAM) [24], proposed by Sanghyun et al. in 2018, is an attention mechanism module within the realm of deep learning. Figure 3 illustrates its structure. CBAM's central concept involves the simultaneous introduction of channel and spatial attention sub-modules, enabling the neural network model to concurrently attend to both channel and spatial information.

Figure 4A illustrates how the channel attention module captures the interdependencies among various channels. This module computes the significance of each channel by processing the input features, subsequently assigning weights to minimize extraneous information across channels, which enhances the feature extraction quality. The computation is described by the following formula:

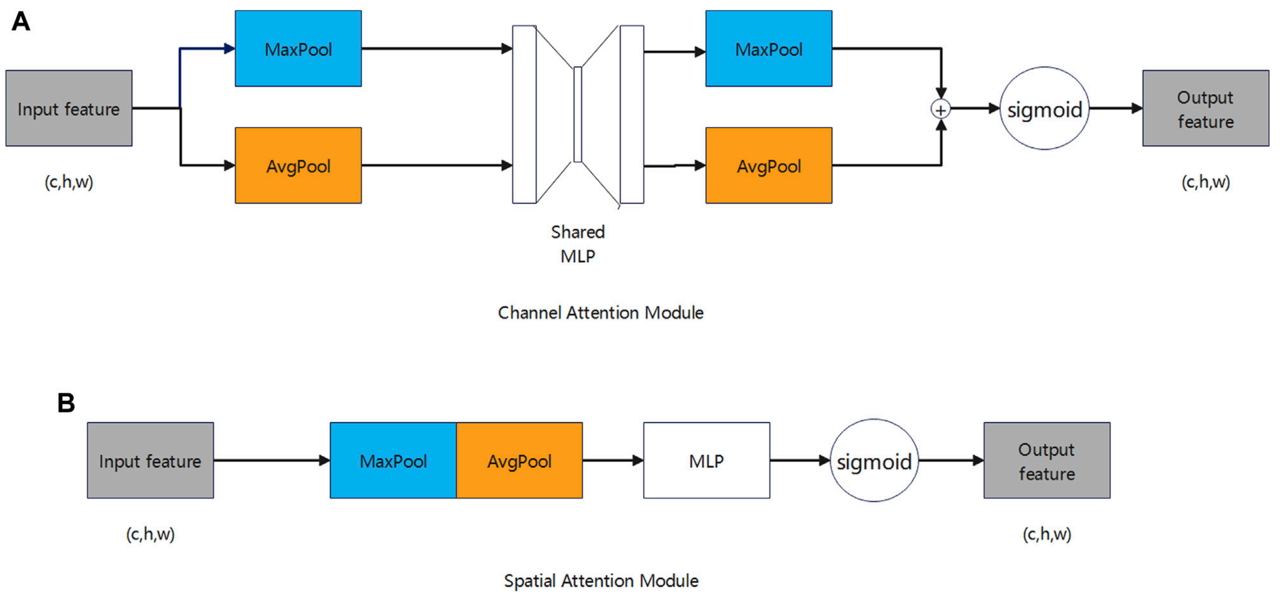


FIGURE 4
SA and CA.

$$Fs = \sigma\left(W_1\left(W_0\left(F^C_{avg}\right)\right) + W_1\left(W_0\left(F^{\,C}_{max}\right)\right)\right)$$

Where σ is the activation function sigmoid, W_1、 W_2 represents two different convolution operations, and F^C_{avg}、 $F^{\,C}_{max}$ represents average pooling and maximum pooling.

Figure 4B illustrates that the spatial attention module captures dependencies across various locations within the feature map. Utilizing channel attention as a basis, the spatial attention module computes the significance weights for each position, subsequently applying these weights to the feature map to attenuate information from irrelevant locations. The computation is described by the following formula:

$$Fs = \sigma\left(f^{7\times 7}\left[F^S_{avg}; F^{\,S}_{max}\right]\right)$$

Where σ is the activation function sigmoid, $f^{7\times 7}$ represents a convolution operation with a convolution kernel size of 7 × 7, and F^S_{avg}、 $F^{\,S}_{max}$ represents average pooling and maximum pooling.

2) Incorporate a Bi-LSTM into the bottleneck layer to capture the inter-slice sequential information within the pulmonary embolism dataset.

Given that medical datasets are challenging to obtain, often contain sparse labeled samples, and limited information, the extraction of maximal information from the dataset for training purposes remains a significant challenge in current neural network research. Considering a dataset of pulmonary embolism obtained through CTPA imaging, conventional convolutional neural networks (CNNs) are limited to extracting intra-slice information from the dataset, failing to capture inter-slice correlations.

In 2018, Chen et al. applied Bi-directional Long Short-Term Memory (Bi-LSTM) [25] to Chinese word segmentation. Figures 5, 6 depict the architectures of Bi-LSTM and LSTM, respectively. This model represents an enhancement over the traditional Long Short-Term Memory (LSTM) architecture [26]. It integrates both forward and reverse LSTM networks, enabling the processing of input sequences progressively at each time step. This allows for the acquisition of information in both forward and reverse temporal directions.

2.2 Dataset

Our study utilized a single dataset: Pulmonary Embolism Dataset 1 (PEA1). This dataset was developed by the Sichuan Provincial People's Hospital.

The PEA1 dataset comprises $15^{-\Delta\Delta CT}$ scans and 1,210 sliced JPG images. The original CT data, stored in DICOM format, undergoes slicing to produce JPG images. Typically, each CT scan encompasses 50–200 axial slices, with dimensions of 512 × 512 pixels for each slice.

The dataset was annotated by professional doctors at the Sichuan Provincial People's Hospital using the LABEL ME software, thus rendering the PEA1 dataset suitable for training and evaluating the segmentation capabilities of our neural network. The dataset was partitioned into two subsets: a training set and a validation set, comprising 1,080 images (90%, 13 cases) and 130 images (10%, 2 cases), respectively.

2.3 Evaluation indicators

The evaluation index is derived from the confusion matrix. The confusion matrix typically comprises four elements: true positives (TP), false negatives (FN), false positives (FP), and true negatives (TN). Each column of the confusion matrix corresponds to a predicted category, with the sum of the column's entries denoting the number of predictions for that category; each row pertains to the actual category, and the sum of the row's entries reflects the total instances within that category.

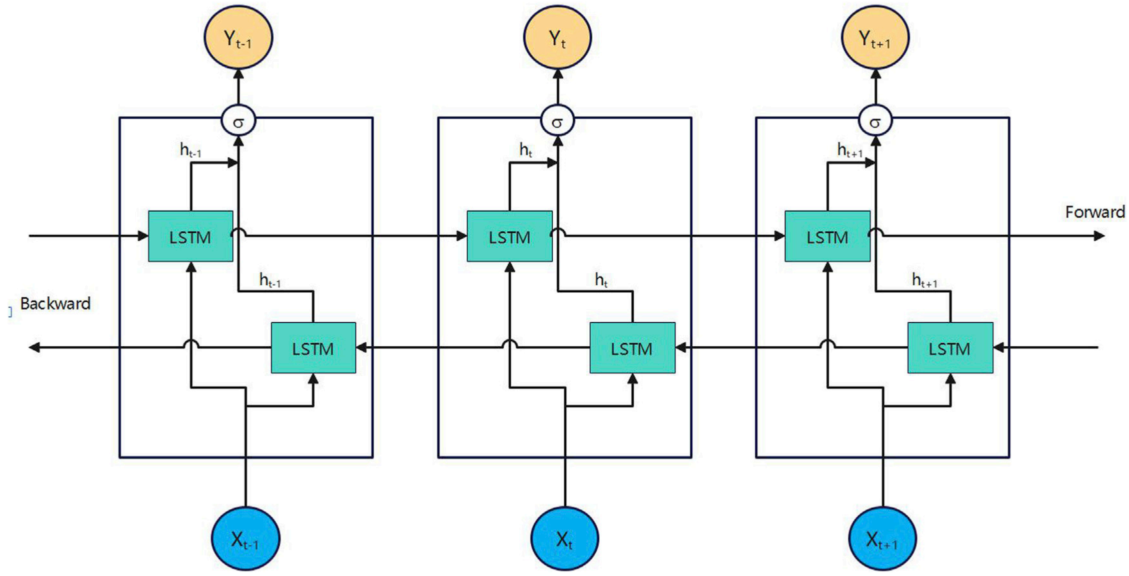


FIGURE 5
Bi-LSTM structure diagram.

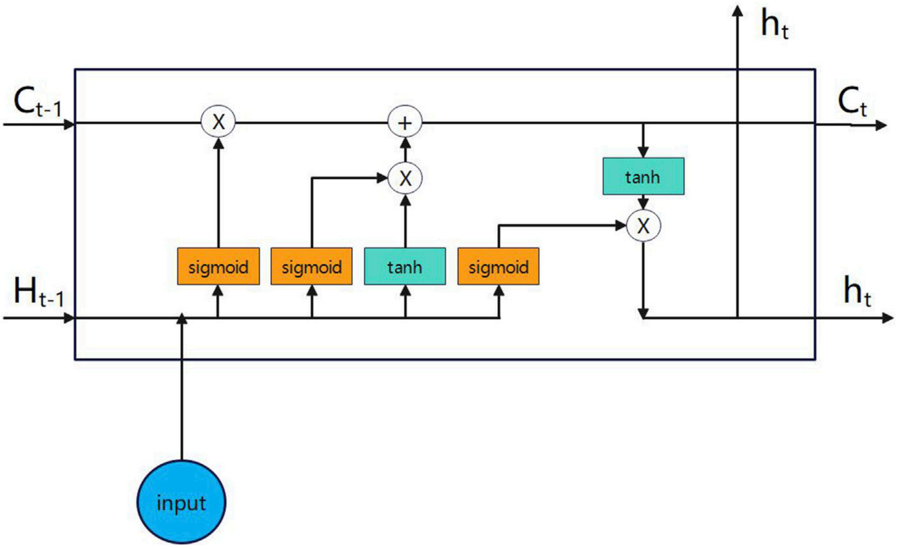


FIGURE 6
LSTM structure diagram.

Table 1 presents the evaluation metrics applied and the corresponding calculation methodologies.

2.4 Loss function

Semantic segmentation of pulmonary embolism presents challenges including severe category imbalance and the difficulty of detecting small-scale targets. Figure 7 illustrates a real CTPA image of a pulmonary embolism patient on the left, and a manually segmented pulmonary embolism mask image by a medical professional on the right. The red area represents the pulmonary embolism lesion, while the black areas denote the background.

Cross-entropy [27] is widely used as a loss function in classification tasks, defined as the measure of disparity between probability distributions for a particular random variable or set of events.

The binary cross-entropy loss function is delineated as:

$$Loss_{BCE}(y, y') = -(ylog(y') + (1 - y)log(1 - y'))$$

Here, y' is the predicted value by the prediction model.

TABLE 1 Evaluation indicators.

Evaluation	Definition	Note
Jac	Jac = (A¡ÉB)/(A¡ÈB)	Jaccard index
DSC	DSC = 2*(A¡ÉB)/(|A|+|B|)	Dice similarity coefficient
Accuracy	Accuracy = (TP + TN)/(TP + TN + FN + FP)	Proportion of samples that predict correctly
mPA@.x	mPA@.x = sum(P(i))/N	Mean pixel accuracy

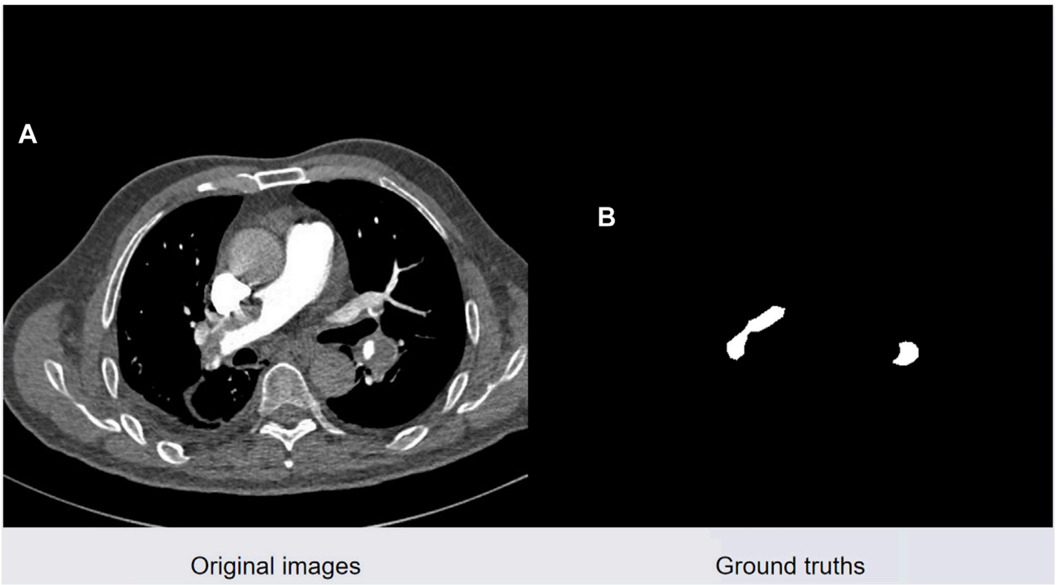


FIGURE 7
Original images and ground truths.

The Focal Tversky Loss [28] combines Focal Loss and Tversky Loss, prioritizing challenging examples by diminishing the influence of simple, common elements, particularly in small regions of interest (ROIs), through the application of a specific coefficient as detailed subsequently:

$$\boldsymbol{FTLoss} = \sum_{C} (\mathbf{1} - \boldsymbol{Tl_c})^{\gamma}$$

This study is the first to integrate BCE (Binary Cross-Entropy) loss and FT (Focal Tversky) loss, aiming to enhance the performance of the neural network model while addressing the challenges of significant category imbalance and small target segmentation in the pulmonary embolism dataset.

$$\boldsymbol{LOSS} = \boldsymbol{Loss_{BCE}} + \boldsymbol{FTLoss}$$

3 Results and discussion

In the course of the study, we performed a series of control experiments, which included the evaluation of established neural network architectures, including U-Net, U-Net++, Attention U-Net, TransU-Net, and the integration of CBAM attention modules into the U-Net framework, as well as the incorporation of Bi-LSTM modules. Additionally, we assessed the employment of a hybrid loss function to address class imbalance and granular target segmentation in the training phase. For each experimental series, the training set was employed to retrain the model, subsequently model segmentation performance on the test set was assessed using the Jaccard index, Dice coefficient, Mean Pixel Accuracy (mPA), and accuracy.

As depicted in Figure 8A, is the original CT slice image of the patient, Figure 8B is the mask image manually segmented by the doctor, Figure 8C is the segmentation image obtained using the original U-Net model, and Figure 8D is the segmentation obtained using the classic model U-Net++ Image, Figure 8E is a segmented image obtained using the classic model Attention U-Net, Figure 8F is a segmented image obtained using the classic model TransU-Net model, Figure 8G is obtained by our method after adding CBAM and Bi-LSTM modules and using a hybrid loss function segmented image.

As Table 2 demonstrates, the integration of the CBAM attention mechanism into the U-Net's backbone, coupled with the replacement of the bottleneck layer by a Bi-LSTM and the adoption of a hybrid loss function, yields the most superior outcomes. When applied to the test set, it yielded a Jaccard index of 0.6958, a Dice score of 0.8193, an MPA of 0.8519, and an accuracy of 0.9993.

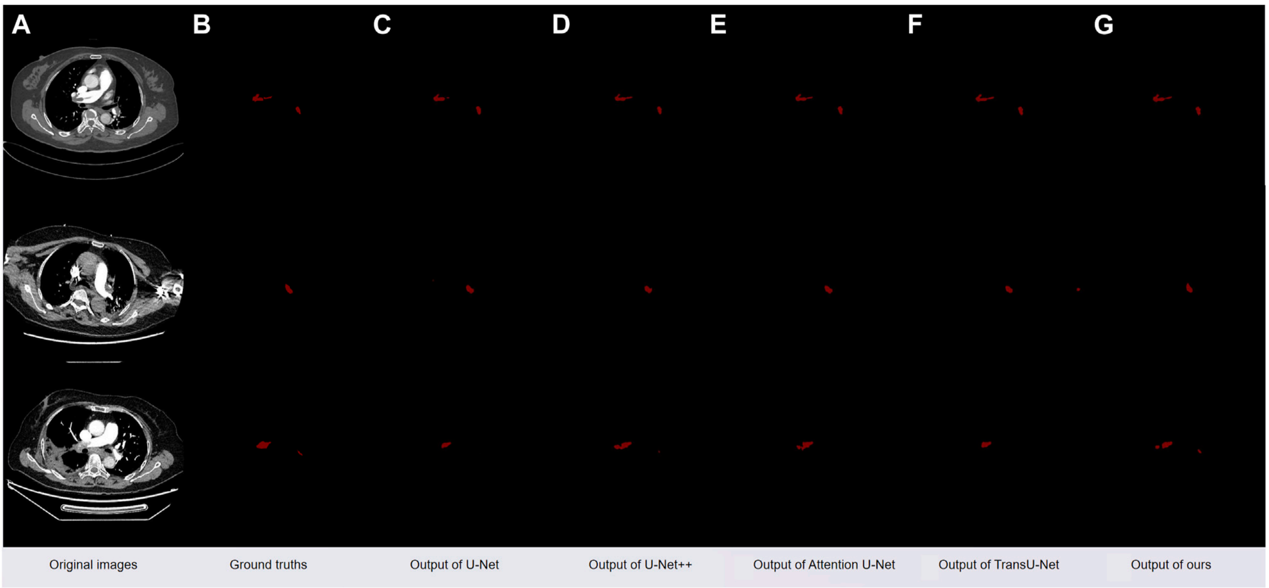


FIGURE 8
Contrast test.

TABLE 2 The results.

Method	Jac	Dice score	Mpa	Accuracy
U-net trained by CEloss	0.6352	0.7769	0.6785	0.995
U-net++ trained by CEloss	0.673	0.8045	0.7915	0.9987
Attention_U-net trained by CEloss	0.6524	0.7896	0.7488	0.9988
Trans U-net trained by CEloss	0.6873	0.8147	0.8005	0.9988
U-net trained by BCE Loss and FT loss	0.6419	0.7819	0.706	0.9986
U-net with Bi-LSTM trained by BCE Loss and FT loss	0.6782	0.8082	0.7767	0.9988
LSCU-Net trained by BCE Loss and FT loss	0.6958	0.8193	0.8519	0.9993

Upon implementing a hybrid loss function tailored for small object segmentation and addressing class imbalance to train the model, the Jaccard index (Jac) exhibited an increase from 0.6352 to 0.6419, the Dice Score improved from 0.7769 to 0.7819, the Mean Pixel Accuracy (MPA) rose from 0.6785 to 0.7060, and the Accuracy enhanced from 0.9950 to 0.9986. This resulted in an overall performance enhancement of the model by approximately 0.6%, demonstrating that the hybrid loss function effectively guides the model in addressing the significant class imbalance issue inherent in the dataset, and in improving the model's performance in segmenting small targets.

Upon integrating the Bi-LSTM module into the original U-Net architecture, the Jaccard index improved from 0.6419 to 0.6782, the Dice coefficient from 0.7819 to 0.8082, the Mean Pixel Accuracy from 0.7060 to 0.7767, and the overall accuracy from 0.9986 to 0.9988. This enhancement bolstered the model's overall performance by approximately 3.6%, demonstrating that replacing the bottleneck layer with the Bi-LSTM module enables the U-Net model to more effectively learn inter-sequential information within the pulmonary embolism dataset, thereby enhancing its feature segmentation capabilities.

Upon integrating the Convolutional Block Attention Module (CBAM) into the U-Net architecture, which already includes the Bi-LSTM module, the Jaccard Index improved from 0.6782 to 0.6958, Dice score from 0.8082 to 0.8193, Mean Pixel Accuracy (MPA) from 0.7767 to 0.8519, and accuracy from 0.9988 to 0.9993. This enhancement resulted in an overall performance improvement of approximately 1.8%, demonstrating that the incorporation of the CBAM module into the U-Net's backbone enables the U-Net model to focus more on critical regions within pulmonary embolism images throughout the training process, thereby enhancing the model's performance and feature learning capabilities.

4 Conclusion

This study integrates computer networking, Internet of Things (IoT), and social networking technologies and utilizes CPSS technology in the medical management of pulmonary embolism patients. We have developed an innovative LSCU-Net model and established a specialized automated segmentation system for pulmonary embolism. The key contributions are summarized as follows:

(1) We integrate the CBAM attention mechanism into the U-net model's backbone to refine the learning strategy, enhancing focus on salient features in both channel and spatial dimensions.
(2) We employ a Bi-LSTM module to supplant the bottleneck layer, enabling the extraction of inter-slice sequential information from the pulmonary embolism dataset.
(3) Throughout the training phase, we utilize a hybrid loss function that merges BCEloss with Focal Tversky Loss, significantly enhancing the model's ability to extract features from highly imbalanced categories and diminutive target datasets.

In future research, we will concentrate our efforts on three primary areas. First, we aim to explore the latest advancements in research on Cyber-Physical-Social Systems (CPSS), which seeks to foster deeper integration among computer networks, the Internet of Things, and social networks, and apply these insights to pulmonary embolism treatment research, with the expectation of achieving significant breakthroughs. Second, considering the limitations of existing pulmonary embolism datasets, particularly in terms of case numbers and significant individual variability, we aim to construct an extensive, detailed, and comprehensive dataset to address these challenges. Finally, we are dedicated to developing a broad spectrum of applications and advanced architectures for medical image segmentation models, while investigating sophisticated and efficient search algorithms, aiming to enhance the models' learning capabilities, and then to promote the integrated application of the Cyber-Physical-Social Systems in the field of medical treatment.

Data availability statement

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

Author contributions

SZ: Conceptualization, Funding acquisition, Project administration, Resources, Supervision, Writing–review and editing. XL: Data curation, Methodology, Software, Validation, Visualization, Writing–original draft. LG: Funding acquisition, Project administration, Resources, Writing–review and editing, Methodology. MX: Data curation, Software, Supervision, Writing–review and editing. TL: Formal Analysis, Funding acquisition, Project administration, Resources, Supervision, Writing–review and editing. SL: Data curation, Formal Analysis, Funding acquisition, Methodology, Project administration, Resources, Writing–review and editing. HY: Data curation, Project administration, Software, Supervision, Writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work has been supported by the Medico-Engineering Cooperation Funds from University of Electronic Science and Technology of China "(No. ZYGX2021YGLH211)" and Sichuan Science and Technology Program "(No. 2022YFG0176)".

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Wang FY. The emergence of intelligent enterprises: from CPS to CPSS. *IEEE Intell Syst* (2010) 25:85–8. doi:10.1109/mis.2010.104

2. Jiang H, Xu W. How to find your appropriate doctor: an integrated recommendation framework in big data context. In: IEEE Symposium on Computational Intelligence in Healthcare and e-health (CICARE); December, 2014; Orlando, FL, USA (2014). p. 92–5.

3. Xu Q, Su ZS, Shui Y. Green social CPS based e-healthcare systems to control the spread of infectious diseases. In: Proc. IEEE International Conference on Communications (ICC); 20-24 May 2018; Kansas City, MO (2018). p. 1–5.

4. Cheng X. *Pulmonary vascular disease*. Beijing: Beijing Medical University and Union Medical College of China Press (1993).

5. Byrnes JR, Wolberg AS. New findings on venous thrombogenesis. *Hamostaseologie* (2017) 37(1):25–35. doi:10.5482/HAMO-16-09-0034

6. Aggarwal A, Fullam L, Brownstein AP, Maynard GA, Ansell J, Varga EA, et al. Deep vein thrombosis (DVT) and pulmonary embolism (PE): awareness and prophylaxis practices reported by patients with cancer. *Cancer Invest* (2015) 33(9):405–10. doi:10.3109/07357907.2015.1048871

7. Samkoff JS, Comstock GW. Epidemiology of pulmonary embolism: mortality in a general population. *Am J Epidemiol* (1981) 114:488–96. doi:10.1093/oxfordjournals.aje.a113214

8. Ferrari E, Baudouy M, Morand P. Epidemiology of pulmonary embolism. *Arch Mal Coeur Vaiss* (1995) 88:1687–91. 11 Suppl.

9. Rathbun S. Cardiology patient pages. The Surgeon General's call to action to prevent deep vein thrombosis and pulmonary embolism. *Circulation* (2009) 119(15): 480–2. doi:10.1161/circulationaha.108.841403

10. Nisio D, van Es N, Büller N. Deep vein thrombosis and pulmonary embolism. *Lancet* (2016) 388(10063):3060–3073. doi:10.1016/S0140-6736(16)30514-1

11. Grosse SD, Nelson RE, NyarkoRichardson KALC, Raskob GE. The economic burden of incident venous thromboembolism in the United States: a review of estimated attributable healthcare costs. *Thromb Res* (2016) 137:3–10. doi:10.1016/j.thromres.2015.11.033

12. Goldhaber SZ. *Deep venous thrombosis and pulmonary thromboembolism* (2008).

13. Goldhaber SZ, Visani L, Rosa MD. Acute pulmonary embolism: clinical outcomes in the international cooperative pulmonary embolism registry (icoper). *Lancet* (1999) 353(9162):1386–9. doi:10.1016/s0140-6736(98)07534-5

14. Nikulina N, Terekhovskaya YV. Epidemiology of pulmonary embolism in today's context: analysis of incidence, mortality and problems of their study. *Russ J Cardiol* (2019)(6) 103–8. doi:10.15829/1560-4071-2019-6-103-108

15. Colin W, Ilan G, Susan S, Scott M, Logr G, Ayman E, et al. Effect of a multidisciplinary pulmonary embolism response team on patient mortality. *Am J Cardiol* (2021) 161:102–7. doi:10.1016/j.amjcard.2021.08.066

16. Sezgin M, Sankur B. Survey over image thresholding techniques and quantitative performance evaluation. *J Electron Imaging* (2004) 13(1):146–68. doi:10.1117/1.1631315

17. Xu J, Tu L, Zhang Z, et al. Tongue quality and tongue coating recognition based on image area segmentation method. *Shanghai:Journal Shanghai Univ Traditional Chin Med* (2009) 23:42–5.

18. Cumani A. Edge detection in multispectral images. *Graphical Models Image Process* (1991) 53(1):40–51. doi:10.1016/1049-9652(91)90018-f

19. Piotr Z. Atlas-based segmentation in extraction of knee joint bone structures from CT and MR. *Sensors* (2022) 22(22):8960. doi:10.3390/s22228960

20. Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition; June 7 2015 to June 12 2015; Boston, MA, USA (2015). p. 3431–40.

21. Ronneberger O, Fischer P, Brox T. U-net: convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention; October 5-9, 2015; Munich, Germany. Cham: Springer (2015). 234–41.

22. Shrivastava A, Gupta A, Girshick R. Training region-based object detectors with online hard example mining. In: IEEE Conference on Computer Vision and Pattern Recognition; June 26 2016 to July 1 2016; Las Vegas, NV, USA. IEEE Computer Society (2016). 761–9.

23. Tian M, Wang W, Chen Y. Attention is all you need: an interpretable transformer-based asset allocation approach. *Int Rev Financial Anal* (2023) 90:102876. doi:10.1016/J.IRFA.2023.102876

24. Woo S, Park J, Lee J-Y, Kweon IS. Cbam: convolutional block attention module. In: Proceedings of theEuropean conference on computer vision (ECCV); September 8-14, 2018; Munich, Germany (2018). 3–19.

25. Chen J, Weihua LI, Chen JI, Xuze J, Yanbu G. Bi-directional long short-term memory neural networks for Chinese word segmentation. *J Chin Inf Process* (2018).

26. Hochreiter S, Schmidhuber J. *Long short-term memory* (1997).

27. Yi-De M, Qing L, Zhi-Bai Q. Automated image segmentation using improved PCNN model based on cross-entropy. In: Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing; October 20-22, 2004; Hong Kong. IEEE (2005). p. 743–6.

28. Abraham N, Khan NM. "A novel focal tversky loss function with improved attention U-Net for lesion segmentation," in IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019), Venice, Italy (2019). 683–687. doi:10.1109/ISBI.2019.8759329

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 20 February 2024
DOI 10.3389/fphy.2024.1349284

Check for updates

OPEN ACCESS

EDITED BY
Amin Ul Haq,
University of Electronic Science and Technology of China, China

REVIEWED BY
Md Belal Bin Heyat,
Westlake University, China
Inam Ullah,
Shandong Jianzhu University, China

*CORRESPONDENCE
Hua Li,
✉ lh1@ustl.edu.cn

RECEIVED 04 December 2023
ACCEPTED 31 January 2024
PUBLISHED 20 February 2024

CITATION
Li B, Li H, Sun Q, Lv R and Yan H (2024), Dynamics analysis and optimal control study of uncertain information dissemination model triggered after major emergencies.
Front. Phys. 12:1349284.
doi: 10.3389/fphy.2024.1349284

Dynamics analysis and optimal control study of uncertain information dissemination model triggered after major emergencies

Bowen Li[1], Hua Li[2]*, Qiubai Sun[2], Rongjian Lv[1] and Huining Yan[1]

[1]School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, China, [2]School of Business Administration, University of Science and Technology Liaoning, Anshan, China

In order to effectively prevent and combat online public opinion crises triggered by major emergencies, this paper explores the dissemination mechanism of uncertain information on online social platforms. According to the decision-making behavior of netizens after receiving uncertain information, they are divided into eight categories. Considering that there will be a portion of netizens who clarify uncertain information after receiving it, this paper proposes a SEFTFbTbMR model of uncertain information clarification behavior. The propagation dynamics equations of the model are given based on the theory of differential equations, the basic regeneration number R_0 of the model is calculated, and the existence and stability of the equilibrium point of the model are analyzed. The theoretical analysis of the model is validated using numerical simulation software, and sensitivity analysis is performed on the parameters related to R_0. In order to reduce the influence caused by uncertain information, the optimal control strategy of the model is proposed using the Hamiltonian function. It is found that the dissemination of uncertain information among netizens can be suppressed by strengthening the regulation of social platforms, improving netizens' awareness of identifying the authenticity of information, and encouraging netizens to participate in the clarification of uncertain information. The results of this work can provide a theoretical basis for future research on the uncertain information dissemination mechanism triggered by major emergencies. In addition, the results can also provide methodological support for the relevant government departments to reduce the adverse effects caused by uncertain information in the future.

KEYWORDS

major emergencies, uncertain information, classical epidemic transmission model, optimal control model, uncertain information clarification behavior

1 Introduction

Digital new media platforms are essentially unbounded, interactive, and anonymous, which brings a significant degree of convenience to users but also introduces certain hidden dangers. When a major emergency occurs, various network clusters form to discuss the event. Although the internet users within these clusters are eager to obtain relevant information about the event, its uncertainty and urgency often mean that the relevant

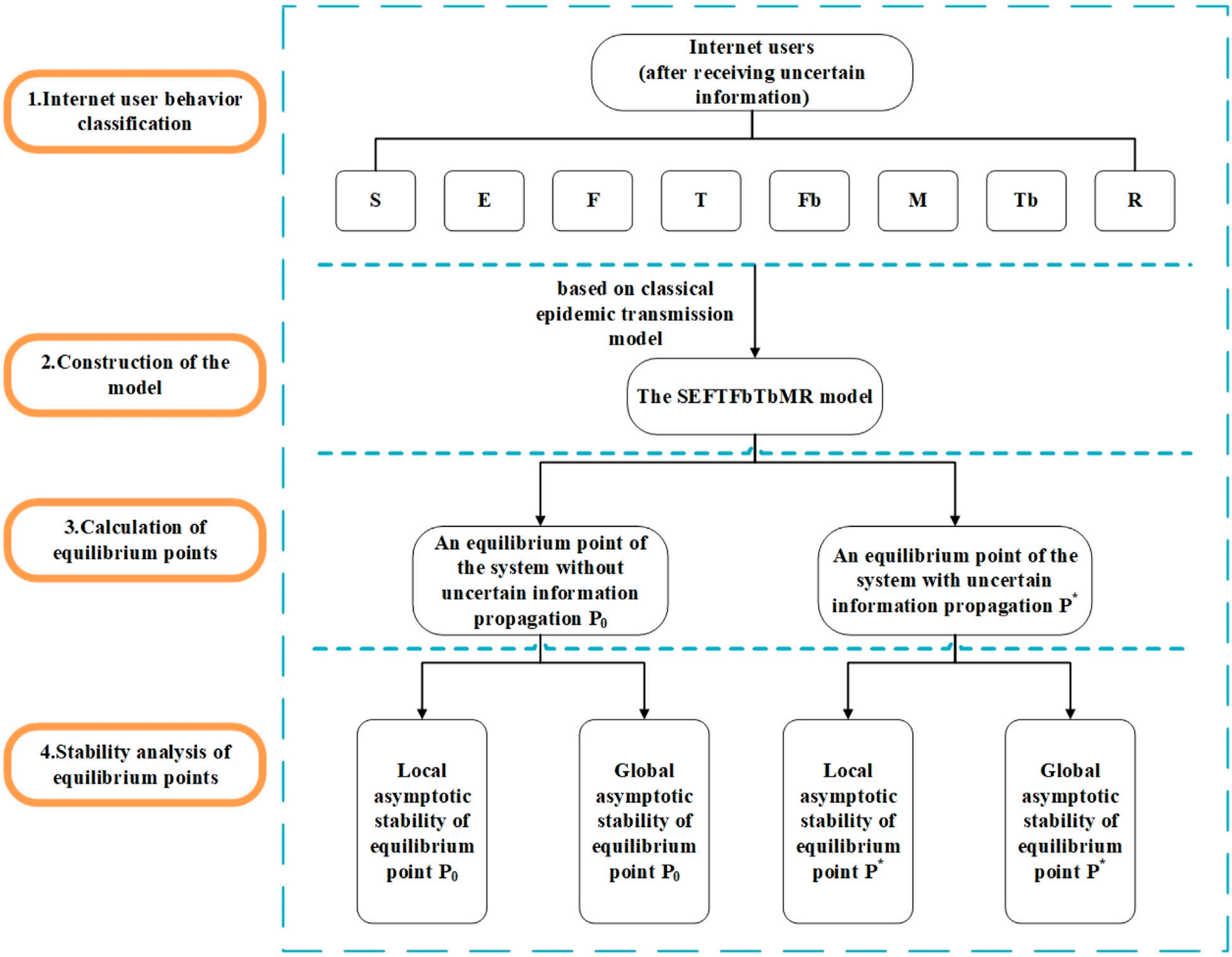

1.Internet user behavior classification
2.Construction of the model
3.Calculation of equilibrium points
4.Stability analysis of equilibrium points
Internet users (after receiving uncertain information)
S
E
F
T
Fb
M
Tb
R
based on classical epidemic transmission model
The SEFTFbTbMR model
An equilibrium point of the system without uncertain information propagation P_0
An equilibrium point of the system with uncertain information propagation P^*
Local asymptotic stability of equilibrium point P_0
Global asymptotic stability of equilibrium point P_0
Local asymptotic stability of equilibrium point P^*
Global asymptotic stability of equilibrium point P^*

FIGURE1
Organizational diagram of the current study.

departments are unable to announce details to the public in the early stages of the response. Thus, during this information window, some internet users use digital new media to disseminate uncertain information, which may cause unnecessary panic among uninformed internet users, possibly leading to social disquiet and unrest. Thus, to reduce the secondary effects caused by the dissemination of uncertain information after major emergencies, it is critical to construct a model of dissemination of uncertain information and analyze the mechanisms whereby such information is transmitted.

This paper develops an epidemic propagation dynamics model and uses optimal control theory to analyze the delivery mechanism of uncertain information dissemination among internet users. First, based on different decision-making behaviors, netizens are categorized into eight groups: unknowns *S*, thinkers *E*, uncertain information publishers *F*, clarifiers of uncertain information *T*, internet users who believe uncertain information *Fb*, internet users who only believe true information *Tb*, internet users who do not believe any online information *M*, and information immunizers *R*. We then construct the SEFTFbTbMR uncertain information dissemination model. Second, the model is solved to find the basic regeneration number R_0 of the system, and the equilibrium points P_0 and P^* that exist without and with uncertain information dissemination, respectively, are calculated. The stability of points P_0 and P^* is then analyzed, and numerical simulations are conducted using Matlab 2017b to verify the theoretical derivations. Finally, to control the scale of uncertain information dissemination and increase the proportion of thinkers and clarifiers of uncertain information, an optimal control model is established based on the SEFTFbTbMR uncertain information dissemination model.

The main contributions of the research reported in this paper are as follows: 1) Considering that there will be some netizens who will exhibit behaviors such as clarifying or re-disseminating the uncertain information after receiving it, this paper divides the netizens into eight categories according to decision-making behavior in the process of uncertain information dissemination. 2) During the construction of the model, we consider not only the dissemination of uncertain information but also the dissemination of true information that clarifies the uncertain information. 3) In order to reduce the influence caused by uncertain information, the Hamiltonian function is utilized to propose the optimal control strategy of the model. The research in this paper provides a theoretical basis for dealing with the uncertain information

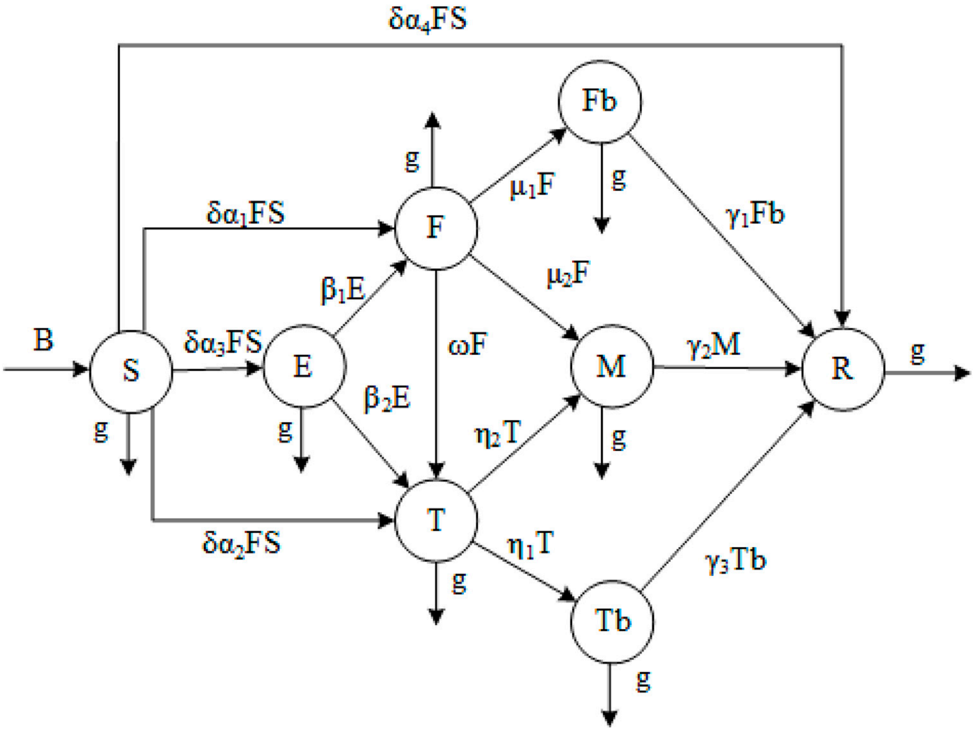


FIGURE 2
Flowchart of the propagation dynamics equations for the SEFTFbTbMR model.

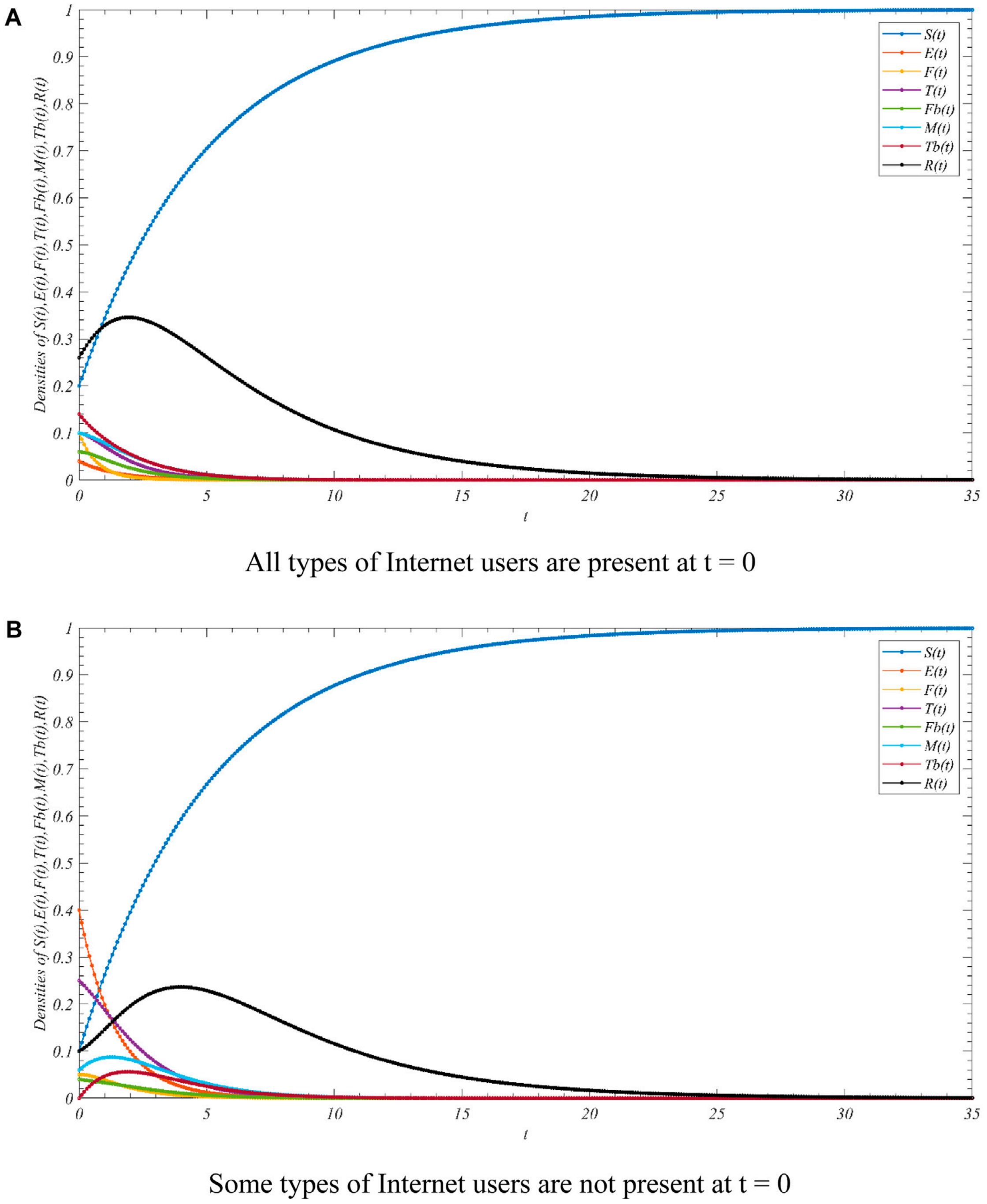
dissemination mechanism triggered by major emergencies, and the related conclusions provide methodological support for reducing the adverse effects of uncertain information.

2 Related work

Since the outbreak of COVID-19, various studies have examined major emergencies from different perspectives. Tan [1], Yao [2], Gao [3], and Zhang [4] conducted studies from the perspective of emergency management after the occurrence of a major emergency. Other scholars have investigated the impact of major emergencies, such as Ukwuoma [5], Benifa [6], Asif [7], and Fazmiya [8], who analyzed the physical effects of major emergencies on humans from a medical perspective, using COVID-19 as an example. Mo et al. [9] argued that major emergencies have both an emotional as impact as well as a huge economic impact. Cheng et al. [10] found that secondary disasters of major emergencies can have an impact on international oil prices. Yang et al. [11] concluded that major emergencies can seriously affect the public's emotions and cause a certain amount of panic. Similarly, De las Heras-Pedrosa et al. [12] argued that major emergencies can also have serious psychological effects on the public. Atehortua et al. [13] reported that the occurrence of major emergencies is followed by large amounts of uncertain information emerging on social networks, leading to the spread of panic. Jalan et al. [14] argued that, after a major emergency, the uncertain information disseminated across new media can cause more panic in the public than traditional media reports. Zhang et al. [15] found that the subsequent control of major emergencies can be hampered by the dissemination of uncertain information after the occurrence of a major emergency.

In the study of uncertain information, it is critical to examine the various actors in the process of information dissemination. Crokidakis [16], Zhao [17], and Yin [18] studied the crucial role of social media in the dissemination of uncertain information. Allington et al. [19] argued that social media platforms are the main disseminators of uncertain information, and Centola [20] showed that netizens tend to believe information when it is received from several different sources. Zhang et al. [21] used the behaviors of online media, internet users, and the government in response to the Chinese COVID-19 Shuanghuanglian incident as an example to examine the dissemination of information. Choi [22] argued that although opinion leaders play a driving role in the dissemination of information, they are not typically its creators. Studies have examined the mechanisms whereby uncertain information is disseminated, including that of Li et al. [23], who examined the information dissemination process under major emergencies, and that of Li et al. [24], who investigated the propagation of uncertain information following an incident. Wei [25] analyzed the propagation process of uncertain information using the theory of heat conduction in physics. Litou et al. [26] studied how to increase the rate of information dissemination at the lowest cost. Wang et al. [27] argued that a higher-status initial disseminator could achieve a faster rate of dissemination, whereas Hong et al. [28] showed that centralizing the release of truthful information effectively reduces the dissemination rate of uncertain information among netizens.

The Susceptible Infected Recovered (SIR) model [29] was first introduced in 1927 by Kermack and McKendrick to study the transmission mechanism of epidemics in populations using a

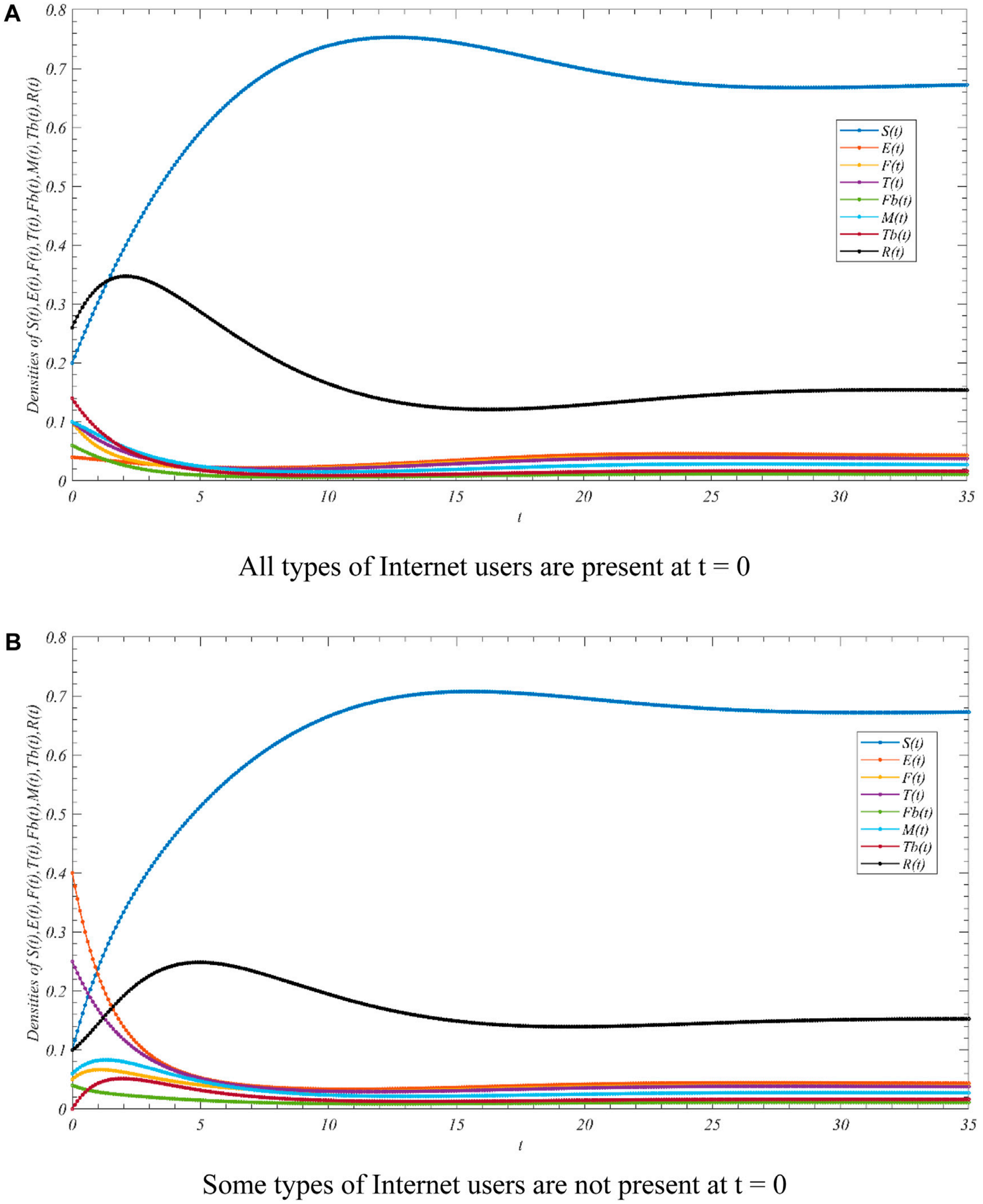


All types of Internet users are present at t = 0

Some types of Internet users are not present at t = 0

FIGURE 3
Evolution of Internet user populations when $R_0 < 1$. **(A)** all types of Internet users are present at t = 0. **(B)** some types of Internet users are not present at t = 0.

kinetic approach. Their mathematical model underwent further enhancement in 1932 [30] and 1933 [31]. Subsequently, researchers have continued to develop and extend this model to account for the dynamics of infectious diseases, and have progressively merged it with disciplines such as mathematics, sociology, complexity science, cybernetics, and computer science [32–34]. The epidemic transmission model has been extensively utilized in studying cross-disciplinary information transmission owing to the similarity of the transmission pattern of information with that of epidemics [35]. [36] developed the M-SDI model, which uses public comments to assess the credibility of online information; in a subsequent study, they introduced the SRFI model [37], which uses numbers of reads and retweets to measure uncertainty in online content. Rui et al. [38] proposed the SPIR model based on discrete-time dynamics. Trpevski et al. [39] developed an uncertain information dissemination model with two different acceptance probabilities based on the SIS model. Zan [40] constructed the DSIR and C-DSIR models by considering the simultaneous existence of multiple uncertain pieces of information in the real world.



All types of Internet users are present at t = 0

Some types of Internet users are not present at t = 0

FIGURE 4
Evolution of Internet user populations when $R_0 > 1$ **(A)** all types of Internet users are present at t = 0. **(B)** some types of Internet users are not present at t = 0.

Based on the above-mentioned studies, we find that most scholars often assume that only uncertain information is disseminated among netizens in the process of researching the dissemination mechanism of uncertain information, and the decision-making behavior of netizens after receiving uncertain information is relatively simple. However, in reality, due to the characteristics of digital media technology, Internet users can not only receive uncertain information but also real information that clarifies uncertain information. Moreover, with the continuous improvement of their own quality, some netizens, when faced with uncertain information, will make a judgment by investigating and collecting evidence or thinking for moment, thus spontaneously clarifying the uncertain information and ultimately choosing to publish real information. Considering the above realities, this paper divides netizens into eight categories according to different decision-making behaviors in the process of uncertain information dissemination. When constructing the model, we consider not only the dissemination of uncertain information but also the

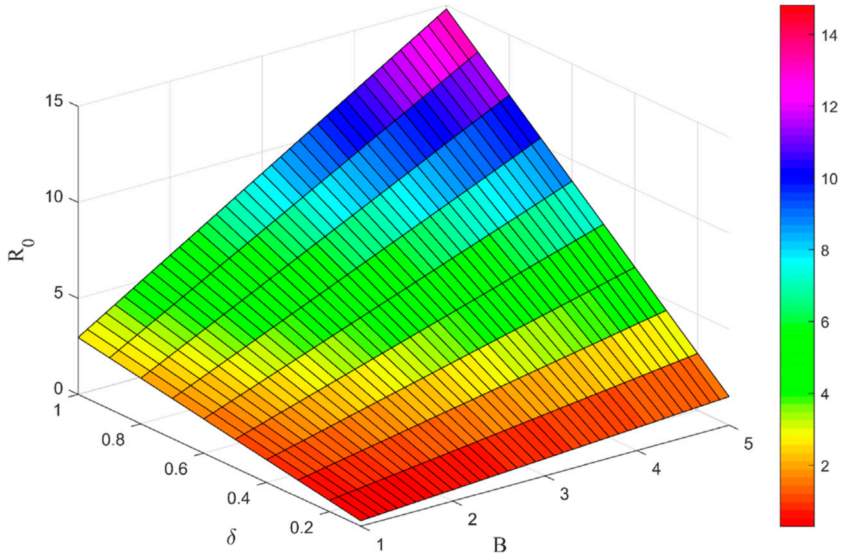


FIGURE 5
Effect of variations in δ and B on R_0.

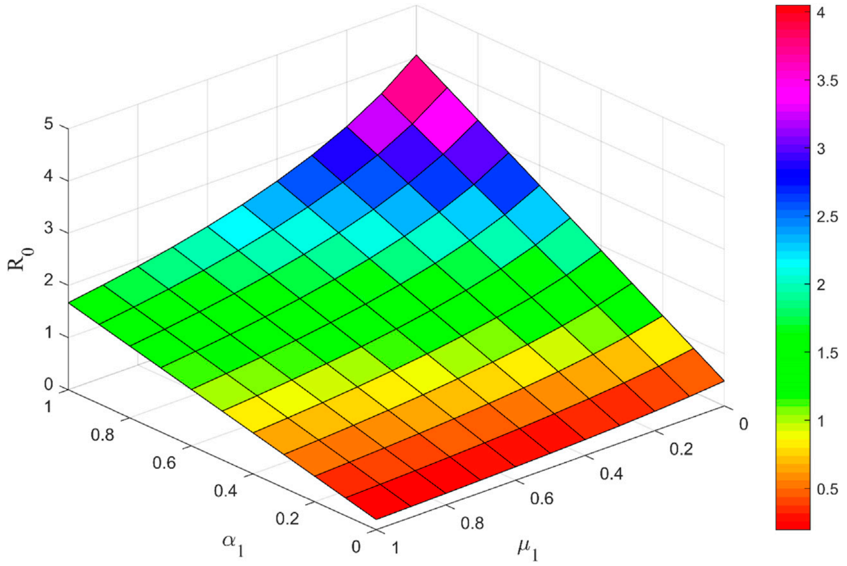


FIGURE 6
Effect of variations in α_1 and μ_1 on R_0.

dissemination of truthful information that clarifies the uncertain information.

3 The model

The workflow of the current study in this paper is shown in Figure 1 below, which consists of four main steps: 1) Internet user behavior classification. 2) Construction of the model. 3) Calculation of equilibrium points. 4) Stability analysis of equilibrium points.

The model construction and derivation process in this paper follows the literature [41]. The specific steps are as follows: First, we construct the SEFTFbTbMR model based on the classical infectious disease model. To find the equilibrium point of the model, we calculate the basic regeneration number of the system, R0, using the next-generation matrix method [42]. We then judge the local asymptotic stability and global asymptotic stability of the equilibrium point using the Routh-Hurwitz criterion [43] and Liapunov's second method [44], respectively. Finally, we conduct numerical simulations of the model.

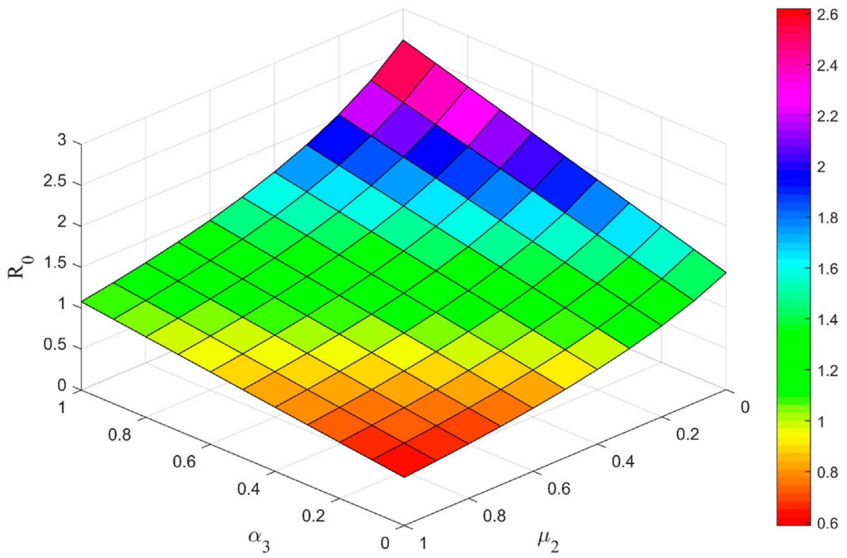


FIGURE 7
Effect of variations in α_3 and μ_2 on R_0.

3.1 Construction of the uncertain information dissemination model

The classical epidemic transmission dynamics model mentioned in the literature [29] is given below:

$$\begin{cases} \frac{dS}{dt} = -\beta SI \\ \frac{dI}{dt} = \beta SI - \gamma I \\ \frac{dR}{dt} = \gamma I \end{cases}, \quad (1)$$

where S refers to susceptible, I refers to infected, R refers to recovered, β is the rate of infection, and γ is the rate of recovery.

In this paper, on the basis of the classical epidemic disease dissemination dynamics model given as Eq. 1, the Internet users in the digital new media platform are divided into eight categories according to their behavior after receiving uncertain information as follows:

(1) The unknowns S are ordinary internet users who have not received uncertain information;
(2) The thinkers E are internet users who, after receiving uncertain information, think about the veracity of this information before acting (i.e., neutral actors);
(3) The uncertain information publishers F are Internet users who, after receiving uncertain information, choose to disseminate the uncertain information;
(4) The clarifiers of uncertain information T are Internet users who, after receiving uncertain information, choose to investigate, obtain evidence, and release true information;
(5) The internet users who believe in uncertain information, Fb;
(6) The internet users who only believe in truthful information, Tb;
(7) The internet users who do not believe any information, M;
(8) The information immunizers R are Internet users who are not interested in either uncertain or true information.

Combining the above eight categories of Internet users with different decision-making behaviors, this paper amends the classical infectious disease SIR model given as Eq. 1 to construct an uncertain information dissemination model, which is defined as the SEFTFbTbMR model. The propagation rules of the SEFTFbTbMR model are as follows:

(I) At moment t, the total number of netizens in the network is $N(t)$, comprising the eight groups identified above, that is,

$$S(t) + E(t) + F(t) + T(t) + Fb(t) + Tb(t) + M(t) + R(t) = N(t). \quad (2)$$

(II) B individuals enter the system per unit time. These individuals are ordinary internet users who have not received uncertain information (i.e., the transfer rate of unknown internet users in the system is B). Individuals in the eight groups exit the system at the same removal rate g.
(III) The propagation rate of uncertain information is δ. When S makes contact with F and receives some item of uncertain information, one of the following four choices is made: S chooses to propagate the uncertain information immediately, thus becoming a new member of population F; S chooses to clarify the uncertain information immediately, thus becoming a new member of population T; S chooses to think appropriately before acting, thus becoming a new member of population E; or S does not take any

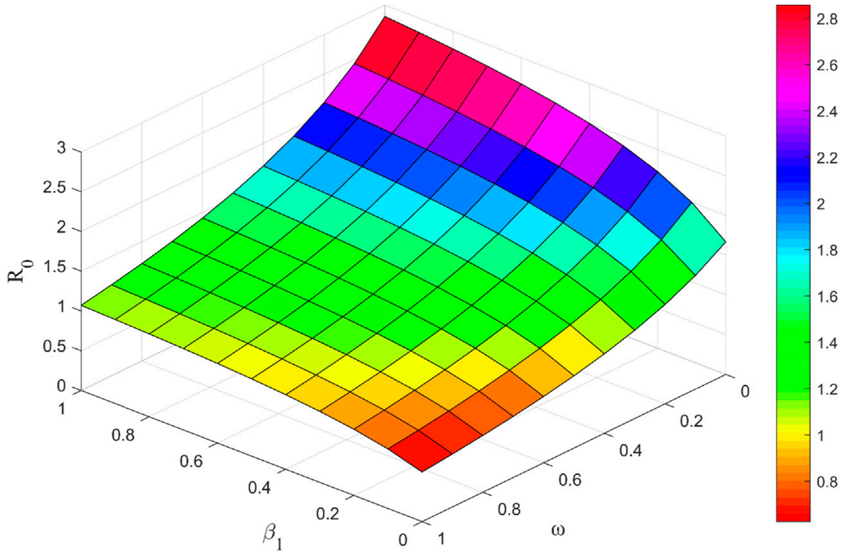


FIGURE 8
Effect of variations in β_1 and ω on R_0.

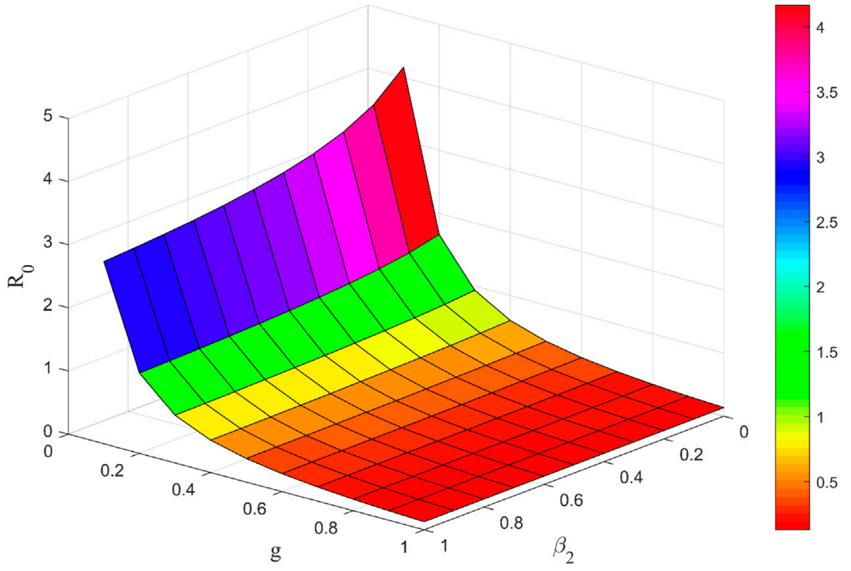


FIGURE 9
Effect of variations in g and β_2 on R_0.

interest in the uncertain information and chooses to withdraw from the discussion, thus becoming a new member of population R. The proportions of transformations into F, T, E, and R are α_1, α_2, α_3, and α_4, respectively; where $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$.

(IV) Members of population E are converted to population F with probability β_1 and to population T with probability β_2. Members of population F are converted to population T with probability ω after learning the true information. As the uncertain information and the true information in the system come into contact, members of population F will be converted to population Fb with probability μ_1 and to population M with probability μ_2. Members of T will be converted to population Tb with probability η_1 and to population M with probability η_2. As the information is time-sensitive, populations Fb, M, and Tb convert to population R with probabilities γ_1, γ_2, and γ_3, respectively.

Based on the above propagation rules, the dynamics for the SEFTFbTbMR model can be written as follows:

$$\begin{cases} \dfrac{dS}{dt} = B - \delta FS - gS \\ \dfrac{dE}{dt} = \alpha_3 \delta FS - \beta_1 E - \beta_2 E - gE \\ \dfrac{dF}{dt} = \alpha_1 \delta FS + \beta_1 E - \omega F - \mu_1 F - \mu_2 F - gF \\ \dfrac{dT}{dt} = \alpha_2 \delta FS + \beta_2 E + \omega F - \eta_1 T - \eta_2 T - gT \\ \dfrac{dFb}{dt} = \mu_1 F - \gamma_1 Fb - gFb \\ \dfrac{dM}{dt} = \mu_2 F + \eta_2 T - \gamma_2 M - gM \\ \dfrac{dTb}{dt} = \eta_1 T - \gamma_3 Tb - gTb \\ \dfrac{dR}{dt} = \gamma_1 Fb + \gamma_2 M + \gamma_3 Tb + \alpha_4 \delta FS - gR \end{cases}. \tag{3}$$

The flow chart of the propagation dynamics equations for the SEFTFbTbMR model is shown in Figure 2.

Based on Equations 2, 3, we obtain

$$\frac{dN(t)}{dt} = B - gN(t). \tag{4}$$

When $N_0 = N(0)$, Eq. 4 yields $N(t) = \left(N_0 - \frac{B}{g}\right)e^{-gt} + \frac{B}{g}$, namely, $\lim_{t\to\infty} N(t) = \frac{B}{g}$. Thus, we can judge the positive invariant set of system (3) to be

$$\Omega = \left\{ \begin{array}{l} S, E, F, T, Fb, M, Tb, R \in \mathbb{R}_8^+: \\ 0 \le S + E + F + T + Fb + M + Tb + R \le \dfrac{B}{g} \end{array} \right\}. \tag{5}$$

3.2 Calculation of equilibrium points

Summing up the equilibrium equations in system (3), it can be concluded that there exists an equilibrium point of the system without uncertain information propagation, which is defined as

$$P_0 = \left(\frac{B}{g}, 0, 0, 0, 0, 0, 0, 0\right). \tag{6}$$

Based on the fundamental regeneration number in propagation dynamics [42], we define the total number of times a member of population F transforms a member of population S into a new member of population F during the average propagation period as the fundamental regeneration number of uncertain information propagation, denoted as R_0. The R_0 of the system can be calculated by the next-generation matrix method. Letting $X = (F, E, T, Fb, M, Tb, R, S)^{\mathrm{T}}$, system (3) can be rewritten as

$$\frac{dX}{dt} = \mathcal{F}(X) - \mathcal{V}(X), \tag{7}$$

where

$$\mathcal{F}(X) = \left(\alpha_1 \delta FS,\ \alpha_3 \delta FS,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\right)^{\mathrm{T}}, \tag{8}$$

$$\mathcal{V}(X) = \begin{pmatrix} -\beta_1 E + \omega F + \mu_1 F + \mu_2 F + gF \\ \beta_1 E + \beta_2 E + gE \\ -\alpha_2 \delta FS - \beta_2 E - \omega F + \eta_1 T + \eta_2 T + gT \\ -\mu_1 F + \gamma_1 Fb + gFb \\ -\mu_2 F - \eta_2 T + \gamma_2 M + gM \\ -\eta_1 T + \gamma_3 Tb + gTb \\ -\gamma_1 Fb - \gamma_2 M - \gamma_3 Tb - \alpha_4 \delta FS + gR \\ -B + \alpha_1 \delta FS + \alpha_2 \delta FS + \alpha_3 \delta FS + \alpha_4 \delta FS + gS \end{pmatrix}. \tag{9}$$

The Jacobian matrix for Equations 8, 9 at equilibrium point (6) is calculated as follows:

$$D\mathcal{F}(X) = \begin{bmatrix} F & 0 \\ 0 & 0 \end{bmatrix}, \tag{10}$$

$$D\mathcal{V}(X) = \begin{bmatrix} V & 0 \\ V_1 & V_2 \end{bmatrix}, \tag{11}$$

where

$$F = \begin{bmatrix} \alpha_1 \delta \dfrac{B}{g} & 0 \\ \alpha_3 \delta \dfrac{B}{g} & 0 \end{bmatrix}, \tag{12}$$

$$V = \begin{bmatrix} \omega + \mu_1 + \mu_2 + g & -\beta_1 \\ 0 & \beta_1 + \beta_2 + g \end{bmatrix}. \tag{13}$$

According to the literature [45], the R_0 of system (3) is equivalent to the spectral radius of the matrix FV^{-1}:

$$R_0 = \rho\left(FV^{-1}\right) = \frac{B\delta\left[\alpha_1\left(\beta_1 + \beta_2 + g\right) + \alpha_3 \beta_1\right]}{g\left(\beta_1 + \beta_2 + g\right)\left(g + \mu_1 + \mu_2 + \omega\right)}. \tag{14}$$

From the definition of R_0, there exists an equilibrium point of the system with uncertain information propagation when $R_0 > 1$, which can be expressed as

$$P^* = (S^*, E^*, F^*, T^*, Fb^*, M^*, Tb^*, R^*). \tag{15}$$

Equilibrium point (15) should satisfy

$$\begin{cases} B - \alpha_1 \delta F^* S^* - \alpha_2 \delta F^* S^* - \alpha_3 \delta F^* S^* - \alpha_4 \delta F^* S^* - gS^* = 0 \\ \alpha_3 \delta F^* S^* - \beta_1 E^* - \beta_2 E^* - gE^* = 0 \\ \alpha_1 \delta F^* S^* + \beta_1 E^* - \omega F^* - \mu_1 F^* - \mu_2 F^* - gF^* = 0 \\ \alpha_2 \delta F^* S^* + \beta_2 E^* + \omega F^* - \eta_1 T^* - \eta_2 T^* - gT^* = 0 \\ \mu_1 F^* - \gamma_1 Fb^* - gFb^* = 0 \\ \mu_2 F^* + \eta_2 T^* - \gamma_2 M^* - gM^* = 0 \\ \eta_1 T^* - \gamma_3 Tb^* - gTb^* = 0 \\ \gamma_1 Fb^* + \gamma_2 M^* + \gamma_3 Tb^* + \alpha_4 \delta F^* S^* - gR^* = 0 \end{cases}. \tag{16}$$

A specific expression for equilibrium point (15) in terms of R_0 can be obtained by performing calculations on system (16):

$$S^* = \frac{\left(\beta_1 + \beta_2 + g\right)\left(g + \mu_1 + \mu_2 + \omega\right)}{\delta\left[\alpha_1\left(\beta_1 + \beta_2 + g\right) + \alpha_3 \beta_1\right]} = \frac{B}{gR_0}, \tag{17}$$

$$E^* = \frac{\delta \alpha_3 F^* S^*}{\beta_1 + \beta_2 + g} = \frac{\alpha_3 B\left(R_0 - 1\right)}{\left(\beta_1 + \beta_2 + g\right)R_0}, \tag{18}$$

$$F^* = \frac{B - gS^*}{\delta S^*} = \frac{g\left(R_0 - 1\right)}{\delta}, \tag{19}$$

$$T^{*} = \frac{\beta_2 E^{*} + \alpha_2 \delta F^{*} S^{*} + \omega F^{*}}{g + \eta_1 + \eta_2} = \frac{(R_0 - 1)}{g + \eta_1 + \eta_2}\left[\frac{\beta_2 \alpha_3 B}{(\beta_1 + \beta_2 + g)R_0} + \frac{\alpha_2 B}{g R_0} + \frac{g\omega}{\delta}\right], \tag{20}$$

$$Fb^{*} = \frac{\mu_1 F^{*}}{\gamma_1 + g} = \frac{g\mu_1 (R_0 - 1)}{\delta(\gamma_1 + g)} \tag{21}$$

$$M^{*} = \frac{\mu_2 F^{*} + \eta_2 T^{*}}{\gamma_2 + g} = \frac{(R_0 - 1)}{\gamma_2 + g}\left\{\frac{g\mu_2}{\delta} + \frac{\eta_2}{g + \eta_1 + \eta_2}\left[\frac{\beta_2 \alpha_3 B}{(\beta_1 + \beta_2 + g)R_0} + \frac{\alpha_2 B}{g R_0} + \frac{g\omega}{\delta}\right]\right\}, \tag{22}$$

$$Tb^{*} = \frac{\eta_1 T^{*}}{\gamma_3 + g} = \frac{\eta_1 (R_0 - 1)}{(g + \eta_1 + \eta_2)(\gamma_3 + g)}\left[\frac{\beta_2 \alpha_3 B}{(\beta_1 + \beta_2 + g)R_0} + \frac{\alpha_2 B}{R_0} + \frac{g\omega}{\delta}\right], \tag{23}$$

$$\begin{aligned} R^{*} &= \frac{\gamma_1 Fb^{*} + \gamma_2 M^{*} + \gamma_3 Tb^{*} + \alpha_4 \delta F^{*} S^{*}}{g} \\ &= \frac{\gamma_1 \mu_1 (R_0 - 1)}{\delta(\gamma_1 + g)} + \frac{B\alpha_3 (R_0 - 1)}{g R_0} \\ &\quad + \frac{\gamma_2 (R_0 - 1)}{g(\gamma_2 + g)}\left\{\frac{g\mu_2}{\delta} + \frac{\eta_2}{g + \eta_1 + \eta_2}\left[\frac{\beta_2 \alpha_3 B}{(\beta_1 + \beta_2 + g)R_0} + \frac{\alpha_2 B}{g R_0} + \frac{g\omega}{\delta}\right]\right\} \\ &\quad + \frac{\eta_1 \gamma_3 (R_0 - 1)}{g(g + \eta_1 + \eta_2)(\gamma_3 + g)}\left[\frac{\beta_2 \alpha_3 B}{(\beta_1 + \beta_2 + g)R_0} + \frac{\alpha_2 B}{R_0} + \frac{g\omega}{\delta}\right]. \end{aligned} \tag{24}$$

3.3 Stability analysis of equilibrium points

Theorem 1. When $R_0 < 1, \beta_1 + \beta_2 + 2g + \mu_1 + \mu_2 + \omega > \frac{\delta\alpha_1 B}{g}$, equilibrium point *P₀* is locally asymptotically stable in the feasible domain *Ω*.

Proof: The Jacobian matrix $J(P_0)$ of system (3) at equilibrium point P_0 is

$$J(P_0) = \begin{bmatrix} -g & 0 & -\frac{\delta B}{g} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\beta_1 - \beta_2 - g & \frac{\alpha_3 \delta B}{g} & 0 & 0 & 0 & 0 & 0 \\ 0 & \beta_1 & \frac{\alpha_1 \delta B}{g} - \mu_1 - \mu_2 - \omega - g & 0 & 0 & 0 & 0 & 0 \\ 0 & \beta_2 & \frac{\alpha_2 \delta B}{g} + \omega & -g - \eta_1 - \eta_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mu_1 & 0 & -g - \gamma_1 & 0 & 0 & 0 \\ 0 & 0 & \mu_2 & \eta_2 & 0 & -g - \gamma_2 & 0 & 0 \\ 0 & 0 & 0 & \eta_1 & 0 & 0 & -g - \gamma_3 & 0 \\ 0 & 0 & \frac{\alpha_4 \delta B}{g} & 0 & \gamma_1 & \gamma_2 & \gamma_3 & -g \end{bmatrix}. \tag{25}$$

Let the eigenvalues of matrix (25) be N_i $(i = 1, 2, 3, \ldots, 8)$. From matrix (25), six of the eigenvalues are negative:

$$N_1 = -g < 0, N_2 = -g - \eta_1 - \eta_2 < 0, N_3 = -g - \gamma_1 < 0, N_4 = -g - \gamma_2 < 0, N_5 = -g - \gamma_3 < 0, N_6 = -g < 0$$

The remaining two eigenvalues are also eigenvalues of matrix A_1, which is

$$A_1 = \begin{bmatrix} -\beta_1 - \beta_2 - g & \frac{\alpha_3 \delta B}{g} \\ \beta_1 & \frac{\alpha_1 \delta B}{g} - \mu_1 - \mu_2 - \omega - g \end{bmatrix}. \tag{26}$$

The eigenvalues of A_1 satisfy the following quadratic equation:

$$\lambda^2 + c_1 \lambda + c_2 = 0, \tag{27}$$

where

$$c_1 = \beta_1 + \beta_2 + 2g + \mu_1 + \mu_2 + \omega - \frac{\delta\alpha_1 B}{g}, \tag{28}$$

$$\begin{aligned} c_2 &= -\delta\alpha_1 B - \frac{\delta B\alpha_1\beta_1}{g} - \frac{\delta B\alpha_3\beta_1}{g} - \frac{\delta B\alpha_1\beta_2}{g} + \beta_1 g + \beta_2 g + g^2 + \beta_1\mu_1 \\ &\quad + \beta_2\mu_1 + g\mu_1 + \beta_1\mu_2 + \beta_2\mu_2 + g\mu_2 + \beta_1\omega + \beta_2\omega + g\omega \\ &= -\delta\alpha_1 B - \frac{\delta B\alpha_1\beta_1}{g} - \frac{\delta B\alpha_3\beta_1}{g} - \frac{\delta B\alpha_1\beta_2}{g} + (\beta_1 + \beta_2 + g) \\ &\quad \times (\mu_1 + \mu_2 + g + \omega). \end{aligned} \tag{29}$$

From $\beta_1 + \beta_2 + 2g + \mu_1 + \mu_2 + \omega > \frac{\delta\alpha_1 B}{g}$, it can be seen that $c_1 > 0$. From $R_0 = \frac{B\delta[\alpha_1(\beta_1 + \beta_2 + g) + \alpha_3\beta_1]}{g(\beta_1 + \beta_2 + g)(g + \mu_1 + \mu_2 + \omega)} < 1$, we have $(\beta_1 + \beta_2 + g)(g + \mu_1 + \mu_2 + \omega) - \delta\alpha_1 B - \frac{\delta B\alpha_1\beta_1}{g} - \frac{\delta B\alpha_3\beta_1}{g} - \frac{\delta B\alpha_1\beta_2}{g} > 0$. Therefore, $c_2 > 0$.

Based on the Routh–Hurwitz criterion [43], it can be concluded that the locally asymptotically stable equilibrium point *P₀* lies within the feasible domain *Ω* when $R_0 < 1, \beta_1 + \beta_2 + 2g + \mu_1 + \mu_2 + \omega > \frac{\delta\alpha_1 B}{g}$, which proves Theorem 1.

Theorem 2. When $R_0 < 1, \delta B \le g^2$, the equilibrium point *P₀* is globally asymptotically stable in the feasible domain *Ω*.

Proof: We construct the Lyapunov function around the equilibrium point *P₀* as follows:

$$L_{P_0}(t) = E(t) + F(t) + T(t) + Fb(t) + Tb(t) + M(t) + R(t). \tag{30}$$

Based on system (3), the derivative of the Lyapunov function (30) at equilibrium point *P₀* is

$$\begin{aligned} L_{P_0}{}'(t) &= E'(t) + F'(t) + T'(t) + Fb'(t) + Tb'(t) + M'(t) + R'(t) \\ &= \alpha_3\delta FS - \beta_1 E - \beta_2 E - gE + \alpha_1\delta FS + \beta_1 E - \omega F - \mu_1 F \\ &\quad - \mu_2 F - gF + \alpha_2\delta FS + \beta_2 E + \omega F - \eta_1 T - \eta_2 T - gT \\ &\quad + \mu_1 F - \gamma_1 Fb - gFb + \mu_2 F + \eta_2 T - \gamma_2 M - gM + \eta_1 T \\ &\quad - \gamma_3 Tb - gTb + \gamma_1 Fb + \gamma_2 M + \gamma_3 Tb + \alpha_4\delta FS - gR \\ &= (\delta S - g)F - g(E + T + Fb + Tb + M + R). \end{aligned} \tag{31}$$

From Eq. 5, we know that $S \le \frac{B}{g}$, and because $\delta B \le g^2$, it follows that

$$L_{P_0}{}'(t) \le \left(\frac{\delta B}{g} - g\right)F - g(E + T + Fb + Tb + M + R) \le 0. \tag{32}$$

Based on Equations 31, 32, it can be concluded that $L_{P_0}{}'(t) = 0$ is only true if $F = E = T = Fb = Tb = M = R = 0$. For system (3), the only solution on Ω that satisfies $L_{P_0}{}'(t) = 0$ is P_0. Based on the LaSalle invariance principle [46], it can be demonstrated that the globally asymptotically stable equilibrium point P_0 exists in the feasible domain Ω when $R_0 < 1, \delta B \le g^2$ is true, which proves Theorem 2.

Theorem 3. When $R_0 > 1, \frac{\alpha_3\beta_1 + \alpha_1(\beta_1 + \beta_2)}{\alpha_1} < \mu_1 + \mu_2 + \omega$, the uncertain information propagation equilibrium point P^* is locally asymptotically stable in the feasible domain Ω.

Proof: The Jacobian matrix $J(P^*)$ of system (3) at equilibrium point P^* with uncertain information propagation is

$$J(P^*) = \begin{bmatrix} -g-\delta F^* & 0 & -\delta S^* & 0 & 0 & 0 & 0 & 0 \\ \alpha_3\delta F^* & -\beta_1-\beta_2-g & \alpha_3\delta S^* & 0 & 0 & 0 & 0 & 0 \\ \alpha_1\delta F^* & \beta_1 & \alpha_1\delta S^*-\mu_1-\mu_2-\omega-g & 0 & 0 & 0 & 0 & 0 \\ \alpha_2\delta F^* & \beta_2 & \alpha_2\delta S^*+\omega & -g-\eta_1-\eta_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mu_1 & 0 & -g-\gamma_1 & 0 & 0 & 0 \\ 0 & 0 & \mu_2 & \eta_2 & 0 & -g-\gamma_2 & 0 & 0 \\ 0 & 0 & 0 & \eta_1 & 0 & 0 & -g-\gamma_3 & 0 \\ \alpha_4\delta F^* & 0 & \alpha_4\delta S^* & 0 & \gamma_1 & \gamma_2 & \gamma_3 & -g \end{bmatrix}. \tag{33}$$

We denote the eigenvalues of matrix (33) as H_i $(i = 1, 2, 3, \ldots, 8)$. It is apparent from matrix (33) that five of the eigenvalues are negative:

$$H_1 = -g < 0, H_2 = -g - \eta_1 - \eta_2 < 0, H_3 = -g - \gamma_1 < 0, H_4 = -g - \gamma_2 < 0, H_5 = -g - \gamma_3 < 0$$

The remaining three eigenvalues are also eigenvalues of matrix A_2, which is

$$A_2 = \begin{bmatrix} -g-\delta F^* & 0 & -\delta S^* \\ \alpha_3\delta F^* & -\beta_1-\beta_2-g & \alpha_3\delta S^* \\ \alpha_1\delta F^* & \beta_1 & \alpha_1\delta S^*-\mu_1-\mu_2-\omega-g \end{bmatrix}. \tag{34}$$

Converting the elements of matrix A_2 to an expression containing R_0(14) yields matrix A^*:

$$A^* = \begin{bmatrix} -gR_0 & 0 & -\frac{\delta B}{gR_0} \\ \alpha_3 g(R_0-1) & -\beta_1-\beta_2-g & \frac{\alpha_3\delta B}{gR_0} \\ \alpha_1 g(R_0-1) & \beta_1 & \frac{\alpha_1\delta B}{gR_0}-\mu_1-\mu_2-\omega-g \end{bmatrix}. \tag{35}$$

For ease of writing and derivation later, we let the elements of matrix (35) be

$$K_1 = gR_0, K_2 = \frac{\delta B}{gR_0}, K_3 = -\alpha_3 g(R_0-1), K_4 = \beta_1+\beta_2+g, K_5 = -\frac{\alpha_3\delta B}{gR_0} K_6 = -\alpha_1 g(R_0-1), K_7 = -\beta_1, K_8 = -\left(\frac{\alpha_1\delta B}{gR_0} - \mu_1 - \mu_2 - \omega - g\right) = \frac{\alpha_3\beta_1(\mu_1+\mu_2+\omega+g)}{\alpha_3\beta_1+\alpha_1(\beta_1+\beta_2+g)}$$

Because $R_0 > 1$, we have that $K_1 > 0, K_2 > 0, K_4 > 0, K_8 > 0, K_3 < 0, K_5 < 0, K_6 < 0, K_7 < 0$.

Writing matrix A^* as

$$A^* = \begin{bmatrix} -K_1 & 0 & -K_2 \\ -K_3 & -K_4 & -K_5 \\ -K_6 & -K_7 & -K_8 \end{bmatrix}, \tag{36}$$

the eigenvalues satisfy the following one-dimensional cubic equation:

$$\lambda^3 + h_1\lambda^2 + h_2\lambda + h_3 = 0, \tag{37}$$

where

$$h_1 = K_1 + K_4 + K_8 > 0, \tag{38}$$

$$h_2 = K_1K_4 - K_2K_6 - K_5K_7 + K_1K_8 + K_4K_8, \tag{39}$$

$$\begin{aligned} h_3 &= -K_2K_4K_6 + K_2K_3K_7 - K_1K_5K_7 + K_1K_4K_8 \\ &= K_1(K_4K_8 - K_5K_7) + K_2(K_3K_7 - K_4K_6). \end{aligned} \tag{40}$$

First, to determine the positive and negative solutions of Eq. 39, recall that $K_6 < 0$. Then, $K_1K_4 - K_2K_6 + K_1K_8 > 0$, and

$$\begin{aligned} K_4K_8 - K_5K_7 &= -(\beta_1+\beta_2+g)\left(\frac{\alpha_1\delta B}{gR_0} - \mu_1 - \mu_2 - \omega - g\right) - \frac{\alpha_3\delta B\beta_1}{gR_0} \\ &= \frac{-\alpha_1\delta B(\beta_1+\beta_2+g) + (\beta_1+\beta_2+g)(\mu_1+\mu_2+\omega+g)gR_0 - \alpha_3\delta B\beta_1}{gR_0} \\ &= \frac{(\beta_1+\beta_2+g)(\mu_1+\mu_2+\omega+g)gR_0 - [\alpha_3\delta B\beta_1 + \alpha_1\delta B(\beta_1+\beta_2+g)]}{gR_0} \\ &= \frac{B\delta[\alpha_1(\beta_1+\beta_2+g) + \alpha_3\beta_1] - [\alpha_3\delta B\beta_1 + \alpha_1\delta B(\beta_1+\beta_2+g)]}{gR_0} \\ &= \frac{(B\delta - B\delta)[\alpha_1(\beta_1+\beta_2+g) + \alpha_3\beta_1]}{gR_0} = 0. \end{aligned} \tag{41}$$

Therefore, $h_2 > 0$.

Second, to determine the positive and negative solutions of Eq. 40, if $K_3 < 0, K_6 < 0, K_7 < 0$, then $K_2(K_3K_7 - K_4K_6) > 0$. Thus, based on Eq. 41, we have that $h_3 > 0$.

From Equations 38–40, the values of h_1h_2 and h_1h_2-h_3 are respectively

$$\begin{aligned} h_1h_2 = {} & K_1{}^2K_4 + K_1K_4{}^2 - K_1K_2K_6 - K_2K_4K_6 - K_1K_5K_7 - K_4K_5K_7 \\ & + K_1{}^2K_8 + 3K_1K_4K_8 + K_4{}^2K_8 - K_2K_6K_8 - K_5K_7K_8 \\ & + K_1K_8{}^2 + K_4K_8{}^2, \end{aligned} \tag{42}$$

$$\begin{aligned} h_1h_2 - h_3 &= K_1{}^2K_4 + K_1K_4{}^2 - K_1K_2K_6 - K_4K_5K_7 + K_1{}^2K_8 \\ &\quad + 2K_1K_4K_8 + K_4{}^2K_8 - K_2K_6K_8 - K_5K_7K_8 + K_1K_8{}^2 \\ &\quad + K_4K_8{}^2 - K_2K_3K_7 \\ &= K_1{}^2(K_4 + K_8) + (K_4 + K_8)(K_4K_8 - K_5K_7) \\ &\quad - K_2(K_3K_7 + K_6K_8) \\ &\quad + K_1(K_4{}^2 - K_2K_6 + 2K_4K_8 + K_8{}^2). \end{aligned} \tag{43}$$

Finally, to determine the positive solutions of Eq. 43, recall that $K_6 < 0$ and $K_4K_8 - K_5K_7 = 0$. Therefore, the following must hold:

$$K_1{}^2(K_4 + K_8) + (K_4 + K_8)(K_4K_8 - K_5K_7) + K_1(K_4{}^2 - K_2K_6 + 2K_4K_8 + K_8{}^2) > 0$$

The calculation of $K_3K_7 + K_6K_8$ is simplified as follows:

$$
\begin{aligned}
K_3K_7 + K_6K_8 &= \alpha_3 g(R_0 - 1)\beta_1 + \alpha_1 g(R_0 - 1)\left(\frac{\alpha_1 \delta B}{gR_0} - \mu_1 - \mu_2 - \omega - g\right)\\
&= g(R_0 - 1)\left[\alpha_3\beta_1 + \frac{{\alpha_1}^2 B\delta}{gR_0} - \alpha_1(g + \mu_1 + \mu_2 + \omega)\right]\\
&= g(R_0 - 1)\frac{\alpha_3\beta_1[\alpha_3\beta_1 + \alpha_1(\beta_1 + \beta_2 - \mu_1 - \mu_2 - \omega)]}{\alpha_3\beta_1 + \alpha_1(\beta_1 + \beta_2 + g)}.
\end{aligned}
\tag{44}
$$

Because $R_0 > 1, \frac{\alpha_3\beta_1+\alpha_1(\beta_1+\beta_2)}{\alpha_1} < \mu_1 + \mu_2 + \omega$, we have that $K_3K_7 + K_6K_8 < 0$. Because $K_2 > 0$, we have that $-K_2(K_3K_7 + K_6K_8) > 0$, and it follows that $h_1h_2 - h_3 > 0$.

Based on the Routh–Hurwitz criterion [43], it can be concluded that the locally asymptotically stable uncertain information propagation equilibrium point P^* lies within the feasible domain Ω when $R_0 > 1, \frac{\alpha_3\beta_1+\alpha_1(\beta_1+\beta_2)}{\alpha_1} < \mu_1 + \mu_2 + \omega$, which proves Theorem 3.

Theorem 4. When $R_0 > 1$, the uncertain information propagation equilibrium point P^* is globally asymptotically stable in the feasible domain Ω.

Proof: We construct the Lyapunov function around the equilibrium point P^* as follows:

$$
L_{P^*}(t) = \left\{\begin{array}{l} [S(t) - S^*] + [E(t) - E^*] + [F(t) - F^*] + [T(t) - T^*] \\ +[Fb(t) - Fb^*] + [Tb(t) - Tb^*] + [M(t) - M^*] + [R(t) - R^*] \end{array}\right\}^2.
\tag{45}
$$

Based on system (3), the derivative of the Lyapunov function (45) at the equilibrium point P^* is

$$
\begin{aligned}
{L_{P^*}}'(t) &= 2\left\{\begin{array}{l} [S(t) - S^*] + [E(t) - E^*] + [F(t) - F^*] + [T(t) - T^*] \\ +[Fb(t) - Fb^*] + [Tb(t) - Tb^*] + [M(t) - M^*] + [R(t) - R^*] \end{array}\right\}\\
&\quad \times [S'(t) + E'(t) + F'(t) + T'(t) + Fb'(t) + Tb'(t) + M'(t) + R'(t)]\\
&= 2\left\{\begin{array}{l} [S(t) - S^*] + [E(t) - E^*] + [F(t) - F^*] + [T(t) - T^*] \\ +[Fb(t) - Fb^*] + [Tb(t) - Tb^*] + [M(t) - M^*] + [R(t) - R^*] \end{array}\right\}\\
&\quad \times [B - g(S + E + F + T + Fb + Tb + M + R)].
\end{aligned}
\tag{46}
$$

From point P* in Eq. 15, it follows that $B - g(S^* + E^* + F^* + T^* + Fb^* + Tb^* + M^* + R^*) = 0$, namely, $B = g(S^* + E^* + F^* + T^* + Fb^* + Tb^* + M^* + R^*)$.

Therefore, Eq. 46 can be expressed as

$$
\begin{aligned}
{L_{P^*}}'(t) &= 2\left\{\begin{array}{l} [S(t) - S^*] + [E(t) - E^*] + [F(t) - F^*] \\ +[T(t) - T^*] + [Fb(t) - Fb^*] + [Tb(t) - Tb^*] \\ +[M(t) - M^*] + [R(t) - R^*] \end{array}\right\}\\
&\quad \left[g(S^* + E^* + F^* + T^* + Fb^* + Tb^* + M^* + R^*) - g\left(\begin{array}{l} S + E + F + T \\ +Fb + Tb + M + R \end{array}\right)\right]\\
&= -2g\left\{\begin{array}{l} [S(t) - S^*] + [E(t) - E^*] + [F(t) - F^*] + [T(t) - T^*] \\ +[Fb(t) - Fb^*] + [Tb(t) - Tb^*] + [M(t) - M^*] \\ +[R(t) - R^*] \end{array}\right\}^2\\
&\leq 0
\end{aligned}
\tag{47}
$$

From Eq. 47, it can be concluded that ${L_{P^*}}'(t) = 0$ only if $S(t) = S^*, E(t) = E^*, F(t) = F^*, T(t) = T^*, Fb(t) = Fb^*, Tb(t) = Tb^*, M(t) = M^*, R(t) = R^*$ all hold. From system (3), the only solution on Ω that satisfies ${L_{P^*}}'(t) = 0$ is P^*. Based on the LaSalle invariance principle [46], it can be demonstrated that the globally asymptotically stable equilibrium point P^* of uncertain information propagation exists in the feasible domain Ω when $R_0 > 1$ holds, which proves Theorem 4.

3.4 Numerical simulation analysis of equilibrium point stability

To verify the theoretical derivations, we now assign values to the parameters and perform numerical simulations using Matlab2017b. As these parameters cannot be obtained directly in practical cases, we use reasonable values within the context of the situation. The relevant parameters are assigned based on the following scenarios:

Scenario 1: To verify the local and global asymptotic stability of equilibrium point P_0 in the feasible domain Ω for $R_0 < 1$, the parameters are assigned as follows:

$$
\left\{\begin{array}{l} B = 1, g = 0.2, \alpha_1 = 0.2, \alpha_2 = 0.2, \alpha_3 = 0.5, \alpha_4 = 0.1, \beta_1 = 0.2, \\ \beta_2 = 0.3, \mu_1 = 0.4, \mu_2 = 0.3, \eta_1 = 0.3, \eta_2 = 0.3, \omega = 0.6, \\ \gamma_1 = 0.5, \gamma_2 = 0.5, \gamma_3 = 0.5, \delta = 0.02 \end{array}\right. .
\tag{48}
$$

Based on the parameters in (48), we have that $R_0 = 0.0229 < 1$, which satisfies the basic assumptions of Theorems 1 and 2. To further explore whether the initial values of the various Internet user populations in the system impact the final stability of the equilibrium point P_0, we maintain the values in (48) and conduct numerical simulations with different initial values. Figure 3 shows the evolution of equilibrium point P_0 over time when $R_0 < 1$.

From Figure 3, it is evident that, regardless of the initial proportions of the Internet population, every Internet user eventually becomes an unknown entity. Thus, equilibrium point P_0 is asymptotically stable within the feasible domain Ω when $R_0 < 1$, which is consistent with the theory.

Based on Scenario 1, we can know that in the real world, when a major emergencies occurs in a certain place, as long as $R_0 < 1$, the uncertain information in the online social platform will be gradually forgotten by the netizens over time. In this case, the relevant government departments do not need to make additional interventions, and the uncertain information does not affect the stability of society.

Scenario 2: To verify the local and global asymptotic stability of equilibrium point P^* in the feasible domain Ω for $R_0 > 1$, the parameters are assigned as follows:

$$
\left\{\begin{array}{l} B = 1, g = 0.2, \alpha_1 = 0.4, \alpha_2 = 0.15, \alpha_3 = 0.4, \alpha_4 = 0.05, \beta_1 = 0.2, \\ \beta_2 = 0.2, \mu_1 = 0.2, \mu_2 = 0.2, \eta_1 = 0.3, \eta_2 = 0.3, \omega = 0.3, \gamma_1 = 0.5, \\ \gamma_2 = 0.5, \gamma_3 = 0.5, \delta = 0.5 \end{array}\right. .
\tag{49}
$$

Based on the parameter values in (49), we have that $R_0 = 1.4815 > 1$, which satisfies the basic assumptions of Theorems 3 and 4. To further explore whether the initial values of the various Internet user populations impact the final stability of equilibrium point P^*, we maintain the values in (49) and conduct numerical simulations with different initial values. Figure 4 shows the evolution of the equilibrium point P^* when $R_0 > 1$.

From Figure 4, it is evident that, regardless of the initial populations of each state in the system, all users eventually become an unknown entity. Thus, the uncertain information propagation equilibrium point

P^* is asymptotically stable within the feasible domain Ω when $R_0 > 1$, which is consistent with the theory.

Based on Scenario 2, we can know that in the real world, when a major emergencies occur in a certain place, when $R_0 > 1$, the uncertain information in the online social platform will keep spreading among netizens over time. In this case, if the relevant government departments do not intervene, it will lead to the continuous spread of panic among netizens, which will eventually affect the stability of society.

4 Optimal control model

Based on the SEFTFbTbMR uncertain information dissemination model, it is recommended that netizens be encouraged to clarify uncertain information as much as possible, or to think and judge uncertain information instead of posting random remarks; this action will reduce the impact of uncertain information. Therefore, the number of thinkers and clarifiers of uncertain information should be increased. Thus, we now examine the effect of modifying the model's proportionality constants α_2, α_3, β_2, and ω into control variable functions $\alpha_2(t)$, $\alpha_3(t)$, $\beta_2(t)$, and $\omega(t)$, respectively.

The objective function is defined as follows:

$$J(\alpha_2, \alpha_3, \beta_2, \omega) = \int_0^{t_f} \begin{bmatrix} E(t) + T(t) - \frac{\psi_1}{2}\alpha_2{}^2(t) - \frac{\psi_2}{2}\alpha_3{}^2(t) - \frac{\psi_3}{2}\beta_2{}^2(t) \\ -\frac{\psi_4}{2}\omega^2(t) \end{bmatrix} dt, \tag{50}$$

where t_f is the end moment, and ψ_1, ψ_2, ψ_3, and ψ_4 are the weight coefficients of each function.

We seek to satisfy the following system constraints:

$$\begin{cases} \frac{dS}{dt} = B - gS - \alpha_1\delta FS - \alpha_2(t)\delta FS - \alpha_3(t)\delta FS - \alpha_4\delta FS \\ \frac{dE}{dt} = \alpha_3(t)\delta FS - \beta_1 E - \beta_2(t)E - gE \\ \frac{dF}{dt} = \alpha_1\delta FS + \beta_1 E - \omega(t)F - \mu_1 F - \mu_2 F - gF \\ \frac{dT}{dt} = \alpha_2(t)\delta FS + \beta_2(t)E + \omega(t)F - \eta_1 T - \eta_2 T - gT \\ \frac{dFb}{dt} = \mu_1 F - \gamma_1 Fb - gFb \\ \frac{dM}{dt} = \mu_2 F + \eta_2 T - \gamma_2 M - gM \\ \frac{dTb}{dt} = \eta_1 T - \gamma_3 Tb - gTb \\ \frac{dR}{dt} = \gamma_1 Fb + \gamma_2 M + \gamma_3 Tb + \alpha_4\delta FS - gR \end{cases}. \tag{51}$$

The initial conditions necessary to satisfy system (60) are

$$\begin{aligned} &S(0) = S_0, E(0) = E_0, F(0) = F_0, T(0) = T_0, \\ &Fb(0) = Fb_0, M(0) = M_0, Tb(0) = Tb_0, R(0) = R_0 \text{'} \end{aligned} \tag{52}$$

where

$$\alpha_2(t), \alpha_3(t), \beta_2(t), \omega(t) \in U \triangleq \left\{ \begin{array}{l} (\alpha_2, \alpha_3, \beta_2, \omega) \big| (\alpha_2(t), \alpha_3(t), \beta_2(t), \omega(t)) \\ measurable, \\ 0 \le \alpha_2(t), \alpha_3(t), \beta_2(t), \omega(t) \le 1, \forall t \in [0, t_f] \end{array} \right\}. \tag{53}$$

Theorem 5. There exists an optimal control tuple $(\alpha_2{}^*, \alpha_3{}^*, \beta_2{}^*, \omega^*) \in U$ such that

$$J(\alpha_2{}^*, \alpha_3{}^*, \beta_2{}^*, \omega^*) = \max\{J(\alpha_2, \alpha_3, \beta_2, \omega): (\alpha_2, \alpha_3, \beta_2, \omega) \in U\}. \tag{54}$$

Proof: Set $X(t) = (S(t), E(t), F(t), T(t), Fb(t), Tb(t), M(t), R(t))^T$ and

$$\begin{aligned} &L(t, X(t), \alpha_2(t), \alpha_3(t), \beta_2(t), \omega(t)) \\ &\quad = E(t) + T(t) - \frac{\psi_1}{2}\alpha_2{}^2(t) - \frac{\psi_2}{2}\alpha_3{}^2(t) - \frac{\psi_3}{2}\beta_2{}^2(t) - \frac{\psi_4}{2}\omega^2(t) \end{aligned}$$

The existence of optimal control tuples is contingent upon fulfilling the following criteria:

1. The set of control variables and corresponding state variables must constitute a nonempty set.
2. The control set U must be closed and convex.
3. The right-hand side of (60) should take the form of a linear system comprising state variables and control variables.
4. The product of the target generalization must be convex on U.
5. There is a constant $k_1 > 0, k_2 > 0, l > 0$ such that the product of the intended generalized function satisfies

$$-L(t, X(t), \alpha_2, \alpha_3, \beta_2, \omega) \ge k_1\left(|\alpha_2|^2 + |\alpha_3|^2 + |\beta_2|^2 + |\omega|^2\right)^{\frac{l}{2}} - k_2. \tag{55}$$

As conditions 1–3 are straightforward, only conditions 4 and 5 are proved.

First, it is easy to obtain inequalities based on system (51):

$$\begin{aligned} &S' \le B, E' \le \alpha_3(t)\delta FS, F' \le \alpha_1\delta FS + \beta_1 E, T' \le \alpha_2(t)\delta FS + \beta_2(t)E \\ &\quad + \omega(t)F, Fb' \le \mu_1 F, M' \le \mu_2 F + \eta_2 T, Tb' \le \eta_1 T, R' \le \gamma_1 Fb + \gamma_2 M \\ &\quad + \gamma_3 Tb + \alpha_4\delta FS.. \end{aligned} \tag{56}$$

Therefore, condition 4 holds.

Second, for any $t \ge 0$, there exists a positive constant Z satisfying $|X(t)| \le Z$. Hence,

$$\begin{aligned} -L(t, X(t), \alpha_2, \alpha_3, \beta_2, \omega) &= \frac{\psi_1}{2}\alpha_2{}^2(t) + \frac{\psi_2}{2}\alpha_3{}^2(t) + \frac{\psi_3}{2}\beta_2{}^2(t) \\ &\quad + \frac{\psi_4}{2}\omega^2(t) - E(t) - T(t) \\ &\ge k_1\left(|\alpha_2|^2 + |\alpha_3|^2 + |\beta_2|^2 + |\omega|^2\right)^{\frac{l}{2}} - 2Z. \end{aligned} \tag{57}$$

Setting $k_1 = \min\left\{\frac{\psi_1}{2}, \frac{\psi_2}{2}, \frac{\psi_3}{2}, \frac{\psi_4}{2}\right\}, k_2 = 2Z, l = 2,$ condition 5 then holds.

At this point, all optimal control tuples have been successfully verified, proving Theorem 5.

Theorem 6. For the optimal control tuple $(\alpha_2{}^*, \alpha_3{}^*, \beta_2{}^*, \omega^*) \in U$ for system (51), there is an associated variable $\rho_i (i = 1, 2, ..., 8)$ such that

$$\begin{cases} \frac{d\rho_1}{dt} = (\rho_1 + \rho_4)\alpha_2(t)\delta F + \rho_1\alpha_4\delta F + \rho_3\alpha_1\delta F + (\rho_1 F + \rho_2 S)\alpha_3(t)\delta \\ \qquad + \rho_8\alpha_4\delta F + \rho_1 g + \rho_1\alpha_1\delta F \\ \frac{d\rho_2}{dt} = 1 + (\rho_2 - \rho_4)\beta_2(t) + \rho_2(\beta_1 + g) - \rho_3\beta_1 \\ \frac{d\rho_3}{dt} = (\rho_1 - \rho_4)\alpha_2(t)\delta S - \rho_5\mu_1 + \rho_8\alpha_4\delta S + (\rho_1 - \rho_2)\alpha_3(t)\delta S \\ \qquad + \rho_1[-\alpha_1\delta S + \alpha_4\delta S] \\ \qquad + (\rho_3 - \rho_4)\omega(t) - \rho_6\mu_2 + \rho_3[-\alpha_1\delta S + \mu_1 + \mu_2 + g] \\ \frac{d\rho_4}{dt} = 1 - \rho_7\eta_1 - \rho_6\eta_2 + \rho_4(\eta_1 + \eta_2 + g) \\ \frac{d\rho_5}{dt} = \rho_5(\gamma_1 + g) - \rho_8\gamma_1 \\ \frac{d\rho_6}{dt} = \rho_6(\gamma_2 + g) - \rho_8\gamma_2 \\ \frac{d\rho_7}{dt} = \rho_7(\gamma_3 + g) - \rho_8\gamma_3 \\ \frac{d\rho_8}{dt} = \rho_8 g, \end{cases} \tag{58}$$

with the following boundary conditions:

$$\rho_1(t_f) = \rho_2(t_f) = \rho_3(t_f) = \rho_4(t_f) = \rho_5(t_f) = \rho_6(t_f) = \rho_7(t_f) \\ = \rho_8(t_f) = 0. \tag{59}$$

Furthermore, the optimal control tuple $(\alpha_2{}^*, \alpha_3{}^*, \beta_2{}^*, \omega^*) \in U$ for the state system can be obtained from the following equation:

$$\begin{cases} \alpha_2^*(t) = \min\left\{1, \max\left\{0, \frac{(\rho_1 - \rho_4)\delta FS}{\psi_1}\right\}\right\} \\ \alpha_3^*(t) = \min\left\{1, \max\left\{0, \frac{(\rho_1 - \rho_2)\delta FS}{\psi_2}\right\}\right\} \\ \beta_2^*(t) = \min\left\{1, \max\left\{0, \frac{(\rho_2 - \rho_4)E}{\psi_3}\right\}\right\} \\ \omega^*(t) = \min\left\{1, \max\left\{0, \frac{(\rho_3 - \rho_4)F}{\psi_4}\right\}\right\}. \end{cases} \tag{60}$$

Proof: To derive the necessary expressions for the optimal control system and control tuple, we define a Hamiltonian function with a penalty term, with the following expression serving as a guideline:

$$\begin{aligned} H = &-L(t, X(t), \alpha_2(t), \alpha_3(t), \beta_2(t), \omega(t)) \\ &+ \rho_1[B - gS - \alpha_1\delta FS - \alpha_2(t)\delta FS - \alpha_3(t)\delta FS - \alpha_4\delta FS] \\ &+ \rho_7[\eta_1 T - \gamma_3 Tb - gTb] + \rho_2[\alpha_3(t)\delta FS - \beta_1 E - \beta_2(t)E - gE] \\ &+ \rho_6[\mu_2 F + \eta_2 T - \gamma_2 M - gM] \\ &+ \rho_3[\alpha_1\delta FS + \beta_1 E - \omega(t)F - \mu_1 F - \mu_2 F - gF] \\ &+ \rho_5[\mu_1 F - \gamma_1 Fb - gFb] \\ &+ \rho_4[\alpha_2(t)\delta FS + \beta_2(t)E + \omega(t)F - \eta_1 T - \eta_2 T - gT] \\ &+ \rho_8[\gamma_1 Fb + \gamma_2 M + \gamma_3 Tb + \alpha_4\delta FS - gR] - \lambda_{11}\alpha_2(t) \\ &- \lambda_{12}(1 - \alpha_2(t)) - \lambda_{21}\alpha_3(t) - \lambda_{22}(1 - \alpha_3(t)) - \lambda_{31}\beta_2(t) \\ &- \lambda_{32}(1 - \beta_2(t)) - \lambda_{41}\omega(t) - \lambda_{42}(1 - \omega(t)), \end{aligned} \tag{61}$$

where the penalty term $\lambda_{ij}(t) \geq 0$ satisfies $\lambda_{11}(t)\alpha_2(t) = \lambda_{12}(t)(1 - \alpha_2(t)) = 0$ at the optimal control point for α_2^*, $\lambda_{21}(t)\alpha_3(t) = \lambda_{22}(t)(1 - \alpha_3(t)) = 0$ at the optimal control point for α_3^*, $\lambda_{31}(t)\beta_2(t) = \lambda_{32}(t)(1 - \beta_2(t)) = 0$ at the optimal control point for β_2^*, and $\lambda_{41}(t)\omega(t) = \lambda_{42}(t)(1 - \omega(t)) = 0$ at the optimal control point for ω^*.

Based on Pontryagin's maximum principle [47], the concomitant system can be expressed as follows:

$$\begin{aligned} &\frac{d\rho_1}{dt} = -\frac{\partial H}{\partial S}, \frac{d\rho_2}{dt} = -\frac{\partial H}{\partial E}, \frac{d\rho_3}{dt} = -\frac{\partial H}{\partial F}, \frac{d\rho_4}{dt} = -\frac{\partial H}{\partial T}, \\ &\frac{d\rho_5}{dt} = -\frac{\partial H}{\partial Fb}, \frac{d\rho_6}{dt} = -\frac{\partial H}{\partial M}, \frac{d\rho_7}{dt} = -\frac{\partial H}{\partial Tb}, \frac{d\rho_8}{dt} = -\frac{\partial H}{\partial R}. \end{aligned} \tag{62}$$

The boundary conditions of this system are

$$\rho_1(t_f) = \rho_2(t_f) = \rho_3(t_f) = \rho_4(t_f) = \rho_5(t_f) = \rho_6(t_f) = \rho_7(t_f) \\ = \rho_8(t_f) = 0. \tag{63}$$

The optimality conditions in terms of α_2^* are

$$\frac{\partial H}{\partial \alpha_2^*} = \psi_1\alpha_2(t) - \rho_1\delta FS + \rho_4\delta FS - \lambda_{11} + \lambda_{12} = 0. \tag{64}$$

Thus, the optimal control equation can be written as

$$\alpha_2^* = \frac{(\rho_1 - \rho_4)\delta FS}{\psi_1} + \lambda_{11} - \lambda_{12}. \tag{65}$$

To obtain the final optimal control equation without λ_{11} or λ_{12}, the following three cases are discussed separately.

1. For $\{t \mid 0 < \alpha_2^*(t) < 1\}$, $\lambda_{11}(t) = \lambda_{12}(t) = 0$, the optimal control equation can be expressed as follows:

$$\alpha_2^* = \frac{(\rho_1 - \rho_4)\delta FS}{\psi_1}. \tag{66}$$

2. For $\{t | \alpha_2^*(t) = 1\}$, $\lambda_{11}(t) = 0$, the optimal control equation can be expressed as follows:

$$1 = \alpha_2^* = \frac{(\rho_1 - \rho_4)\delta FS - \lambda_{12}}{\psi_1}. \tag{67}$$

Because $\lambda_{12}(t) \geq 0$, we have that $\frac{(\rho_1 - \rho_4)\delta FS - \lambda_{12}}{\psi_1} \geq 1$.

3. For $\{t | \alpha_2^*(t) = 0\}$, $\lambda_{12}(t) = 0$, the optimal control equation can be expressed as follows:

$$0 = \alpha_2^* = \frac{(\rho_1 - \rho_4)\delta FS + \lambda_{11}}{\psi_1}. \tag{68}$$

Based on these three cases, the final optimal control equation for $\alpha_2^*(t)$ can be written as

$$\alpha_2^*(t) = \min\left\{1, \max\left\{0, \frac{(\rho_1 - \rho_4)\delta FS}{\psi_1}\right\}\right\}. \tag{69}$$

Similarly, the final optimal control equation for $\alpha_3^*(t)$ is

$$\alpha_3^*(t) = \min\left\{1, \max\left\{0, \frac{(\rho_1 - \rho_2)\delta FS}{\psi_2}\right\}\right\}, \tag{70}$$

and that for $\beta_2^*(t)$ is

$$\beta_2^*(t) = \min\left\{1,\ \max\left\{0, \frac{(\rho_2 - \rho_4)E}{\psi_3}\right\}\right\}. \tag{71}$$

Finally, the optimal control equation for $\omega^*(t)$ can be written as

$$\omega^*(t) = \min\left\{1,\ \max\left\{0, \frac{(\rho_3 - \rho_4)F}{\psi_4}\right\}\right\}. \tag{72}$$

We have now obtained system (51), which includes the initial conditions (52), and the accompanying system (58), which includes the boundary conditions. The optimal control system can now be expressed as follows:

$$\begin{cases}
\frac{dS}{dt} = B - gS - \alpha_1\delta FS - \min\left\{1,\ \max\left\{0, \frac{(\rho_1 - \rho_4)\delta FS}{\psi_1}\right\}\right\}(t)\delta FS \\
\qquad - \min\left\{1,\ \max\left\{0, \frac{(\rho_1 - \rho_2)\delta FS}{\psi_2}\right\}\right\}(t)\delta FS - \alpha_4\delta FS \\
\frac{dE}{dt} = \min\left\{1,\ \max\left\{0, \frac{(\rho_1 - \rho_2)\delta FS}{\psi_2}\right\}\right\}(t)\delta FS - \beta_1 E \\
\qquad - \min\left\{1,\ \max\left\{0, \frac{(\rho_2 - \rho_4)E}{\psi_3}\right\}\right\}(t)E - gE \\
\frac{dF}{dt} = \alpha_1\delta FS + \beta_1 E - \min\left\{1,\ \max\left\{0, \frac{(\rho_3 - \rho_4)F}{\psi_4}\right\}\right\}(t)F - \mu_1 F - \mu_2 F - gF \\
\frac{dT}{dt} = \min\left\{1,\ \max\left\{0, \frac{(\rho_1 - \rho_4)\delta FS}{\psi_1}\right\}\right\}(t)\delta FS \\
\qquad + \min\left\{1,\ \max\left\{0, \frac{(\rho_2 - \rho_4)E}{\psi_3}\right\}\right\}(t)E \\
\qquad + \min\left\{1,\ \max\left\{0, \frac{(\rho_3 - \rho_4)F}{\psi_4}\right\}\right\}(t)F - \eta_1 T - \eta_2 T - gT \\
\frac{dFb}{dt} = \mu_1 F - \gamma_1 Fb - gFb \\
\frac{dM}{dt} = \mu_2 F + \eta_2 T - \gamma_2 M - gM \\
\frac{dTb}{dt} = \eta_1 T - \gamma_3 Tb - gTb \\
\frac{dR}{dt} = \gamma_1 Fb + \gamma_2 M + \gamma_3 Tb + \alpha_4\delta FS - gR \\
\frac{d\rho_1}{dt} = (\rho_1 + \rho_4)\min\left\{1,\ \max\left\{0, \frac{(\rho_1 - \rho_4)\delta FS}{\psi_1}\right\}\right\}(t)\delta F + \rho_1\alpha_4\delta F + \rho_3\alpha_1\delta F \\
\qquad + (\rho_1 F + \rho_2 S)\min\left\{1,\ \max\left\{0, \frac{(\rho_1 - \rho_2)\delta FS}{\psi_2}\right\}\right\}(t)\delta + \rho_8\alpha_4\delta F + \rho_1 g + \rho_1\alpha_1\delta F \\
\frac{d\rho_2}{dt} = 1 + (\rho_2 - \rho_4)\min\left\{1,\ \max\left\{0, \frac{(\rho_2 - \rho_4)E}{\psi_3}\right\}\right\}(t) + \rho_2(\beta_1 + g) - \rho_3\beta_1 \\
\frac{d\rho_3}{dt} = (\rho_1 - \rho_4)\min\left\{1,\ \max\left\{0, \frac{(\rho_1 - \rho_4)\delta FS}{\psi_1}\right\}\right\}(t)\delta S - \rho_5\mu_1 + \rho_8\alpha_4\delta S \\
\qquad + (\rho_1 - \rho_2)\min\left\{1,\ \max\left\{0, \frac{(\rho_1 - \rho_2)\delta FS}{\psi_2}\right\}\right\}(t)\delta S + \rho_1[-\alpha_1\delta S + \alpha_4\delta S] \\
\qquad + (\rho_3 - \rho_4)\min\left\{1,\ \max\left\{0, \frac{(\rho_3 - \rho_4)F}{\psi_4}\right\}\right\}(t) - \rho_6\mu_2 + \rho_3[-\alpha_1\delta S + \mu_1 + \mu_2 + g] \\
\frac{d\rho_4}{dt} = 1 - \rho_7\eta_1 - \rho_6\eta_2 + \rho_4(\eta_1 + \eta_2 + g) \\
\frac{d\rho_5}{dt} = \rho_5(\gamma_1 + g) - \rho_8\gamma_1 \\
\frac{d\rho_6}{dt} = \rho_6(\gamma_2 + g) - \rho_8\gamma_2 \\
\frac{d\rho_7}{dt} = \rho_7(\gamma_3 + g) - \rho_8\gamma_3 \\
\frac{d\rho_8}{dt} = \rho_8 g
\end{cases}, \tag{73}$$

where

$$\begin{aligned} S(0) &= S_0, E(0) = E_0, F(0) = F_0, T(0) = T_0, Fb(0) = Fb_0, M(0) \\ &= M_0, Tb(0) = Tb_0, R(0) = R_0\rho_1(t_f) = \rho_2(t_f) = \rho_3(t_f) \\ &= \rho_4(t_f) = \rho_5(t_f) = \rho_6(t_f) = \rho_7(t_f) = \rho_8(t_f) = 0. \end{aligned} \tag{74}$$

This completes the proof of Theorem 6.

5 Discussion

This paper investigates the dissemination mechanism of uncertain information triggered by major emergencies on online social platforms. Based on the construction of the SEFTFbTbMR model of uncertain information clarification behavior, the optimal control strategy of the model is proposed using the Hamiltonian function. It is known from the analysis of the model that the size of the basic regeneration number plays a crucial role in predicting whether uncertain information can eventually die out. When the basic regeneration number is < 1, uncertain information can die out automatically over time. When the basic regeneration number is > 1, uncertain information will always exist on the online social platform, which can significantly disrupt society.

During major emergencies, due to limited resources for public opinion control, uncertain information may spread unchecked on online social platforms. This can lead to some netizens unintentionally or intentionally becoming disseminators of such information. Currently, government departments are responsible for investigating and managing major emergencies. However, it may not be possible for them to effectively control and remove all sources of uncertain information within a short period of time. The value of this study lies in its ability to provide theoretical support for relevant government departments to reduce the adverse effects caused by the propagation of uncertain information in the future. The experimental results of this article help to deepen our understanding of the propagation mechanism of uncertain information among Internet users and further enrich the related theories and methods of uncertain information propagation research.

5.1 Sensitivity analysis of the basic regeneration number R_0

The experimental results of this paper show that the value of the basic regeneration number determines whether uncertain information can be disseminated in the online social platform. The basic regeneration number R_0 is jointly composed of different parameters in the model. Therefore, this section focuses on the influence of related parameters on the basic regeneration number.

To analyze the influence of parameter value changes on the basic regeneration number R_0, we obtained the first-order partial derivatives for each parameter in R_0. A positive sign in the partial derivative function indicates a positive influence of the parameter on R_0. If the sign of the partial derivative function is negative, it indicates that the parameter has a negative impact on the basic reproduction number R_0.

$$\frac{\partial R_0}{\partial B}=\frac{\delta[\alpha_1(\beta_1+\beta_2+g)+\alpha_3\beta_1]}{g(\beta_1+\beta_2+g)(g+\mu_1+\mu_2+\omega)}>0 \tag{75}$$

$$\frac{\partial R_0}{\partial \delta}=\frac{B[\alpha_1(\beta_1+\beta_2+g)+\alpha_3\beta_1]}{g(\beta_1+\beta_2+g)(g+\mu_1+\mu_2+\omega)}>0 \tag{76}$$

$$\frac{\partial R_0}{\partial \alpha_1}=\frac{B\delta}{g(g+\mu_1+\mu_2+\omega)}>0 \tag{77}$$

$$\frac{\partial R_0}{\partial \alpha_3}=\frac{B\delta\beta_1}{g(\beta_1+\beta_2+g)(g+\mu_1+\mu_2+\omega)}>0 \tag{78}$$

$$\frac{\partial R_0}{\partial \beta_1}=\frac{B\delta\alpha_3(\beta_2+g)}{g(\beta_1+\beta_2+g)^2(g+\mu_1+\mu_2+\omega)}>0 \tag{79}$$

$$\frac{\partial R_0}{\partial \beta_2}=-\frac{B\delta\alpha_3\beta_1}{g(\beta_1+\beta_2+g)^2(g+\mu_1+\mu_2+\omega)}<0 \tag{80}$$

$$\frac{\partial R_0}{\partial g}=-\frac{B\delta\left\{\begin{array}{l}\alpha_1(\beta_1+\beta_2+g)^2(2g+\mu_1+\mu_2+\omega)\\+\alpha_3\beta_1[(\beta_1+\beta_2)(2g+\mu_1+\mu_2+\omega)\\+g(3g+2\mu_1+2\mu_2+2\omega)]\end{array}\right\}}{g^2(\beta_1+\beta_2+g)^2(g+\mu_1+\mu_2+\omega)^2}<0 \tag{81}$$

$$\frac{\partial R_0}{\partial \mu_1}=-\frac{B\delta[\alpha_1(\beta_1+\beta_2+g)+\alpha_3\beta_1]}{g(\beta_1+\beta_2+g)(g+\mu_1+\mu_2+\omega)^2}<0 \tag{82}$$

$$\frac{\partial R_0}{\partial \mu_2}=-\frac{B\delta[\alpha_1(\beta_1+\beta_2+g)+\alpha_3\beta_1]}{g(\beta_1+\beta_2+g)(g+\mu_1+\mu_2+\omega)^2}<0 \tag{83}$$

$$\frac{\partial R_0}{\partial \omega}=-\frac{B\delta[\alpha_1(\beta_1+\beta_2+g)+\alpha_3\beta_1]}{g(\beta_1+\beta_2+g)(g+\mu_1+\mu_2+\omega)^2}<0 \tag{84}$$

To visualize the impact of various parameter values on R_0, we used Matlab 2017b numerical simulation software. Based on the assignment result (49), we kept the remaining parameters constant and varied the value range of B from 1 to 5, δ from 0.1 to 1, and the rest of the parameters from 0 to 1. We conducted numerical simulations in groups of two by two to determine the effects of different parameter values on R_0.

Figure 5 shows that the basic regeneration number R_0 increases as parameters B and δ increase. 1) The larger the number of new Internet users in the social platform per unit of time, the more conducive to the spread of uncertain information. The larger the base of Internet users, the greater the number of individuals who may be concerned about uncertain information, and the more likely it is that such information will become widely known. 2) The speed at which uncertain information spreads affects its propagation. In other words, the more Internet users on a social platform who come into contact with the publisher of uncertain information, the more likely it is to spread. Thus, the spread of uncertain information can be curbed by limiting the speech flow of certain netizens on social media platforms or blocking specific keywords.

Figure 6 shows that the basic regeneration number R_0 increases with parameter α_1 and decreases with parameter μ_1. 1) When unknown people receive uncertain information, they promote the spread of uncertain information in social platforms if they choose to believe in the content of the uncertain information and spontaneously spread it. 2) Publishers of uncertain information who receive true information on social platforms and choose not to publish their own statements, regardless of whether they believe in the true information or not, inhibit the spread of uncertain information on social platforms.

Figure 7 shows that the basic regeneration number R_0 increases with parameter α_3 and decreases with parameter μ_2. 1) When the unknown person receives the uncertain information, if he does not spread the uncertain information, but keeps a wait-and-see attitude, it will promote the spread of uncertain information in the social platform. 2) If an uncertain information publisher receives real information on a social platform, they may choose not to publish their own speech, regardless of whether they believe the real information or not. This can help inhibit the spread of uncertain information on the platform.

Figure 8 shows that the basic regeneration number R_0 increases with an increase in parameter β_1 and decreases with an increase in parameter ω. 1) The dissemination of uncertain information in social platforms is facilitated when the thinker still chooses to believe in the content of the uncertain information and disseminates it after forensically examining and thinking about the uncertain information. 2) The spread of uncertain information on social media is hindered when the person who originally shared the uncertain information realizes that it is false after receiving accurate information and decides to clarify it.

According to Figure 9, the basic regeneration number R_0 decreases as parameters g and β_2 increase. 1) The higher the number of netizens exiting in the social platform per unit time, the more conducive to suppressing the spread of uncertain information. 2) The spread of uncertain information on social media can be reduced when individuals recognize that the information is false and take the time to clarify it after gathering evidence and carefully considering the information.

5.2 Comparison with existing research on uncertain information dissemination

This section compares the research in this paper with existing research on uncertain information dissemination. In their study of uncertain information dissemination [21], only distinguish between the decision-making behavior of online media and that of Internet users. However, they fail to consider that different Internet users may exhibit various decision-making behaviors when faced with uncertain information. Some may even adopt the same decision-making behaviors as online media [24, 43]. Only considered two decision-making behaviors of Internet users: dissemination or thinking. However, in reality, some Internet users choose to clarify uncertain information when faced with it, while others maintain their behavior after contacting other Internet users. In a previous study [46], observed this effect but did not consider that some netizens may not immediately change their minds after interacting with others, but rather become more thoughtful.

Compared to the studies conducted by the aforementioned scholars on uncertain information dissemination, this paper considers not only the fact that Internet users exhibit multiple decision-making behaviors when faced with uncertain information, but also that some of them choose to clarify such information. Additionally, it acknowledges that Internet users are influenced not only by uncertain information, but also by real information. This paper categorizes Internet users into eight groups based on their decision-making behaviors during uncertain information dissemination. The model considers both the dissemination of uncertain information and the dissemination of real information that clarifies uncertain information, making it highly innovative.

5.3 Limitations and future prospects

The research presented in this paper has the following limitations. First, in constructing the uncertain information dissemination model, the impact of time lags on uncertain information dissemination was not considered. In reality, information dissemination has a certain degree of lag, and Internet users receive uncertain information at inconsistent times. Therefore, in future research, we will add a time lag to our model. Second, this paper does not differentiate the communication ability of Internet users, whereas, in reality, the information released by opinion leaders is more likely to be trusted by ordinary Internet users. Therefore, in the future, we will combine complex networks with the uncertain information dissemination model to study the propagation mechanism based on different network structures. Finally, this study only used Matlab for the numerical simulations, without any real data. Therefore, in future research, we will integrate real data where possible and simulate real cases.

6 Conclusion

This paper has described the SEFTFbTbMR uncertain information dissemination model, which is based on the classical SIR epidemic dynamics model. The next-generation matrix method was used to calculate the basic regeneration number and equilibrium points of the model, and the local stability and global stability of the equilibrium points were theoretically analyzed according to the Routh–Hurwitz criterion and the Lyapunov function, respectively. The accuracy of the theoretical derivation was verified through numerical simulations, and the sensitivity of the basic regeneration number to various parameters was analyzed. Finally, to reduce the influence of uncertain information, optimal control theory was applied to the model, and a strategy was proposed. This will further enrich the relevant theories and methods for the propagation of uncertain information. The main results of this study are as follows:

(1) Strengthen the supervision of social platforms to block the dissemination of uncertain information (i.e., reduce the value of δ in the model, and increase the values of μ_1 and μ_2). When major emergencies occur, social platforms can use their own authority to supervise related information so as to reduce the emergence of uncertain information at the source. After uncertain information has emerged, the flow of some published remarks should be limited on the platform, or certain keywords should be blocked to reduce the dissemination rate. This suppresses the dissemination of uncertain information.

(2) Improve the ability of internet users to determine the authenticity of information, improve the reward and punishment mechanism, and encourage users to participate in the clarification of uncertain information (i.e., increase the values of ω and β_2 in the model and reduce the values of β_1, α_1, and α_3). Following a major emergency, the relevant governmental departments should release the real information related to the events in a timely manner and provide materials to support Internet users in carrying out independent investigations. The government should also seek to punish Internet users who release uncertain information to reduce its spread. Internet users who publish true information should be rewarded so as to encourage expression of users' own opinions and greater participation in clarifying uncertain information.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary Material, further inquiries can be directed to the corresponding author.

Author contributions

BL: Writing–original draft, Writing–review and editing. HL: Supervision, Writing–review and editing. QS: Writing–review and editing. RL: Writing–review and editing. HY: Writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. The research was supported by the Project of Liaoning Provincial Federation Social Science Circles of China (No. L20BGL047) and the Liaoning Provincial Social Science Planning Fund (No. L18AJY001).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Tan KY, Liu WH, Xu F. Optimization model and algorithm of logistics vehicle routing problem under major emergency. *Mathematics* (2023) 11(5):1274. doi:10.3390/math11051274

2. Yao J, Jin Y, Tang X, Wu J, Hou S, Liu X, et al. Development of intelligent response to public health emergencies. *Strateg Study CAE* (2021) 23(5):34–40. doi:10.15302/j-sscae-2021.05.005

3. Gao Y, Fan YX, Wang J, Duan Z. Evaluation of governmental safety regulatory functions in preventing major accidents in China. *Saf Sci* (2019) 120:299–311. doi:10.1016/j.ssci.2019.07.002

4. Zhang X, Zhou M. Emergency management planning for major epidemic disease prevention and control. *J Saf Sci Tech* (2020) 16(6):37–42.

5. Ukwuoma CC, Qin ZG, Heyat MBB, Akhtar F, Smahi A, Jackson JK, et al. Automated lung-related pneumonia and COVID-19 detection based on novel feature extraction framework and vision transformer approaches using chest X-ray images. *Bioengineering-basel* (2022) 9(11):709. doi:10.3390/bioengineering9110709

6. Benifa JVB, Chola C, Muaad AY, Hayat MAB, Bin Heyat MB, Mehrotra R, et al. FMDNet: an efficient system for face mask detection based on lightweight model during COVID-19 pandemic in public areas. *Sensors* (2023) 23(13):6090. doi:10.3390/s23136090

7. Asif S, Zhao M, Tang FX, Zhu Y. A deep learning-based framework for detecting COVID-19 patients using chest X-rays. *Multimedia Syst* (2022) 28(4):1495–513. doi:10.1007/s00530-022-00917-7

8. Fazmiya MJA, Sultana A, Rahman K. Current insights on bioactive molecules, antioxidant, anti-inflammatory, and other pharmacological activities of cinnamomum camphora linn. *Oxidative Med Cell Longevity* (2022) 2022:9354555.

9. Mo TC, Xie C, Li KL, Ouyang Y, Zeng Z. Transmission effect of extreme risks in China?s financial sectors at major emergencies: empirical study based on the GPD-CAViaR and TVP-SV- VAR approach. *Electron Res Archive* (2022) 30(12):4657–73. doi:10.3934/era.2022236

10. Cheng A, Chen TH, Jiang GG, Han X. Can major public health emergencies affect changes in international oil prices?. *Int J Environ Res Public Health* (2021) 18(24):12955. doi:10.3390/ijerph182412955

11. Yang G, Wang ZD, Chen L. Investigating the public sentiment in major public emergencies through the complex networks method: a case study of COVID-19 epidemic. *Front Public Health* (2022) 10:847161. doi:10.3389/fpubh.2022.847161

12. De Las Heras-Pedrosa C, SáNCHEZ-NúñEZ P, PeláEZ JI. Sentiment analysis and emotion understanding during the COVID-19 pandemic in Spain and its impact on digital ecosystems. *Int J Environ Res Public Health* (2020) 17(15):5542. doi:10.3390/ijerph17155542

13. Atehortua NA, Patino S. COVID-19, a tale of two pandemics: novel coronavirus and fake news messaging. *Health Promot Int* (2021) 36(2):524–34. doi:10.1093/heapro/daaa140

14. Jalan M, Riehm K, Agarwal S, Gibson D, Labrique A, Thrul J. Burden of mental distress in the US associated with trust in media for COVID-19 information. *Health Promot Int* (2022) 37(6):daac162. doi:10.1093/heapro/daac162

15. Zhang H, Zhou H, Li J. Research on the situational awareness of a major emergency under incomplete information. *J China Soc Scientific Tech Inf* (2021) 40(9):903–13.

16. Crokidakis N. Effects of mass media on opinion spreading in the Sznajd sociophysics model. *Physica a-Statistical Mech Its Appl* (2012) 391(4):1729–34. doi:10.1016/j.physa.2011.11.038

17. Zhao LJ, Wang Q, Cheng JJ, Zhang D, Ma T, Chen Y, et al. The impact of authorities' media and rumor dissemination on the evolution of emergency. *Physica a-Statistical Mech Its Appl* (2012) 391(15):3978–87. doi:10.1016/j.physa.2012.02.004

18. Yin FL, Shao XY, Wu JH. Nearcasting forwarding behaviors and information propagation in Chinese Sina-Microblog. *Math Biosciences Eng* (2019) 16(5):5380–94. doi:10.3934/mbe.2019268

19. Allington D, Duffy B, Wessely S, Dhavan N, Rubin J. Health-protective behaviour, social media usage and conspiracy belief during the COVID-19 public health emergency. *Psychol Med* (2021) 51(10):1763–9. doi:10.1017/s003329172000224x

20. Centola D. The spread of behavior in an online social network experiment. *Science* (2010) 329(5996):1194–7. doi:10.1126/science.1185231

21. Zhang X, Zhou Y, Zhou FL, Pratap S. Internet public opinion dissemination mechanism of COVID-19: evidence from the Shuanghuanglian event. *Data Tech Appl* (2022) 56(2):283–302. doi:10.1108/dta-11-2020-0275

22. Choi S. The two-step flow of communication in twitter-based public forums. *Soc Sci Comp Rev* (2015) 33(6):696–711. doi:10.1177/0894439314556599

23. Li SY, Liu ZX, Li YL. Temporal and spatial evolution of online public sentiment on emergencies. *Inf Process Manag* (2020) 57(2):102177. doi:10.1016/j.ipm.2019.102177

24. Li L, Wan YJ, Plewczynski D, Zhi M. Simulation model on network public opinion communication model of major public health emergency and management system design. *Scientific Programming* (2022) 2022:1–16. doi:10.1155/2022/5902445

25. Wei Y. Network public opinion propagation control model of major emergencies based on heat conduction theory. *Wireless Commun Mobile Comput* (2022) 2022:1476231.

26. Litou I, Boutsis I, Kalogeraki V. Efficient techniques for time-constrained information dissemination using location-based social networks. *Inf Syst* (2017) 64:321–49. doi:10.1016/j.is.2015.12.002

27. Wang J, Yu H, Wang X. Dissemination and control model of public opinion in online social networks based on users' relative weight. *Syst Engineering-Theory Pract* (2019) 39(6):1565–79.

28. Hong W, Li Q, Wu L. Food safety internet public opinion transmission simulation and management countermeasures considering information authenticity. *Syst Engineering-Theory Pract* (2017) 37(12):3253–69.

29. Kermack WO, Mckendrick AG. Contributions to the mathematical theory of epidemics—I. *Bull Math Biol* (1991) 53(1-2):33–55. doi:10.1007/bf02464423

30. Kermack WO, Mckendrick AG. Contributions to the mathematical theory of epidemics—II. The problem of endemicity. *Bull Math Biol* (1991) 53(1-2):57–87. doi:10.1007/bf02464424

31. Kermack WO, Mckendrick AG. Contributions to the mathematical theory of epidemics—III. Further studies of the problem of endemicity. *Bull Math Biol* (1991) 53(1-2):89–118. doi:10.1007/bf02464425

32. Duan W, Fan ZC, Zhang P, Guo G, Qiu X. Mathematical and computational approaches to epidemic modeling: a comprehensive review. *Front Comp Sci* (2015) 9(5):806–26. doi:10.1007/s11704-014-3369-2

33. Chowell G, Sattenspiel L, Bansal S, Viboud C. Mathematical models to characterize early epidemic growth: a review. *Phys Life Rev* (2016) 18:66–97. doi:10.1016/j.plrev.2016.07.005

34. Li MY, Muldowney JS. Global stability for the SEIR model in epidemiology. *Math biosciences* (1995) 125(2):155–64. doi:10.1016/0025-5564(95)92756-5

35. Haihong E, Hu YX, Peng HP. Theme and sentiment analysis model of public opinion dissemination based on generative adversarial network. *Chaos Solitons and Fractals* (2019) 121:160–7.

36. Yin FL, Lv JH, Zhang XJ, Xia X, Wu J. COVID-19 information propagation dynamics in the Chinese Sina-microblog. *Math Biosciences Eng* (2020) 17(3):2676–92. doi:10.3934/mbe.2020146

37. Yin FL, Pang HY, Xia XY. COVID-19 information contact and participation analysis and dynamic prediction in the Chinese Sina-microblog. *Physica a-Statistical Mech Its Appl* (2021) 570:125788. doi:10.1016/j.physa.2021.125788

38. Rui XB, Meng FR, Wang ZX, Yuan G, Du C. SPIR: the potential spreaders involved SIR model for information diffusion in social networks. *Physica a-Statistical Mech Its Appl* (2018) 506:254–69. doi:10.1016/j.physa.2018.04.062

39. Trpevski D, Tang WKS, Kocarev L. Model for rumor spreading over networks. *Phys Rev E* (2010) 81(5):056102. doi:10.1103/physreve.81.056102

40. Zan YL. DSIR double-rumors spreading model in complex networks. *Chaos Solitons and Fractals* (2018) 110:191–202. doi:10.1016/j.chaos.2018.03.021

41. Tian Y, Ding XJ. Rumor spreading model with considering debunking behavior in emergencies. *Appl Math Comput* (2019) 363:124599. doi:10.1016/j.amc.2019.124599

42. Diekmann O, Heesterbeek JA, Metz JA. On the definition and the computation of the basic reproduction ratio R0 in models for infectious diseases in heterogeneous populations. *J Math Biol* (1990) 28(4):365–82. doi:10.1007/bf00178324

43. Kang SD, Hou XL, Hu YH, Liu H. Dynamical analysis and optimal control of the developed information transmission model. *Plos One* (2022) 17(5):e0268326. doi:10.1371/journal.pone.0268326

44. Hu YH, Pan QH, Hou WB, He M. Rumor spreading model considering the proportion of wisemen in the crowd. *Physica a-Statistical Mech Its Appl* (2018) 505:1084–94. doi:10.1016/j.physa.2018.04.056

45. Lv R, Li H, Sun Q. Panic spreading model with different emotions under emergency. *Mathematics* (2021) 9(24):3190. doi:10.3390/math9243190

46. Kang SD, Hou XL, Hu YH, Liu H. Dynamic analysis and optimal control considering cross transmission and variation of information. *Scientific Rep* (2022) 12(1):18104. doi:10.1038/s41598-022-21774-4

47. He ZB, Cai ZP, Yu JG, Wang X, Sun Y, Li Y. Cost-efficient strategies for restraining rumor spreading in mobile social networks. *Ieee Trans Vehicular Tech* (2017) 66(3):2789–800. doi:10.1109/tvt.2016.2585591

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 06 March 2024
DOI 10.3389/fphy.2024.1364140

Check for updates

OPEN ACCESS

EDITED BY
Yuanyuan Huang,
Chengdu University of Information Technology,
China

REVIEWED BY
Peilin He,
University of Pittsburgh, United States
Tingting Song,
Jinan University, China

*CORRESPONDENCE
Min Hou,
✉ houmin@scujj.edu.cn

RECEIVED 01 January 2024
ACCEPTED 20 February 2024
PUBLISHED 06 March 2024

CITATION
Hou M and Wu Y (2024), Single-photon-based quantum secure protocol for the socialist millionaires' problem.
Front. Phys. 12:1364140.
doi: 10.3389/fphy.2024.1364140

Single-photon-based quantum secure protocol for the socialist millionaires' problem

Min Hou[1,2]* and Yue Wu[1]

[1]School of Computer Science, Sichuan University Jinjiang College, Meishan, China, [2]Network and Data Security Key Laboratory of Sichuan Province, School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, China

The socialist millionaires' problem, emanating from the millionaires' problem, allows two millionaires to determine whether they happen to be equally rich while remaining their riches undisclosed to each other. Most of the current quantum solutions to the socialist millionaires' problem have lower efficiency and are theoretically feasible. In this paper, we introduce a practical quantum secure protocol for the socialist millionaires' problem based on single photons, which can be easily implemented and manipulated with current technology. Our protocol necessitates the involvement of a semi-honest third party (TP) responsible for preparing the single-photon sequences and transmitting them to Alice who performs Identity or Hadamard operations on the received quantum sequences via her private inputs and the secret keys, producing new quantum sequences that are subsequently sent to Bob. Similarly, Bob encodes his private inputs into the received quantum sequences to produce new quantum sequences, which are then sent to TP. By conducting single-particle measurements on the quantum sequences received from Bob, TP can ascertain the equality of private inputs between Alice and Bob, and subsequently communicate the comparison result to them. To assess the feasibility, the proposed protocol is simulated on IBM Quantum Cloud Platform. Furthermore, security analysis demonstrates that our protocol can withstand attacks from outsiders, such as eavesdroppers, and from insider participants attempting to grab the private input of another participant.

KEYWORDS

single photons, quantum secure protocol, the socialist millionaires' problem, semi-honest third party, quantum cryptography

1 Introduction

Since Bennett and Brassard [1] introduced the pioneering quantum key distribution (QKD) protocol in 1984, leveraging the distinctive properties of quantum mechanics instead of relying on computational complexity problems and demonstrating its unconditional security, a multitude of quantum cryptographic protocols have since been developed. These include quantum secret sharing [2–4], quantum secure direct communication [5–7], and quantum key agreement [8, 9], aiming to address various cryptographic tasks. Quantum cryptography offers significant security advantages compared to classical cryptography, which is vulnerable to attacks from quantum algorithms (e.g., Shor's algorithm [10]).

In 1982, Andrew Yao [11] proposed the concept of the millionaires' problem, with the aim of solving the following task: two millionaires, each possessing their own wealth, seek to ascertain the wealthier party without revealing their financial status. Boudot et al. [12]

introduced an efficient scheme for the socialist millionaires' problem, relying on three standard assumptions: discrete logarithm, the Diffie–Hellman, and the Decision Diffie–Hellman. In this problem, two millionaires aim to ascertain the equality of their wealth. Nevertheless, as noted by Lo [13], securely evaluating an equality function in a two-party setting is deemed impossible. Consequently, the involvement of a third party (TP) becomes imperative to address the millionaires' problem. Indeed, addressing the socialist millionaires' problem is tantamount to formulating a private comparison protocol for confidentially comparing secrets. The reliability of the third party (TP) can be categorized into three types: completely honest, semi-honest, and dishonest. Since completely honest TP involvement in real life is hard to find, and implementing dishonest TP is difficult, semi-honest TP, who may misbehave but cannot collude with the participants, is a more reasonable and widely used approach in designing private comparison protocols up to now.

Quantum private comparison (QPC), which combines quantum mechanics and classical private comparison, can be used to solve the socialist millionaires' problem that achieves the comparison of the equality or inequality of two secrets while ensuring the security of information transmission. The first QPC protocol, incorporating EPR pairs and decoy photons, was suggested by Yang et al. [14] in 2009, which allows the equality relationship of two secrets to be determined by involving a TP who is barred from accessing either the comparison result or the private inputs. To conserve quantum resources, Chen et al. [15] introduced a QPC protocol using triplet entangled states. In this protocol, the classical message can be divided into multiple groups, and comparison results can be obtained even if not all data are completely compared. Lin et al. [16] identified vulnerabilities in the protocol described in Ref. [15], noting its susceptibility to intercept-resend attacks and emphasizing the need for improvements. Afterward, several QPC protocols were proposed using different quantum states as carriers of quantum information, such as single photons [17], Bell states [18, 19], multi-qubit entangled states [20–24], and multi-qubit cluster states [25–28]. In addition, Ye [29] proposed a QPC protocol using cavity quantum electrodynamics (QED), which requires two-atom product states as carriers of quantum information, and one two-atom product state can be utilized to perform the equality comparison of 1 bit in each round. Chen et al. [30] introduced a QPC protocol utilizing quantum walks on a circle. This protocol requires a two-particle quantum walk state and a quantum walk operator, and it can improve efficiency by allowing private inputs to be compared all at once rather than bit by bit. In order to compare the relationship of arbitrary single-qubit states, Huang et al. [31] constructed a QPC protocol by utilizing the special property of rotation encryption and swap test.

The QPC protocols mentioned above mainly utilize low-dimensional quantum states as carriers of quantum information, with the classical message encoded on these quantum states. In most quantum states, a single quantum state can only convey 1 bit of information, limiting the transmission efficiency of quantum states. To address the issue, some scholars have focused on developing QPC protocols based on high-dimensional quantum states instead of low-dimensional quantum states since high-dimensional quantum states can encode a greater amount of information. In 2011, Jia et al. [32] introduced d-level GHZ states to solve the millionaire problem. The private inputs are encoded into the phase of the initial quantum entangled states by performing local operations, and the phase information can be obtained by performing collective measurements. In 2013, Yu et al. [33] introduced d-level single particles to construct the QPC protocol, with the aim of comparing the size relationship of private inputs. Guo et al. [34] used entanglement swapping of d-level Bell states to determine the equality and size relationship of two secrets. Since the particles can be used multiple times, the scheme has an advantage in efficiency. After that, Li and Shi [35] proposed a QPC protocol utilizing quantum Fourier transforms, wherein the encoding of private inputs into the phase of the quantum state sent from the third party is employed. This protocol achieves higher communication efficiency by employing secret-by-secret comparisons rather than bit-by-bit comparisons. Ji et al. [36] used (n+1)-qubit GHZ states as quantum resources to compare the participants' secrets, and the requirement of quantum devices can be reduced as the protocol only employs quantum states and quantum measurements without the need for any entanglement swapping and unitary operations. Wu and Zhao [37] proposed a QPC based on d-level Bell states to determine the equality and size relationship of two secrets.

Based on the analysis of the aforementioned protocols, it can be deduced that QPC protocols utilizing low-dimensional quantum states as quantum information carriers, have lower transmission efficiency. In contrast, implementing high-dimensional quantum states-based QPC protocols poses challenges with current quantum technologies. In this paper, we introduce a practical QPC protocol to address the socialist millionaires' problem utilizing single photons, as they are easier to implement and manipulate with current technology. This protocol utilizes single photons as carriers of quantum information, with TP tasked with preparing groups of quantum sequences and transmitting them to Alice who performs Identity or Hadamard operations on the received quantum sequences via her private inputs and the secret keys to obtain new quantum sequences, which are then sent to Bob. Similarly, Bob encodes his private inputs into the received quantum sequences to produce new quantum sequences, which are then sent to TP. By conducting single-particle measurements on the quantum sequence received from Bob, TP can ascertain the equality of private inputs between Alice and Bob, and subsequently communicate the comparison results to them. Two simulation experiments are conducted on IBM Quantum Experience to showcase the feasibility of the proposed protocol. Additionally, the incorporation of decoy photons enables the detection of any potential eavesdropping during the eavesdropping detection process.

The remaining sections of this paper are structured as follows: Section 2 introduces preliminary knowledge, Section 3 outlines the detailed steps of the proposed quantum secure protocol for the socialist millionaires' problem, Section 4 conducts two simulation experiments, and Section 5 provides the corresponding analysis for the proposed protocol. Finally, Section 6 concludes the paper.

2 Preliminary knowledge

In this section, we will primarily introduce the Identity and Hadamard operations, which are equivalent to two quantum gates. In essence, a quantum gate can be represented as a unitary matrix. When performing a quantum gate on an n-qubit quantum state, the unitary matrix is of size $2^n \times 2^n$. For a single photon, also known as a single qubit, the unitary matrix is of size 2×2. Therefore, Identity or Hadamard operations can be represented as a 2×2 unitary matrix, as shown in the following equation.

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{1}$$

For a single qubit, performing the Identity operation will not change its state, while the state will change when performing the Hadamard operation. That is $|0\rangle \leftrightarrow |+\rangle, |1\rangle \leftrightarrow |-\rangle$.

Theorem 1. When using the Z-basis to measure $|0\rangle$ and $|1\rangle$ respectively, the measurement results yield $|0\rangle$ and $|1\rangle$ respectively with a probability of 1. However, when using Z-basis to measure $|+\rangle$ and $|-\rangle$ respectively, the measurement results yield $|0\rangle$ and $|1\rangle$ respectively, with an equal probability of 0.5.

Proof. The measurement operators of Z-basis can be represented as $M_0 = |0\rangle\langle 0|$ and $M_1 = |1\rangle\langle 1|$, where M_0 and M_1 are Hermitian matrices and satisfy the completeness equation, that is,

$$I = M_0^\dagger M_0 + M_1^\dagger M_1 \tag{2}$$

When performing the measurement on $|0\rangle$ with the Z-basis, the probabilities that the measurement results yield $|0\rangle$ and $|1\rangle$ respectively can be given by

$$p_1(|0\rangle) = \langle 0|M_0^\dagger M_0|0\rangle = \langle 0||0\rangle\langle 0||0\rangle = 1 \tag{3}$$

$$p_1(|1\rangle) = \langle 0|M_1^\dagger M_1|0\rangle = \langle 0||1\rangle\langle 1||0\rangle = 0 \tag{4}$$

When performing the measurement on $|1\rangle$ with the Z-basis, the probabilities that the measurement results yield $|0\rangle$ and $|1\rangle$ respectively can be given by

$$p_2(|0\rangle) = \langle 1|M_0^\dagger M_0|1\rangle = \langle 1||0\rangle\langle 0||1\rangle = 0 \tag{5}$$

$$p_2(|1\rangle) = \langle 1|M_1^\dagger M_1|1\rangle = \langle 1||1\rangle\langle 1||1\rangle = 1 \tag{6}$$

When performing the measurement on $|+\rangle$ with the Z-basis, the probabilities that the measurement results yield $|0\rangle$ and $|1\rangle$ respectively can be given by

$$p_3(|0\rangle) = \langle +|M_0^\dagger M_0|+\rangle = \frac{\langle 0| + \langle 1|}{\sqrt{2}}|0\rangle\langle 0|\frac{|0\rangle + |1\rangle}{\sqrt{2}} = \frac{1}{2} \tag{7}$$

$$p_3(|1\rangle) = \langle +|M_1^\dagger M_1|+\rangle = \frac{\langle 0| + \langle 1|}{\sqrt{2}}|1\rangle\langle 1|\frac{|0\rangle + |1\rangle}{\sqrt{2}} = \frac{1}{2} \tag{8}$$

When performing the measurement on $|-\rangle$ with the Z-basis, the probabilities that the measurement results yield $|0\rangle$ and $|1\rangle$ respectively can be given by

$$p_4(|0\rangle) = \langle -|M_0^\dagger M_0|-\rangle = \frac{\langle 0| - \langle 1|}{\sqrt{2}}|0\rangle\langle 0|\frac{|0\rangle - |1\rangle}{\sqrt{2}} = \frac{1}{2} \tag{9}$$

$$p_4(|1\rangle) = \langle -|M_1^\dagger M_1|-\rangle = \frac{\langle 0| - \langle 1|}{\sqrt{2}}|1\rangle\langle 1|\frac{|0\rangle - |1\rangle}{\sqrt{2}} = \frac{1}{2} \tag{10}$$

From Eqs 3–6, we can conclude that when using the Z-basis to measure $|0\rangle$ and $|1\rangle$ respectively, the measurement results are $|0\rangle$ and $|1\rangle$ respectively with a probability of 1. From Eqs 7–10, we can also conclude that when using the Z-basis to measure $|+\rangle$ and $|-\rangle$ respectively, the measurement results are $|0\rangle$ and $|1\rangle$ respectively with the same probability of 0.5.

Theorem 2. When using the X-basis to measure $|0\rangle$ and $|1\rangle$ respectively, the measurement results are $|+\rangle$ and $|-\rangle$ respectively with an equal probability of 0.5. However, when using the X-basis to measure $|+\rangle$ or $|-\rangle$ respectively, the measurement results yield $|+\rangle$ and $|-\rangle$ respectively with a probability of 1.

Proof. The measurement operators of X-basis can be represented as $\boldsymbol{M}_+ = |+\rangle\langle +|$ and $M_- = |-\rangle\langle -|$, where M_+ and M_- are also Hermitian matrices and satisfy the completeness equation as well, that is,

$$I = M_+^\dagger M_+ + M_-^\dagger M_- \tag{11}$$

When performing the measurement on $|0\rangle$ with the X-basis, the probabilities that the measurement results yield $|+\rangle$ and $|-\rangle$ respectively can be given by

$$p_5(|+\rangle) = \langle 0|M_+^\dagger M_+|0\rangle = \langle 0||+\rangle\langle +||0\rangle = \frac{1}{2} \tag{12}$$

$$p_5(|-\rangle) = \langle 0|M_-^\dagger M_-|0\rangle = \langle 0||-\rangle\langle -||0\rangle = \frac{1}{2} \tag{13}$$

When performing the measurement on $|1\rangle$ with the X-basis, the probabilities that the measurement results yield $|+\rangle$ and $|-\rangle$ respectively can be given by

$$p_6(|+\rangle) = \langle 1|M_+^\dagger M_+|1\rangle = \langle 1||+\rangle\langle +||1\rangle = \frac{1}{2} \tag{14}$$

$$p_6(|-\rangle) = \langle 1|M_-^\dagger M_-|1\rangle = \langle 1||-\rangle\langle -||1\rangle = \frac{1}{2} \tag{15}$$

When performing the measurement on $|+\rangle$ with the X-basis, the probabilities that the measurement results yield $|+\rangle$ and $|-\rangle$ respectively can be given by

$$p_7(|+\rangle) = \langle +|M_+^\dagger M_+|+\rangle = \langle +||+\rangle\langle +||+\rangle = 1 \tag{16}$$

$$p_7(|-\rangle) = \langle +|M_-^\dagger M_-|+\rangle = \langle +||-\rangle\langle -||+\rangle = 0 \tag{17}$$

When performing the measurement on $|-\rangle$ with the X-basis, the probabilities that the measurement results yield $|+\rangle$ and $|-\rangle$ respectively can be given by

$$p_8(|+\rangle) = \langle -|M_+^\dagger M_+|-\rangle = \langle -||+\rangle\langle +||-\rangle = 0 \tag{18}$$

$$p_8(|-\rangle) = \langle -|M_-^\dagger M_-|-\rangle = \langle -||-\rangle\langle -||-\rangle = 1 \tag{19}$$

From Eqs 12–15, we can also conclude that when using the X-basis to measure $|0\rangle$ and $|1\rangle$ respectively, the measurement results are $|+\rangle$ and $|-\rangle$ respectively with the same probability of

0.5. From Eqs 16–19, we can also conclude that when using the X-basis to measure $|+\rangle$ and $|-\rangle$ respectively, the measurement results are $|+\rangle$ and $|-\rangle$ respectively with a probability of 1.

3 Quantum secure protocol for the socialist millionaires' problem

The quantum secure protocol for the socialist millionaires' problem is run between two participants, each of whom possesses two secret inputs, A and B, respectively. The two participants aim to determine the equality relationship between A and B. The binary representations of A and B in F_2^L can be represented as $A' = (a_1, a_2, \cdots, a_L)$ and $B' = (b_1, b_2, \cdots, b_L)$, where L is the length of A' and B'. If the length of A' and B' is less than L, Alice and Bob fill in the high digit with adequate zeros. A semi-honest third party is engaged in the preparation of the sequence of single photons. In the entire process, TP may have access to some immediate computation processes, but she cannot collude with any participant. Before the protocol is executed, TP shares a secret key $TA = (ta_1, ta_2, \cdots, ta_L)$ and $TB = (tb_1, tb_2, \cdots, tb_L)$ between Alice and Bob via a secure QKD protocol, respectively. Additionally, Alice and Bob also share a secret key $AB = (ab_1, ab_2, \cdots, ab_L)$ using a secure QKD protocol.

The detailed steps of the proposed protocol are described in the following procedure.

Step 1: TP prepares λ groups of quantum sequences denoted as $S = (\otimes_{i=1}^{L} s_i^1; \otimes_{i=1}^{L} s_i^2; \cdots \otimes_{i=1}^{L} s_i^{\lambda})$, with each group being equivalent and containing L photons randomly selected from $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. Then, she prepares δ decoy photons and inserts them into the sequence S at random positions to produce a new sequence S' and notes the positions of the decoy photons in S' and each quantum state in sequence S. Finally, TP sends S' to Alice.

Step 2: Upon receiving S', Alice and TP perform the eavesdropping detection to identify the presence of any eavesdropper. When TP knows that Alice has received S', TP securely conveys the positions of the decoy photons and their corresponding measurement bases to Alice through a classical channel. Subsequently, Alice measures the decoy photons using the provided measurement bases and communicates the measurement results back to TP. TP then compares these results with the originally prepared δ decoy photons. If they are different, the process is returned to Step 1. Otherwise, they proceed with the following steps.

Step 3: Alice discards the decoy photons to get S. If $a_i \oplus ta_i \oplus ab_i = 0$, Alice applies the Identity operation to each photon within the λ groups in S. Otherwise, Alice applies the Hadamard operation to each photon within the λ groups in S. Let the resultant sequence be S_A. To detect the eavesdropper, Alice adds δ decoy photons into S_A to produce a fresh sequence $S_A^{'}$, which is then sent to Bob.

Step 4: Upon receiving $S_A^{'}$, Alice and Bob perform the eavesdropping detection in the same manner as TP. If no eavesdropper is detected, Bob removes the decoy photons from $S_A^{'}$ to get S_A. If $b_i \oplus tb_i \oplus ab_i = 0$, Bob performs the Identity operation. Otherwise, Bob performs the Hadamard operation. Let the resultant sequence be S_B. To prevent eavesdropping, Bob adds δ decoy photons into S_B to produce a fresh sequence $S_B^{'}$, which is then sent to TP.

Step 5: Upon receiving $S_B^{'}$, TP interacts with Bob in the same manner as Alice and Bob to check whether the eavesdropper exists. If not, TP gets S_B by removing the decoy photons from $S_B^{'}$. In the following, TP applies the Identity or Hadamard operation to each photon within the λ groups in S_B to produce a new sequence S_{TP}. If $ta_i \oplus tb_i = 0$, TP performs the Identity operation. Otherwise, TP performs the Hadamard operation. TP measures each photon of the λ groups in S_{TP} with the measurement basis determined by the initial prepared quantum state in S to get the measurement results. If the photon stays in $|0\rangle$ or $|1\rangle$, the measurement basis is the Z-basis, Otherwise, the measurement basis is the X-basis.

Step 6: TP communicates the comparison results to both Alice and Bob. If all measurement results in S_{TP} are the same as the initially prepared quantum state in S, A and B are identical. Otherwise, A and B are different.

4 Simulation experiments

Since single photons are easier to implement and manipulate compared to low-dimensional and high-dimensional quantum states, we simulate the aforementioned protocol on IBM Quantum Experience using two concrete examples to demonstrate its feasibility and correctness. The specifics of two simulation experiments are outlined below.

4.1 Simulation I. Alice and Bob desire to compare their private inputs, with A = 12 and B = 12, respectively

A and B can be denoted as $A' = 1100$ and $B' = 1100$ in the form of binary representations in F_2^L. For the sake of simplicity, any eavesdropping or attacks will not be considered in the simulation experiments. We assume that TP shares the secret keys $TA = 1011$ and $TB = 1001$ respectively, and then Alice and Bob also share a secret key $AB = 1101$.

Suppose that the initial quantum sequence prepared by TP is $S = \{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. After that, Alice performs the operators $\{H, I, H, I\}$ on each photon of S to get $S_A = \{H|0\rangle, I|1\rangle, H|+\rangle, I|-\rangle\}$ and then she sends S_A to Bob. In the same way, Bob performs the operator $\{H, I, I, I\}$ on each photon of S_A to get $S_B = \{HH|0\rangle, II|1\rangle, IH|+\rangle, II|-\rangle\}$ and then she sends S_B to TP. Finally, TP performs the operators $\{I, I, H, I\}$ on each photon of S_B to get $S_{TP} = \{IHH|0\rangle, III|1\rangle, HIH|+\rangle, III|-\rangle\} = \{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$ and then measures S_{TP} with the measurement bases determined by the initially prepared quantum state in S to get the measurement results. That is, TP measures S_{TP} with basis $\{Z, Z, X, X\}$ to get the measurement results denoted as $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. Therefore, we can see that all measurement results are the same as the initially prepared quantum state, indicating that A and B are identical.

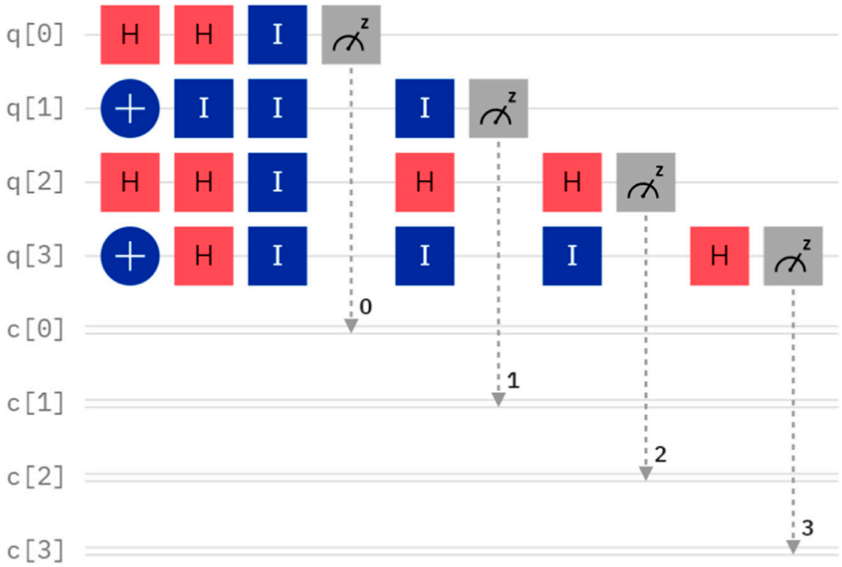


FIGURE 1
The quantum circuit of Simulation I.

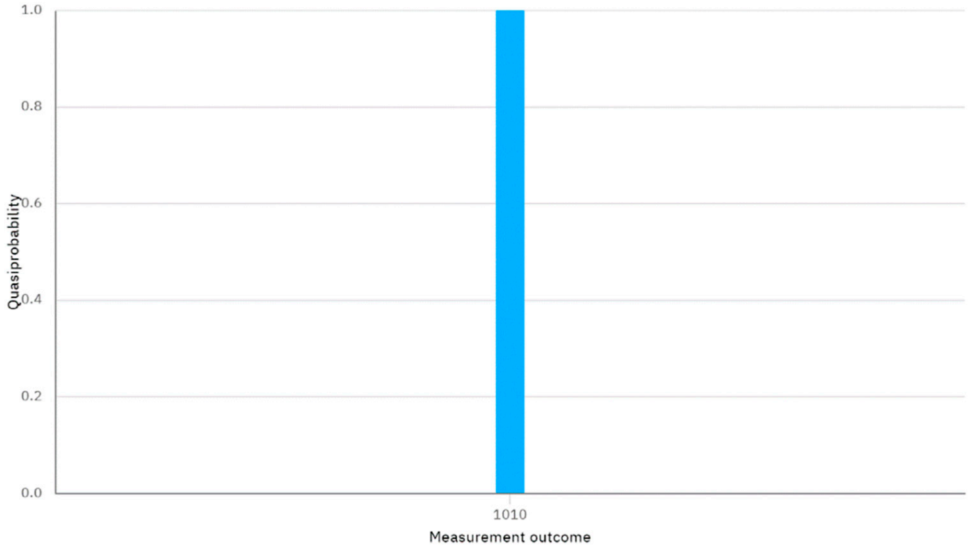


FIGURE 2
The measurement outcome in Figure 1.

The quantum circuit for Case I is depicted in Figure 1. By executing the quantum circuit on IBM Quantum Experience, we can obtain the measurement results shown in Figure 2. In Figure 2, the string on the horizontal axis represents the measurement outcome, corresponding to q [0]-q [3] from right to left. The value on the vertical axis represents the quasiprobability. It is important to note that both the measurement bases selected in q [2] and q [3] are the X basis, and the measurement outcome 1 and 0 are considered as $|+\rangle$ and $|-\rangle$ respectively. From Figure 2, we can see that the final measurement outcome is $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$, which is the same as the initial prepared quantum state. This indicates that A and B are identical.

4.2 Simulation II. Alice and Bob desire to compare their private inputs, with A = 55 and B = 22, respectively

A and B can be represented as $A' = 110111$ and $B' = 10110$ in the form of binary representations in F_2^L. Suppose that $L = 6$, we can see that the length of B' is less than L, Bob will fill in the necessary 0s

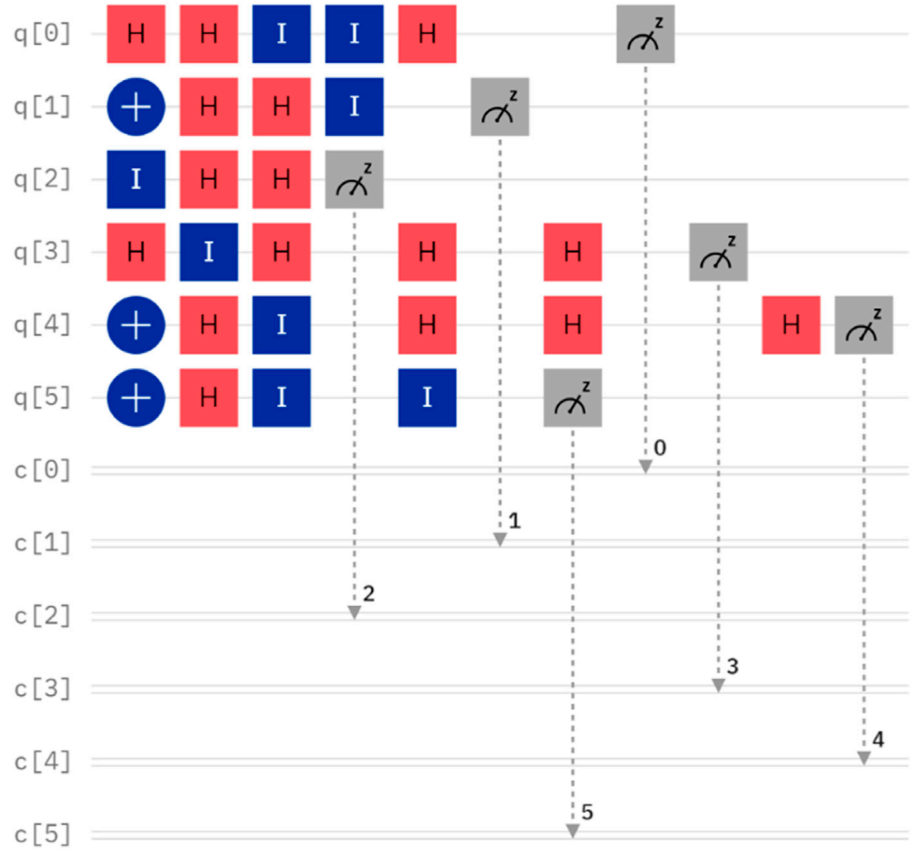


FIGURE 3
The quantum circuit of Simulation II.

at the higher digits and thus $B' = 010110$. We assume that TP shares the secret keys $TA = 101011$ and $TB = 100101$ between Alice and Bob, respectively, and Alice and Bob also share a secret key $AB = 101101$.

Suppose that the initial quantum sequence prepared by TP is $S = \{|+\rangle, |1\rangle, |0\rangle, |+\rangle, |-\rangle, |1\rangle\}$. Afterward, Alice performs the operators $\{H, H, I, I, I, H\}$ on each photon of S to get $S_A = \{H|+\rangle, H|1\rangle, I|0\rangle, I|+\rangle, I|-\rangle, H|1\rangle\}$, which is then sent to Bob. In the same way, Bob performs the operators $\{I, H, H, H, H, I\}$ on each photon of S_A to get $S_B = \{IH|+\rangle, HH|1\rangle, HI|0\rangle, HI|+\rangle, HI|-\rangle, IH|1\rangle\}$, which is then sent to TP. Finally, TP performs the operators $\{I, I, H, H, H, I\}$ on each photon of S_B to get $S_{TP} = \{IIH|+\rangle, IHH|1\rangle, HHI|0\rangle, HHI|+\rangle, HHI|-\rangle, IIH|1\rangle\} = \{|0\rangle, |1\rangle, |0\rangle, |+\rangle, |-\rangle, |-\rangle\}$ and then TP measures S_{TP} using the measurement bases determined by the initially prepared quantum state in S to obtain the measurement results. That is, TP measures S_{TP} with basis $\{X, Z, Z, X, X, Z\}$ to get the measurement results denoted as $\{|+\rangle or |-\rangle, |1\rangle, |0\rangle, |+\rangle, |-\rangle, |0\rangle\, or\, |1\rangle\}$. Therefore, we can see that not all measurement results are the same as the initially prepared quantum state, indicating that A and B are different.

The quantum circuit for Case II is depicted in Figure 3. By executing the quantum circuit on IBM Quantum Experience, we can obtain the measurement results shown in Figure 4. It is important to note that the measurement basis selected in q [0], q [3], and q [4] are all based on the X basis. The measurement outcome 1 and 0 can be considered as $|+\rangle$ and $|-\rangle$ respectively. From Figure 2, we can see that the measurement outcome is $\{|+\rangle or |-\rangle, |1\rangle, |0\rangle, |+\rangle, |-\rangle, |0\rangle\, or\, |1\rangle\}$, which corresponds to the measurement outcome q [0]-q [5] from right to left. Since the measurement outcome is not the same as the initial quantum state, A and B are different.

5 Analysis

5.1 Correctness analysis

In the proposed protocol, TP prepares λ groups of quantum sequences denoted as $S = (\otimes_{i=1}^{L} s_i^1; \otimes_{i=1}^{L} s_i^2; \cdots \otimes_{i=1}^{L} s_i^{\lambda})$, which is sent to Alice. Then, Alice applies the Identity or Hadamard operation to each photon within the λ groups in S_{TP} according to her private inputs and the secret keys. Thus, we can get

$$S_A = \left(\otimes_{i=1}^{L} (IorH) s_i^1; \otimes_{i=1}^{L} (IorH) s_i^2; \cdots \otimes_{i=1}^{L} (IorH) s_i^{\lambda}\right) \tag{20}$$

After that, S_A is sent to Bob. Bob also applies the Identity or Hadamard operation to each photon within the λ groups in S_A according to her private inputs and the secret keys. Thus, we can also get

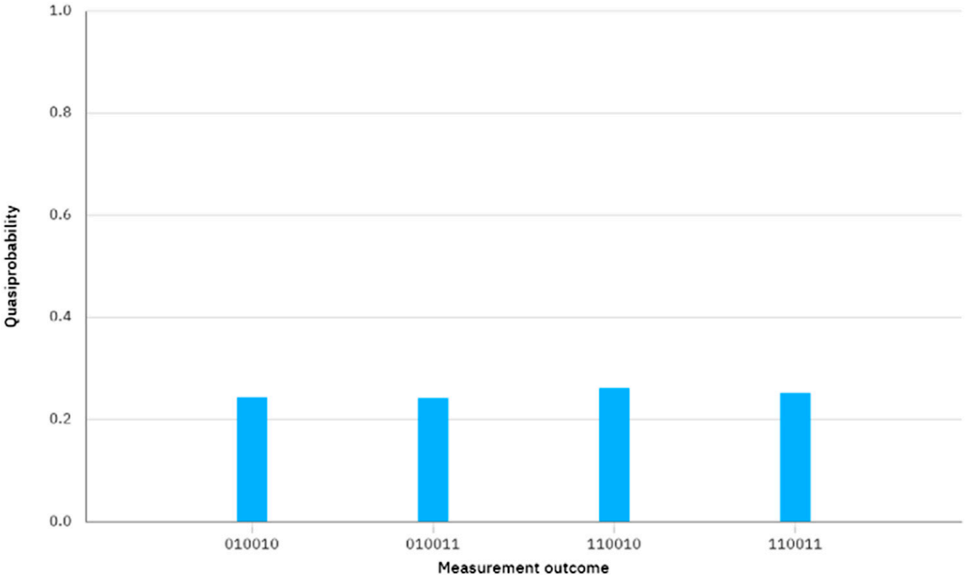


FIGURE 4
The measurement outcome in Figure 3.

$$S_B = \left(\otimes_{i=1}^{L} (IorH)(IorH)s_i^1; \otimes_{i=1}^{L} (IorH)(IorH)s_i^2; \cdots \otimes_{i=1}^{L} (IorH)(IorH)s_i^\lambda\right) \quad (21)$$

After that, S_B is sent to TP. TP also applies the Identity or Hadamard operation to each photon within the λ groups in S_B to produce a new sequence S_{TP}. If $ta_i \oplus tb_i = 0$, Bob performs the Identity operation. Otherwise, Bob performs the Hadamard operation.

There are four cases that should be considered.

Case I: If $a_i = 0$ and $b_i = 0$, then

$$S_{TP} = \left(\otimes_{i=1}^{L} s_i^1; \otimes_{i=1}^{L} s_i^2; \cdots \otimes_{i=1}^{L} s_i^\lambda\right) \quad (22)$$

When TP measures each group of quantum states in S_{TP} with the measurement bases determined by the initially prepared quantum state in S. We can easily observe that all the ith qubits in each group of S_{TP} are the same as the initially prepared ith qubits in each group of S, indicating that A and B are identical.

Case II: If $a_i = 1$ and $b_i = 0$, then

$$S_{TP} = \left(\otimes_{i=1}^{L} Hs_i^1; \otimes_{i=1}^{L} Hs_i^2; \cdots \otimes_{i=1}^{L} Hs_i^\lambda\right) \quad (23)$$

When TP measures each group of quantum states in S_{TP} with the measurement bases determined by the initially prepared quantum state in S. It is easy to see that not all the ith qubits in each group of S_{TP} are the same as the initially prepared ith qubits in each group of S, indicating that A and B are not identical.

Case III: If $a_i = 0$ and $b_i = 1$, then

$$S_{TP} = \left(\otimes_{i=1}^{L} Hs_i^1; \otimes_{i=1}^{L} Hs_i^2; \cdots \otimes_{i=1}^{L} Hs_i^\lambda\right) \quad (24)$$

We can see that S_{TP} in Case III is the same as in the Case II, and thus we can deduce that A and B are not identical.

Case IV: If $a_i = 1$ and $b_i = 1$, then

$$S_{TP} = \left(\otimes_{i=1}^{L} s_i^1; \otimes_{i=1}^{L} s_i^2; \cdots \otimes_{i=1}^{L} s_i^\lambda\right) \quad (25)$$

Similarly, we can also observe that S_{TP} in Case VI is the same as in Case I, and thus we can deduce that A and B are not identical.

Therefore, the above results reveal that our protocol is correct.

5.2 Security analysis

5.2.1 External attacks

External attacks involve an outsider eavesdropper, Eve, who may attempt to obtain valuable information about Alice's or Bob's private inputs during the transmission of the quantum sequence between the participants. Unfortunately, decoy photons are used during the transmission of each quantum sequence. Both the sender and receiver of the quantum sequences will perform the eavesdropping detection to verify the presence of any eavesdropper. This technique guarantees the security of the quantum sequence transmission, and any external attacks including intercept-resend attack, auxiliary particle attack, the man-in-the-middle attack and denial-of-service (Dos) attacks are invalid. In this context, we primarily delve into the security aspects of the proposed protocol concerning intercept-resend attacks, entanglement-measure attacks, and Trojan-Horse attacks in detail.

5.2.1.1 The intercept-resend attack

The intercept-resend attack refers to the outsider eavesdropper, Eve, intercepting the sequence sent from the previous participant during the transmission of each quantum sequence. Once Eve obtains the quantum sequence that carries the private inputs, she has the option to measure them using the Z-basis and send a fake sequence whose states match the measurement results instead of the initial quantum sequences to the original receiver. We assume that when a sender's initial quantum state is $|0\rangle$ or $|1\rangle$, and Eve intercepts and measures it with the Z-basis, she will evade eavesdropping detection. If Eve measures it using the X-basis, she will successfully evade eavesdropping detection with a probability of

1/2. For any selected decoy photon, the probability that Eve can correctly choose the measurement basis is 1/2. Therefore, the error rate of a decoy state that Eve introduced in the eavesdropping detection is $\left(1-\frac{1}{2}\times 1-\frac{1}{2}\times\frac{1}{2}\right)=\frac{1}{4}$. Since the number of decoy photons is δ, the probability of detecting the decoy states that Eve resends incorrectly is $1-\left(\frac{3}{4}\right)^{\delta}$. It is important to note that if δ is sufficiently large, the error rate introduced by Eve in the eavesdropping detection will approach 1, indicating that Eve's eavesdropping will be detected by the sender and the receiver, and the entire protocol process will need to be restarted. Therefore, the intercept-resend attack carried out by Eve is invalid, and her attempts to pilfer any valuable information regarding Alice's or Bob's private inputs prove futile.

5.2.1.2 The entanglement-measure attack

The entanglement-measure attack involves an outsider eavesdropper, Eve, intercepting the sequence sent from the previous participant during the transmission of each quantum sequence. She then performs unitary operations to entangle the prepared auxiliary particle sequence $E=\{|E_0\rangle, |E_1\rangle, \cdots, |E_n\rangle\}$ with the intercepted single-photon sequence. And the unitary operations performed on each single photon can be denoted as

$$U|E_i\rangle|0\rangle = a|e_{00}\rangle|0\rangle + b|e_{01}\rangle|1\rangle \tag{26}$$

$$U|E_i\rangle|1\rangle = c|e_{10}\rangle|0\rangle + d|e_{11}\rangle|1\rangle \tag{27}$$

$$\begin{aligned} U|E_i\rangle|+\rangle &= U|E_i\rangle\otimes\frac{|0\rangle+|1\rangle}{\sqrt{2}} \\ &= \frac{1}{\sqrt{2}}\left(a|e_{00}\rangle|0\rangle + b|e_{01}\rangle|1\rangle + c|e_{10}\rangle|0\rangle + d|e_{11}\rangle|1\rangle\right) \\ &= \frac{1}{\sqrt{2}}\begin{pmatrix} a|e_{00}\rangle\otimes\frac{|+\rangle+|-\rangle}{\sqrt{2}} + b|e_{01}\rangle\otimes\frac{|+\rangle-|-\rangle}{\sqrt{2}} \\ +c|e_{10}\rangle\otimes\frac{|+\rangle+|-\rangle}{\sqrt{2}} + d|e_{11}\rangle\otimes\frac{|+\rangle-|-\rangle}{\sqrt{2}} \end{pmatrix} \\ &= \frac{1}{2}\begin{bmatrix} |+\rangle\left(a|e_{00}\rangle + b|e_{01}\rangle + c|e_{10}\rangle + d|e_{11}\rangle\right) \\ +|-\rangle\left(a|e_{00}\rangle - b|e_{01}\rangle + c|e_{10}\rangle - d|e_{11}\rangle\right) \end{bmatrix} \end{aligned} \tag{28}$$

$$\begin{aligned} U|E_i\rangle|-\rangle &= U|E_i\rangle\otimes\frac{|0\rangle-|1\rangle}{\sqrt{2}} \\ &= \frac{1}{\sqrt{2}}\left(a|e_{00}\rangle|0\rangle + b|e_{01}\rangle|1\rangle - c|e_{10}\rangle|0\rangle - d|e_{11}\rangle|1\rangle\right) \\ &= \frac{1}{\sqrt{2}}\begin{pmatrix} a|e_{00}\rangle\otimes\frac{|+\rangle+|-\rangle}{\sqrt{2}} + b|e_{01}\rangle\otimes\frac{|+\rangle-|-\rangle}{\sqrt{2}} \\ -c|e_{10}\rangle\otimes\frac{|+\rangle+|-\rangle}{\sqrt{2}} - d|e_{11}\rangle\otimes\frac{|+\rangle-|-\rangle}{\sqrt{2}} \end{pmatrix} \\ &= \frac{1}{2}\begin{bmatrix} |+\rangle\left(a|e_{00}\rangle + b|e_{01}\rangle - c|e_{10}\rangle - d|e_{11}\rangle\right) \\ +|-\rangle\left(a|e_{00}\rangle - b|e_{01}\rangle - c|e_{10}\rangle + d|e_{11}\rangle\right) \end{bmatrix} \end{aligned} \tag{29}$$

$\{|e_{00}\rangle, |e_{01}\rangle, |e_{10}\rangle, |e_{11}\rangle\}$ are four pure quantum states that are determined by the unitary operations U, and they satisfy the following condition.

$$\sum_{\alpha,\beta}\langle e_{\alpha,\beta}|e_{\alpha,\beta}\rangle = 1 \tag{30}$$

Moreover, the parameters a, b, c, and d satisfy the condition, e.g., $|a|^2+|b|^2=1$ and $|c|^2+|d|^2=1$. In the proposed protocol, the eavesdropping detection is performed between each transmission of the quantum sequence. If the decoy photon is in state $|0\rangle$ or $|1\rangle$ and Eve wants to avoid detection, the parameters b and c must satisfy $b=c=0$. Similarly, if the decoy photon is in state $|+\rangle$ or $|-\rangle$ and Eve wants to avoid detection, then $a|e_{00}\rangle - b|e_{01}\rangle + c|e_{10}\rangle - d|e_{11}\rangle = \vec{0}$ and $a|e_{00}\rangle + b|e_{01}\rangle - c|e_{10}\rangle - d|e_{11}\rangle = \vec{0}$. Therefore, we can easily deduce that

$$a|e_{00}\rangle = d|e_{11}\rangle \tag{31}$$

When Substituting Eq. 31 and $b=c=0$ into Eqs 26–29, we can get

$$U|E_i\rangle|0\rangle = a|e_{00}\rangle|0\rangle \tag{32}$$

$$U|E_i\rangle|1\rangle = a|e_{00}\rangle|1\rangle \tag{33}$$

$$U|E_i\rangle|+\rangle = a|e_{00}\rangle|+\rangle \tag{34}$$

$$U|E_i\rangle|-\rangle = a|e_{00}\rangle|-\rangle \tag{35}$$

From Eqs 32–35, we can easily see that the auxiliary particles are not related to the intercepted ones. No matter what the intercept particles are, the auxiliary particles will always be in $|e_{00}\rangle$. As a result, Eve will fail to evade eavesdropping detection by performing the entanglement-measure attack, and her attempts to pilfer any valuable information regarding Alice's or Bob's private inputs also prove futile.

5.2.1.3 The Trojan-Horse attacks

The Trojan-Horse attacks [38] mainly include the delay-photon attack and the invisible photon eavesdropping attack. These attacks may occur in a two-way communication protocol where quantum states are returned to the sender. Since our protocol is a two-way communication protocol, the initial quantum sequence prepared by TP is returned to TP and the quantum sequence is transmitted in a circular mode. Therefore, the Trojan-Horse attacks should be considered. In order to prevent these attacks, both the Wavelength Quantum Filter (WQF) and the Photons Number Splitter (PNS) should be equipped to remove invisible photons and separate legitimate photons from delayed photons, respectively.

5.2.2 Participants' attack

Since the participants have the legal capacity to access more information compared to an outside eavesdropper, the dishonest individual has a high probability of obtaining the private input of the dishonest participant without being detected. Therefore, participants' attack as high security risk should be prevented by taking appropriate measures. Here, we analyze three types of attacks by participants that are aimed at obtaining the private input of the participants.

5.2.2.1 The attack from TP

As a semi-honest party, TP may exhibit improper behavior, but she cannot collude with either Alice or Bob. If TP intends to usurp the private input of Alice or Bob, she may perform external attacks similar to Eve. Unfortunately, this action will be detected, as discussed in Section 5.2.1, and TP cannot avoid detection by eavesdropping. Although TP has some advantages in generating the initial quantum sequences used for information transmission and receiving the sequences encoded with private inputs and secret keys, TP can only gain knowledge about the comparison result. In

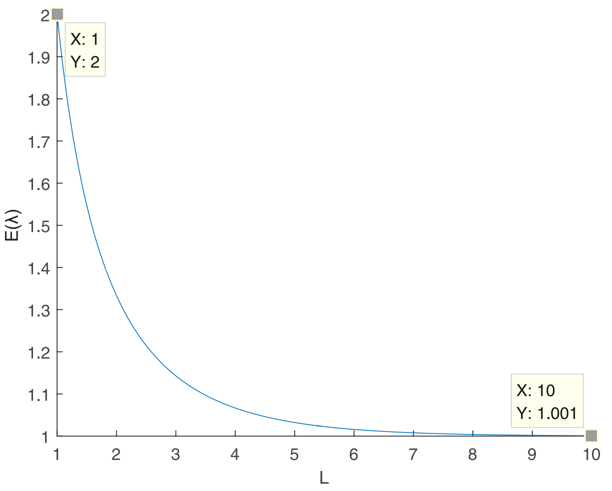


FIGURE 5
The relationship between L and E(λ).

other words, TP is able to determine whether a bit of Alice and Bob is identical or not, but it will not disclose whether the bit of Alice or Bob is 0 or 1. In addition, both S_A and S_B are encoded with the private inputs and the secret keys shared, TP remains unable to access any information regarding the private inputs of Alice and Bob without knowledge of the key AB. Therefore, the proposed protocol is resistant to TP's attack.

5.2.2.2 The attack from Alice

When TP sends S to Alice, Alice can intercept and measure it directly. And then she sends carefully prepared quantum sequences denoted as S_A'' to Bob. When Bob applies the Identity or Hadamard operation to each photon within S_A'' via his private inputs and the secret keys to obtain new quantum sequences denoted as S_B', which is sent to TP. Afterward, Alice launches the intercept-resend attack on S_B' that Bob sends to TP. In other words, Alice can intercept S_B' and send a fake sequence S_B'' to TP. Once TP receives the counterfeit sequence S_B'', Bob will convey the positions of the decoy photons and their corresponding measurement bases. Simultaneously, Alice is aware of the positions of the decoy photons in S_B'' and she can discard them. Then Alice measures the remaining particles in S_B'' to obtain the measurement result. Although this attack can be identified through the eavesdropping detection mechanism, Alice has already obtained the final states, allowing her to deduce the operations that Bob performs. However, Bob's actions are influenced by his private inputs and the confidential key TB shared exclusively between TP and Bob. Alice remains unable to access any information regarding Bob's secrets without knowledge of the key TB.

5.2.2.3 The attack from Bob

When Alice sends S_A to Bob, Bob can measure each particle in S_A directly and obtain the measurement result. Bob can also infer which operations that Alice performs. However, this attack will not work. Firstly, the sequence S prepared by TP will not be disclosed to Bob due to the simi-honesty of TP. Once Bob intends to know S by performing outside attacks just like Eve does, he will be detected in the eavesdropping detection. In addition, S_A is encoded with the private inputs of Alice and the secret key TA shared between TP and Alice, and Alice also remains unable to access any information regarding Alice's secrets without knowledge of the key TA.

In summary, the proposed protocol remains resilient against attacks from the participants, ensuring that the secrets of both Alice and Bob are not compromised.

5.3 Efficiency analysis and comparison

In most of QPC protocol, the qubit efficiency is an important indicator for evaluating the utilization rate of quantum states. However, it does not take into account the decoy photons used in eavesdropping detection, which can be considered as an independent process.

The qubit efficiency [39] η_e is given by

$$\eta_e = \frac{\eta_c}{\eta_t} \tag{36}$$

Where η_c represents the total number of bits that Alice and Bob want to compare, and η_t represents the total number of qubits used, excluding the decoy photons. In our protocol, L-length classical-bit information needs to be encoded using λL single photons as the information carriers to encode them. Therefore, the qubit efficiency of the proposed protocol is $\eta_e = \frac{1}{\lambda}$, where λ represents the number of repetitively prepared quantum sequences.

Next, we will discuss the value of $E(\lambda)$, which represents the average number of times the quantum sequences are repetitively prepared. In Section 5.1, we can conclude that for all ith qubits in each group of quantum states in S_{TP}, the measurement result of S_{TP}

TABLE 1 Comparison among some typical two-party QPC protocols.

	[14]	[15]	[17]	[18]	[28]	Ours
Quantum state used	EPR pairs	GHZ state	Single photons	Bell states	Five-particle cluster state	Single photons
Quantum measurement	Bell-basis	Single-particle	Single-particle	GHZ-basis	Single-particle	Single-particle
Entanglement swapping	No	No	No	Yes	Yes	No
Unitary operation	Yes	Yes	Yes	No	Yes	Yes
QKD used	No	No	Yes	Yes	No	Yes
Qubit efficiency	25%	33%	25%	50%	40%	[50%, 100%)

is the same as the initial prepared quantum state S if and only if $a_i = 0$ and $b_i = 0$ as well as $a_i = 1$ and $b_i = 1$. Therefore, the probability that the measurement result matches the initial prepared quantum state for a qubit is $\frac{1}{2}$. We denote the measurement result of one qubit being different from the initially prepared quantum state as *Situation I*. For a L-length sequence, the probability of *Situation I* appearing once is $1-(\frac{1}{2})^L$. How many times should TP prepare the initial quantum state to make Situation I appear once? We denote X as the event. Suppose that in *Situation I*, when preparing λ groups of quantum sequences, the distribution of X is denoted as

$$P(X=\lambda) = \left(\left(\frac{1}{2}\right)^L\right)^{\lambda-1}\left(1-\left(\frac{1}{2}\right)^L\right) \tag{37}$$

$E(\lambda)$ can be calculated as

$$\begin{aligned} E(\lambda) &= \sum_{\lambda=1}^{\infty}\lambda P(X=\lambda) = \sum_{\lambda=1}^{\infty}\lambda\left(\left(\frac{1}{2}\right)^L\right)^{\lambda-1}\left(1-\left(\frac{1}{2}\right)^L\right) \\ &= \frac{\left(1-\left(\frac{1}{2}\right)^L\right)}{\left(\frac{1}{2}\right)^L}\lim_{n\longrightarrow\infty}\sum_{\lambda=1}^{n}\lambda\left(\left(\frac{1}{2}\right)^L\right)^{\lambda} = \frac{1}{1-\left(\frac{1}{2}\right)^L} \end{aligned} \tag{38}$$

When L is large, we can obtain $(\frac{1}{2})^L \to 0$ and $E(\lambda) \to 1$. The relationship between $E(\lambda)$ and L can be seen in Figure 5. From Fig.8, it is evident that $E(\lambda) = 2$ when $L = 1$, and $E(\lambda) = 1.001$ when $L = 10$. Meanwhile, as L gradually increases, $E(\lambda)$ approaches 1. Therefore, the value of $E(\lambda) = (1,2]$, and $\eta_e = \frac{1}{E(\lambda)} = [0.5, 1)$.

Table 1 illustrates a comparison between the proposed protocol and previous two-party QPC protocols.

Table 1 reveals that our protocol utilizes single photons as carriers of quantum information, which is more feasible than Bell states and multi-particle states. Although both Ref. [17] and our protocol utilize single photons as quantum resources, the qubit efficiency in Ref. [17] is only 25%, which is lower with our protocol with the qubit efficiency of $[0.5, 1)$. Additionally, our protocol only utilizes unitary operations, which are relatively easier to implement compared to the entanglement swapping technology. The QKD technology does not used in Refs. [14, 15, 28] to share the secret key, but it is performed before the protocol begins and its cost can be ignored. Therefore, our protocol is more practical and efficient compared to the previous protocols [14, 15, 17, 18, 28].

6 Conclusion

A single-photon-based quantum secure protocol for the socialist millionaires' problem is presented in this article. By utilizing single photons as quantum information carriers, encoding the private input through Identity or Hadamard operations, and obtaining the classical outcome via single-particle measurement, the protocol is easier to implement and manipulate compared to other existing protocols. By executing the protocol, TP can ascertain the equality of Alice and Bob's private inputs and subsequently communicates the result to them. Furthermore, the protocol's feasibility is tested through simulation on IBM Quantum Cloud Platform. Security analysis demonstrates that any attempt by eavesdroppers or insider parties to grab the private input of another participant is invalid. Currently, the quantum protocols for the socialist millionaires' problem are primarily designed assuming that all users, including TP, have complete quantum capabilities. In the future, we aim to investigate the development of a quantum protocol that accommodates classical users who can only reflect or measure the received quantum states.

Data availability statement

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

Author contributions

MH: Formal Analysis, Investigation, Methodology, Writing–original draft. YW: Funding acquisition, Writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work is supported by the Gongga Plan for the "Double World-class Project".

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Bennett CH, Brassard G. Quantum cryptography: public key distribution and coin tossing. *Theor Comput Sci* (2014) 560:7–11. doi:10.1016/j.tcs.2014.05.025

2. Hillery M, Bužek V, Berthiaume A. Quantum secret sharing. *Phys Rev A* (1999) 59(3):1829–34. doi:10.1103/PhysRevA.59.1829

3. Li F, Hu H, Zhu S, Yan J, Ding J. A verifiable (k, n)-threshold dynamic quantum secret sharing schemen-threshold dynamic quantum secret sharing scheme. *Quan Inf Process* (2022) 21(7):259. doi:10.1007/s11128-022-03617-3

4. Kuo SY, Tseng KC, Yang CC, Chou YH. Efficient multiparty quantum secret sharing based on a novel structure and single qubits. *EPJ Quan Tech* (2023) 10(1):29. doi:10.1140/epjqt/s40507-023-00186-x

5. Sheng YB, Zhou L, Long GL. One-step quantum secure direct communication. *Sci Bull* (2022) 67(4):367–74. doi:10.1016/j.scib.2021.11.002

6. Zhou L, Sheng YB. One-step device-independent quantum secure direct communication. *Sci China Phys Mech Astron* (2022) 65(5):250311. doi:10.1007/s11433-021-1863-9

7. Huang X, Zhang S, Chang Y, Yang F, Hou M, Chen W. Quantum secure direct communication based on quantum homomorphic encryption. *Mod Phys Lett A* (2021) 36(37):2150263. doi:10.1142/S0217732321502631

8. Yang YG, Gao S, Li D, Zhou YH, Shi WM. Two-party quantum key agreement over a collective noisy channel. *Quan Inf Process* (2019) 18:74–17. doi:10.1007/s11128-019-2187-8

9. Huang X, Zhang SB, Chang Y, Qiu C, Liu DM, Hou M. Quantum key agreement protocol based on quantum search algorithm. *Int J Theor Phys* (2021) 60:838–47. doi:10.1007/s10773-020-04703-x

10. Shor PW. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev* (1999) 41(2):303–32. doi:10.1137/S0036144598347011

11. Yao AC. Protocols for secure computations. In: 23rd annual symposium on foundations of computer science (sfcs 1982); 03-05 November 1982; Chicago, IL, USA. IEEE (1982). p. 160–4. doi:10.1109/SFCS.1982.38

12. Boudot F, Schoenmakers B, Traore J. A fair and efficient solution to the socialist millionaires' problem. *Discrete Appl Maths* (2001) 111(1-2):23–36. doi:10.1016/S0166-218X(00)00342-5

13. Lo HK. Insecurity of quantum secure computations. *Phys Rev A* (1997) 56(2):1154–62. doi:10.1103/PhysRevA.56.1154

14. Yang YG, Wen QY. An efficient two-party quantum private comparison protocol with decoy photons and two-photon entanglement. *J Phys A: Math Theor* (2009) 42(5):055305. doi:10.1088/1751-8113/42/5/055305

15. Chen XB, Xu G, Niu XX, Wen QY, Yang YX. An efficient protocol for the private comparison of equal information based on the triplet entangled state and single-particle measurement. *Opt Commun* (2010) 283(7):1561–5. doi:10.1016/j.optcom.2009.11.085

16. Lin J, Tseng HY, Hwang T. Intercept–resend attacks on Chen et al.'s quantum private comparison protocol and the improvements. *Opt Commun* (2011) 284(9):2412–4. doi:10.1016/j.optcom.2010.12.070

17. Huang W, Wen QY, Liu B, Gao F, Sun Y. Robust and efficient quantum private comparison of equality with collective detection over collective-noise channels. *Sci China Phys Mech Astron* (2013) 56:1670–8. doi:10.1007/s11433-013-5224-0

18. Huang X, Zhang SB, Chang Y, Hou M, Cheng W. Efficient quantum private comparison based on entanglement swapping of bell states. *Int J Theor Phys* (2021) 60:3783–96. doi:10.1007/s10773-021-04915-9

19. Liu W, Wang YB, Cui W. Quantum private comparison protocol based on Bell entangled states. *Commun Theor Phys* (2012) 57(4):583–8. doi:10.1088/0253-6102/57/4/11

20. Lin S, Guo GD, Liu XF. Quantum private comparison of equality with χ-type entangled states. *Int J Theor Phys* (2013) 52(11):4185–94. doi:10.1007/s10773-013-1731-z

21. Ye TY, Ji ZX. Two-party quantum private comparison with five-qubit entangled states. *Int J Theor Phys* (2017) 56:1517–29. doi:10.1007/s10773-017-3291-0

22. Ji ZX, Zhang HG, Fan PR. Two-party quantum private comparison protocol with maximally entangled seven-qubit state. *Mod Phys Lett A* (2019) 34(28):1950229. doi:10.1142/S0217732319502298

23. Ji Z, Zhang H, Wang H. Quantum private comparison protocols with a number of multi-particle entangled states. *IEEE Access* (2019) 7:44613–21. doi:10.1109/ACCESS.2019.2906687

24. Fan P, Rahman AU, Ji Z, Ji X, Hao Z, Zhang H. Two-party quantum private comparison based on eight-qubit entangled state. *Mod Phys Lett A* (2022) 37(05):2250026. doi:10.1142/S0217732322500262

25. Sun Z, Long D. Quantum private comparison protocol based on cluster states. *Int J Theor Phys* (2013) 52:212–8. doi:10.1007/s10773-012-1321-5

26. Zha XW, Yu XY, Cao Y, Wang SK. Quantum private comparison protocol with five-particle cluster states. *Int J Theor Phys* (2018) 57:3874–81. doi:10.1007/s10773-018-3900-6

27. Xu GA, Chen XB, Wei ZH, Li MJ, Yang YX. An efficient protocol for the quantum private comparison of equality with a four-qubit cluster state. *Int J Quan Inf* (2012) 10(04):1250045. doi:10.1142/S0219749912500451

28. Chang Y, Zhang WB, Zhang SB, Wang HC, Yan LL, Han GH, et al. Quantum private comparison of equality based on five-particle cluster state. *Commun Theor Phys* (2016) 66(6):621–8. doi:10.1088/0253-6102/66/6/621

29. Ye TY. Quantum private comparison via cavity QED. *Commun Theor Phys* (2017) 67(2):147. doi:10.1088/0253-6102/67/2/147

30. Chen FL, Zhang H, Chen SG, Cheng WT. Novel two-party quantum private comparison via quantum walks on circle. *Quan Inf Process* (2021) 20(5):178. doi:10.1007/s11128-021-03084-2

31. Huang X, Chang Y, Cheng W, Hou M, Zhang SB. Quantum private comparison of arbitrary single qubit states based on swap test. *Chin Phys B* (2022) 31(4):040303. doi:10.1088/1674-1056/ac4103

32. Jia HY, Wen QY, Song TT, Gao F. Quantum protocol for millionaire problem. *Opt Commun* (2011) 284(1):545–9. doi:10.1016/j.optcom.2010.09.005

33. Yu CH, Guo GD, Lin S. Quantum private comparison with d-level single-particle states. *Physica Scripta* (2013) 88(6):065013. doi:10.1088/0031-8949/88/06/065013

34. Guo FZ, Gao F, Qin SJ, Zhang J, Wen QY. Quantum private comparison protocol based on entanglement swapping of d-level Bell states. *Quan Inf Process* (2013) 12(8):2793–802. doi:10.1007/s11128-013-0536-6

35. Li L, Shi R. A novel and efficient quantum private comparison scheme. *J Korean Phys Soc* (2019) 75:15–21. doi:10.3938/jkps.75.15

36. Ji Z, Fan P, Zhang H, Wang H. Greenberger-Horne-Zeilinger-based quantum private comparison protocol with bit-flipping. *Physica Scripta* (2020) 96(1):015103. doi:10.1088/1402-4896/abc980

37. Wu WQ, Zhao YX. Quantum private comparison of size using d-level Bell states with a semi-honest third party. *Quan Inf Process* (2021) 20(4):155. doi:10.1007/s11128-021-03059-3

38. Gisin N, Fasel S, Kraus B, Zbinden H, Ribordy G. Trojan-horse attacks on quantum-key-distribution systems. *Phys Rev A* (2006) 73(2):022320. doi:10.1103/PhysRevA.73.022320

39. Huang X, Zhang WF, Zhang SB. Efficient multiparty quantum private comparison protocol based on single photons and rotation encryption. *Quan Inf Process* (2023) 22(7):272. doi:10.1007/s11128-023-04027-9

TYPE Original Research
PUBLISHED 07 March 2024
DOI 10.3389/fphy.2024.1357209

OPEN ACCESS

EDITED BY
Amin Ul Haq,
University of Electronic Science and Technology of China, China

REVIEWED BY
G. Palai,
Gandhi Institute For Technological Advancement, India
Riaz Ullah Khan,
University of Electronic Science and Technology of China, China
Jalaluddin Khan,
KL University, India

*CORRESPONDENCE
Ridha Ouni,
rouni@ksu.edu.sa

RECEIVED 17 December 2023
ACCEPTED 06 February 2024
PUBLISHED 07 March 2024

CITATION
Saleem K, Zinou MF, Mohammad F, Ouni R, Elhendi AZ and Almuhtadi J (2024), End-to-end security enabled intelligent remote IoT monitoring system.
Front. Phys. 12:1357209.
doi: 10.3389/fphy.2024.1357209

End-to-end security enabled intelligent remote IoT monitoring system

Kashif Saleem[1], Mohammed Farouk Zinou[1,2], Farah Mohammad[1], Ridha Ouni[2]*, Ahmed Zohier Elhendi[3] and Jalal Almuhtadi[1,4]

[1]Center of Excellence in Information Assurance (CoEIA), King Saud University, Riyadh, Saudi Arabia, [2]Department of Computer Engineering, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia, [3]Science Technology and Innovation Department, King Saud University, Riyadh, Saudi Arabia, [4]Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia

Introduction: Internet of things (IoT) compose of million of devices connected together over the internet. IoT plays a vital role now a days and especially in future, the most of the monitoring and data collection. The data should be secure while collection and as well in the process of transferring till the destination whether Service Organization Control (SOC) or to cloud for storage. In this paper, a secure IoT based intelligent monitoring system is proposed.

Methods: An intelligent IoT station that interacts via cellular connection to relay data to the cloud is constructed using the Waspmote platform. The algorithm is injected to automatically filter and only keep the new data for transfer to avoid redundancy. The advanced encryption standard (AES) 256-bit method is enabled for onboard data encryption and then the generated cipher text is transmitted. The encrypted data is then stored over the cloud to ensure privacy. Moreover, the mobile application (mApp) is developed to be installed on handheld devices for calling the secure data from the cloud, decrypting it, and displaying it as per user input, whether real-time or historical.

Results and Discussion: The encryption algortihm helps in securing the proposed monitoring system from brute force, man in the middle, phishing, spoofing, and denial of service (DoS) attacks. The results of the real testbed experimentation demonstrate the complexity evaluation and reliability of IoT monitoring systems with end-to-end data security in terms of encryption algorithm delay and data rate, respectively.

KEYWORDS

cloud, eHealthcare, intelligent, IoT, privacy, security, smart cities, smart grid

1 Introduction

The Internet of Things (IoT), which is widely used in daily life, is built by any device that has been enabled with internet protocol (IP) and can establish communication through the Internet. The cellular networks further boost this IoT technology that eventually enables the smart city concept [1]. The IoT is predicted to grow at an exponential rate by 2023, with approximately 30 billion linked devices. This equates to more than three gadgets for every person on Earth, and given recent events that have resulted in a more dramatic surge in online engagement, this number is likely to be substantially higher [2].The IoT has become more important in mobility, monitoring technologies, and modern data communication.

Everyday things may now connect to the Internet and communicate with each other thanks to IoT. IoT allows for the interoperability of a large number of applications and devices [3,4]. Security, throughput, large-scale connection, and ultra-reliability are all new criteria to evaluate the performance of IoT. Cellular networks play a critical role in enabling IoT by connecting and supporting communication of a large number of things via the Internet [5,6].

Several studies have investigated issues including scalability, availability, mobility, reliability, and flexibility [7]. Mobility management, interoperability of hybrid networks, and large network volumes are all design considerations for Internet Protocol version 6 (IPv6). In IoT devices, for real-time routing with IPv6, the Routing Protocols for Low-Power and Lossy Networks (RPL) were introduced to communicate over the Internet directly, without packet translation. However, because of the limited IoT node resources, an algorithm with high complexity is not a good option [8] and therefore requires some intelligent mechanisms.

Furthermore, the massive mobile interconnectivity, the IP-based open architecture, the cloud, the dynamic heterogeneity device kinds, and the diversity of the underlying access network technologies that employ information sharing and data pre-processing, together raise security concerns [5,9,10]. The immense use of these systems at Gb/s causes a lot of problems. Because of the gigantic complexity of large-scale security, data privacy and IoT deployment are the most important concerns, particularly for vital applications [11,12].

In order to provide remote monitoring with end-to-end security and user privacy, this article provides a useful IoT station-based monitoring system that is equipped with data encryption. Starting from reviewing the related literature and analyzing the issues to decide and choose the best possible options such as routing protocol, cloud setup, and encryption algorithm according to the components' limitations. Furthermore, various circumstances have distinct libraries and technical information accessible. As a result, there are several stages involved in the reconfigurations: Waspmote scripting, server configuration handling HTML pages, data transmission over UDP, and, last but not least, mApp development. The Waspmote IoT device consists of a smart city board equipped with temperature, humidity, Carbone Monoxide – CO, and Carbon dioxide – CO2 sensors. The miniaturized IoT station encrypts every communication before sending it to the cloud and gathering sensor readings in real time. The encrypted data is kept on the cloud and accessed on the portable device using an iOS native app. The main contribution of this paper includes:

- Propose a secure intelligent IoT monitoring system by integrating multiple devices and flashing them with developed program.
- Enable system with most simplest end-to-end security architecture starting from the IoT device to the user side.
- Develop a mobile application to call, decrypt, and show the real time and/or historical data.
- Conduct real-life scenario experiments to check the complete system reliability.

The proposed system could help in many different scenarios, for example, it can be linked with smart home accessories to control the air conditioner temperature to reduce energy consumption and assist the smart grid. Knowing the air quality in the user house is another benefit of the system. Moreover, it could be installed in different locations across the city to measure air quality which enables smart cities to help in multiple health situations such as people with respiratory disease. The rest of the paper is organized as follows. Section 2 presents a background of IoT applications and reviews some related work. Section 3 describes the methodology of designing a secure intelligent IoT monitoring system. The implementation and results are presented in Section 4. Finally, Section 5 concludes this paper with future work.

2 Literature review

As our world becomes more linked, new IoT applications are appearing in every industry, whether large or small. 5G also plays a crucial role in enabling new IoT capabilities [6]. We have a unique vantage point as a major provider of storage solutions from endpoints to edge devices through the core because storage is a critical component in enabling various IoT use cases. We'll go over some of the current IoT application cases in this article as follows [13].

Smart Cities: Parking, Transportation, Energy [7], and More: Creating smarter, more efficient cities is one of the most exciting IoT use cases [10]. Healthcare: Whether it is a mobile device gathering patient information at an emergency room, or an on-body monitoring system for continuous glucose, IoT devices at the edge are transforming patients' healthcare experiences. Autonomous and Connected Vehicles: Vehicles will eventually achieve Level 5 autonomy and drive autonomously without the need for human intervention. To achieve this, almost 1 terabyte of data will be stored onboard soon, with the number rising to 2+ terabytes in the next decade. Smart Agriculture: Farmers today are harnessing the potential of the Internet of Things to streamline their operations. Connected technology can track animals while they graze in open pastures as the usage of free-range livestock grows more widespread. Smart sensors can also be used in irrigation systems to save water by ensuring the proper moisture level in the soil for a certain crop [14]. AR/VR: Augmented reality (AR) and virtual reality (VR) are gaining popularity in a variety of industries, including entertainment, commerce, gaming, and medical procedures, to provide "extended reality" experiences. Wearables, Fitness Trackers, and Smart Watches: The wearables business is booming, thanks to an infusion of new personal devices.

Indeed, according to a recent analysis, the sector is predicted to reach 520.1 million units by 2025, up from 181.5 million units in 2019, representing a 19.9% CAGR over the projection period (2020–2025). Companion Robots: Companion robots are an IoT use case that has emerged in tandem with the 2020 pandemic. Proposed secure intelligent IoT monitoring system. Many of the researchers used machine learning and deep learning algorithms on the server or the where the computing is perform to make the system optimized. For example, in [15] presents a new methodology that uses state-of-the-art predictive models like Artificial Neural Network and Blockchain-based traceable mechanisms to prevent the spread of new variants of COVID-19 based on a Blockchain-based traceable model that tracks and traces. Another [16] outlines a novel approach to statistically analyze the current state of affairs and predict COVID-19 breakouts in the future. The technique analyzes the present state of affairs in nations all around the world using weekly mobility data. To create a prediction framework, the approach is assessed using a multi-layer perceptron neural network (MLPNN), a deep learning model. Cronbach's alpha, the Case Fatality Ratio (CFR), and other measures were calculated to assess the forecasting's success.

Several approaches have been discussed in the literature for efficient monitoring. The devices have a linkage with other devices in the era of the internet of things (IoT). The prediction and monitoring systems have become more efficient and integrated with different applications to notify people robustly in real time. The connectivity with different APIs and applications requires data security, data integrity, and end-to-end privacy. The presented approaches lack data security and end-to-end security. Several approaches and techniques have been presented in this section highlighting monitoring systems [17].

In [18] an IoT-based smart environment analyzing system was presented to map the humidity, CO2 level, and temperature intensity. The data in the presented system is transmitted from the sender nodes to the receiver nodes and organized in the database. Android application and LabVIEW were utilized to monitor and share the weather information efficiently but the information sharing and data transmission did not follow the end-to-end secure channel for communication. The sleep time of the microcontroller and the power consumption of the sensors are the limitations of the proposed monitoring system.

An efficient API [19] is designed to secure the communication between applications hosted on the cloud and smart sensors using middleware secure communication. The presented approach secures communication using end-to-end security protocols by overcoming the challenges of devices like network attenuation, computation power, network buffer, and energy. Optimal scheme decider and session resumption algorithm utilized to secure the communication between devices and sensors. The session resumption algorithm resumes the device connection, disconnected due to network glitches. Supervised machine learning was employed to connect with the secure network using the optimal scheme decider method. The results of the presented technique show that secure communication can be performed robustly using the certificate and pre-shared keys. The main security concern of the proposed model is replaying checksum and collision attacks.

An advanced [20] IoT based-environment monitoring model is presented for real-time weather prediction as well as wind speed, humidity, and UV index measurement. Several sensors are utilized to collect data from the environment and data is transmitted to web pages to plot the real time graphs of weather changes. The monitoring system also comprises a monitoring app that is utilized to send alerts to people about sudden weather changes through a notification. The data collected from sensors using a web page for statistical graph plotting can be accessed anywhere using an API. The API is utilized to analyze and predict accurate predictions from past data as well as real-time weather analysis. The collected data from sensors for weather prediction can be utilized for future prediction due to compact data size. The present monitoring system did not comply with end-to-end data privacy and security which is a drawback of the implemented system.

A wireless sensor network (WSN) [21] was utilized to design an IoT-based weather monitoring system. The main goal of the presented approach is to monitor the weather in remote areas and provide access to the data collected in these areas through the Internet. The presented model contains two types of nodes. One is for information extractions with access through webpages anywhere on the internet and the other sends alerts about the harshness of weather to people when the parameter of data collected for weather prediction exceeds the threshold limit. The proposed system collects data and provides access over the internet but does not deal with the data security required for data access and data transmission. IoT systems normally are based on wireless network sensors (WSN). WSNs are utilized for the weather monitoring system to measure and predict the weather conditions and data is transmitted over the network.

The article [22] stress the significance of patient data security and privacy in IoT-enabled healthcare systems. The authors provide an IoT sensor-enabled medical healthcare (SMSH) system's safe surveillance technique. The suggested method entails keyframe picture encryption and intelligently recorded video summaries onto the server. First, a well-organized keyframe extraction process is triggered by the visual sensor to extract relevant picture frames, after which an alarm is delivered to the appropriate authority within the healthcare system. Second, the ultimate determination of what transpired with the keyframes that were taken is kept secure from any hackers. In [23] the authors conduct further research by using cosine-transform encryption to make it more secure from any adversary.

A novel WSN-based weather monitoring is introduced in [24]. The implemented system collects data from the environment of different conditions like air, humidity, heat and cold and stores all the data on the cloud-based storage system. The stored data can be provided to a single client or multiple users on their smartphones when people want to know the weather conditions. The presented WSN-based system has been implemented in real time to evaluate the adaptability, sustainability, and scalability of the weather monitoring architecture. The limitation of the proposed model includes power availability in remote areas which can be improved using solar panels in the wild.

The authors in [25] offers a process for IIoT ecosystem validating efficacy at every step. The technique is built on perceptively extracted keyframes, gorgeous monitoring, and lightweight cosine functions processed by hybrid approach chaotic map keyframe image encryption. Compared to the earlier keyframes image encryption approach, the generated result has a shorter execution time, is more resilient, and can be implemented at a reasonable cost while maintaining security [23].

In [26], the authors introduce a class of smart electronic gadget for automotive applications that manages an anti-theft IoTs security system. The gadget contributes significantly to the development of the smart city framework and is intended to reduce the workload for security officers. The way the gadget operates is to prevent unauthorized individuals from accessing the car until an authorized password is provided or an SMS is delivered to the car. The gadget may also be used to remotely operate the vehicle's engine-stopping system. It has been demonstrated that the produced prototype is both affordable and appropriate for real-world use. An effective IoT alert

TABLE 1 Realted work comparison.

References	Hardware						Sensors			
	Mote	Connectivity	Algorithm	Cloud	Encryption	App	Temp	Hum	CM	CD
[22]	-	-	YOLOv3 MATLAB Sim based	-	CTC-IES 256	-	-	-	-	-
[21]	Arduino Uno	WiFi	Exp	✓	-	✓	✓	✓	-	✓
[16]	-	-	YOLOv3 MATLAB Sim based	-	STC-IES 256	-	-	-	-	-
[25]	-	-	MLPNN - Sim based	-	-	-	-	-	-	-
[24]	Arduino Uno	XBee Pro + 2G Cellular	Exp	✓	-	✓	-	✓	✓	✓
[27]	-	-	ACO Sim based	✓	-	-	-	✓	-	-
Proposed Design	Waspmote	WiFi + 3G Cellular	Exp	✓	AES 256	iOS	✓	✓	✓	✓

where: MLPNN, Multi-layer Perceptron Neural Network; Sim, Simulation; Exp, Real Testbed Experimentation; CTC, Cosine-Transform-based Chaotic Sequence; STC, sine tent cosine; IES, Image Encryption System; ACO, Ant Colony Optimization; Temp, Temperature Sensor; Hum, Humidity Sensor; CM, Carbone Monoxide Sensor; CD, Carbone Dioxide Sensor.

system based on MQTT protocol was introduced that enables compact and quick communication. To sum up, the article presents a novel device that is efficient, cost-effective, and suitable for practical implementation, and it is a significant contribution to the field of automobile security.

In [27] presents a new kind of monitoring system that is based on an ant colony optimization algorithm. The system is designed to monitor the rural environment and provide real-time data to the users. The system is composed of three parts: the sensor network, the data processing unit, and the user interface. The sensor network is responsible for collecting data from the environment, while the data processing unit processes the data and sends it to the user interface. The user interface is responsible for displaying the data to the user in a user-friendly way. The ant colony optimization algorithm is used to optimize the routing of the data from the sensor network to the data processing unit. The algorithm is designed to minimize the energy consumption of the sensor nodes and to maximize the lifetime of the network. The experimental results show that the proposed system is efficient and effective in monitoring the rural environment. The summarized comparison of the related work is presented in Table 1.

3 Methodology

Since IoT is quickly growing, the number of connected devices is increasing and getting more powerful. There are lots of devices that could be used as an IoT development platform, including Arduino Genuino UNO, Raspberry Pi 3, WeIO, BeagleBone, and Nanode. Each one has its advantages and disadvantages. One of these platforms, Waspmote has the most battery life that lasts from 1 to 5 years depending on the application. Moreover, it simplifies the hardware connection because it has an API to deal with different parts of the platform modules rather than interact directly with the pins [28].

Waspmote is a sensor gadget that may be used to create IoT projects. The IoT hardware architecture has been specifically engineered to operate at exceptionally low power consumption. Any of the sensor interfaces, as well as the radio modules, can be turned on and off using digital switches. Waspmote is the lowest consumption IoT platform on the market (7 µA) thanks to three different sleep modes [29] Figure 1 shows the hardware layout for Waspmote.

Another major feature of Waspmote is over the air programming, in recent years, the notion of Wireless Programming, also known as Programming Over the Air (OTAP), has been utilized to reprogram mobile devices such as cell phones. With the new concepts of IoT, M2M, and the Wireless Sensor Networks where networks consist of hundreds or thousands of nodes, OTAP is taken in a new direction, and for the first time it is used with both licenses such as 5G and unlicensed protocols such as WiFi [29,30]. The main benefits of OTAP are:

- Allow for firmware upgrades or changes without requiring physical access;
- The latest firmware is installed by requesting an FTP server, which helps to preserve battery life;
- Upgrade an entire network in a matter of minutes.

In order to implement the project, the following hardware parts are required:

- 1x Waspmote 802.15.4 uFL;
- 1x Waspmote Gas Sensors Board v20;
- 1x Temperature Sensor;
- 1x Humidity Sensor;
- 1x Carbone Monoxide Sensor;
- 1x Carbone Dioxide Sensor;
- 1x 2300 mAh LiPo Battery;
- 1x miniUSB Cable;
- 1x Cellular Module.

The required software are:

- Waspmote IDE (https://development.libelium.com/waspmote-ide-v06/download-ide-windows).

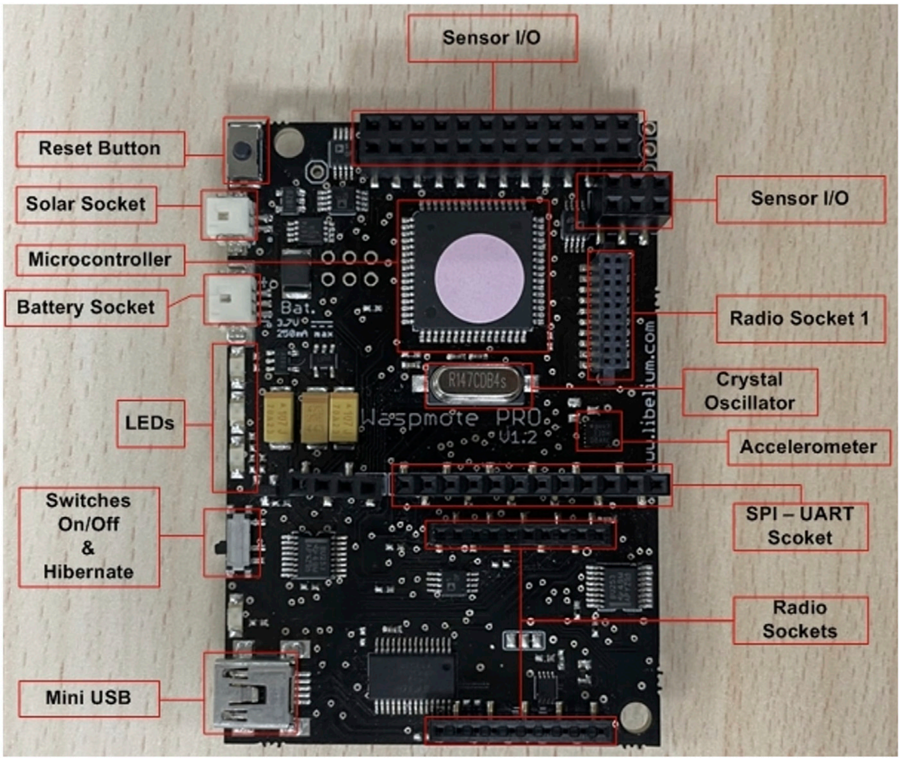


FIGURE 1
Waspmote layout.

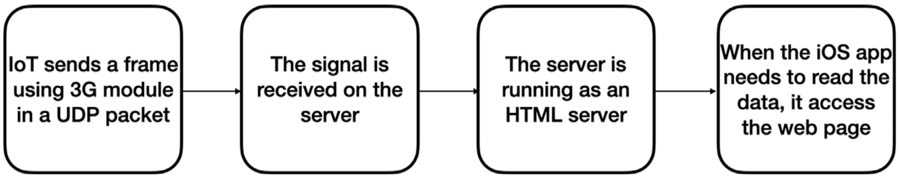

IoT sends a frame using 3G module in a UDP packet
The signal is received on the server
The server is running as an HTML server
When the iOS app needs to read the data, it access the web page

FIGURE 2
Data flow.

- Waspmote API v32 (http://downloads.libelium.com/waspmote-pro-api-v032.zip).

Note that downloading the IDE will come with the latest API version, but it is needed to downgrade into v32 because the Gas Sensors Board v20 requires an API version of 32.

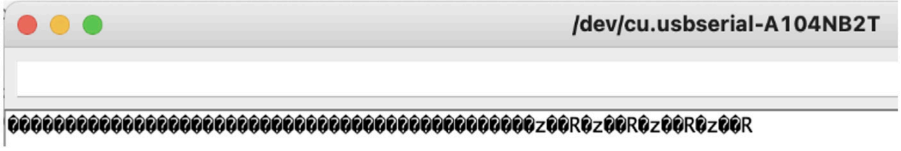
Networking plays a main role in IoT where any IoT device needs to communicate to perform the purpose that was intended to serve.

Several possibilities could be used to connect with the Waspmote IoT platform. We list them below:

- Connect using a WiFi module.
- Connect using a Xbee module.
- Connect using a Cellular module.

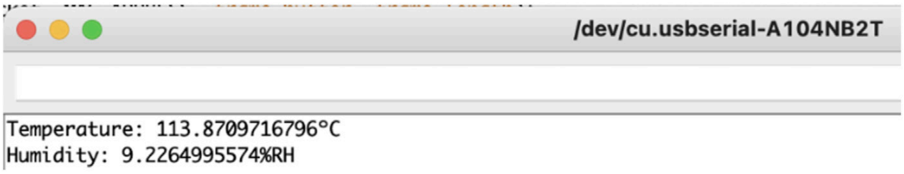
The first two options limit the connection to the local area network (LAN). To connect and transfer the IoT measurements over the internet, the Cellular module is used. This module is enabled with 3G (Third Generation) cellular networks that operates on specific frequency bands. The 3G network primarily uses the UMTS (Universal Mobile Telecommunications System) technology for which the frequency range is 2100 MHz band 12 and the data transfer rates of up to 2 Mbps. In two ways mainly 3G dealt with the high-frequency noise and interference, firstly 3G base stations use smart antennas (such as MIMO) to enhance signal reception and reduce interference. These antennas dynamically adjust their beam patterns to focus on specific users or areas. Secondly, 3G employs robust error correction techniques that detect and correct errors in transmitted data, ensuring reliable communication. In the next step, the signal is generated from the IoT device towards the end user's mobile phone to view it in the developed mobile iOS application.

The flowchart, given in Figure 2, shows how the data is transferred.



```
����������������������������������������������z��R�z��R�z��R�z��R
```

FIGURE 3
Message not recognized by IDE terminal.



```
Temperature: 113.8709716796°C
Humidity: 9.2264995574%RH
```

FIGURE 4
Message recognized by IDE terminal.

Yet, the data is transferred as plain text, and if someone accesses the website or hijacks the forwarded packets using a man-in-a-middle attack, the hacker can easily monitor or even manipulate the actual data or even launch other attacks [11]. Therefore, the end-to-end security is implemented from the onboard frame encryption based on an advanced encryption standard (AES) algorithm with a key length of 256 bits [31] to ensure information authenticity and confidentiality. The onboard AES algorithm encrypts the message m based on electronic codebook (ECB) cipher mode with ZERO padding at Waspmote to generate cipher c as given in (1) [32].

$$c_n = e(k, m_n) \tag{1}$$

Where e represents the encryption process, k is the predefined key, and n is the sequence of the plaintext blocks from the beginning till the end.

The c is transferred to the given destination over the cloud as per configuration for storage in an encrypted form to maintain data security and privacy. The user can call the data from the cloud on the handheld device through the mApp. The data c transfers from the cloud to mApp is still encrypted, until and unless the user inputs the given key for decryption d. The real-time data is called, decrypted with k based on (2) [32], and displayed as soon as the c reaches the server from the IoT station. In addition to the real-time data, the history can be viewed by the user based on the duration entered by the user. The history data also reaches the user side in encrypted form c and is displayed on mApp; after getting d by the mApp when a user unlocks it by k.

$$m_n = d(k, c_n) \tag{2}$$

Furthermore, the Waspmote IoT station is enabled with an intelligent mechanism to check and prevent messages with similar environmental values. The mote periodically senses the environmental values and tally with the previous memorized record. If the values are similar, the mote copies the new values in the memory and erases the old record, locally. Otherwise, the message is generated, encrypted, and forwarded with only the new values. In this way, frequent messaging can be avoided to help minimise massive network traffic and especially traffic overload. Secondly, the reduction in communication maximizes the battery lifetime.

On the security side, the intelligent mechanism makes it very hard for a cryptanalyst to exploit the irregular communication and as well to analyze the similarities due to the newer value in each transmission.

4 Implementation and results

We have started by installing the Waspmote platform IDE (Integrated Development Environment) on our computer. It is needed to connect and deploy code into the hardware itself through a significant capability in supporting Windows, macOS, and even Linux. After making the hardware connections and installing the battery, we run the default program that comes with the hardware. It reports every 5 s the battery level, MAC address, and the temperature of the board. However, the IDE terminal was not able to recognize the message as shown in Figure 3.

The baud rate should be set to 115200 instead of 9600 because this value is used by Waspmote v1.2. Now, the message is decoded and shown correctly in Figure 4. The next step is to install the Waspmote Smart Cities Board to equip the required sensors. Then, we wrote and installed the program which reads both humidity and temperature, and the output is presented in Figure 4.

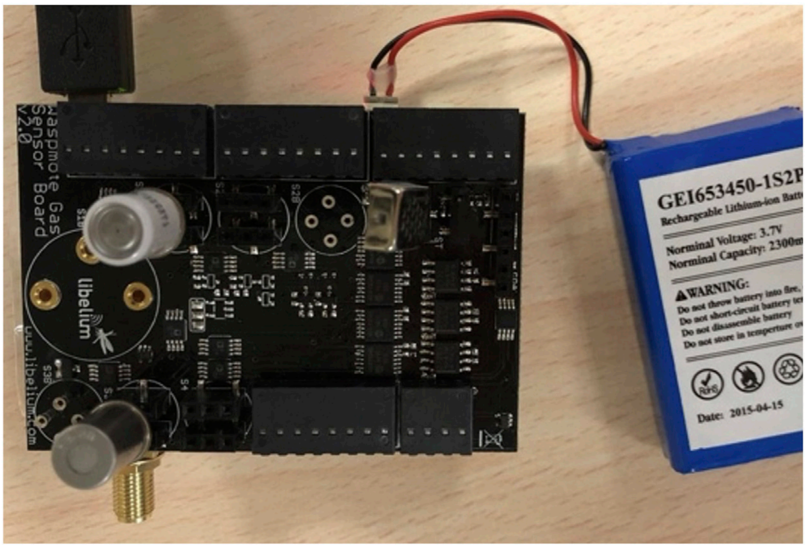
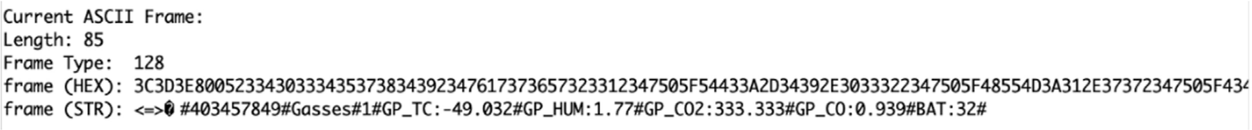

Waspmote Gas Sensor Board v2.0
libelium
www.libelium.com
GEI653450-1S2P
Rechargeable Lithium-ion Batt
Norminal Voltage: 3.7V
Norminal Capacity: 2300m
WARNING:
Do not throw battery into fire,
Do not short-circuit battery
Do not disassemble battery
Do not store in temperture
Date: 2015-04-15

FIGURE 5
Fully equipped smart sensor board for Waspmote.



```
Current ASCII Frame:
Length: 85
Frame Type:  128
frame (HEX): 3C3D3E800523343033343537383439234761737365732331234750 5F54433A2D34392E3033322347505F48554D3A312E37372347505F434
frame (STR): <=>� #403457849#Gasses#1#GP_TC:-49.032#GP_HUM:1.77#GP_CO2:333.333#GP_CO:0.939#BAT:32#
```

FIGURE 6
The code output.

The next step consisted of deploying the rest of the sensors, which are CO2 (Carbon Dioxide) and CO (Carbon Monoxide) sensors. Figure 5 shows the fully equipped Smart Cities Board.

However, when working on the code that reads all the sensor readings from the board, some major errors appeared and the code was not compiling at all. After investigation, we detected that the signal names were turned out (renamed) in the Interrupt Vector Table. Unfortunately, the given solution did not solve the issue and introduced a new range of errors! After multiple searches, the problem was identified. The Smart Sensors City Board requires an API of version 32, as mentioned in Section 3, which leads to a downgrade of the API. Finally, the problem is resolved and can compile the code and read the sensor values as shown in Figure 6.

4.1 Networking

The Cellular module configuration is set up in the first step including the APN, username, password, and PIN for the sim. Then, the Waspmote can send user datagram protocol (UDP) and/or transmission control protocol (TCP) packets. Due to the traffic overhead issues and other reasons as listed below the UDP is enabled.

- UDP packet is much simpler than TCP.
- UDP is a connectionless protocol, so we do not need to deal with establishing a connection process.
- According to the selected application, the packet loss does not matter, but the massive traffic overhead can reduce the lifetime of the device.

In the second step, cloud service is availed with the information as given in Figure 7.

After running the Ubuntu Linux distribution, the server is connected by using a secure shell protocol (SSH), and then a UDP connection is configured. The firewall settings are configured in the system according to the traffic requirements. The UDP on port 6000 is opened by entering the following command:

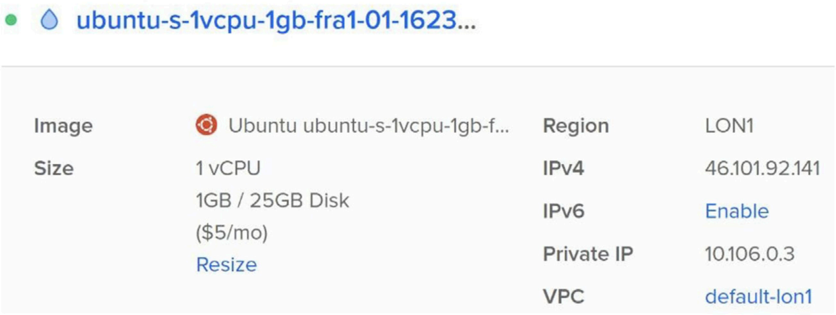
```
sudo ufw allow 6000/udp.
```

The firewall is then restarted to update the settings:

```
sudo ufw disable.
sudo ufw enable.
```

The UDP part is completed at this point. The third step in the experimentation consists of running the server as an HTML server, which is achieved through an Apache server using the following commands:

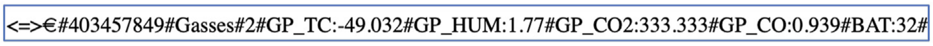
```
sudo apt install apache2.
```



ubuntu-s-1vcpu-1gb-fra1-01-1623...

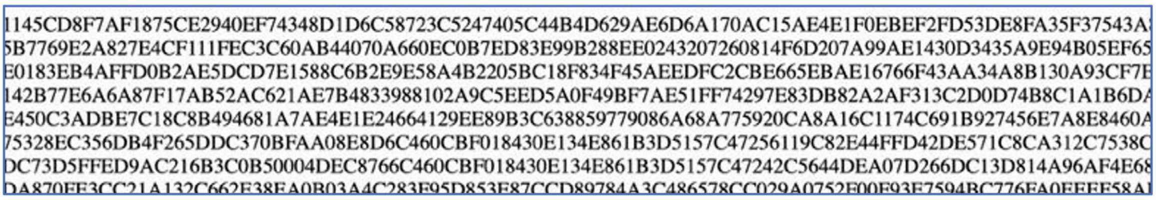
Image	Ubuntu ubuntu-s-1vcpu-1gb-f...	Region	LON1
Size	1 vCPU 1GB / 25GB Disk ($5/mo) Resize	IPv4	46.101.92.141
		IPv6	Enable
		Private IP	10.106.0.3
		VPC	default-lon1

FIGURE 7
Cloud Configuration.



```
<=>€#403457849#Gasses#2#GP_TC:-49.032#GP_HUM:1.77#GP_CO2:333.333#GP_CO:0.939#BAT:32#
```

FIGURE 8
Single record data stored on the cloud and displayed on the HTML page.



```
1145CD8F7AF1875CE2940EF74348D1D6C58723C5247405C44B4D629AE6D6A170AC15AE4E1F0EBEF2FD53DE8FA35F37543A
5B7769E2A827E4CF111FEC3C60AB44070A660EC0B7ED83E99B288EE0243207260814F6D207A99AE1430D3435A9E94B05EF65
E0183EB4AFFD0B2AE5DCD7E1588C6B2E9E58A4B2205BC18F834F45AEEDFC2CBE665EBAE16766F43AA34A8B130A93CF7E
142B77E6A6A87F17AB52AC621AE7B4833988102A9C5EED5A0F49BF7AE51FF74297E83DB82A2AF313C2D0D74B8C1A1B6DA
E450C3ADBE7C18C8B494681A7AE4E1E24664129EE89B3C638859779086A68A775920CA8A16C1174C691B927456E7A8E8460A
75328EC356DB4F265DDC370BFAA08E8D6C460CBF018430E134E861B3D5157C47256119C82E44FFD42DE571C8CA312C7538C
DC73D5FFED9AC216B3C0B50004DEC8766C460CBF018430E134E861B3D5157C47242C5644DEA07D266DC13D814A96AF4E68
DA870EE3CC21A132C662E38EA0B03A4C283E95D853E87CCD89784A3C486578CC029A0752E00E93E7594BC776EA0EEEE58A
```

FIGURE 9
The ciphertext transferred and stored on the cloud.

systemctl enable apache2.

systemctl start apache2.

The server is now configured to run as an HTML web server, that can be accessed directly by entering the server IP in the web browser. One last step to enable the server to receive UDP packets and write their content directly to the HTML page is enabled by entering these commands.

First, the directory is changed to the path where the Apache server is running:

cd/var/www/html

Second, UDP packets are captured by using NetCat tool:

nc -u -l 6000 -k > index.html

where

nc: the command to use NetCat tool.

-u: use UDP.

-l: listening mode.

6000: is the port number to allow the UDP traffic early in the firewall.

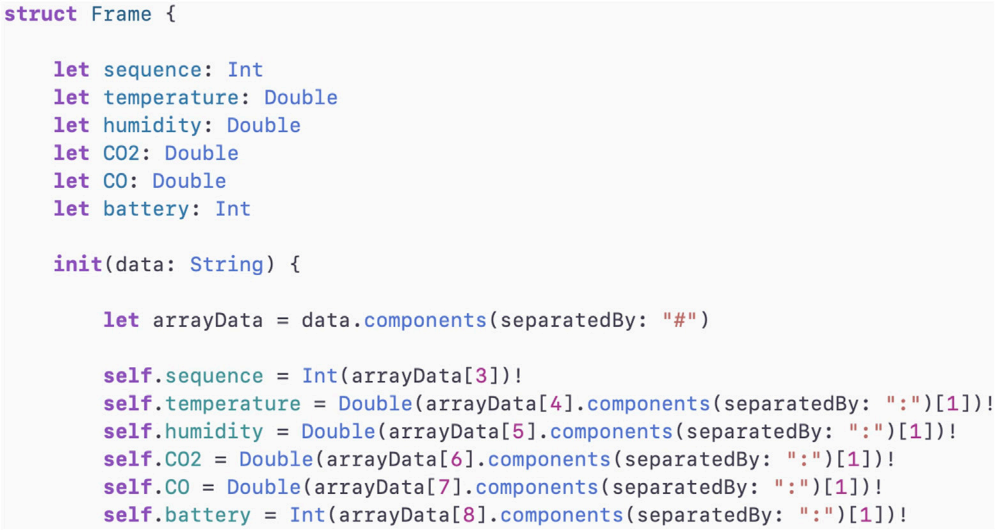
-k: keep the connection alive. This command is very important to keep the tool continuously listening. Otherwise, the tool will stop listening after receiving the first packet and needs to enter the command manually multiple times.

> index.html: write the received content into the webpage defined by the directory that we are running the Apache web server.

Figure 8 shows the received data from the IoT device and is displayed on the HTML page.

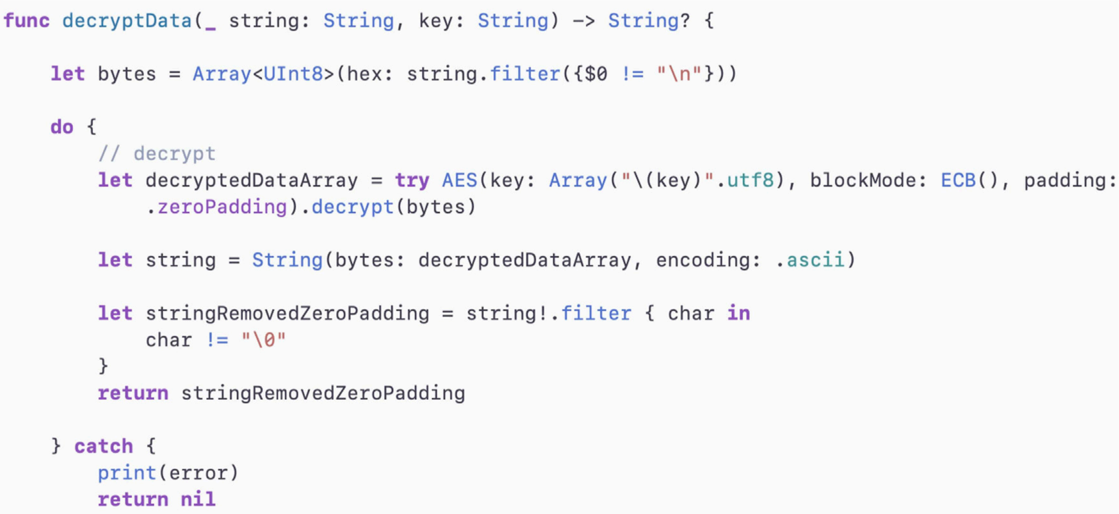
4.2 Encryption

In order to protect the data, encryption has been enabled in the intelligent monitoring system before sending the data to ensure secure transfer and storage. This is one of the reasons for choosing the Waspmote platform as it supports encrypting frames before sending. Therefore, the AES 256 encryption algorithm is included for the secure transfer as shown in the generated ciphertext in Figure 9.



```
struct Frame {

    let sequence: Int
    let temperature: Double
    let humidity: Double
    let CO2: Double
    let CO: Double
    let battery: Int

    init(data: String) {

        let arrayData = data.components(separatedBy: "#")

        self.sequence = Int(arrayData[3])!
        self.temperature = Double(arrayData[4].components(separatedBy: ":")[1])!
        self.humidity = Double(arrayData[5].components(separatedBy: ":")[1])!
        self.CO2 = Double(arrayData[6].components(separatedBy: ":")[1])!
        self.CO = Double(arrayData[7].components(separatedBy: ":")[1])!
        self.battery = Int(arrayData[8].components(separatedBy: ":")[1])!
```

FIGURE 10
Frame structure.



```
func decryptData(_ string: String, key: String) -> String? {

    let bytes = Array<UInt8>(hex: string.filter({$0 != "\n"}))

    do {
        // decrypt
        let decryptedDataArray = try AES(key: Array("\(key)".utf8), blockMode: ECB(), padding:
            .zeroPadding).decrypt(bytes)

        let string = String(bytes: decryptedDataArray, encoding: .ascii)

        let stringRemovedZeroPadding = string!.filter { char in
            char != "\0"
        }
        return stringRemovedZeroPadding

    } catch {
        print(error)
        return nil
```

FIGURE 11
Decryption function.

4.3 Mobile application and data decryption

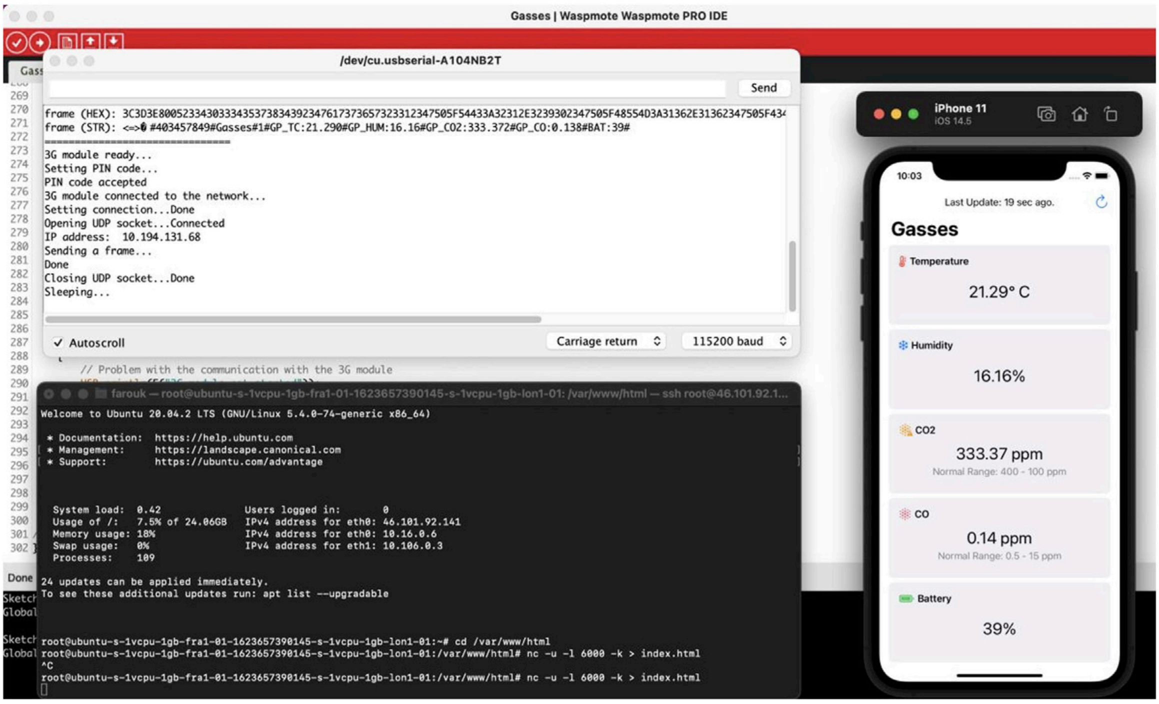
The native iOS application is built to enable the user to discover the environmental readings globally in real-time, which were sent from the IoT device over the internet.

Every message "send", as shown in Figure 8 from the IoT device, begins with <=> using # separator. This information is utilized to create an object called a frame to map the data in the app. Figure 10 shows the frame structure declaration and initialization.

In the initialization process, each frame is mapped to the corresponding value that it represents. For example, the 3rd element in the original frame represents the frame sequence number, the 4th element represents the temperature sensor value, and the 5th element represents the sensor humidity value.

The encrypted data needs to be decrypted for further analysis or viewing. The decryption is performed by using the function inside the app as shown in Figure 11.

Furthermore, the rest of the app code uses the user interface (UI) and manages different states of the app, including data encryption, decryption, or in case the key is wrong. The screenshot of the sensed data output along with the mobile application coding is shown in Figure 12. In addition, a video recording of the app in action can be accessed via https://youtu.be/Vt5Jz_cZ_xk.


Gasses | Waspmote Waspmote PRO IDE
/dev/cu.usbserial-A104NB2T
Send
iPhone 11
iOS 14.5

```
frame (HEX): 3C3D3E8005233430333435373834392347617373657323312347505F54433A32312E3239302347505F48554D3A31362E31362347505F434
frame (STR): <=>� #403457849#Gasses#1#GP_TC:21.290#GP_HUM:16.16#GP_CO2:333.372#GP_CO:0.138#BAT:39#
================================
3G module ready...
Setting PIN code...
PIN code accepted
3G module connected to the network...
Setting connection...Done
Opening UDP socket...Connected
IP address:  10.194.131.68
Sending a frame...
Done
Closing UDP socket...Done
Sleeping...
```

✓ Autoscroll   Carriage return   115200 baud

```
    // Problem with the communication with the 3G module
```

farouk — root@ubuntu-s-1vcpu-1gb-fra1-01-1623657390145-s-1vcpu-1gb-lon1-01: /var/www/html — ssh root@46.101.92.1...

```
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-74-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage



  System load:  0.42              Users logged in:       0
  Usage of /:   7.5% of 24.06GB   IPv4 address for eth0: 46.101.92.141
  Memory usage: 18%               IPv4 address for eth0: 10.16.0.6
  Swap usage:   0%                IPv4 address for eth1: 10.106.0.3
  Processes:    109

24 updates can be applied immediately.
To see these additional updates run: apt list --upgradable


root@ubuntu-s-1vcpu-1gb-fra1-01-1623657390145-s-1vcpu-1gb-lon1-01:~# cd /var/www/html
root@ubuntu-s-1vcpu-1gb-fra1-01-1623657390145-s-1vcpu-1gb-lon1-01:/var/www/html# nc -u -l 6000 -k > index.html
^C
root@ubuntu-s-1vcpu-1gb-fra1-01-1623657390145-s-1vcpu-1gb-lon1-01:/var/www/html# nc -u -l 6000 -k > index.html
```

10:03

Last Update: 19 sec ago.

Gasses

Temperature: 21.29° C

Humidity: 16.16%

CO2: 333.37 ppm
Normal Range: 400 - 100 ppm

CO: 0.14 ppm
Normal Range: 0.5 - 15 ppm

Battery: 39%

FIGURE 12
The sensed data output with mobile application coding.

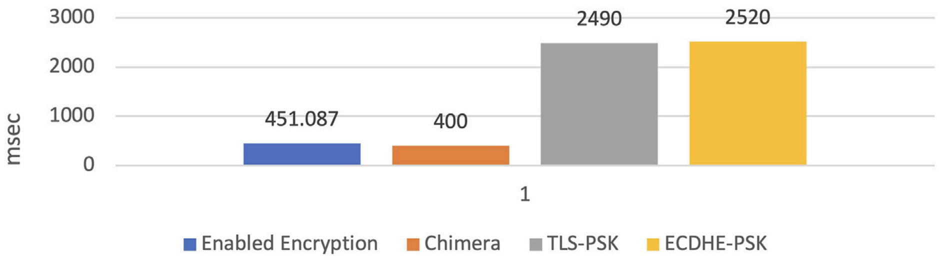
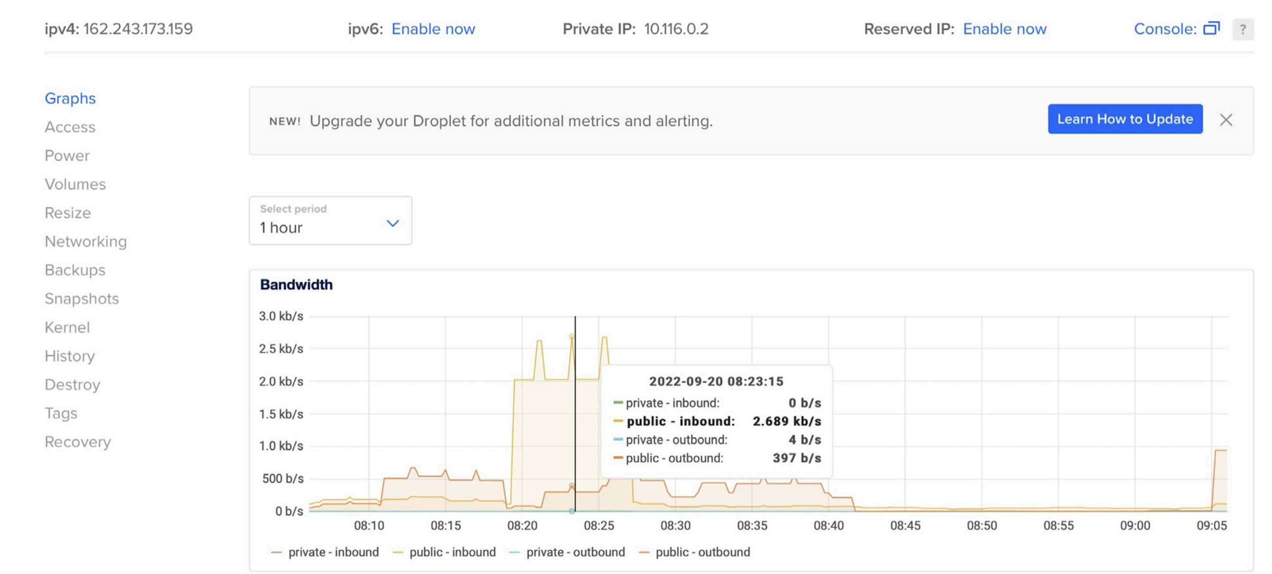


FIGURE 13
Delay comparison.

4.4 Result analysis

The proposed IoT based monitoring system has been experimented with in real time. In the first scenario, both normal data and encrypted data are collected locally through a WLAN connection for about 96 h. Figure 13 shows the delay (in ms) given for various encryption algorithms. The average encryption delay at the node from the collected 617 records is 451.1 msec.

In the second scenario, the IoT Waspmote is configured to send encrypted data over a cellular network and store it on the cloud. The ciphertext is then called on the mobile side by the developed iOS application. The legitimate user can access the data by entering the correct passcode. The process on the backend starts when the application calls the ciphertext and decrypts it based on the passcode entered by the user. In this scenario, the experiment run four times, in first the packet is generated every 5 s and sent as normal open data. In the second, the packet is generated every 50 s and sent as normal data. In the third, the packet is generated every 5 s but the Waspmote encrypts the data and sends it as ciphertext. In the last experimentation, the packet is generated every 50 s and sent as ciphertext. When data is transferred to the cloud, the configured droplet on the configured cloud shows the data rate of transferred ciphertext with 5 milliseconds packet rate shown in Figure 14 and with 50 milliseconds in Figure 15. While observing Figure 14, the high bandwidth of about 2.7 kb/s can be seen, because of the high packet rate generated on the Waspmote side. Figure 15 shows very little bandwidth of about 48 bits per second in front of Figure 14, due to the packets generated by Waspmote with long duration.

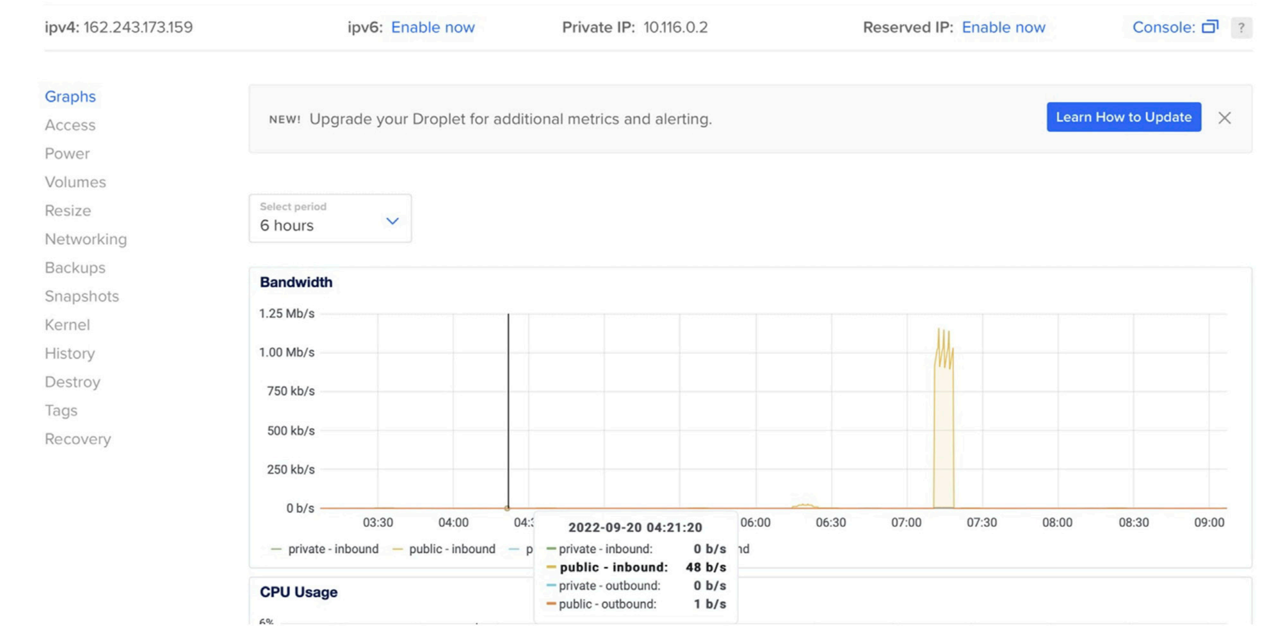

ipv4: 162.243.173.159 ipv6: Enable now Private IP: 10.116.0.2 Reserved IP: Enable now Console:

NEW! Upgrade your Droplet for additional metrics and alerting.

Select period: 1 hour

2022-09-20 08:23:15
- private - inbound: 0 b/s
- **public - inbound: 2.689 kb/s**
- private - outbound: 4 b/s
- public - outbound: 397 b/s

FIGURE 14
Cloud receiving ciphertext with high bandwidth.


ipv4: 162.243.173.159 ipv6: Enable now Private IP: 10.116.0.2 Reserved IP: Enable now Console:

NEW! Upgrade your Droplet for additional metrics and alerting.

Select period: 6 hours

2022-09-20 04:21:20
- private - inbound: 0 b/s
- **public - inbound: 48 b/s**
- private - outbound: 0 b/s
- public - outbound: 1 b/s

FIGURE 15
Cloud receiving Ciphertext with Less Bandwidth.

5 Conclusion

The intelligent monitoring secure IoT station has been proposed, implemented and tested experimentally in this research paper. The related literature is reviewed and according to the most recent IoT requirements, the important privacy and security issues in remote sensing are addressed. The device has been coded and transformed into a complete secure remote monitoring system along with a developed mApp to fetch data from the cloud in a secure manner. The Waspmote based IoT station is intelligent in a manner to eliminate frequent communication and data packet redundancy. End-to-end security is enabled, starting from the onboard data encryption on the IoT station. The testbed of the intelligent IoT station is experimentally tested which shows the performance with the high data rate of up to 200 packets per second that comes up with a bandwidth of 2.7kilobit per second.In future, multiple IoT remote monitoring stations will be programmed and deployed to check the compatibility and performance. The encrytion algorithm will be enhanced with better security because AES 256 is not guaranteed against quantum attacks. Moreover, the system will be enabled with mobile monitoring nodes to enable eagle eye viewing and as well better connectivity between multiple nodes especially in the case of multi hop communication.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

KS: Conceptualization, Formal Analysis, Methodology, Supervision, Visualization, Writing–original draft, Writing–review and editing, Funding acquisition, Project administration, Resources, Software, Validation. MFZ: Conceptualization, Software, Validation, Visualization, Writing–original draft, Data curation. FM: Conceptualization, Visualization, Formal Analysis, Methodology, Supervision, Writing–review and editing. RO: Conceptualization, Formal Analysis, Methodology, Supervision, Visualization, Writing–review and editing, Writing–original draft. AE: Formal Analysis, Investigation, Methodology, Validation, Visualization, Writing–review and editing. JA: Funding acquisition, Investigation, Project administration, Supervision, Visualization, Writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work was supported by the Deputyship for Research & Innovation, “Ministry of Education” in Saudi Arabia for funding this research work through the project number (IFKSUDR_D104).

Acknowledgments

The authors extend their appreciation to the Deputyship for Research & Innovation, “Ministry of Education” in Saudi Arabia for funding this research work through the project number (IFKSUDR_D104).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Zhao Y, Li S, Chen H, Xu Y Application of smart city construction in a new data environment. *Front Energ Res* (2022) 10. doi:10.3389/fenrg.2022.908338

2. Cisco. *Cisco annual internet report (2018–2023) white paper*. San Jose, CA, USA: Cisco (2020).

3. Bayılmış C, Ebleme MA, Küçük K, Sevin A A survey on communication protocols and performance evaluations for internet of things. *Digital Commun Networks* (2022) 8: 1094–104. doi:10.1016/j.dcan.2022.03.013

4. Vilela PH, Rodrigues JJPC, Solic P, Saleem K, Furtado V Performance evaluation of a fog-assisted iot solution for e-health applications. *Future Generation Comp Syst* (2019) 97:379–86. doi:10.1016/j.future.2019.02.055

5. Saleem K, Alabduljabbar GM, Alrowais N, Al-Muhtadi J, Imran M, Rodrigues JJPC Bio-inspired network security for 5g-enabled iot applications. *IEEE Access* (2020) 8: 229152–60. doi:10.1109/ACCESS.2020.3046325

6. Shen Y, He T, Wang Q, Zhang J, Wang Y Secure transmission and intelligent analysis of demand-side data in smart grids: a 5g nb-iot framework. *Front Energ Res* (2022) 10. doi:10.3389/fenrg.2022.892066

7. Gu Q, Qu Q Towards an internet of energy for smart and distributed generation: applications, strategies, and challenges. *J Comput Des Eng* (2022) 9:1789–816. doi:10.1093/jcde/qwac087

8. Saleem K, Chaudhry J, Orgun MA, Al-Muhtadi J A bio-inspired secure ipv6 communication protocol for internet of things. In: 2017 Eleventh International Conference on Sensing Technology (ICST); December, 2017; Sydney, NSW, Australia (2017). p. 1–6. doi:10.1109/ICSensT.2017.8304428

9. Snow S, Happa J, Horrocks N, Glencross M Using design thinking to understand cyber attack surfaces of future smart grids. *Front Energ Res* (2020) 8. doi:10.3389/fenrg.2020.591999

10. IbneÂ Hossain NU, Nagahi M, Jaradat R, Shah C, Buchanan R, Hamilton M Modeling and assessing cyber resilience of smart grid using bayesian network-based approach: a system of systems problem. *J Comput Des Eng* (2020) 7:352–66. doi:10.1093/jcde/qwaa029

11. Stergiou C, Psannis KE, Kim B-G, Gupta B Secure integration of iot and cloud computing. *Future Generation Comp Syst* (2018) 78:964–75. doi:10.1016/j.future.2016.11.031

12. Yaseen M, Saleem K, Orgun MA, Derhab A, Abbas H, Al-Muhtadi J, et al. Secure sensors data acquisition and communication protection in ehealthcare: review on the state of the art. *Telematics Inform* (2018) 35:702–26. doi:10.1016/j.tele.2017.08.005

13. Iarovici Y *Top 10 IoT use cases. Report* (2022).

14. Ma B, Wang Q, Xue B, Hou Z, Jiang Y, Cai W Application of uav remote sensing in monitoring water use efficiency and biomass of cotton plants adjacent to shelterbelt. *Front Plant Sci* (2022) 13:894172. doi:10.3389/fpls.2022.894172

15. Khan RU, Haq AU, Hussain SM, Ullah S, Almakdi S, Kumar R, et al. Analyzing and battling the emerging variants of covid-19 using artificial neural network and blockchain. In: 2021 18th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP); December, 2021; Chengdu, China (2021). p. 101–5. doi:10.1109/ICCWAMTIP53232.2021.9674142

16. Khan RU, Almakdi S, Alshehri M, Kumar R, Ali I, Hussain SM, et al. Probabilistic approach to covid-19 data analysis and forecasting future outbreaks using a multi-layer perceptron neural network. *Diagnostics* (2022) 12:2539. doi:10.3390/diagnostics12102539

17. Mabrouki J, Azrour M, Dhiba D, Farhaoui Y, Hajjaji SE Iot-based data logger for weather monitoring using arduino-based wireless sensor networks with remote graphical application and alerts. *Big Data Mining and Analytics* (2021) 4:25–32. doi:10.26599/BDMA.2020.9020018

18. Shah J, Mishra B Iot enabled environmental monitoring system for smart cities. In: 2016 International Conference on Internet of Things and Applications (IOTA); January, 2016; Pune, India (2016). p. 383–8. doi:10.1109/IOTA.2016.7562757

19. Mukherjee B, Wang S, Lu W, Neupane RL, Dunn D, Ren Y, et al. Flexible iot security middleware for end-to-end cloud–fog communication. *Future Generation Comp Syst* (2018) 87:688–703. doi:10.1016/j.future.2017.12.031

20. Rahut Y, Afreen R, Kamini D, Gnanamalar SS Smart weather monitoring and real time alert system using iot. *Int Res J Eng Tech* (2018) 5:848–54.

21. kumar AS, Murugan S, Elngar AA, Garg L, Kanmani R, Malar ACJ *A novel scheme for an IoT-based weather monitoring system using a wireless sensor network*. Cham: Springer International Publishing (2020). doi:10.1007/978-3-030-38516-3_10

22. Khan J, Li JP, Ahamad B, Parveen S, Ul Haq A, Khan GA, et al. Smsh: secure surveillance mechanism on smart healthcare iot system with probabilistic image encryption. *IEEE Access* (2020) 8:15747–67. doi:10.1109/ACCESS.2020.2966656

23. Khan J, Li JP, Haq AU, Khan GA, Ahmad S, Abdullah Alghamdi A, et al. Efficient secure surveillance on smart healthcare iot system through cosine-transform encryption. *J Intell Fuzzy Syst* (2021) 40:1417–42. doi:10.3233/jifs-201770

24. Ouni R, Saleem K Framework for sustainable wireless sensor network based environmental monitoring. *Sustainability* (2022) 14:8356. doi:10.3390/su14148356

25. Khan J, Khan GA, Li JP, AlAjmi MF, Haq AU, Khan S, et al. Secure smart healthcare monitoring in industrial internet of things (iiot) ecosystem with cosine function hybrid chaotic map encryption. *Scientific Programming* (2022) 2022:1–22. doi:10.1155/2022/8853448

26. Tripathy SK, Mondal SR, Nayak MR, Palai G Experimental studies on electronic smart device for automobiles application. *Opt Quan Elect* (2023) 55:550. doi:10.1007/s11082-023-04789-7

27. Li S, Ye J Design of rural ambient intelligence monitoring system based on ant colony optimization algorithm. In: 2023 International Conference on Ambient Intelligence, Knowledge Informatics and Industrial Electronics (AIKIIE); November, 2023; Ballari, India (2023). p. 1–5. doi:10.1109/AIKIIE60097.2023.10390270

28. Lima J *Arduino vs Raspberry vs libelium vs WeIO vs rascal: what IoT board should you buy? Report*. London, United Kingdom: New Statesman Media Group Ltd (2022).

29. Sl LCD *Waspmote technical guide*. Aragon, Spain: Libelium (2020).

30. Atif M, Muralidharan S, Ko H, Yoo B Wi-esp—a tool for csi-based device-free wi-fi sensing (dfws). *J Comput Des Eng* (2020) 7:644–56. doi:10.1093/jcde/qwaa048

31. Libelium. *Encryption libraries. Report*. Aragon, Spain: Libelium (2017).

32. Stallings W *Cryptography and network security principles and practices*. London, United Kingdom: Pearson (2006).

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 12 March 2024
DOI 10.3389/fphy.2024.1324719

Check for updates

OPEN ACCESS

EDITED BY
Yuanyuan Huang,
Chengdu University of Information Technology, China

REVIEWED BY
Changxin Bai,
Kettering University, United States
Dekun Hu,
Chengdu University, China

*CORRESPONDENCE
Siyu Zhan,
✉ zhansy@uestc.edu.cn

RECEIVED 20 October 2023
ACCEPTED 26 January 2024
PUBLISHED 12 March 2024

CITATION
Zhang W, Cheng H, Zhan S, Luo M, Wang F and Huang Z (2024), Dimensionality reduction and machine learning based model of software cost estimation.
Front. Phys. 12:1324719.
doi: 10.3389/fphy.2024.1324719

Dimensionality reduction and machine learning based model of software cost estimation

Wei Zhang[1], Haixin Cheng[2,3], Siyu Zhan[2,3]*, Ming Luo[4], Feng Wang[1] and Zhan Huang[4]

[1]Research Institute of Natural Gas Gathering and Transmission Engineering Technology, PetroChina Southwest Oil and Gasfield Company, Chengdu, China, [2]Laboratory of Intelligent Collaborative Computing, University of Electronic Science and Technology of China, Chengdu, China, [3]Trusted Cloud Computing and Big Data Key Laboratory of Sichuan Province, Chengdu, China, [4]Capital Construction Department, PetroChina Southwest Oil and Gasfield Company, Chengdu, China

Software Cost Estimation (SCE) is one of the research priorities and challenges in the construction of cyber-physical-social systems (CPSSs). In CPSS, it is urge to process environmental and social information accurately and use it to guide social practice. Thus, in response to the problems of low prediction accuracy, poor robustness, and poor interpretability in SCE, this paper proposes a SCE model based on Autoencoder and Random Forest. First, preprocess the project data, remove outliers, and build regression trees to fill in missing attributes in the data. Second, construct a Autoencoder to reduce the dimensionality of factors that affect software cost. Subsequently, the performance of the model was trained and validated using the XGBoost framework on three datasets: COCOMO81, Albrecht, and Desharnais, and compared with common cost prediction models. The experimental results show that the MMRE, MdMRE, and PRED (0.25) values of the proposed model on the COCOMO81 dataset reached 0.21, 0.16, and 0.71, respectively. Compared with other models, the proposed model achieved significant improvements in accuracy and robustness.

KEYWORDS

software cost estimation, Autoencoder, random forest, COCOMO, dimensionality reduction

1 Introduction

In the era of big data, the new paradigm of computer-based platforms and people-oriented approaches has gradually demonstrated its strong vitality and potential value, triggering a new form of research on complex system modeling, analysis, control, and management, which is known as cyber-physical-social systems (CPSSs). One of the main issues in CPSS research is how to use data as a guide and construct accurate models to regulate social relationships between people, which is also an important issue in software engineering research. As a part of software engineering, Software Cost Estimation (SCE) not only needs to collect and analyze multi-dimensional information about software development needs, but also needs to consider the team's collaborative ability and personnel management costs [1–4]. The predicted results generated through computer algorithms will provide managers with unprecedented efficient management capabilities and improve the team's resource allocation efficiency, building an efficient communication bridge between engineering development and personnel management, thus improving the quality of software development and reducing the risk of research and development failure. Therefore, how to accurately predict the costs of software development has been one of the most important topics studied in software engineering in recent years [5, 6].

However, in practical applications, due to the large number of indicators used for project evaluation and unclear functional requirements in the early stages of development, managers can hardly accurately predict the cost of software development in most cases, resulting in erroneous decisions and unnecessary losses for the company. In addition, with the continuous development of software development technology, object-oriented programming has become the dominant paradigm of software development. Object-oriented design principles such as the single responsibility principle, low coupling, and high cohesion also increase the difficulty of cost estimation [7]. The long development cycle of large-scale software engineering, the significant differences between different projects, and the limited availability of previous project data for cost evaluation hinder the feature learning and data fitting of the constructed model, further limiting the accuracy of the results.

In the process of software development, many cost estimation tasks are completed manually by managers. However, with the expansion of software engineering, the difficulty of implementing this method and the accuracy of the results are unsatisfactory. Therefore, many studies have proposed more automated and intelligent techniques to complete this task. Esteve and Aparicio [8] used the ID3 algorithm to generate a large number of decision trees to classify software modules with high development intensity. In [9], the author studied the application of fuzzy ID3 decision tree. This method is designed by combining the concepts of ID3 algorithm and fuzzy set theory, and uses MMRE and Pred as the criteria for measuring prediction accuracy. The above algorithms all use weak classifiers or regressors to generate prediction models. Although the convergence speed of the models are fast, due to the large differences in mathematical features between different projects, the robustness of the models are poor, making them difficult to obtain reliable prediction results [10, 11]. While deep learning based methods can explore the potential correlations between various attributes better, they require a large amount of data to train the weights of neural networks. Considering the small size of the dataset used in SCE, the model obtained by this method cannot converge well, resulting in low prediction accuracy [12].

To improve the problem of low prediction accuracy and insufficient model robustness in SCE, this paper proposes a SCE model based on Autoencoder and Random Forest. By using neural networks to non-linearly recombine some attributes, various factors that affect software cost are comprehensively reflected from different perspectives, making the new attributes have stronger interpretability and reduce the losses in the final prediction results due to deviations in some attribute values. At the same time, using the Random Forest model to achieve SCE avoids the problem of low model accuracy caused by insufficient data sets, resulting in a model with strong generalization and robustness, which can achieve more reliable prediction results in practical applications.

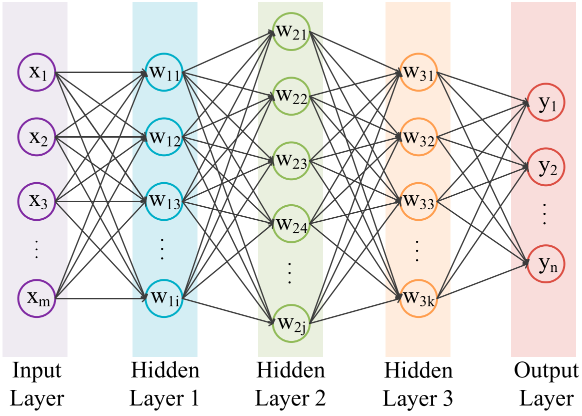


FIGURE 1
Structure of the Autoencoder uesd for dimensionality reduction.

2 Theory and methods

2.1 Dimensionality reduction

Usually, several attributes are used to describe different characteristics of the projects from multiple dimensions in SCE. Richer dimensional information can more comprehensively characterize the cost of a project and improve the accuracy of predictions, but to some extent it also increases the difficulty of data collection, making the prediction results vulnerable to noise [13]. In addition, there may be strong correlations among attributes, resulting in certain attributes affecting the result of prediction together from a single dimension, reducing the robustness and interpretability of the model. To solve this problem, we will use dimensionality reduction method on the original data, reorganizing some variables with complex relationships into a few comprehensive factors, so that the recombined factors can reflect the cost of software development from different perspectives, avoiding the problem of low model accuracy caused by large estimation bias of single attribute.

Principal Component Analysis and Factor Analysis methods have been widely applied in the field of software engineering [14, 15]. However, as linear dimensionality reduction methods, they often fail to achieve good dimensionality reduction effects in scenarios where complex data and high data structure preservation requirements are present. In recent years, with the widespread application of artificial neural networks in the field of data dimensionality reduction [16, 17], Autoencoder, as a nonlinear dimensionality reduction method, can more accurately identify and reorganize data attributes, fully explore the potential correlations between data, and has strong anti-interference ability for noise in data, which is suitable for dimensionality reduction of data in software engineering [18, 19]. The Autoencoder used for dimensionality reduction only contains the encoding part, which consists of an input layer, several hidden layers, and an output layer. Its structure is shown in Figure 1. In the encoder, input data is passed through a series of hidden layers for transformation and mapping to the output layer. Each hidden layer consists of multiple neurons, each of which is connected to neurons in the previous layer and undergoes nonlinear transformation through an activation function. The goal of the encoder is to learn an encoding function that maps input data to a low-dimensional representation in the encoding layer. This encoding process is usually achieved through optimization methods such as backpropagation and gradient descent. By adjusting the network's weights and biases, the encoder gradually learns a set of features that can effectively represent the input data. The training process of Autoencoder usually uses unsupervised learning methods, which only use the input data itself without requiring label information. This allows Autoencoder to be trained on unlabeled data, thereby better adapting to the complex data distribution in software engineering.

2.2 Random forest model

At present, existing prediction models mainly include methods based on the function point method and neural network-based methods. The prediction results of the former are more subjective and have lower prediction accuracy, because in the early stages of a project, there is usually only a user requirement document, lacking a complete software system specification document. Neural network-based evaluation methods require a large amount of sample data to train the neural network, but historical SCE data is often limited, resulting in models that cannot converge to good results. In addition, the poor interpretability of deep learning models is not conducive to evaluating the quality and stability of the model. To achieve the desired prediction accuracy and convergence speed, we used a Random Forest model to implement the task. Considering that historical data is often limited in practical applications, we adopted the XGBoost algorithm framework to build the model to accelerate the model's convergence process.

The XGBoost algorithm [20] uses second-order Taylor expansion to calculate the loss function, adds a regularization term to the GBDT objective function, and uses first and second-order derivatives to approximate the objective function. This approach simplifies the model and effectively reducing the risk of overfitting. The objective function of XGBoost consists of two parts: the loss function and the regularization term:

$$L(\phi) = \sum_{i=1}^{n} l(\hat{y}_i, y_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Where i represents the i th sample in the dataset, n is the total number of samples, and k represents the k th regression tree. $l(\hat{y}_i, y_i)$ represents a traditional differentiable convex loss function, which measures the difference between the true label and the predicted label. $\Omega(f_k)$ is a regularization term that helps smooth learning weights and avoid overfitting the model. Its calculation formula is as follows:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\|^2$$

γ and λ are the regularization parameters, w is the weight vector of the leaf node. When the regularization parameter is set to zero, it becomes a traditional gradient boosting tree.

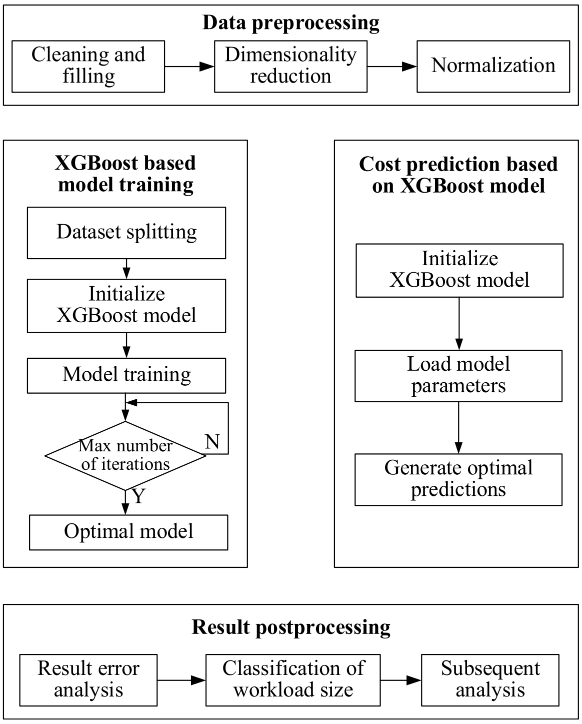


FIGURE 2
Framework of our SCE model.

Since all CART trees are binary trees, the difference between the objective function and the structural score after branching in the algorithm can be measured using the following formula:

$$Gain = \frac{1}{2}\left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}\right] - \gamma$$

γ is a punishment item. G_L and H_L are calculated from the left child node, G_R and H_R are calculated from the right child node. $(G_L + G_R)$ and $(H_L + H_R)$ are calculated through intermediate nodes.

As a tree model, XGBoost simplifies the modeling process while preserving as much original data information as possible. This algorithm performs well in regression tasks, with higher fitting accuracy, robustness, and generalization ability than other traditional machine learning regression algorithms, and is widely used in data prediction tasks. Therefore, in the practical application of SCE, even with fewer training samples, a model with high prediction accuracy can still be obtained from XGBoost algorithm.

3 Model building strategy based on XGBoost

3.1 Framework of the model

Figure 2 shows the framework of the SCE model proposed in this article, including data preprocessing, XGBoost-based prediction model training, cost prediction, and prediction result analysis stages. The specific process is as follows:

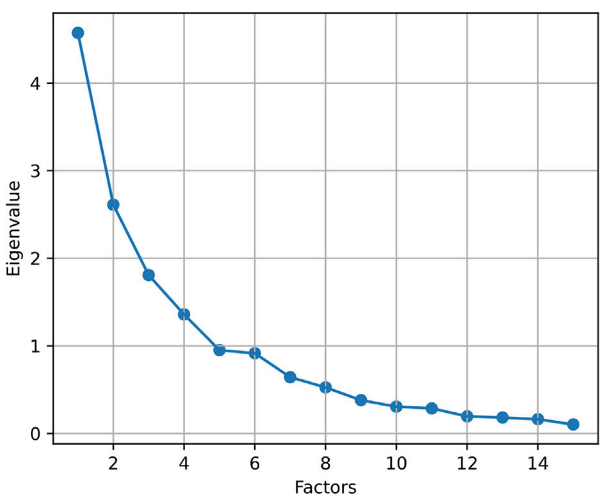


FIGURE 3
Scree plot uesd to determine the number of dimensions.

1) Project data preprocessing stage. Clean the data, remove abnormal values from the data, and fill in missing attributes; The Autoencoder is used to reduce the dimensionality of the cleaned data and eliminate redundant information in the dataset. Finally, normalize the data to unify the dimensions of different features.
2) Model training stage. Use the preprocessed data to train the XGBoost Random Forest model. After training reaches the maximum iteration number, the optimal prediction model is output.
3) Software cost prediction stage. Input the data to be evaluated into the model to generate the predicted cost value.
4) Prediction result analysis stage. Determine whether the error of the prediction result is within an acceptable range, convert the numerical value of the result to different levels, and use the results in subsequent software development evaluations.

3.2 Data preprocessing

In practical applications, some data attributes used for software cost evaluation may have missing values or significant deviations from the true values, which can affect the accuracy of the prediction results. To solve this problem, we use the box plot method to remove outliers from the data to avoid the impact of extreme values on the prediction results. Then for all missing values, we use a linear regression tree model to fill them in. Specifically, we select an attribute with missing values as the dependent variable and other attributes as the independent variables to construct a regression tree model. We use the constructed regression tree model to predict the values of missing values, and repeat this step until all missing values are filled.

Then, the Autoencoder is used to reduce the dimensionality of the cleaned data. By using neural networks to transform high-dimensional data into a new low-dimensional coordinate system, the purpose of eliminating redundant information in the data is achieved. If the reduced-dimensional data has too many factors, it will make the model more susceptible to noise and reduce its robustness. On the other hand, having too few factors will lead to a low expression rate of the data and prevent the effective extraction of potential information from the original data. Therefore, we determined the optimal number of factors based on the scree plot in factor analysis and the practical significance of SCE, and the final reduced-dimensional data contained six dimensions, as shown in Figure 3.

3.3 Model training

After completing data preprocessing, each data sample will consist of six independent variables and one dependent variable, which is the actual cost value. Predicting the cost of software is essentially finding the mapping relationship between independent variables and dependent variables. To enhance the accuracy and robustness of the model, we will construct several linear regression trees and use gbtree as a booster to construct a Random Forest model. First, shuffle the dataset and split it into a training set and a testing set. Then, generate the optimal model using the training dataset. The specific algorithm is shown in Table 1. After generating the model, the test dataset will be used to evaluate the effectiveness of the model. By comparing the gap between the true value and the model's estimated value, it can be determined whether the model has overfitting and whether the prediction error is within an acceptable range.

3.4 Model prediction

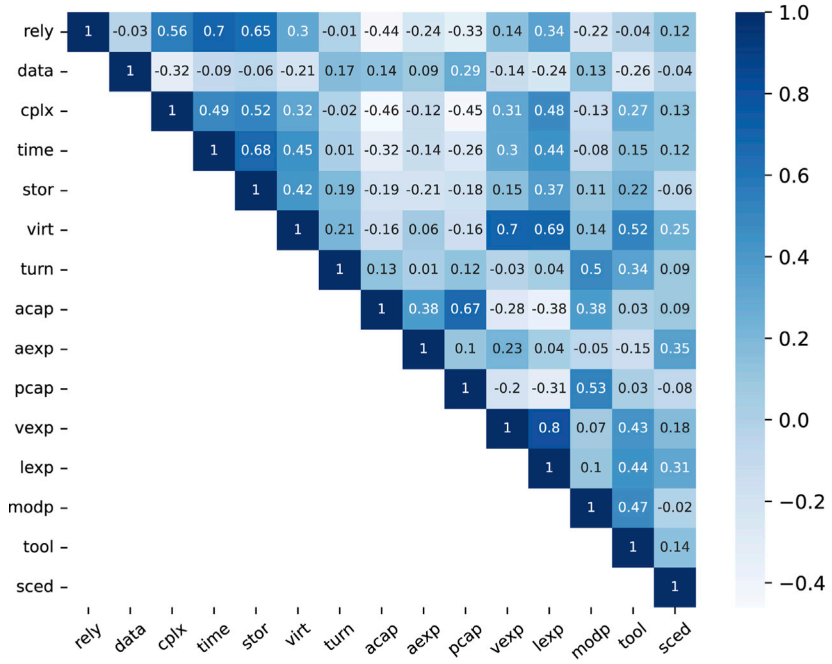
After the training of the Random Forest model based on XGBoost, a mapping relationship between factors that affect software cost and the value of that is established, thus providing the ability to assess future software engineering cost. The relevant attributes of the new software engineering project are passed into the model, and after data cleaning and filling, dimensionality reduction and normalization, a column vector with 6 factors is obtained. By inputting it into the trained Random Forest model, the predicted cost value can be obtained.

3.5 Result postprocessing

In order to use the obtained prediction results to guide the actual software development work, we also need to further analyze and process the results. In practical software development, in order to reduce the risk of project failure caused by excessive deviation in cost estimation, it is necessary to further provide a confidence interval for the prediction results, with a confidence level generally taken as 0.80. If the confidence interval is too large, it indicates that the reliability of the results obtained by using this model to predict is poor, and other methods should be used for prediction. If the confidence interval size is within a reasonable range, it indicates that the model's prediction effect is good. At this point, in order to highlight the practical significance of the prediction effect, the specific numerical value can be converted into five levels (very low, low, moderate, high, very high) to represent different cost extents. Finally, the prediction

TABLE 1 Model training algorithm steps based on XGBoost.

Training algorithm for SCE model based on XGBoost	
Input	Training dataset $\Phi = \{(X_1, y_1), (X_2, y_2), \ldots, (X_m, y_m)\}$. Initialize weights W, bias b, learning rate lr and hyperparameters such as the number of Random Forest trees and the maximum depth of each tree
Output	The Random Forest model with the best prediction accuracy
Dependency	Loss function Loss
while	the preset number of iterations has not been reached **do** Feed the training dataset into the model and generate output values Compare the output value with the actual value and calculate the error E; Calculate the partial derivative of the weight W and bias b for errors respectively Update weight W and bias b by using the following formula: $W \rightarrow W - lr \times \frac{\partial E}{\partial W}, b \rightarrow b - lr \times \frac{\partial E}{\partial b}$
end	
Output model	



	rely	data	cplx	time	stor	virt	turn	acap	aexp	pcap	vexp	lexp	modp	tool	sced
rely	1	-0.03	0.56	0.7	0.65	0.3	-0.01	-0.44	-0.24	-0.33	0.14	0.34	-0.22	-0.04	0.12
data		1	-0.32	-0.09	-0.06	-0.21	0.17	0.14	0.09	0.29	-0.14	-0.24	0.13	-0.26	-0.04
cplx			1	0.49	0.52	0.32	-0.02	-0.46	-0.12	-0.45	0.31	0.48	-0.13	0.27	0.13
time				1	0.68	0.45	0.01	-0.32	-0.14	-0.26	0.3	0.44	-0.08	0.15	0.12
stor					1	0.42	0.19	-0.19	-0.21	-0.18	0.15	0.37	0.11	0.22	-0.06
virt						1	0.21	-0.16	0.06	-0.16	0.7	0.69	0.14	0.52	0.25
turn							1	0.13	0.01	0.12	-0.03	0.04	0.5	0.34	0.09
acap								1	0.38	0.67	-0.28	-0.38	0.38	0.03	0.09
aexp									1	0.1	0.23	0.04	-0.05	-0.15	0.35
pcap										1	-0.2	-0.31	0.53	0.03	-0.08
vexp											1	0.8	0.07	0.43	0.18
lexp												1	0.1	0.44	0.31
modp													1	0.47	-0.02
tool														1	0.14
sced															1

FIGURE 4
Covariance thermogram of various attributes in the COCOMO81 dataset.

results will be handed over to the project managers to guide the subsequent software development work.

4 Results and analysis

4.1 Dataset introduction and preprocessing

To verify the effectiveness of the model on different datasets, we will use three different datasets to test the accuracy of the model, namely, COCOMO81, Albrecht, and Desharnais. The COCOMO81 dataset [21] is one of the most popular datasets for SCE, containing data from 63 projects. Each project contains 17 attributes, 15 of which are independent variables and 2 of which are actual cost sizes. The Albrecht dataset contains data from 24 projects implemented by the IBM DP Services organization. These data include the count of four types of external input/output elements of the entire software application, the number of Source Lines Of Code (SLOC) including annotations, and the number of functional points per project; The Desharnais dataset consists of data from 81 software projects at a Canadian software company. These 81 projects are subdivided into 46 projects

TABLE 2 Performance of the model on different datasets.

Dataset	Maximum depth	Tree number	Learning rate
COCOMO81	8	10	0.1
Albrecht	8	6	0.1
Desharnais	7	5	0.1

TABLE 3 Model evaluation indicators and their meanings.

Evaluation Criteria	Description
$\mathrm{MMRE} = \frac{1}{n}\sum_{i=1}^{n}\mathrm{MRE_i}$	Mean MRE (Mean MRE, MMRE) is one of the most commonly used model prediction criteria
$\mathrm{MdMRE} = \mathrm{Median}\,(\mathrm{MRE})$	Median MRE (Median MRE, MdMRE) is often used in conjunction with MMRE to measure the degree of dispersion of prediction results. MdMRE is not sensitive to outliers and can more accurately reflect the overall distribution of data
$\mathrm{PRED}\,(x) = \frac{1}{n}\times\sum_{i=1}^{n}\begin{cases}1, \mathrm{MRE_i} \le x\\ 0, \mathrm{otherwise}\end{cases}$	Where n denotes the total number of projects and k denotes the number of projects whose MRE is less than or equal to x. normally, x is set to be 0.25

in traditional environments, 25 projects that “improve” traditional environments, and 10 projects in micro-environments based on their technical environments. It is one of the most classic datasets that can be used for SCE.

The preprocessing of the three datasets includes outlier removal, missing value filling, data dimensionality reduction, and normalization. After analysis using the box plot method, 7 attribute values from COCOMO81, 3 attribute values from Albrecht, and 4 attribute values from Desharnais were eliminated. Then, linear regression tree models were built on the three datasets to predict the removed outliers and original missing values. Then, a correlation analysis was conducted on the datasets, as shown in Figure 4. There was a strong data correlation between the attributes of the three datasets, making it suitable for using Autoencoder for data dimensionality reduction. The hidden layer of the Autoencoder contains three fully connected layers. The first layer contains 20 neurons, the second layer contains 30 neurons, and the third layer contains 10 neurons. The activation function after each layer uses ReLu, and the output layer contains 6 neurons, meaning that the output data contains six dimensions. Finally, the dimension-reduced data is normalized using the Sigmoid function.

4.2 Model training

To ensure that the model can fully converge, we divide the dataset into a training set and a testing set, with the training set accounting for 90% and the testing set accounting for 10%. For the training set data, we use the XGBoost distributed gradient boosting framework to train the model. In order to determine the parameters that can generate the best Random Forest, this experiment uses the method of adjusting hyperparameters rather than theoretical analysis. During the hyperparameter tuning process, different parameter value combinations are used to establish Random Forest models. Then, the parameters that generate the best predictive model are considered to be the most appropriate hyperparameters for that model. The hyperparameters obtained for each model are shown in Table 2.

4.3 Evaluation criteria in SCE

This article will use three indicators, MMRE, MdMRE, and PRED to evaluate the model [22]. The calculation methods and their meanings of each indicator are shown in Table 3, all of which are based on the Magnitude of Relative Error (MRE):

$$MRE = \frac{|act - est|}{act}$$

Where *act* represents the actual software cost and *est* represents the software cost predicted by the model.

4.4 Evaluation and discussion

The performance of the model trained by using the Autoencoder and Random Forest methods on the three test sets is shown in Table 4. As can be seen from the results, the difference between the MMRE and MdMRE indicators for different data sets is quite small, indicating that the prediction results are relatively stable, with no individual prediction result showing significant deviation from the true values. Although Albrecht’s training set only contains 21 training samples, the model still has high prediction accuracy on this dataset, which also indicates that the random forest model has a high convergence rate and can obtain accurate prediction results when there is insufficient historical software evaluation data. The performance of the model on the Desharnais dataset is lower than the previous two models, mainly due to the presence of some missing values in the data. The attributes after filling in the data using regression trees still have some differences from the true values, which reduces the accuracy of the model prediction to some extent.

TABLE 4 Performance of the model on different datasets.

Dataset	MMRE	MdMRE	PRED(0.25)
COCOMO81	0.21	0.16	0.71
Albrecht	0.37	0.36	0.33
Desharnais	0.38	0.37	0.22

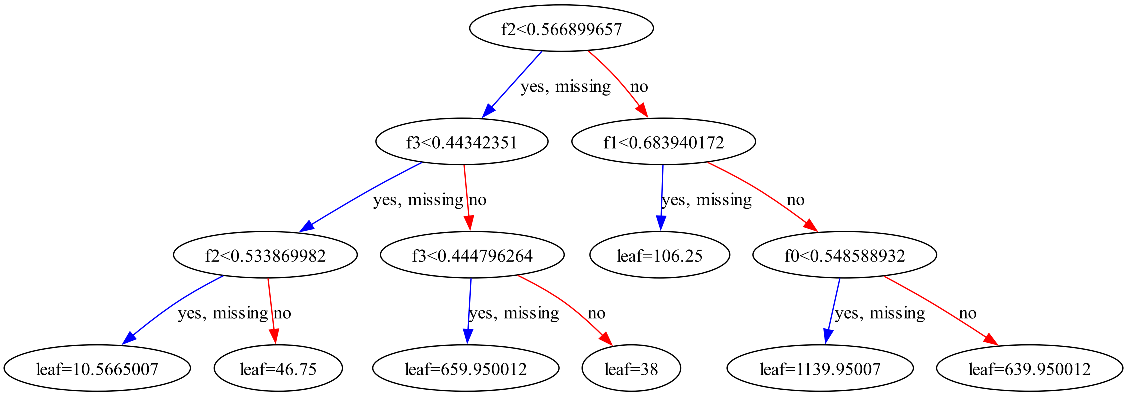


FIGURE 5
Visualization of decision tree prediction process.

TABLE 5 Performance of different models on the COCOMO81 dataset.

Model	MMRE	MdMRE	PRED(0.25)
HACO-BA	3.47	4.47	0.06
PSO-FLANN	0.38	0.33	0.43
Ours	0.21	0.16	0.71

The prediction results on three datasets show that the SCE method using a combination of Autoencoder and Random Forest has strong generalization ability, and can still better fit the results in different engineering projects. The Autoencoder can identify a small number of independent common factors that govern the relationships between multiple attributes, and predict the state of the common factors by establishing a quantitative relationship between the common factors and the original variables. This can help discover some objective regularity between different software engineering projects, and thus abstract a common model for evaluating the size of software costs. At the same time, the Random Forest composed of several regression trees can clearly demonstrate the process of model prediction, as shown in Figure 5, which enhances the interpretability of the model and provides a reliable basis for subsequent management to analyze project costs.

To further compare the performance differences of different SCE models, Table 5 lists the performance of the three models on the COCOMO81 dataset. As can be seen from the table, deep learning-based algorithms such as HACO-BA performed poorly, mainly due to insufficient training data sets resulting in underfitting of the model. Compared to other algorithms, the model proposed in this article has a lower MdMRE value and a higher PRED value, indicating that the model has good consistency in prediction results across different project data, stable model performance, and high prediction accuracy. The cost evaluation in practical software engineering is mainly aimed at reducing development risks and promoting the rational allocation of resources, so the robustness of the prediction model is even more important. Comprehensively evaluated by various indicators, the model proposed in this article based on Autoencoder and Random Forest has better performance.

In subsequent engineering analysis, factor analysis methods can be used to draw radar charts to further analyze the factors that affect the size of software cost, and rational allocation of resources can be used to make up for development shortcomings, thereby accelerating the software development process. We combine the scree plot method and practical significance of SCE to comprehensively determine the optimal number of factors. When the number of factors is 6, the eigenvalue of the matrix reaches the inflection point, and the expression rate of these factors reaches 81%. Therefore, the number of factors after dimensionality reduction is determined to be 6. By looking at the contribution rates of the original attributes to each factor, we named the six factors according to their practical significance. The radar chart of common factors

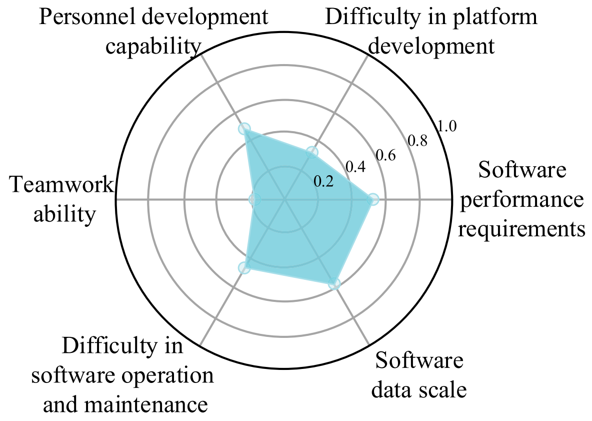


FIGURE 6
Radar chart of common factors for a project.

after dimensionality reduction of a data sample is shown in Figure 6. It can be seen from the figure that the development environment of the project is relatively simple, and the R&D personnel have strong abilities. However, the software performance requirements and data scale are high, and the team's collaboration ability is poor. Team managers should strengthen team communication and collaboration, and focus on algorithm design to reduce the spatial and temporal complexity of software, thereby achieving a multiplier effect.

5 Conclusion

In order to adapt to the issue of comprehensive processing of social information and use it to improve production efficiency in CPSS, this paper proposes a novel SCE model based on Autoencoders and Random Forest, and evaluates its feasibility and performance through theoretical and experimental analysis. This article first introduces the improvement of Autoencoder and Random Forest algorithms on model accuracy and robustness, and analyzes the advantages of these two methods compared to traditional methods and neural network algorithms. Then, the overall framework and algorithm flow of the model are introduced, which are divided into four stages: data preprocessing, model training, cost prediction and result analysis. Finally, the performance of the model on three datasets, COCOMO81, Albrecht, and Desharnais, is introduced, and it is compared with common SCE algorithms to analyze the advantages and disadvantages of different algorithms. Compared with other algorithms, the proposed algorithm model has better accuracy and astringency, and can better complete the cost prediction task in practical software engineering.

At present, there is still much room for improvement in the evaluation models based on Autoencoder and Random Forest, such as low accuracy on datasets with ordinal attributes and significant influence by dataset on model accuracy. Future work should focus more on data processing and data imbalance issues to further improve the performance of the model.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary Material, further inquiries can be directed to the corresponding author.

Author contributions

WZ: Conceptualization, Data curation, Formal Analysis, Investigation, Methodology, Software, Writing–original draft. HC: Funding acquisition, Resources, Supervision, Writing–review and editing. SZ: Data curation, Writing–review and editing. ML: Formal Analysis, Writing–review and editing. FW: Software, Writing–review and editing. ZH: Funding acquisition, Writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This study is supported by Sichuan Science and Technology Program (NO. 2022YFG0176), and Fundamental Research Funds for the Central Universities (No. ZYGX2021YGLH211).

Conflict of interest

Authors WZ, ML, FW and ZH were employed by PetroChina Southwest Oil and Gasfield Company.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fphy.2024.1324719/full#supplementary-material

References

1. Keung J. Software development cost estimation using analogy: a review. In: 2009 Australian Software Engineering Conference; 14-17 April 2009; Gold Cost, Australia (2009). doi:10.1109/ASWEC.2009.32

2. Chirra SMR, Reza H. A survey on software cost estimation techniques. *J Softw Eng Appl* (2019) 12(06):226–48. doi:10.4236/jsea.2019.126014

3. Saleem MA, Ahmad R, Alyas T, Idrees M, Farooq A Systematic literature review of identifying issues in software cost estimation techniques. *Int J Adv Comp Sci Appl* (2019) 10(8):10. doi:10.14569/ijacsa.2019.0100844

4. Jorgensen M, Shepperd M. A systematic review of software development cost estimation studies. *IEEE Trans Softw Eng* (2007) 33(1):33–53. doi:10.1109/TSE.2007.256943

5. Latif AM, Khan KM, Duc AN. *Software cost estimation and capability maturity model in context of global software engineering*. IGI Global (2022). 910–928. doi:10.4018/978-1-6684-3702-5.ch045

6. Martinez-Fernandez S, Bogner J, Franch X, Oriol M, Siebert J, Trendowicz A, et al. Software engineering for ai-based systems: a survey. *ACM Trans Softw Eng Methodol* (2022) 31(2):1–59. doi:10.1145/3487043

7. Tomasevic N, Gvozdenovic N, Vranes S. An overview and comparison of supervised data mining techniques for student exam performance prediction. *Comput Edu* (2020) 143:103676. doi:10.1016/j.compedu.2019.103676

8. Esteve M, Aparicio J, Rabasa A, Rodriguez-Sala JJ. Efficiency analysis trees: a new methodology for estimating production Frontiers through decision trees. *Expert Syst Appl* (2020) 162:113783. doi:10.1016/j.eswa.2020.113783

9. Elyassami S, Idri A. Applying fuzzy Id3 decision tree for software effort estimation (2011). Available at: https://arxiv.org/abs/1111.0158 (Accessed October 1, 2023).

10. Asheeri MMA, Hammad M. Machine learning models for software cost estimation. In: 2019 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT); 2019 22-23 September; Bahrain (2019). doi:10.1109/3ICT.2019.8910327

11. abdelali Z, Mustapha H, Abdelwahed N. Investigating the use of random forest in software effort estimation. *Proced Comp Sci* (2019) 148:343–52. doi:10.1016/j.procs.2019.01.042

12. Priya Varshini AG, Anitha Kumari K, Janani D, Soundariya S. Comparative analysis of machine learning and deep learning algorithms for software effort estimation. *J Phys Conf Ser* (2021) 1767(1):012019. doi:10.1088/1742-6596/1767/1/012019

13. Grattarola D, Alippi C. Graph neural networks in tensorflow and keras with spektral [application notes]. *IEEE Comput Intelligence Mag* (2021) 16(1):99–106. doi:10.1109/mci.2020.3039072

14. Prabha CL, Shivakumar N. Software defect prediction using machine learning techniques. In: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184); 15-17 June 2020; Tirunelveli, India (2020). doi:10.1109/ICOEI48184.2020.9142909

15. Sharma D, Chandra P. Identification of latent variables using, factor analysis and multiple linear regression for software fault prediction. *Int J Syst Assur Eng Manag* (2019) 10(6):1453–73. doi:10.1007/s13198-019-00896-5

16. Hamada MA, Abdallah A, Kasem M, Abokhalil M. Neural network estimation model to optimize timing and schedule of software projects. In: 2021 IEEE International Conference on Smart Information Systems and Technologies (SIST); 2021 28-30 April (2021). doi:10.1109/SIST50301.2021.9465887

17. Heiat A. Comparison of artificial neural network and regression models for estimating software development effort. *Inf Softw Tech* (2002) 44(15):911–22. doi:10.1016/s0950-5849(02)00128-3

18. Xie W, Liu B, Li Y, Lei J, Du Q. Autoencoder and adversarial-learning-based semisupervised background estimation for hyperspectral anomaly detection. *IEEE Trans Geosci Remote Sensing* (2020) 58(8):5416–27. doi:10.1109/tgrs.2020.2965995

19. Yu W, Kim IIY, Mechefske C. Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme. *Mech Syst Signal Process* (2019) 129:764–80. doi:10.1016/j.ymssp.2019.05.005

20. Chen T, Guestrin C. Xgboost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; San Francisco, California, USA: Association for Computing Machinery; August 13 - 17, 2016; San Francisco, California, USA (2016). p. 785–94. doi:10.1145/2939672.2939785

21. Musilek P, Pedrycz W, Nan S, Succi G. On the sensitivity of cocomo ii software cost estimation model. In: Proceedings Eighth IEEE Symposium on Software Metrics; 2002; 4-7 June 2002; Ottawa, Canada (2002). doi:10.1109/METRIC.2002.1011321

22. Sahin CB. The role of vulnerable software metrics on software maintainability prediction. *Avrupa Bilim ve Teknoloji Dergisi* (2021)(23) 686–96. doi:10.31590/ejosat.858720

frontiers | Frontiers in Physics

TYPE Review
PUBLISHED 28 March 2024
DOI 10.3389/fphy.2024.1349463

Deep learning-powered malware detection in cyberspace: a contemporary review

Ananya Redhu[1], Prince Choudhary[1], Kathiravan Srinivasan[1] and Tapan Kumar Das[2]*

[1]School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India, [2]School of Computer Science Engineering and Information Systems, Vellore Institute of Technology, Vellore, India

OPEN ACCESS

EDITED BY
Amin Ul Haq,
University of Electronic Science and
Technology of China, China

REVIEWED BY
Rajesh Kumar,
University of Electronic Science and
Technology of China, China
Ikram Ud Din,
The University of Haripur, Pakistan
Nuzhat Naqvi,
Mohi-ud-Din Islamic University, Pakistan

*CORRESPONDENCE
Tapan Kumar Das,
tapan.das@vit.ac.in

RECEIVED 04 December 2023
ACCEPTED 15 March 2024
PUBLISHED 28 March 2024

CITATION
Redhu A, Choudhary P, Srinivasan K and Das TK (2024), Deep learning-powered malware detection in cyberspace: a contemporary review.
Front. Phys. 12:1349463.
doi: 10.3389/fphy.2024.1349463

This article explores deep learning models in the field of malware detection in cyberspace, aiming to provide insights into their relevance and contributions. The primary objective of the study is to investigate the practical applications and effectiveness of deep learning models in detecting malware. By carefully analyzing the characteristics of malware samples, these models gain the ability to accurately categorize them into distinct families or types, enabling security researchers to swiftly identify and counter emerging threats. The PRISMA 2020 guidelines were used for paper selection and the time range of review study is January 2015 to Dec 2023. In the review, various deep learning models such as Recurrent Neural Networks, Deep Autoencoders, LSTM, Deep Neural Networks, Deep Belief Networks, Deep Convolutional Neural Networks, Deep Generative Models, Deep Boltzmann Machines, Deep Reinforcement Learning, Extreme Learning Machine, and others are thoroughly evaluated. It highlights their individual strengths and real-world applications in the domain of malware detection in cyberspace. The review also emphasizes that deep learning algorithms consistently demonstrate exceptional performance, exhibiting high accuracy and low false positive rates in real-world scenarios. Thus, this article aims to contribute to a better understanding of the capabilities and potential of deep learning models in enhancing cybersecurity efforts.

KEYWORDS

artificial intelligence, cyberspace data security, deep learning, malware detection, network security

1 Introduction

This comprehensive review delves into the burgeoning role of deep learning (DL) models in the face of the ever-evolving menace of malware in cyberspace. Malware represents a continuously evolving cybersecurity threat, and traditional detection technologies often struggle to keep up with the rapid creation of new malware types [1]. However, deep learning models have gained significance in this field due to their ability to automatically learn features from large datasets [2]. Deep learning models also possess the remarkable capability to adapt to emerging threats by learning from extensive and diverse datasets [3]. They excel in extracting intricate and subtle features within malware samples, a task that may be challenging for rule-based or signature-based systems. This feature extraction prowess contributes to heightened accuracy in distinguishing between benign and malicious files, thereby reducing false positives that can disrupt legitimate operations. Moreover, deep learning models offer speed, efficiency, scalability, and continuous improvement, making them invaluable tools for real-time or near-real-time detection and response in the dynamic landscape of cybersecurity [4]. Figure 1 illustrates different categories in malware analysis.

DL models can reliably categorize malware samples into numerous families or types by analyzing their individual properties, assisting security researchers and practitioners in recognizing and responding to emerging threats more efficiently [5]. This review aims to provide an in-depth understanding of various DL architectures utilized in this field, including Recurrent Neural Networks (RNNs), Deep Autoencoders (DAEs), Long Short-Term Memory (LSTM) networks, Deep Neural Networks (DNNs), Deep Belief Networks (DBNs), Deep Convolutional Neural Networks (CNNs), and Deep Generative Models (such as Generative Adversarial Networks or GANs). RNNs are designed for sequential data processing and can capture dependencies in data over time. They are commonly used in tasks like natural language processing and speech recognition. Deep DAEs are utilized for unsupervised learning and data compression. They comprise an encoder and a decoder and find applications in feature learning and anomaly detection. LSTMs, a type of RNN, have specialized memory cells that capture long-term dependencies in data. They are particularly effective in sequential tasks where retaining context is crucial. DNNs consist of multiple layers of interconnected neurons and are employed for supervised learning tasks like image and speech recognition. They form the core of many deep learning applications. DBNs are generative models composed of multiple layers of stochastic, latent variables. They are used in tasks such as feature learning, collaborative filtering, and dimensionality reduction. CNNs are designed for processing grid-like data, such as images, and use convolutional layers to automatically learn spatial hierarchies of features. They find wide applications in image and video analysis. Deep Generative Models, including GANs, are capable of generating data rather than classifying it. GANs, for example, consist of a generator and a discriminator that compete in a game, resulting in the generation of realistic data. They are often used in image generation and data augmentation. This review investigates these models in terms of their unique capabilities and applications in the field of cybersecurity.

1.1 Limitations of previous reviews

In recent research, various issues and challenges related to malware detection using data mining have been extensively explored [6]. One significant challenge is the imbalance of classes within datasets, which affects the accuracy and robustness of malware detection models. Additionally, the need for open and public benchmarks, the emergence of concept drift, and the concerns surrounding adversarial learning techniques all pose significant obstacles to the effectiveness of these detection mechanisms. Furthermore, the interpretability of models remains a critical concern, impeding the deployment of reliable and understandable solutions. When it comes to Cyber-Physical System (CPS) malware detection, the complexity of different malware classes and their numerous variants ma kes detection even more challenging [7]. The rise of Advanced Persistent Threats (APTs) adds another layer of sophistication, demanding advanced strategies to combat coordinated and purposeful attacks. Analyzing malware, including static and dynamic aspects, presents difficulties in understanding and identifying malware, necessitating robust detection strategies. While signature-based and behavior-based methods offer distinct advantages, they also face challenges related to accuracy and efficiency in classifying programs as malicious or benign [6].

An examination of the strengths and weaknesses of signature-based and behavior-based malware detection reveals that each method has its own merits and shortcomings [6]. Signature-based detection is fast and efficient but struggles to detect polymorphic malware, whereas behavior-based detection excels in identifying unconventional attacks but faces challenges regarding storage and time complexity. Ransomware detection and prediction techniques have received significant attention, particularly in the context of machine learning methods [8]. However, there has been a lack of emphasis on predicting ransomware, and identified shortcomings in real-time protection and 0-day ransomware identification highlight the need for more comprehensive approaches. Adversarial machine learning exploitation and concept drift further complicate the landscape of machine learning models in this domain.

Deep Learning (DL)-based malware detection frameworks encounter several challenges, including data imbalance, interpretability issues, susceptibility to adversarial attacks, the need for regular updates, and difficulties in achieving cross-platform detection [9]. Efficient feature extraction techniques and the recognition of new characteristics in 0-day malware add further complexity to the development and deployment of DL models. Deep learning for 0-day malware detection and classification focuses on learning paradigms, feature types, benchmark datasets, and evaluation metrics. API/System calls are the most common type of feature, and prevalent benchmark datasets include Drebin. Evaluation metrics encompass Accuracy, Precision, Recall, F1-score, False Positive Rate, False Negative Rate, Area Under the Curve, and Evasion rate.

In the domain of Android malware detection using machine learning, challenges and advancements in static analysis have been explored [10]. Machine learning techniques applied to features extracted through static analysis have shown varying degrees of success, relying on tools like APK Tool and Androguard for decompiling and analyzing APK files. The challenges posed by adversarial attacks for PE (Portable Executable) malware are multifaceted. Adversarial attacks in both feature-space and problem-space encounter difficulties in maintaining the format, executability, and maliciousness of PE files. The taxonomy of attacks includes white-box attacks, where the attacker has full knowledge of the model, and black-box attacks, where limited or no knowledge of the model's internals presents additional challenges [11]. Feature-space attacks involve direct manipulation of features, while problem-space attacks entail altering the actual inputs, such as PE files. These challenges highlight the need for robust defenses against adversarial threats in the context of malware detection. Table 1 provides a comparison with previous review papers with a similar focus.

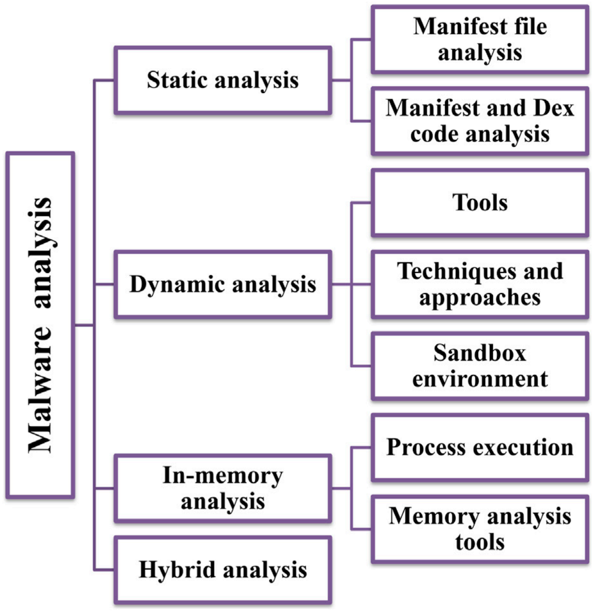


FIGURE 1
Categories in malware analysis.

1.2 Motivation and objectives of this review

The rapidly evolving landscape of cybersecurity presents an ongoing challenge, particularly in the realm of malware detection. Traditional methods struggle to keep pace with the relentless creation of new malware variants. They struggle to keep up with evolving threats, making deep learning's ability to autonomously extract features from vast datasets crucial. Deep Learning models excel in discerning intricate patterns within malware, offering scalability, efficiency, and continuous improvement. As a result, there's a pressing need for innovative solutions that can adapt to emerging threats and provide robust protection against cyberattacks.

This review aims to delve into the burgeoning role of deep learning models in combating malware threats in cyberspace. It provides a thorough exploration of various deep learning architectures and their applications in malware detection. By analyzing the strengths and limitations of each model, the review offers valuable insights to researchers and practitioners seeking to harness deep learning techniques for cybersecurity. Recognizing the dynamic nature of cyber threats, the review also sheds light on the evolving landscape of malware and the increasing sophistication of cyber-attacks. It identifies future research directions, emphasizing the need for innovative DL-based solutions that can adapt to dynamic malware behavior and effectively counter adversarial attacks.

1.3 Contributions of this review

The main contributions of this article are as follows:

a) This review provides a critical assessment of the existing literature in the field of deep learning-powered malware detection in cyberspace. Further, it helps to identify gaps and areas for improvement, guiding future research directions and ensuring a more comprehensive understanding of the subject matter.
b) This review extends the scope of traditional approaches to encompass the rapidly growing threat landscape targeting mobile devices. This expansion of focus ensures that the review remains relevant and up-to-date with emerging trends in cybersecurity, providing insights into the unique challenges and opportunities presented by mobile malware.
c) By including recent tools in malware analysis and detection, this review offers readers a comprehensive overview of the current state-of-the-art technologies and methodologies available for combating malware threats. This enables researchers and practitioners to stay abreast of the latest advancements in the field and make informed decisions when selecting and implementing detection tools and techniques.
d) By incorporating a diverse range of tools in malware detection, including behavioral analysis tools, threat intelligence platforms, deception tools, and memory forensic tools, this review provides a holistic perspective on the multifaceted nature of malware detection. This ensures that readers gain insights into the various approaches and methodologies employed in the detection and analysis of malware, enhancing their understanding of the complexities involved in combating cyber threats.
e) By highlighting open challenges in the field of malware detection using deep learning, this review identifies areas where further research and development are needed to address existing gaps and limitations. This stimulates discussion and collaboration within the research community, fostering innovation and driving progress towards more effective and robust solutions for malware detection using deep learning techniques.

2 Survey methodology

Figure 2 illustrates the process of article selection for this review, adhering to the PRISMA guidelines [13]. A comprehensive search for deep learning models in malware analysis and detection was conducted in three databases, namely, Google Scholar, Scopus, and Web of Science, spanning from January 2015 to December 2023. The search string “Cyberspace, Deep Learning, and Malware Detection” was employed to collect relevant articles. Inclusion and exclusion criteria were applied to determine the articles to be included in the review. Specifically, the articles had to be written in English, published in peer-reviewed journals or conferences, and relevant to both malware analysis and detection and deep learning. During the initial stage, 900 non-duplicate articles were obtained, as depicted in Figure 2. Following the screening of titles and abstracts, 457 articles were excluded. Subsequently, 171 articles for which full-text reports could not be retrieved were also removed from consideration. Additionally, 272 articles were assessed for eligibility, leading to the removal of 133 articles with incomplete information. Finally, a total of 139 articles met all the criteria and were selected for this review.

TABLE 1 Comparison with previous review articles with a similar background. (✓: Yes and ✗: No).

Reference	Year	Summary of the main contributions	Deep learning	Open challenges	Future directions
Our Review	-	The review examines the effectiveness of Deep Learning models in detecting malware, as well as highlighting the existing challenges and potential future opportunities	✓	✓	✓
[9]	2023	This work presents a survey of deep learning techniques for 0-day malware detection and classification, elaborating on the taxonomy of resilient techniques for 0-day attacks	✓	✓	✗
[7]	2023	The review discusses the significant impacts of malware threats on Cyber-Physical Systems and explores the application of nature-inspired metaheuristic algorithms as a means to counter these threats	✗	✗	✓
[8]	2023	This work offers a thorough overview of the evolution, taxonomy, and research related to ransomware. It specifically focuses on the challenges and detection techniques within the realm of cybersecurity	✓	✗	✓
[10]	2022	This work provides a critical review of machine learning approaches used for Android malware detection. It covers various learning methods and their organization based on feature use	✓	✗	✓
[11]	2022	The review discusses the utilization of Indicators of Compromise (IOCs), machine learning methods, and deep learning-based methods in tools and anti-malware products	✓	✓	✓
[12]	2020	This survey provides a detailed overview of traditional machine learning methods, including their challenges and limitations in the field. It also highlights recent trends, particularly in deep learning, and discusses open research issues	✓	✓	✗
[6]	2018	This work offers a systematic and detailed survey of malware detection mechanisms that utilize data mining techniques. It classifies the approaches into two categories: signature-based methods and behavior-based methods	✓	✓	✗

3 Deep learning-powered malware detection in cyberspace

Deep learning (DL) models are highly proficient in autonomously learning features from extensive datasets, making them particularly suitable for detecting malware in the digital realm. By thoroughly analyzing malware samples, DL models acquire the capability to accurately categorize them into distinct families or types.

DL models undergo training using comprehensive sets of extracted attributes, including elements such as opcode sequences, API calls, and system calls. This training empowers the models to differentiate intricate patterns that distinguish malware from benign software. Consequently, these well-trained models can be deployed to classify new and previously unknown samples, providing a powerful tool for robust detection and in-depth analysis of malware. Figure 3 illustrates the current taxonomy of deep learning models for malware detection in cyberspace. Additionally, Table 2 provides a summary of research conducted on deep learning models for malware detection in cyberspace.

3.1 Recurrent neural networks

Recurrent Neural Networks (RNNs) play a significant role in the field of malware detection in cyberspace due to their ability to handle sequential input data. In the context of malware detection, RNNs are useful for assessing system calls, API calls, and network traffic generated by software applications to identify potentially harmful activities. System calls provide insights into a program's actions within the system, allowing the detection of deviations from normal software behavior that may indicate malicious activity. API calls reveal how a program interacts with the underlying system, enabling the identification of specific APIs used for malicious purposes, such as modifying system settings. Network traffic data is crucial for detecting malware that communicates with external servers, attempts data exfiltration, or engages in suspicious data exchange over the network.

RNNs excel at analyzing sequences of data and capturing temporal dependencies in the behavior of potentially malicious software. They are particularly effective at processing sequential data encountered in malware analysis, such as sequences of system calls, API calls, or network traffic generated by software. By being exposed to sequential data, RNNs become adept at discerning correlations and patterns that indicate malware behavior. They take sequential data as input, working through it one element at a time and updating their internal state based on the observed data. This mechanism allows RNNs to capture temporal dependencies and patterns in the data, which is essential for understanding the dynamic nature of malware. The hidden state within RNNs serves as a form of memory, retaining information about previous observations and enabling the contextualization of past events while predicting the current one. However, these strengths are counterbalanced by challenges inherent in its application. There is a limitation that Kaspersky malware family classification criteria of the malware sample used for analysis in this paper may not be

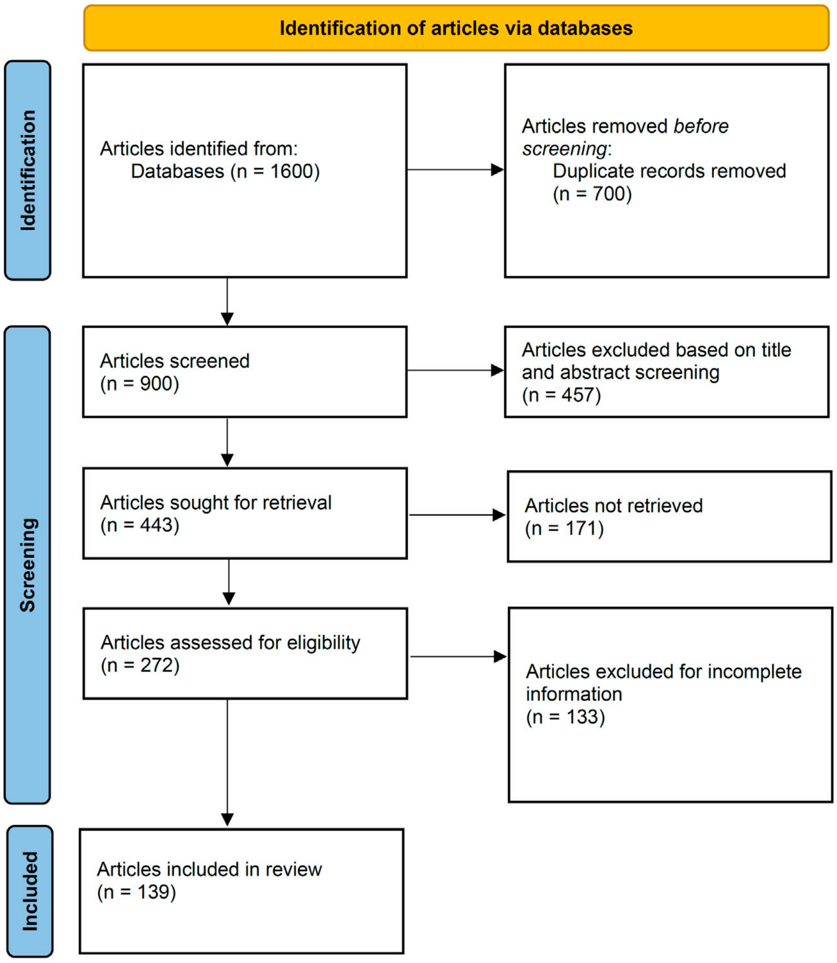

Identification of articles via databases
Identification
Articles identified from:
Databases (n = 1600)
Articles removed *before screening*:
Duplicate records removed
(n = 700)
Screening
Articles screened
(n = 900)
Articles excluded based on title and abstract screening
(n = 457)
Articles sought for retrieval
(n = 443)
Articles not retrieved
(n = 171)
Articles assessed for eligibility
(n = 272)
Articles excluded for incomplete information
(n = 133)
Included
Articles included in review
(n = 139)

FIGURE 2
Selection of articles adhering to the PRISMA guidelines.

accurate. Only the types and order of the APIs were taken into consideration that were called when extracting patterns for APIs called by malware and evaluating them. Since the API itself is higher-level than the machine code or assembly in the computer, the performance may be improved if the semantic criteria and semantic distinction of malware API to be extracted [1].

To train RNNs for malware detection, historical data is used to adjust their internal parameters. Backpropagation through time is employed to update the model's weights and biases based on prediction errors, allowing the them to learn patterns associated with malware. Additionally, their temporal modeling capabilities enable the identification of anomalous behavior trends within an application over time, facilitating the early detection of novel or previously undiscovered strains of malware. The ability of RNNs to capture nuanced and evolving behavioral patterns makes them a valuable tool in malware detection. They enhance security by providing a dynamic, adaptable, and context-aware approach to identifying malicious software, especially in the face of rapidly changing cybersecurity threats. RNNs are effective at identifying evasive and polymorphic malware, which employ techniques to avoid detection and continually change their code to generate different variants. RNNs can tackle these challenges by recognizing deviations from normal behavior and analyzing the evolving patterns in the code.

Addressing data imbalance in training RNNs demands a strategic approach. One method involves data augmentation, wherein synthetic data is generated by introducing variations to the existing minority class samples, thereby enriching the dataset. Additionally, employing sampling techniques such as oversampling (replicating minority class samples) or undersampling (reducing the number of majority class samples) can help balance the dataset distribution. Moreover, integrating cost-sensitive learning proves effective by assigning varying costs to misclassification errors across different classes, thereby accommodating the imbalance and enhancing model performance. These strategies collectively empower RNNs to navigate the challenges posed by skewed data distributions, ultimately fostering more robust and accurate predictions.

Experiments were conducted with 787 malware samples belonging to nine families. In the experiments that were carried out, representative API call patterns of nine malware families on 551 samples were extracted as a training set and performed

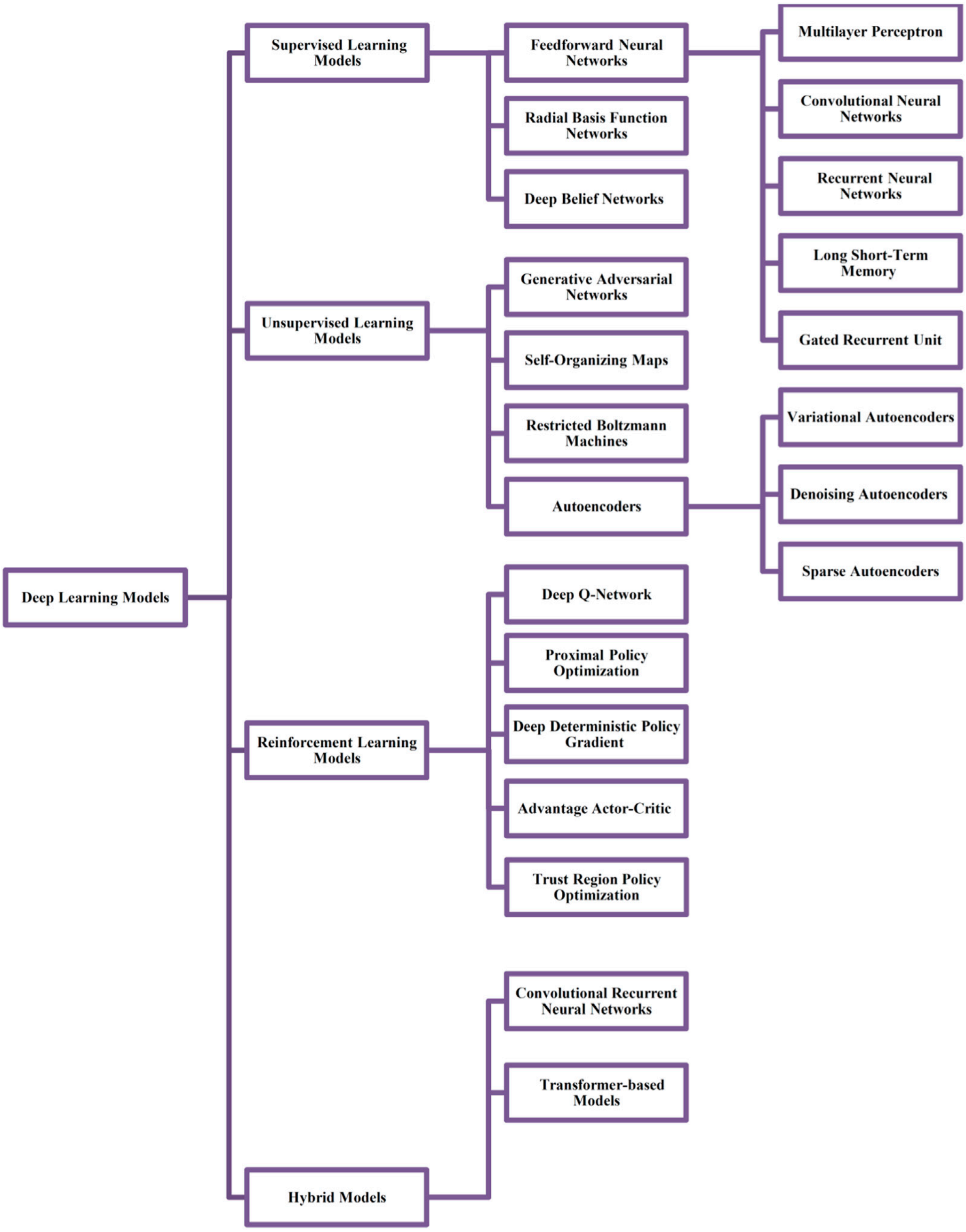

Deep Learning Models
Supervised Learning Models
Unsupervised Learning Models
Reinforcement Learning Models
Hybrid Models
Feedforward Neural Networks
Radial Basis Function Networks
Deep Belief Networks
Generative Adversarial Networks
Self-Organizing Maps
Restricted Boltzmann Machines
Autoencoders
Deep Q-Network
Proximal Policy Optimization
Deep Deterministic Policy Gradient
Advantage Actor-Critic
Trust Region Policy Optimization
Convolutional Recurrent Neural Networks
Transformer-based Models
Multilayer Perceptron
Convolutional Neural Networks
Recurrent Neural Networks
Long Short-Term Memory
Gated Recurrent Unit
Variational Autoencoders
Denoising Autoencoders
Sparse Autoencoders

FIGURE 3
Current Deep learning Models for Malware Detection in Cyberspace–Taxonomy.

classification on the 236 samples as a test set. Classification accuracy results using API call patterns extracted from RNN were measured as 71% on average. The results show the feasibility of our approach using RNN to extract representative API call pattern of malware families for malware family classification. First, the similarities of the representative API call patterns with extracted from each family and the API call sequences of the malware belonging to the test set are compared. Then, top three representative API call patterns were selected with the highest similarity compared to each malware in the test set and compare the top three family results with the correct answer. Jaccard similarity coefficient was used as the similarity measure [1].

Experimental results using a balanced dataset showed 83% accuracy and a 0.44 loss, which outperformed the baseline model in terms of the minimum loss. The imbalanced dataset's accuracy was 98%, and the loss was 0.10, which exceeded the state-of-the-art model's accuracy. This demonstrates how well the suggested model can handle malware classification [23].

One successful application of RNNs in malware detection is dynamic behavioral analysis. This involves analyzing software

TABLE 2 Details of works on deep learning-powered models for malware detection in cyberspace.

Reference	Malware type	DL models used	Brief focus	Key contributions	Limitations	Performance metrics
[5]	Smart Vehicular Network malware	Duelling Deep Q Learning	Detect abnormal network traffic and classification of attack	● Improve detection accuracy using a supervised machine learning task based on agent-environment interaction using a modified duelling DQN model ● The algorithm is modified such that it can pick a supervisor to implement an interaction mechanism as a supervised algorithm for action, state, and reward ● Demonstrates the efficacy of the suggested invasion detection strategy and the enhancement in classification performance over conventional ML algorithms	● Low Recall compared to existing systems	● The experimental results showed that SIMPLE achieves an accuracy of 90% in a 5-way classification task on new malware families
[14]	IoT botnet	CCR-ELM	Lightweight framework to detect IoT botnet and botnet clusters	● Framework for detecting IoT botnets and botnet clusters. The framework works with data regarding automated behaviour ● The Zeek Network Analysis Framework is used for reassembling the network flow. Every network flow produces 27 statistical and behavioural records pertaining to network communication, application-level protocols, and payload exchanged ● For the botnet detection portion of ELM, overall efficacy improves as the number of concealed nodes rises. A botnet family's behaviour may vary based on the device it infects and the stage at which it is deployed	● Requires more memory space with the addition of data	● The proposed ensemble method achieves the best outcome with the highest accuracy 99.9%, compared to state-of-the-art machine learning, deep learning, and ensemble models
[15]	Worms	CNN Model	Anomaly based intrusion detection, classification of an event as malignant or benign	● In context of accuracy and recall, the devised model outperforms popular models such as NB, J48, RF, Bagging, and Adaboost	● Cannot distinguish fuzzy attacks ● Multiclass classification accuracy has room for improvement	● Support Vector Machine (SVM), and AdaBoost algorithms and they achieved the highest accuracy rate of 99,80% with the Decision Tree classifier
[16]	Malware Detection in Fog Computing	CNN	Optimization of detection and classification mechanism for malware detection in Fog Computing	● Structured and powerful malware detection system can be deployed in fog computing by utilising a feature reduction ability that takes a screenshot of a file and converts it into an image and gives a new way for feature reduction by reading only a specific number of bytes per 1 KB of data and splitting an image into chunks, which divides a large file into fixed-size output images ● The training of a model involves the inclusion of disturbance to improve the model's accuracy. This method obtained a 97.2% success rate, used 16 times fewer features than other approaches, and was able to manage enormous files	● When detecting files with specific extensions such as.7zip, which have a fewer number of files in the training dataset, the model performs poorly ● Model detects chunks of large file as malware even if it is benign due to lack of adequate samples in training dataset.	● They used Convolutional Neural Network (CNN) algorithm for classification and achieved an accuracy rate of 94%
[17]	Android malware	● SERLA ● SimHash and CNN	The framework uses disassembly technology to produce bytes file and asm file for each executable file and a special matrix generation technique to produce three 256x 256 square matrices. The three matrices are then utilised as the three channels of an RGB image and combined to create a colour image. Furthermore, to improve the discriminative power of the RGB images, we apply adaptive histogram equalization processing utilizing the CLAHE (Contrast Limited Adaptive Histogram Equalization) data augmentation technique. In conjunction with the oversampling method for training neural networks, trained models for malware detection and family classification are ultimately obtained	● Proposed a comprehensive detection and classification framework for malware that can convert executable files into their corresponding bytes and asm files. Therefore, we create a steady dataset containing both normal software samples and malware samples. This dataset can be utilised for a wider range of malware detection experiment categories. ● A novel approach is introduced for data representation, which leverages binaries and word vectors derived from both bytes' files and asm files. This innovative method aims to extract comprehensive information from software samples, enabling a more holistic understanding of the data. It considers the characteristics of more aspects of the data samples and can provide more valuable assistance for the training of the detection model, thereby enhancing the detection performance.	● The labelling method used cannot filter normal software with 100% accuracy. ● Compilation configuration is also one of the noticeable issues. Different compilation configurations will make the code with the same function compiled into different assembly files, which can cause wrong classification of the detection model	● The experimentation on a recent data set which includes 11,120 applications showed that an accuracy of 97% on average can be achieved. ● SWORD obtained an accuracy of 94.2% in experiments on a data set containing 2000 Android samples from various sources

(Continued on following page)

TABLE 2 (*Continued*) Details of works on deep learning-powered models for malware detection in cyberspace.

Reference	Malware type	DL models used	Brief focus	Key contributions	Limitations	Performance metrics
				● In conjunction with the data augmentation technique in computer vision, an optimised deep neural network SERLA based on SEResNet50, Bi-LSTM, and an attention mechanism is developed for malware detection. Compared to other neural network malware detection models and even state-of-the-art methods, our model is superior in all evaluation metrics, as demonstrated by experimental results		
[18]	Malware	Dragonfly-based DGDBN	Cloud security and malware detection	● Development of a novel Dragonfly-based Genetic Deep Belief Network (DGDBN) technique for safeguarding VMs in cloud environments	● Limited information on dataset and real-world testing	● The authors prove that such gradient approximation mechanism allows the objective function to converge to optima with probability 1, where in their experiments only a 2% accuracy loss is observed on average on GoogleLeNet training
[19]	Android malware	DAE-CNN (Deep Autoencoder-CNN)	Large-scale Android malware detection	● The research proposes a hybrid model combining a Deep Autoencoder (DAE) and Convolutional Neural Network (CNN) to enhance large-scale Android malware detection ● To enhance efficiency, the research introduces DAE as a pre-training method for CNN. This approach significantly reduces training time, specifically an 83% reduction compared to CNN-S ● The model incorporates ReLU activation functions, sparsity rules, and the combination of convolutional and pooling layers with the full-connection layer to enhance feature extraction capability	● The research discusses the model's performance in terms of accuracy and training time reduction but does not specify if it was tested in a real-world environment or against a broader range of Android malware. ● Real-world testing is crucial to validate the model's practical effectiveness	● Showed an excellent performance with an overall detection accuracy of 99.3% for Probe, Remote to Local, Denial of Service and User to Root type of attacks
[20]	Botnet Malware	Khaos (DGA Model)	Domain Generation Algorithm (DGA) for botnets	● The research introduces Khaos, a novel Domain Generation Algorithm (DGA) for generating domain names used in botnet command and control (C&C) servers. Khaos is designed to enhance anti-detection capabilities ● The study leverages neural language models and the Wasserstein Generative Adversarial Network (WGAN) to design Khaos. By mimicking real domain names through the arrangement of syllables and acronyms, Khaos aims to create domain names that are challenging for detection	● The study does not address the adaptability of Khaos to evolving malware threats ● The dynamic nature of botnets and malware requires continuous innovation to stay ahead of detection methods	● Two image conversion methods, byteplot and space-filling curves, were used to represent the malware samples, and a ResNet-50 architecture was used to train models on the image datasets. The models were then tested against a projected gradient descent attack. It was found that without GAN-generated data, the models' prediction performance drastically decreased from 93%–95%–4.5% accuracy
[4]	Cybersecurity Threats	EDRBM (Ensemble Deep Restricted Boltzmann Machine)	Classification of cybersecurity threats in large-scale networks	● The research introduces Ensemble Deep Restricted Boltzmann Machine (EDRBM) as a novel deep learning model for the classification of cybersecurity threats in large-scale network environments. This represents a significant advancement in the field of threat detection ● EDRBM is applied to classify cybersecurity threats, with a specific emphasis on malware attacks. It serves as a classification model capable of differentiating between benign and malicious network traffic flowsets	● The content does not delve into the data and computational resources needed for implementing EDRBM in large-scale network environments ● Real-world implementation may require significant resources, and these requirements should be considered	● When tested on all the features of the NSL-KDD data set, the deep learning method obtaines very low result compared with the mentioned methods, but when it is tested on six features, the method in terms of accuracy metric gets the high result and composed of 75.75%
[21]	Malware Families	Shallow Convolutional Neural Network (CNN)	Assessing the vulnerability of malware classifier to dead code insertion and adversarial attacks	● Introducing a Double Q-network-powered framework to induce misclassification in malware families ● Training an AI agent to insert dead code instructions into malware samples ● Demonstrating significant classification accuracy reduction in malware classifier ● Achieving 100% evasion rate for specific malware families	● Lack of discussion about the impact of these attacks in real-world settings ● No information on the resource requirements for deploying the proposed framework ● Ethical and legal aspects related to malware manipulation and evasion are not addressed ● Specific limitations associated with the proposed framework are not discussed	● The model outperforms other CNN architecture by a significant margin on the accuracy metric, where 98.85% was achieved on both datasets, Benign and Malicious PE Files dataset and MalwareDataSet, and 98.37% was achieved on the Classification of Malwares dataset.

(Continued on following page)

TABLE 2 (*Continued*) Details of works on deep learning-powered models for malware detection in cyberspace.

Reference	Malware type	DL models used	Brief focus	Key contributions	Limitations	Performance metrics
				● Enhancing evasion success rates through deep reinforcement learning with the Double Q-learning algorithm ● Validating results on the Portable Executable files dataset for reproducibility		
[22]	Steganography	Convolutional Neural Networks (CNN) and Extreme Learning Machines (ELM)	Training machine learning models for malware classification based on features obtained without disassembly or code execution	● Introducing a method to visualize malware samples as images for classification ● Evaluating two machine learning techniques: CNNs and ELMs ● Demonstrating that ELMs achieve comparable accuracy to CNNs with significantly faster training times ● Showing that ELMs and CNNs perform well with both one-dimensional and two-dimensional data	● Lack of specific information about the malware types or dataset used in the experiments ● No discussion of the real-world applicability or performance of the proposed approach ● The paper does not address the potential limitations of image-based malware classification in terms of accuracy or security	● The average accuracy for the unweighted model is 96.5%, while for the weighted model we obtain a slight improvement at 97.7%

behavior, such as file operations, system calls, registry accesses, and network interactions, to identify potentially malicious activities. They can create behavioral profiles of software by observing and analyzing its interactions with the operating system and external resources. These profiles can be compared against known malware behavior patterns to identify potential threats. RNNs excel in anomaly detection, crucial for identifying malware that exhibits unusual or unexpected behavior. By continuously learning and adapting to evolving malware behavior, RNN-based systems can update their models to detect novel threats, making them effective against 0-day attacks. They also reduce false positives by focusing on behavioral analysis, prioritizing potential threats for security professionals to investigate. RNNs can consider the context of each action within a software's behavior, distinguishing legitimate activities from malicious ones. Given their proficiency in processing ordered data, RNNs are well-suited for tasks involving time series data. They adapt to various application domains, including speech recognition, language modeling, translation, and image captioning, where sequential data analysis is crucial [1].

3.2 Deep autoencoder

Deep Autoencoders (DAEs) have ascended as potent tools in the fight against malware, particularly within the realm of unsupervised learning. DAEs are a type of neural network that encodes high-dimensional input into a lower-dimensional representation and then decodes it back into its original format. They serve as valuable tools for uncovering the inherent characteristics and patterns exhibited by benign software applications. By training on extensive datasets of benign applications, DAEs learn and internalize the defining traits of harmless programs.

Labeled datasets are crucial in training DAEs for malware detection. These datasets play a pivotal role in imparting the model with the ability to discern nuanced patterns that distinguish benign software from malicious counterparts. Operating within a supervised learning framework, they are trained on input-output pairs derived from labeled datasets. Each input represents features extracted from both benign and malicious samples, enabling the model to comprehend the distinctive characteristics associated with each class. This process contributes to the model's generalization ability, allowing it to recognize common patterns indicative of malware across different variations and instances, ensuring effectiveness on previously unseen data.

Addressing data imbalance in training deep DAEs and variational autoencoders (VAEs) involves employing several strategic approaches. One method involves leveraging the inherent generative capacity of VAEs to counter data scarcity by generating synthetic data. This technique helps balance the class distribution, enhancing the model's ability to learn from underrepresented classes. Additionally, adversarial training can be utilized to foster robustness against imbalances. By exposing the model to adversarial examples, it learns to create more resilient representations, mitigating the impact of data imbalance. These strategies collectively empower them to handle skewed datasets effectively, improving their capacity to generalize and learn meaningful representations across all classes.

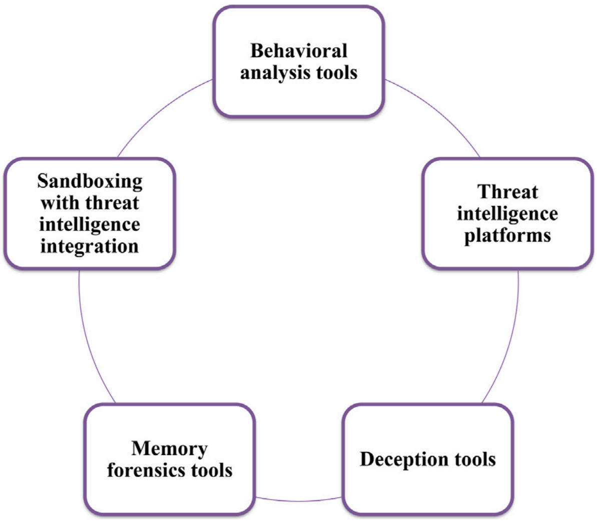

Behavioral analysis tools
Threat intelligence platforms
Deception tools
Memory forensics tools
Sandboxing with threat intelligence integration

FIGURE 4
Categories of recent tools in malware analysis and detection.

When presented with novel applications, DAEs can assess whether they deviate significantly from the learned benign patterns. This assessment is made possible by evaluating the reconstruction error generated during the decoding process. A high reconstruction error indicates a substantial departure from the expected benign behavior, raising suspicion of potential malicious activity. They can also be seamlessly integrated with other machine learning algorithms, enhancing the comprehensiveness of malware detection strategies.

The approach without autoencoder, both precision and recall are 99 Percentage for just the Bi-LSTM model in detecting malicious activities in cyber security. Average precision and recall of the performed model with autoencoder is 93% [24].

The architectural framework of DAEs consists of two pivotal stages: encoding and decoding. In the decoding phase, the compressed representation is reconstructed back to its original form, with each network layer performing a distinctive transformation on the input data. Their adaptability benefits applications like natural language processing (NLP), picture recognition, identity verification, and data reduction.

While DAEs have ascended as potent tools in the fight against malware, deploying them for real-world malware detection comes with various challenges. Scalability is a significant challenge, as training and deploying DAEs at scale can strain organizational infrastructure. Efficient scaling becomes essential as datasets grow in size and models become more complex. The demand for computational resources, especially during training, poses another challenge. Organizations must address the need for processing power, which can lead to longer inference times and increased operational costs. Real-time analysis, crucial for timely malware identification, can be challenging with DAEs, particularly those with complex architectures that struggle to achieve low-latency predictions. Imbalances in real-world malware datasets pose challenges related to biased models favoring the majority class, resulting in suboptimal detection of less common or emerging malware variants. The interpretability and explainability of DAEs, often seen as black-box models, become critical in a production setting to gain the trust of security analysts and stakeholders [24].

In malware detection, a comparative analysis reveals distinctive strengths and weaknesses among DAEs, traditional machine learning algorithms, and other deep learning approaches. DAEs excel in unsupervised feature learning and automatically capture intricate representations vital for complex tasks. Their ability to detect anomalies without labeled data is advantageous for identifying new malware variants. Traditional machine learning algorithms, like Decision Trees, offer superior interpretability and computational efficiency but rely on manual feature engineering and have limitations in anomaly detection [19]. Deep learning approaches, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), excel in spatial and temporal contexts but require large labeled datasets and can be computationally intensive. Factors like interpretability, data availability, and analysis requirements play a crucial role in choosing among DAEs, traditional algorithms, or other deep learning models. DAEs, with their focus on unsupervised feature learning and anomaly detection, stand out in the malware detection toolkit, each approach presenting unique strengths tailored to the demands of the cybersecurity landscape.

Deep Autoencoders (DAEs) find practical application in unsupervised learning for discerning inherent traits within benign software. They leverage extensive datasets of benign applications to learn characteristic features. Through this learning process, DAEs become adept at identifying deviations from these established norms, effectively flagging potential malware presence. By analyzing and detecting anomalies in software behavior, DAEs serve as a valuable tool in the continuous battle against cybersecurity threats, enabling proactive identification of suspicious activities and potential threats [25].

Unsupervised learning presents a promising avenue for discerning the inherent traits of malware, and its efficacy lies in the ability to perform effective feature learning without relying on labeled data. However, despite its potential, DAEs encounter notable challenges.

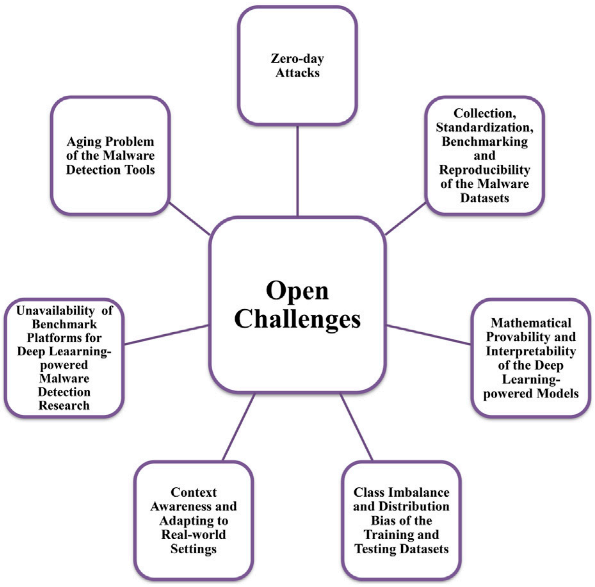

Zero-day Attacks
Aging Problem of the Malware Detection Tools
Collection, Standardization, Benchmarking and Reproducibility of the Malware Datasets
Open Challenges
Unavailability of Benchmark Platforms for Deep Leaarning-powered Malware Detection Research
Mathematical Provability and Interpretability of the Deep Learning-powered Models
Context Awareness and Adapting to Real-world Settings
Class Imbalance and Distribution Bias of the Training and Testing Datasets

FIGURE 5
Open challenges—deep learning-powered malware detection in cyberspace.

Scalability poses a significant hurdle during both training and deployment phases, demanding innovative solutions to handle the complexities of large-scale data. Additionally, the vulnerability to adversarial attacks presents a pressing concern, necessitating robust defense mechanisms to fortify these models. Furthermore, integrating unsupervised learning methodologies with existing security infrastructure proves to be a challenging task, demanding a concerted effort to harmonize these disparate elements effectively. Thus, while holding considerable promise, the practical implementation of unsupervised learning in identifying malware characteristics necessitates a strategic approach to mitigate these formidable challenges.

3.3 LSTM

The Long Short-Term Memory (LSTM) architecture has demonstrated its effectiveness in virus detection due to its ability to identify long-term dependencies within sequential data. LSTMs are a type of recurrent neural network (RNN) that use memory cells and gates to control the flow of information within the network. This makes them valuable for analyzing sequences of system calls, API calls, or network traffic generated by applications, especially in the context of malware detection in cyberspace. By processing sequential data using LSTM networks, patterns and correlations indicative of malicious behavior can be discovered. The following techniques are used in this scenario:

a) Sequence Encoding: System call sequences are encoded into numerical vectors, where each system call is represented as an integer or a one-hot encoded vector. This encoding enables effective processing by the LSTM network.
b) Sequence Padding: Sequences are often padded or truncated to a fixed length to ensure uniform input lengths. This step is crucial for creating consistent input for the LSTM.
c) LSTM Architecture: LSTM layers are utilized to capture the temporal dependencies and the order of system calls within the encoded sequences. LSTMs excel at modeling long-range dependencies, making them well-suited for this task.
d) Output Classification: The output of the LSTM layer is typically connected to a classification layer responsible for distinguishing between benign and malicious behavior based on the patterns learned from the system call sequences.

Like system call analysis, LSTM networks can be used for analyzing sequences of API calls to detect malware in cyberspace. The techniques involved are similar to those used in system call analysis, including sequence encoding, padding, LSTM architecture, and output classification. Additionally, they can also be utilized to model the behavior of an application over time, enabling the identification of anomalous activities that may indicate the presence of a new or previously undiscovered malware strain [25]. For enhanced malware detection strategies, LSTMs can also be seamlessly integrated with other machine learning techniques, such as Convolutional Neural Networks (CNNs).

The development of the LSTM architecture was primarily motivated by the need to address the vanishing gradient problem present in standard neural networks [26]. This problem arises when each connection within a network has its individual weight that remains unchanged over time, leading to training difficulties. As a type of RNN, LSTMs leverage memory cells and gating mechanisms to effectively control the flow of information, making them well-suited for the analysis of sequences involving system calls, API calls, or network traffic while identifying patterns and correlations indicative of malicious behavior.

A high-quality dataset consisting of 2,060 benign and memory-resident programs was created. In other words, the dataset contains 1,287,500 labeled sub-images cut from the MRm-DLDet transformed ultra-high resolution RGB images. MRm-DLDet was implemented for Windows 10, and it performs better than the latest methods, with a detection accuracy of up to 98.34%. Twelve diferent neural networks were trained and the F-measure up to 99.97% [27].

To address the challenges posed by data imbalance in LSTM training, several strategies are employed. One approach involves employing data augmentation techniques, which entail generating synthetic data by introducing variations to the existing minority class samples. This method aids in balancing the dataset and providing the model with more diverse instances to learn from. Another valuable strategy involves leveraging sampling techniques such as oversampling or undersampling. Oversampling involves replicating minority class samples to balance the class distribution, while undersampling focuses on reducing the number of majority class samples. These methods help create a more equitable representation of classes within the dataset, enabling the LSTM to learn effectively from both the majority and minority classes. Furthermore, adopting cost-sensitive learning techniques proves beneficial. By assigning varying costs to misclassification errors in different classes, the model can account for the imbalance and prioritize accurate classification of the minority class. This approach ensures that the LSTM places appropriate emphasis on correctly identifying instances from both the majority and minority classes, thereby enhancing overall performance despite data imbalances [26].

LSTM's mastery in capturing temporal dependencies enables a deep comprehension of malware's dynamic behavior over time. However, this strength comes with inherent challenges. The resource-intensive nature of training poses a significant obstacle, while vulnerability to adversarial attacks is a critical concern. Additionally, the constant need for updates to align with the ever-evolving array of malware variants presents an ongoing demand. Despite these hurdles, the methodology's proficiency in unraveling intricate data sequences remains a promising frontier in the realm of deciphering and combating malware conduct.

3.4 Deep neural network

Deep Neural Networks (DNNs) have gained prominence in virus detection due to their remarkable capacity to comprehend complex patterns and data characteristics. DNNs represent a class of artificial neural networks characterized by numerous interconnected layers of nodes. These layers collaboratively process incoming data, ultimately yielding predictions or classifications. In the realm of malware detection in cyberspace, DNNs are trained on extensive datasets encompassing both benign and malicious programs, enabling them to discern the fundamental attributes and patterns inherent to each class.

Leveraging the knowledge acquired from this training, DNNs can effectively categorize new applications as either benign or malicious based on their intrinsic characteristics. They exhibit versatility in assessing diverse forms of input data, spanning system calls, API calls, and network traffic, making them adaptable to various malware detection scenarios. For instance, in API call pattern recognition, Convolutional Neural Networks (CNNs), a type of DNN, can effectively analyze sequences for anomaly detection or malware identification by representing each API call as a feature vector. In network traffic analysis, DNNs, including CNN architectures, excel at detecting spatial patterns within data for intrusion detection, often extracting features from packet headers or payloads. Additionally, CNNs prove valuable in image-based malware detection, where they process images of executable files to identify malicious code patterns. Furthermore, the ability of DNNs to integrate multiple forms of input data in multimodal threat analysis, combining features from system calls, API patterns, and network traffic, highlights their capability for comprehensive threat assessment [28].

The utilization of transfer learning allows DNNs trained on one malware classification task to be fine-tuned for related challenges, showcasing their adaptability and knowledge transfer capabilities in cybersecurity applications. The training process for DNNs typically involves backpropagation, a technique that seeks to minimize the loss function's value through the gradient descent approach. The training process involves several key stages, starting with the initialization of weights and biases, a critical step that establishes the foundation for effective learning [29]. As the input data undergoes forward propagation, traversing through the network's layers, activations are computed, and the output is generated based on the current parameters. Simultaneously, the loss function calculates the disparity between the predicted output and actual labels, providing a quantifiable metric for the network's performance. Backpropagation follows, utilizing the chain rule to compute gradients and propagate errors backward through the network. This process enables the network to discern the contribution of each weight to the overall error. Subsequently, gradient descent optimization adjusts the weights and biases to minimize the loss function, guiding the network toward optimal configurations for proficient malware detection. The entire sequence of forward propagation, loss calculation, backpropagation, and weight updates iterate over multiple epochs, allowing the network to progressively refine its parameters. These iterative adjustments enhance the network's capacity to generalize and effectively identify previously unseen malware variants, underscoring its effectiveness in the realm of cybersecurity. Resampling methods such as oversampling, undersampling, or using techniques like SMOTE (Synthetic Minority Over-sampling Technique) can rebalance the dataset, ensuring equal representation of classes. Additionally, adjusting class weights during training serves as a means to penalize misclassifications of the minority class more heavily, allowing the model to prioritize learning from the underrepresented data. These strategies collectively aim to mitigate the impact of data imbalance, enabling DNNs to better generalize and make more accurate predictions across all classes in the dataset.

A deep neural network based malware detection system that Invincea has developed, achieves a usable detection rate at an extremely low false positive rate and scales to real world training example volumes on commodity hardware. Their system achieves a 95% detection rate at 0.1% false positive rate (FPR), based on more than 400,000 software binaries sourced directly from our customers and internal malware databases [28].

To generate predictions or classifications, DNNs meticulously process data through their interconnected layers of nodes. When applied to malware detection, these networks can be trained on extensive datasets containing both benign and malicious programs, equipping them with the knowledge needed to distinguish between the two categories. By synergizing DNNs with other machine learning techniques, such as Deep Autoencoders (DAEs) or Long Short-Term Memory (LSTM) networks, a more comprehensive and robust approach to malware identification can be achieved. Deep Autoencoders, as unsupervised models, play a pivotal role in feature learning and extraction, providing compact and meaningful representations of input data, particularly valuable in high-dimensional spaces like raw system call sequences or network traffic patterns. This enhances the model's robustness against various malware variants by capturing latent features and anomalies during pre-training on unlabeled data. On the other hand, Long Short-Term Memory Networks excel in capturing temporal dependencies and sequences, crucial for understanding dynamic aspects of malware behavior over time. Integrated into a DNN architecture, LSTMs contribute temporal context awareness, enabling the model to discern evolving patterns exhibited by sophisticated malware. Ensemble learning techniques, such as stacking or bagging, further amplify the model's robustness by combining the strengths of DNNs, DAEs, and LSTMs [28]. The ensemble approach leverages the diversity of information captured by each component, resulting in a more accurate and resilient model less sensitive to noise and outliers. Additionally, the utilization of transfer learning facilitates knowledge transfer from related tasks, such as feature learning or sequential modeling, enhancing the DNN's generalization performance in malware identification.

While leveraging DNNs has significantly propelled the field of malware detection towards greater accuracy and efficiency, their application comes with inherent limitations and challenges. Firstly, scalability issues pose a substantial hurdle, as training large-scale DNNs demands significant computational resources and can be financially burdensome. The complexity of DNN architectures, coupled with the extensive data required for effective training, exacerbates this challenge, particularly for organizations with limited computational capabilities. Secondly, interpretability challenges impede the widespread adoption of DNNs in malware detection in cyberspace. DNNs are often considered black-box models, lacking transparency in their decision-making processes. In intricate tasks like malware detection, understanding the rationale behind a specific decision is crucial for building trust and ensuring alignment with the expectations of security experts. Adversarial attacks constitute another formidable challenge. DNNs are susceptible to intentional manipulations of input data by malicious actors, leading to misclassifications and compromising the reliability of malware detection systems. Such attacks pose a significant security risk, requiring robust defenses to mitigate their impact [29]. Furthermore, the issue of data imbalance within malware datasets complicates the generalization performance of DNNs. Imbalances, where certain types of malware are underrepresented, can result in model biases towards prevalent classes, leading to suboptimal detection of less common or emerging malware variants. Lastly, the lack of explainability in DNNs' decision-making processes hinders their integration into security workflows. The opacity of these models makes it challenging for security analysts to comprehend the basis for a classification, impeding effective collaboration between automated systems and human experts.

3.5 Deep Belief Network

Deep Belief Networks (DBNs) are powerful tools in the realm of malware detection in cyberspace. These neural networks excel at capturing intricate patterns and features within vast datasets, making them invaluable for identifying malicious software. DBNs are particularly effective in analyzing software behavior and identifying anomalies or suspicious activities. They have the ability to autonomously discover relevant features, which is advantageous in the context of rapidly evolving malware. By processing various aspects of software behavior, such as system calls, API calls, or network traffic patterns, DBNs can differentiate between normal software operations and potentially harmful ones. This approach allows for the detection of previously unseen malware strains or novel attack techniques, making them a critical component of modern cybersecurity systems. The versatility and adaptability of DBNs in handling large and diverse datasets make them an essential tool for protecting against the ever-growing landscape of malware threats [2].

End-to-end deep learning architectures, specifically Bidirectional Long Short-Term Memory (BiLSTM) neural networks, are employed for the static behavior analysis of Android bytecode. Unlike conventional malware detectors that rely on handcrafted features, this system autonomously extracts insights from opcodes. This approach demonstrates the superiority of deep learning models over traditional machine learning methods, offering a promising solution to safeguard Android users from malicious applications [30].

Researchers have also explored the suitability of deep learning models for mobile malware detection. They utilize a deep neural network (DNN) implementation called DeepLearning4J (DL4J), which successfully identifies mobile malware with high accuracy rates. The study suggests that adding more layers to the DNN models improves their accuracy in detecting mobile malware, showcasing the feasibility of using DNNs for continuous learning and anticipating new types of attacks [22].

An anti-malware system that uses customized learning models, which are sufficiently deep, and are end to end deep learning architectures report an accuracy of 0.999 and an F1-score of 0.996 on a large dataset of more than 1.8 million Android applications [2]. The SOFS-OGCNMD system achieves system's average accuracy is 98.28%, average precision is 98.65%, recall is 98.53%, and F1-Score is 98.47 [22].

In addition, a method has been proposed to address the challenges of malware detection in Cyber-Physical Systems (CPS) within the Internet of Things (IoT). The model, called Snake optimizer-based feature selection with optimum graph convolutional network for malware detection (SOFS-OGCNMD), demonstrates remarkable results in accuracy, precision, recall, and F1-Score, outperforming recent models and contributing to the protection of CPS and IoT systems from evolving cyber threats [18].

Furthermore, a system has been designed to enhance the security of power systems through a Deep Belief Network (DBN)-based malware detection system. This system deconstructs malicious code into opcode sequences, extracts feature vectors, and utilizes DBN classifiers to categorize malicious code. It effectively utilizes unlabeled data for training and outperforms other classification algorithms in terms of accuracy. The research showcases the potential of DBNs for enhancing malware detection accuracy and reducing feature vector dimensions, thereby contributing to safeguarding power systems from cyber threats.

3.6 Deep convolutional neural network

The realm of malware detection in cyberspace, particularly in the context of evolving cyber threats, is experiencing a surge in innovative approaches driven by deep learning and convolutional neural networks (CNNs). Deep Convolutional Neural Networks (DCNNs) have emerged as robust and efficient technologies for detecting malware. Their ability to automatically extract complex features from various forms of data makes them exceptionally well-suited for this task. In the context of malware analysis, CNNs excel at processing and identifying malicious patterns within binary code, enabling the detection of known malware strains and even the discovery of novel threats. These networks are particularly effective in detecting malware through the analysis of file content and structure, which includes identifying suspicious code segments and unusual behaviors. Deep CNNs offer a significant advantage in terms of adaptability as they can be trained on diverse and evolving datasets to keep up with the continuous evolution of malware. Additionally, their capacity to handle large-scale data and discern subtle variations in binary files enables the identification of both prominent and more subtle malicious patterns. They are at the forefront of malware detection, contributing to the defense against the ever-growing sophistication of cyber threats [3].

DCNNs have proven to be highly effective in tasks related to image processing, excelling in capturing intricate spatial hierarchies and patterns. Their performance, often measured through precision, recall, and F1-score, is particularly notable in image classification tasks, especially when trained on extensive and diverse datasets. When compared to traditional machine learning models like SVM or Random Forest, DCNNs consistently outshine them in image-related tasks, showcasing a superior ability to discern complex patterns. Transfer learning models, such as VGG16 or ResNet, compete strongly with them, benefiting from pre-trained networks on large datasets. However, DCNNs, especially those incorporating transfer learning architectures, often emerge as leaders, demonstrating heightened precision, recall, and F1-score by leveraging their effective feature extraction capabilities. In tasks involving sequential data, where Recurrent Neural Networks (RNNs) excel in capturing temporal dependencies, DCNNs maintain their superiority in scenarios where spatial features hold more significance, as seen in image-related tasks. Ensemble models, combining various techniques, present a competitive alternative, sometimes matching or exceeding DCNNs' performance, particularly in cases where diverse models contribute to improved generalization. These models find a significant application in the financial sector, particularly in credit scoring. In credit scoring, financial institutions aim to predict the probability of a loan applicant defaulting on a loan. This prediction is based on a myriad of factors including credit history, income, employment status, and others. Ensemble models combine various machine learning models like Decision Trees, Logistic Regression, and Neural Networks to assess these factors. In the real world, this translates to more accurate credit scoring, which helps financial institutions in reducing the risk of loan defaults while approving more loans for credit-worthy applicants.

The evaluation of DCNNs on diverse datasets is indispensable for gauging their generalizability and robustness across various malware types. Assessing these models on multiple datasets provides crucial insights into their adaptability and real-world performance. Key considerations include exploring malware variability, addressing imbalances in datasets, accounting for temporal aspects in malware evolution, cross-domain evaluation to assess adaptability, examining scenarios involving transfer learning, evaluating resilience against adversarial attacks, accounting for geographical variations in malware prevalence, ensuring versatility in handling different feature representations, and maintaining consistent evaluation metrics such as precision, recall, F1-score, and area under the ROC curve. This comprehensive approach to evaluation enables researchers and practitioners to develop DCNN models that can effectively navigate the dynamic and complex landscape of malware detection, ensuring their efficacy across diverse and evolving cybersecurity scenarios.

An advanced intelligent IoT malware detection model proposed based on deep learning and ensemble learning algorithms, called DEMD-IoT achieves the best outcome with the highest accuracy 99.9%, compared to state-of-the-art machine learning, deep learning, and ensemble models [3]. The 4L-DNN model outperforms other DNN architecture by a significant margin on the accuracy metric, where 98.85% was achieved on both datasets, Benign and Malicious PE Files dataset and Malware Dataset, and 98.37% was achieved on the Classification of Malwares dataset [31].

To address the challenges related to malware detection, the DEMD-IoT model leverages the power of deep learning and ensemble learning techniques. It comprises a stack of three one-dimensional convolutional neural networks (1D-CNNs) tailored to analyze IoT network traffic patterns. The model also features a meta-learner, utilizing the Random Forest algorithm, to integrate results and produce the final prediction. DEMD-IoT's advantages lie in its ensemble strategy to enhance performance and the use of hyperparameter optimization to fine-tune base learners. Notably, it employs 1D-CNNs, avoiding the complexity of preprocessing phases. Empirical evaluation on the IoT-23 dataset demonstrates that this ensemble method outperforms other models, achieving a remarkable accuracy of 99.9% [31].

Another study introduces a web-based malware detection system centered on deep learning, specifically a one-dimensional convolutional neural network (1D-CNN). Unlike traditional methods, it focuses on static features within portable executable files, making it ideal for real-time detection. The 1D-CNN architecture, tailored for these executable files, facilitates efficient feature extraction. Comparisons with state-of-the-art methods across diverse datasets confirm the model's superiority. As malware poses a significant security threat, this web-based system offers user-friendly malware detection, reducing vulnerability to cyberattacks and benefiting individuals and organizations alike. The study emphasizes the importance of deploying deep learning models in web-based applications to enhance usability and accessibility [32].

In another paper, the authors propose an efficient neural network model, EfficientNetB1, for classifying malware families using image representations of malware at the byte level. By employing computer vision techniques, they aim to detect sophisticated and evolving malware. The evaluation of various pretrained CNN models highlights the importance of minimizing computational resource consumption during training and testing. EfficientNetB1 achieves an impressive accuracy of 99% in classifying malware classes, requiring fewer network parameters compared to other models. This work contributes to the field of cybersecurity by providing a novel approach that combines efficient neural network models and diverse image representation methods for accurate and resource-efficient malware classification [33].

To address the persistent challenge of malware detection in Windows systems, a Convolutional Neural Network (CNN)-based approach is used. It leverages the execution time behavioral features of Portable Executable (PE) files to identify and classify elusive malware. The approach was evaluated using a dataset comprising MIST files, generating images from N-grams selected by various Feature Selection Techniques. Results from 10-fold cross-validation tests showcase the remarkable malware detection accuracy, particularly when employing N-grams recommended by the Relief Feature Selection Technique. In comparison to other machine learning-based classifiers, this CNN-based approach outperforms them, offering a promising solution to enhance malware detection in Windows systems.

3.7 Deep generative models

Deep Generative Models offer a promising avenue for enhancing malware detection techniques in cyberspace. These models operate by generating synthetic data that mimics the characteristics of malicious code, thereby providing an innovative approach to detect malware. By leveraging techniques such as Variational Autoencoders (VAEs) or Generative Adversarial Networks (GANs), deep generative models can create artificial malware samples to diversify training datasets. This augmentation helps improve the robustness of malware detection systems, enabling them to recognize new and evolving threats. These models can also be employed in anomaly detection, identifying deviations from normal software behavior, which often indicates the presence of malware. Furthermore, they can generate features that enhance feature-based malware detection in cyberspace. Their adaptability and ability to generate data like malicious code samples contribute to strengthening the overall cybersecurity landscape, offering a proactive approach to identifying and combating malware threats [34].

In contrast, state-of-the-art methods and alternative approaches encompass signature-based detection, heuristic-based detection, and traditional machine learning models. Signature-based methods are efficient in identifying known malware through predefined patterns but face limitations in detecting novel threats. Heuristic approaches rely on rules and behavioral patterns, demonstrating adaptability but may produce false positives or negatives. Traditional machine learning models, while interpretable and computationally efficient, are constrained by the need for manual feature engineering and may struggle with high-dimensional data. Interpretability favors signature-based and heuristic-based methods, as well as certain traditional machine learning models, over deep generative models. However, the adaptability to novel threats is shared by deep generative models and heuristic-based approaches, distinguishing them from the limitations of signature-based methods. Deep generative models, with their strengths in unsupervised learning and anomaly detection, offer a promising avenue for addressing challenges posed by evolving and novel malware threats.

The two features extracted from the data with their respective characteristics are concatenated and entered into the malware detector of a hybrid deep generative model. By using both features, the proposed model achieves an accuracy of 97.47%, resulting in the state-of-the-art performance. [34]. A model which was verified by extensive experiments on the benchmark datasets KDD'99 and NSL-KDD effectively identifies normal and abnormal network activities. It achieves 99.73% accuracy on the KDD'99 dataset and 99.62% on the NSL-KDD dataset [35].

To tackle the challenge of detecting obfuscated malware, which often employs techniques like null value insertion and code reordering to evade traditional detection methods, a deep generative model is proposed. This model combines both global and local features by transforming malware into images to capture global characteristics efficiently and extracting local features from binary code sequences. By fusing these two types of features, the model achieves an impressive accuracy of 97.47% [36]. A novel approach is also introduced that leverages generative adversarial networks (GANs) for plausible malware training and augmentation. By training a discriminator using malware images generated by GAN models, the framework enhances the robustness of detection against 0-day malware. This eliminates the need for inefficient malware signature analysis, reducing signature complexity. The study emphasizes the importance of understanding 0-day malware features through explainable AI techniques and suggests future work on expanding the framework's applicability [35]. Despite the inherent black-box nature of these models, Explainable AI (XAI) provides a set of methodologies to shed light on their decision-making processes. Layer-wise Relevance Propagation (LRP) assigns relevance scores to input features, aiding in the identification of crucial patterns for 0-day malware features. Saliency maps highlight significant regions in input data, offering interpretability by emphasizing key areas in images or sequences. Integrated Gradients calculates feature attribution, providing nuanced insights into how variations in input features contribute to the identification of 0-day malware characteristics. Local Interpretable Model-agnostic Explanations (LIME) generates faithful interpretations by perturbing input instances, creating surrogate models for better understanding. Attention mechanisms focus on relevant parts of input sequences, aiding in the interpretation of the importance of different elements, particularly beneficial for sequential data. Counterfactual explanations generate alternative instances, showcasing the impact of input feature variations on model predictions, enhancing understanding of 0-day malware identification. Rule-based explanations extract decision rules approximating the behavior of deep generative models, offering a simplified representation for accessibility and understanding by security analysts. These explainability methods collectively contribute to a more transparent and interpretable framework, allowing analysts to dissect and comprehend the decision-making processes of deep generative models in the complex domain of 0-day malware detection.

To tackle network intrusion detection, where high-dimensional data, the scarcity of labeled samples, and real-time detection pose challenges, the proposed solution utilizes deep learning. It employs a multichannel Simple Recurrent Unit (SRU) model that outperforms traditional LSTM algorithms in efficiency and accuracy. To address the scarcity of labeled samples, a generative adversarial model (DCGAN) is used to generate training data, significantly improving system detection rates and reducing false alarms. The paper introduces efficient data preprocessing and demonstrates an impressive detection accuracy of 99.73% on KDD datasets. The SRU-based approach offers real-time intrusion detection capabilities and enhances network security [20].

While offering unique strengths, Deep Generative Models also come with several drawbacks in the context of malware detection in cyberspace. One key limitation lies in their interpretability, as these models are often perceived as black-box systems, making it challenging for security analysts to comprehend and trust their decision-making processes. Moreover, the computational intensity required for training, stemming from complex architectures and large datasets, poses practical challenges for deployment, particularly in resource-constrained environments. Data dependency is another drawback, with deep generative models relying on substantial amounts of labeled data for effective training. Acquiring diverse and representative datasets for various malware types can be logistically challenging, considering the dynamic nature of the cybersecurity landscape. Additionally, these models are vulnerable to adversarial attacks, similar to other deep learning approaches, which pose a threat to their reliability in real-world scenarios. The need for large-scale training data is a practical concern, as optimal performance often hinges on access to extensive and diverse datasets. Adapting to the dynamic nature of cybersecurity threats is another limitation, requiring frequent updates and retraining to effectively address new malware variants. Incorporating domain knowledge or expert-defined rules into the learning process can be difficult for deep generative models, hindering their ability to leverage human expertise in refining malware detection.

3.8 Deep Boltzmann machine

Deep Boltzmann Machines (DBMs) are powerful tools in malware detection in cyberspace. These deep learning algorithms excel in capturing intricate patterns within large datasets. When used for malware detection, they analyze binary code or behavioral data to identify malicious patterns and anomalies. By modeling the complex relationships between features, they effectively distinguish between benign and malicious software. DBMs offer the advantage of unsupervised learning, making them adept at uncovering novel and previously unseen malware variants. They can identify subtle and evolving threat vectors, making them crucial in the battle against constantly changing malware. Additionally, DBMs can be used for feature extraction, reducing data dimensionality and enhancing the efficiency of other detection algorithms.

Compared to other malware detection techniques, including traditional machine learning algorithms, Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs), each approach brings its unique attributes. Traditional algorithms are known for their interpretability but may require manual feature engineering. CNNs excel in spatial feature extraction for image-based tasks. RNNs outperform DBMs in handling sequential data and capturing temporal dependencies.

In the field of cybersecurity, DBMs play a pivotal role in bolstering defenses and ensuring the early identification of emerging malware threats [37]. A multi-objective RBM model aims to improve robustness and data classification accuracy. This study addresses challenges such as dataset imbalance, complex deep learning network models, and the need for multiple objectives. It leverages non-dominated sorting genetic algorithms (NSGA-II) to tackle imbalanced malware families. The proposed model, in conjunction with NSGA-II, significantly enhances data classification accuracy within HetNets, demonstrating its effectiveness in safeguarding data fusion processes [38].

To tackle dimensionality, Subspace-based Restricted Boltzmann Machines (SRBM) introduce a novel approach that combines RBMs with subspace learning. SRBM efficiently reduces feature dimensionality while considering non-linear feature relationships. Compared to other methods like PCA and Stacked Auto Encoder (SAE), SRBM stands out with significant improvements in performance metrics, enhancing efficiency and accuracy in Android malware detection [4].

To explore the application of deep learning in the detection of Denial of Service (DoS) attacks, a deep Gaussian-Bernoulli-type RBM is introduced with additional layers, optimizing hyperparameters for improved detection accuracy. This deep RBM model supports continuous data and demonstrates superior accuracy when compared to alternative RBM models, such as Bernoulli-Bernoulli RBM. The study underscores the importance of developing systems capable of detecting malicious behavior within network traffic, particularly in the context of DoS attacks [39].

In order to confirm the effect of the proposed method (RBM + NSGA-II) on the accuracy of data classification, the recall rate values with five other methods are compared. The methods are GIST + KNN, GIST + SVM, GLCM + KNN, GLCM + SVM, and DRBA, and the recall rates are 91.7, 91.4, 92.3, 93, and 94.5 percent, respectively [37]. The method proposed here has a recall rate of 95.83 percent. Next, by comparing the values of loss, recall rate, and false alarm rate with, it was found that the proposed multi-objective RBM model has loss values (loss = 0.083, 0.080, and 0.086) and recall rate values (recall = 88.64, 93.48, and 95.83 percent) are all better in three different resolutions, and the value of FPR at 50 × 50 resolution is slightly worse (FPR = 12.5 percent is greater than 11.50 percent) [37].

Obtaining and labeling appropriate training data for Deep Boltzmann Machines (DBMs) in malware detection presents multifaceted challenges with implications for model generalizability and real-world applicability. Firstly, imbalances in class distribution within malware datasets pose a challenge, potentially leading to biased model training and diminished effectiveness in detecting less common malware types. Annotating malware samples is resource-intensive, and the dynamic cybersecurity landscape introduces new variants regularly, contributing to limitations in dataset size and timeliness. This can hinder the model's capacity to generalize to evolving threats. Moreover, inherent biases in malware datasets from different sources create potential limitations. Models trained on biased datasets may struggle to generalize across different contexts, impacting performance when faced with malware variants from underrepresented sources or regions. The active involvement of malicious actors in crafting adversarial samples further complicates the training process. Adversarial samples, intentionally manipulated to deceive the model, can compromise the robustness and reliability of the DBM in real-world scenarios. Additionally, the heterogeneous nature of malware, ranging from simple to highly sophisticated attacks, presents a challenge in capturing this diversity within a single training dataset. A lack of diversity may result in a model that struggles to identify novel and sophisticated malware types, further constraining its efficacy in practical, real-world scenarios.

3.9 Deep reinforcement learning

Deep reinforcement learning (DRL) plays a crucial role in enhancing malware detection by introducing innovative approaches to address evolving cybersecurity challenges. This advanced technique utilizes artificial intelligence and deep learning algorithms to train intelligent agents that learn to make decisions based on interactions with malware samples. These agents can determine optimal sequences of actions to modify malware, making it more difficult for anti-malware engines to detect. DRL is particularly effective in scenarios where traditional machine learning approaches struggle, especially in dealing with adversarial attacks. By allowing the agents to iteratively interact with malware, it is possible to enhance the agility and evasiveness of malware, making detection more challenging. This approach empowers researchers and cybersecurity professionals to proactively combat cyber threats, adapt to new evasion techniques, and continuously strengthen their malware detection systems [40].

In comparison to established methods, signature-based detection techniques prove effective in identifying known malware patterns, offering computational efficiency and a well-established presence in cybersecurity practices. Heuristic-based approaches leverage rules and behavioral patterns, adapting to new threats through heuristic updates. While computationally efficient, heuristics may generate false positives or negatives based on predefined rules. Traditional machine learning models, such as Support Vector Machines (SVMs) or Random Forests, provide interpretability and efficiency but may struggle with complex relationships in data due to their reliance on manual feature engineering. Analysis of these approaches reveals the superiority of DRL techniques in sequential decision-making tasks and adaptability to dynamic environments, addressing limitations seen in signature-based and traditional machine learning methods.

One method explores the evolution from traditional signature-based methods to machine learning-based algorithms for malware detection. While machine learning approaches have significantly improved detection accuracy, they remain vulnerable to adversarial attacks. The study delves into the creation of adversarial samples to test the resilience of these systems, particularly focusing on binary file modification. It discusses the complexities involved in avoiding corruption of the binary and the need to strengthen the defenses of machine learning models. The research highlights the ongoing need to enhance the robustness of malware classifiers against adversarial attacks. Another study introduces a novel framework called DQEAF (Deep Q-Learning for Evading Anti-Malware Engines), which employs DRL to bypass anti-malware engines. This framework trains an artificial intelligence agent to iteratively interact with malware samples and determine optimal sequences of non-destructive actions that modify the samples, enabling them to evade detection. The study emphasizes the effectiveness of this approach, achieving a 75% success rate in evading detection by anti-malware engines, particularly in the context of Portable Executable (PE) samples [40].

Alternative approaches delve into network security and leverage Software-Defined Networking (SDN) to optimize traffic analysis through Deep Packet Inspection (DPI). One such approach utilizes deep reinforcement learning, specifically Deep Deterministic Policy Gradient (DDPG), to intelligently allocate sampling resources in SDN-capable networks. The goal is to capture malicious network flows while minimizing the load on multiple traffic analyzers. The study showcases the efficacy of this approach in achieving more efficient traffic monitoring and cyber threat detection, highlighting the importance of data-driven decisions in traffic sampling [41]. Additionally, the vulnerability of a leading malware classifier to dead code insertion is explored, and a framework employing deep reinforcement learning, specifically a Double Q-network, is introduced to induce misclassification in the classifier. An intelligent agent, trained through a convolutional Q-network, strategically inserts NOP instructions into malware code sequences. The results demonstrate a significant reduction in the classifier's accuracy, showcasing the potential for evasion using the dead code insertion technique.

One of the primary performance metrics of DRLs lies in the rewards earned over time, depicting the agent's learning progress by maximizing cumulative rewards through interactions with an environment. The reported values for the performance comparison were an average of 500 iterations with a 95% confidence interval. Parameter values used for network topologies. traffic steering overheads than other methods while maintaining a load-balancing of traffic analyzers over 88% [41].

DRL models exhibit remarkable adaptability to new and unknown malware samples, making them valuable assets in the ever-changing landscape of cybersecurity. Their adaptability arises from the models' ability to learn optimal strategies through dynamic interactions with their environment, mirroring real-world cybersecurity scenarios effectively. In the realm of malware detection, DRL models such as Deep Q Networks (DQN) or Proximal Policy Optimization (PPO) excel in sequential decision-making tasks. This capability proves vital in scenarios where the identification process involves a series of actions and responses, enabling these models to learn optimal sequences of actions for effective detection and response to emerging threats. The models exhibit a remarkable feature learning capability, automatically extracting relevant patterns from raw input data. This reduces the reliance on predefined features or signatures, facilitating adaptability as the models can discover novel patterns associated with new malware samples without explicit feature engineering.

DRL models shine in 0-day threat detection, showcasing their prowess in identifying previously unseen and unknown threats. By learning from the dynamics of the environment and comprehending the underlying patterns of normal and malicious behavior, DRL models can adeptly adapt to emerging threats that lack historical data or predefined signatures. The support for continuous learning allows the models to stay current with the evolving threat landscape, ensuring they can effectively counter emerging risks. Reinforcement learning agents within DRL can dynamically adjust their policies based on feedback from the environment, enabling the model to update its knowledge as it encounters new malware samples. This dynamic policy adjustment significantly enhances the model's ability to handle unknown threats effectively.

Tackling data imbalance is pivotal for effective model training. One key strategy involves reward balancing, a technique aimed at adjusting reward mechanisms to address the imbalance between minority and majority classes. This approach seeks to ensure that the learning process does not disproportionately favor the majority class while neglecting the minority. By fine-tuning the reward system, the algorithm can be guided to allocate appropriate attention to underrepresented scenarios, encouraging the model to learn from these instances as rigorously as from the dominant ones. This balance fosters a more comprehensive understanding of the environment, enabling the reinforcement learning agent to make informed decisions across diverse situations. By strategically adjusting reward structures, DRL algorithms can overcome data imbalance challenges, ultimately enhancing their adaptability and performance in complex real-world scenarios.

While this model holds promise for malware detection in cyberspace, its practical use faces notable challenges. High computational requirements, especially for complex models like Deep Q Networks and Proximal Policy Optimization, pose a constraint, particularly in resource-constrained environments. Additionally, the demand for substantial training data raises concerns about data efficiency, affecting performance when labeled malware samples are limited. Lengthy training times of DRL models, particularly deep neural networks, can hinder timely deployment in dynamic cybersecurity scenarios. The black-box nature of DRL models presents interpretability challenges, making it difficult to understand the decision-making processes and the features crucial for malware detection in cyberspace. Moreover, sample inefficiency, sensitivity to hyperparameters, and difficulties in generalizing across diverse malware variants further limit the effectiveness of DRL. Vulnerability to adversarial attacks adds another layer of concern, as intentional manipulations could compromise the reliability of the model. Deploying DRL models at scale in complex network environments requires addressing scalability challenges. Ethical considerations, especially regarding privacy and potential misuse, necessitate compliance with regulatory frameworks for responsible deployment. Balancing these challenges is crucial for unlocking the full potential of DRL in the realm of cybersecurity.

3.10 Extreme Learning Machine

Extreme Learning Machine (ELMs) are increasingly used in malware detection due to their versatility and efficiency. ELMs excel in feature extraction, making them suitable for processing various data types crucial for malware analysis. Their single hidden layer with randomized weight assignments enables them to process many features quickly, which is beneficial for comprehensive malware detection in cyberspace. While this allows for quick processing of many features, it may not capture complex relationships and dependencies in the data as effectively as models with multiple hidden layers and optimized weight assignments. ELMs are particularly favored for their fast-training process, as they do not involve iterative weight optimization. This speed and their ability to handle diverse data types make ELMs a valuable tool in the ongoing battle against malware [42].

While this may be beneficial for efficiency, it could also limit the model's ability to fine-tune and improve its performance over time. Iterative weight optimization techniques, such as backpropagation, are commonly used in other machine learning models to refine the model's predictions and achieve higher accuracy. Research in this field addresses the pressing challenges posed by malware, with a primary focus on improving accuracy, automation, and efficiency in the detection process. Data Imbalance can pose a significant problem in ELM training but several strategies can help address this challenge. Resampling techniques like oversampling or undersampling methods are effective approaches. Oversampling involves increasing the instances of the minority class, while undersampling reduces the instances of the majority class, aiming to balance the dataset's representation. This helps prevent the model from being biased toward the majority class. Another valuable strategy is weighted learning, where different weights are assigned to samples based on their class. By assigning higher weights to minority class samples and lower weights to majority class samples, the learning process becomes more balanced, allowing the model to better discern patterns from the less represented class.

Notably, one of these studies introduces an innovative approach in the form of a Two-hidden-layered Extreme Learning Machine (TELM), which departs from conventional backpropagation techniques to offer a streamlined and faster approach to malware detection in cyberspace. This approach incorporates dependencies of malware sequence elements, effectively enhancing the accuracy of classification while dramatically reducing both training and detection time. The practical implications of this study are profound, particularly in safety-critical systems such as healthcare and the Internet of Things (IoT), where rapid and reliable malware detection is imperative [43].

While the Two-hidden-layered Extreme Learning Machine (TELM) and the Gauss-Mapping Black Widow Optimization with Deep Learning Enabled Android Malware Classification (GBWODL-AMC) demonstrate efficacy in malware detection, they exhibit specific limitations and trade-offs. TELM, characterized by its two-hidden-layer extreme learning architecture, faces challenges in interpretability due to the inherent complexity of deep learning models. The model's decision-making processes might be challenging to decipher, potentially impacting the trust users place in its outputs. Additionally, TELM's performance is contingent on the availability of sufficient labeled training data, making it susceptible to constraints in scenarios where obtaining diverse datasets is difficult. The computational intensity of training TELM models, especially with larger datasets and intricate architectures, can pose challenges in resource-constrained environments.

On the other hand, GBWODL-AMC, which integrates Gauss-Mapping Black Widow Optimization, introduces its own set of limitations. The model's effectiveness is tied to the appropriateness of the chosen optimization technique, and its dependency on this specific strategy may limit its applicability across diverse problem domains. Hyperparameter sensitivity poses a trade-off, requiring careful experimentation to select optimal values and avoid reduced model performance. Deep learning components within GBWODL-AMC are susceptible to potential overfitting, especially when dealing with complex datasets, which may hinder the model's generalization to new, unseen data. Similar to TELM, the model's interpretability may be compromised due to the inherent complexity of deep learning architectures.

It is common for Android malware to employ code obfuscation techniques to evade detection. In response, a cutting-edge model, the Gauss-Mapping Black Widow Optimization with Deep Learning Enabled Android Malware Classification (GBWODL-AMC), is introduced. This model combines novel feature selection techniques with deep extreme learning, and through meticulous parameter optimization, it achieves a remarkable accuracy rate of up to 98.95%. The significance of this research extends to the realm of mobile device security, providing a promising solution for more effectively combating Android malware.

Another technique delves into the critical domain of detecting obfuscated malware within network traffic. It introduces the MalHyStack hybrid classification model, a powerful fusion of machine learning algorithms and deep learning [44]. This model, through the incorporation of feature subset selection and a balanced dataset, achieves exceptional accuracy rates that surpass existing models. The broader implication of this research is evident in its ability to combat obfuscated malware efficiently while maintaining a high degree of accuracy.

In comparing Extreme Learning Machines (ELMs) and Convolutional Neural Networks (CNNs) for malware detection across a broader range of datasets, each approach exhibits distinct strengths and weaknesses. ELMs are characterized by their fast training times, simplicity, and non-iterative training, making them efficient for large datasets and resource-constrained scenarios. However, ELMs may face challenges in capturing hierarchical features and may require manual feature engineering, limiting their suitability for complex data, such as images or sequences. On the other hand, CNNs excel in spatial feature extraction, hierarchical representation learning, and end-to-end learning, making them particularly effective for image-based malware detection tasks [42]. Their ability to automatically learn hierarchical representations from raw data eliminates the need for extensive manual feature engineering. Nonetheless, CNNs come with computational intensity during training, interpretability challenges due to their black-box nature, and a dependency on large labeled datasets, posing challenges in data acquisition. When applied to image-based malware detection, ELMs may perform well under resource constraints, while CNNs are likely to outperform ELMs due to their proficiency in spatial feature extraction. In handling sequential data like API calls or network traffic, ELMs may struggle with temporal dependencies and might require additional feature engineering, whereas CNNs, with modifications like 1D convolutions or recurrent layers, offer a more robust performance. For multimodal data encompassing a combination of images and sequences, ELMs may need careful feature engineering, while CNNs, capable of processing both types of data, provide a more comprehensive solution. In terms of generalization across diverse malware datasets, ELMs might face challenges, especially with complex hierarchical features. In contrast, CNNs, with their inherent capacity for hierarchical representation learning, demonstrate potential for better generalization across a broad spectrum of malware datasets [43].

In the realm of malware detection, enhancing the interpretability of deep learning models holds immense importance for establishing trust, comprehending model predictions, and gaining insights into classification decisions. To achieve this, several key approaches have emerged. Feature visualization techniques enable the understanding of the specific characteristics or patterns that the model identifies as indicative of malware. Techniques like activation maximization or gradient-based methods visualize salient features, such as sequences of system calls or network traffic patterns. Saliency maps, generated through methods like Grad-CAM, highlight crucial regions in the input data that influence the model's output, shedding light on the importance of different input features in model decisions. Attention mechanisms, prevalent in models like transformers, aid in understanding the model's processing of sequential data by visualizing attention weights, indicating the elements of input sequences that hold more significance [44]. Transforming complex models into interpretable rule-based systems, accomplished through rule extraction algorithms or decision tree induction, simplifies model logic and facilitates comprehension. Layer-wise relevance propagation helps attribute relevance to input features by discerning the contribution of different model layers to the final decision. Model distillation aims to simplify complex models while retaining performance, training smaller, more interpretable models to mimic the behavior of intricate deep learning models. Techniques like Integrated Gradients or SHAP values quantify the impact of each input feature on model output, providing a clear understanding of feature importance. Moreover, domain-specific visualization tools tailored for malware analysis offer interactive dashboards or tools for security analysts, enabling intuitive exploration of model decisions and deeper insights into malware behavior.

3.11 Attention models

In the ongoing battle against malware, traditional detection methods are struggling to keep pace with the constant innovation of cybercriminals. Attention models, a powerful deep learning technique, are emerging as a game-changer. These models do not treat all aspects of a file equally; instead, they learn from vast datasets of malware and benign software to identify the most critical features–the red flags that scream "malicious." The polymorphism in malicious components has deteriorate the situation, as malicious files, which essentially belong to the same malware "family" and have the same form of malicious behavior, are constantly modified, or obfuscated using various strategies to make them appear to be many different files [45]. By focusing on these key elements, attention models can achieve higher accuracy in detecting both known and unknown malware strains, while also reducing the number of innocent files mistakenly flagged as threats. This ability to adapt and learn makes them invaluable in the fight against ever-evolving cyber threats. Traditional methods often suffer from a high rate of false positives, mistakenly quarantining harmless files. Existing gray image based malware detection and classification approaches are primarily based on conventional machine learning or deep learning with Convolutional Neural Networks (CNNs). GIST + kNN pioneers the application of machine learning in malware classification on the Malimg dataset. Subsequent studies relied on features extracted from PCA features, N-gram application programming interface (API) sequences, opcodes, control flow graphs, text semantics of network traffic and URLs, system calls, OS-level information flow and the network activities of the malware. While these advancements have broadly elevated the field, they require manual feature design, deep foundational knowledge, and even the construction of complex network system environments for detection and classification [45]. Attention models, by placing less weight on irrelevant features, can significantly reduce these false alarms. This translates to less wasted time and resources for security teams, allowing them to focus on genuine threats. Additionally, some attention models offer a degree of explainability. They can provide insights into why a particular file was classified as malware, helping security professionals understand the attacker's techniques and potentially identify vulnerabilities that need to be addressed.

The attention-based feature extraction method allows malicious code analysts to only analyze parts of malicious code based on the features extracted by the attention-based feature extraction method, rather than analyzing the entire malicious code. This is expected to considerably reduce the efforts required by malicious code analysts [46]. An implementation of the ARI cell with LSTM networks, called ARI-LSTM enhances the LSTM cell by incorporating ARI mechanism within the cell, and use sthe resulting neural network for sequence learning with ransomware. Through evaluation on a ransomware dataset for the Windows operating system environment, it is seen that ARI-LSTM improves the performance of an LSTM in detecting ransomware from emulation sequences [47].

Cross-dataset experiments conducted on the Windows and Android datasets, with an accuracy of 90.64% on cross-dataset detection of the android [45]. The attention-based model yielded an accuracy that was approximately 12% and 5% higher than those of the CNN-based and SC-LSTM-based models, respectively [46].

However, attention models are not without their challenges. Training these complex algorithms requires significant resources. Large, diverse datasets of malware samples are essential for them to learn and adapt effectively. Additionally, the computational cost of training and deploying these models can be substantial. Finally, while some models offer explanations, their inner workings can be intricate, requiring expertise to fully comprehend. Despite these challenges, the potential of attention models is undeniable. Their ability to learn, adapt, and focus on the most critical features makes them a powerful weapon in the fight against malware. As these models continue to evolve and become more accessible, they hold the promise of a future where cyber defenses are more agile and effective, constantly learning and adapting to the ever-changing threat landscape.

3.12 Summary and interpretability of deep learning models

Choosing the optimal model for malware detection hinges on several factors, including dataset characteristics, feature requirements, and performance expectations. Among the considered models, each possesses unique strengths. Recurrent Neural Networks (RNNs) excel in capturing temporal dependencies, making them suitable for sequential data. Deep Autoencoders prove effective in learning hierarchical representations, particularly for anomaly detection. Long Short-Term Memory (LSTM) networks, designed for sequential data, demonstrate prowess in handling long-term dependencies. Deep Neural Networks (DNNs) are versatile, capable of learning complex non-linear mappings. Deep Belief Networks (DBNs) are adept at unsupervised learning and hierarchical representation learning.

Deep Convolutional Neural Networks (CNNs) are well-suited for image-based data, capturing spatial hierarchies effectively. Deep Generative Models can generate new samples, aiding in understanding data distribution. Deep Boltzmann Machines are suitable for unsupervised learning and complex dependency modeling. Deep Reinforcement Learning is designed for tasks involving agent-environment interactions and policy learning. Extreme Learning Machines (ELMs) stand out for their fast training, simplicity, and good generalization.

For malware detection, a combination or ensemble approach may prove effective. Models like CNNs can extract features from binary files or images, while RNNs or LSTMs capture temporal dependencies in malware behavior. Unsupervised learning models like autoencoders or DBNs can aid in anomaly detection, identifying novel malware patterns. Experimentation and evaluation on specific datasets are crucial to determine the most effective model, and regular updates are essential to adapt to evolving malware threats.

4 Comparisons with non deep learning models

In the realm of malware detection, traditional non-deep learning methods like signature-based detection heavily rely on predefined patterns, making them susceptible to 0-day threats. Heuristic-based approaches utilize rules but can struggle to adapt to evolving tactics. Behavioural analysis, while effective, faces scalability issues and might overlook subtle anomalies. Non-deep learning machine learning algorithms such as Decision Trees and SVMs demand expert feature engineering and struggle with high-dimensional data complexity [7].

In contrast, deep learning models provide a range of distinct advantages over these traditional methods. They autonomously learn intricate features from raw data, bypassing the limitations of handcrafted features in traditional methods. Their adaptability to diverse data types and capacity to generalize to new, unseen malware variants outshine the rigidity of traditional approaches, often reliant on frequent updates [8]. Deep learning excels in capturing complex patterns and relationships, particularly in large-scale datasets, surpassing traditional methods in nuanced pattern recognition tasks.

Furthermore, deep learning's scalability and automation in handling large datasets streamline feature extraction, whereas traditional methods may encounter scalability limitations. However, while traditional methods often boast interpretability due to explicit rule-based decisions, deep learning models' complex architectures render them less interpretable, though ongoing efforts aim to enhance interpretability through emerging techniques. Additionally, while deep learning can learn from imbalanced data, it requires specific strategies to effectively manage class imbalance, a challenge that traditional methods also encounter, often necessitating sophisticated sampling or weighting techniques [6, 10–12].

5 Smartphone applications in malware analysis and detection

Smartphone applications have emerged as valuable assets in the field of malware analysis and detection. Numerous tools have been designed for both Android and iOS platforms, leveraging the computing power and connectivity of smartphones to enhance the capabilities of security professionals and organizations. These applications play a pivotal role in scanning and analyzing mobile apps for known malware signatures, identifying behavioral anomalies and vulnerabilities. This aids in the early detection of potentially harmful applications and helps prevent device compromise [48].

One key feature of these applications is real-time monitoring, which keeps a vigilant eye on network traffic, system activity, and app behavior. This continuous surveillance is crucial for identifying suspicious or malicious activities on mobile devices, enabling prompt alerts to users or administrators about unusual behaviors or interactions with known malicious domains. Integration with Mobile Device Management (MDM) solutions further enhances the functionality of these applications. MDM allows organizations to manage and secure mobile devices remotely, enforce security policies, deploy updates, and, if necessary, remotely wipe compromised devices. This integration is particularly beneficial for enterprises looking to safeguard their mobile device ecosystem [49].

Furthermore, some of these smartphone applications incorporate threat intelligence feeds, providing access to the latest information on mobile-specific threats and indicators of compromise. This integration significantly improves detection and response capabilities against emerging threats. Another aspect of these applications is app reputation scanning [50]. They assess mobile applications based on various factors, including the source, required permissions, and code behavior. This enables users and administrators to make informed decisions about app installation and usage.

Behavioral analysis is another advanced feature offered by some applications, where they monitor the interactions of mobile apps with device resources and the network. This method effectively unveils hidden or obfuscated malicious activities that may not be evident through static analysis alone. Additionally, there is a focus on user education within these applications. They provide tips and information about potential security risks and best practices for safe mobile device usage, empowering users to become more aware and vigilant regarding their security [51].

These applications often include the capability to detect rooting or jailbreaking of devices. Rooted or jailbroken devices are more susceptible to security risks, and detecting such modifications is crucial for alerting users and administrators to potential tampering or compromise. The contribution of smartphone applications to malware analysis and detection is increasingly significant, especially as mobile devices become more prevalent targets for cybercriminals. These applications empower users and organizations to proactively defend their smartphones and the sensitive data they contain. As the mobile threat landscape evolves, the importance of these applications in ensuring mobile security and privacy becomes even more critical [52].

6 Recent tools in malware analysis and detection

Machine learning has become a popular approach for malware detection due to its proficiency in identifying patterns and anomalies in large datasets [53–55]. Various algorithms, such as Random Forest, Support Vector Machines, and Neural Networks, are employed to analyze features extracted from executable files, including binary code, API calls, and file metadata, effectively detecting malware. Dynamic analysis tools, which involve executing malware in a controlled environment to observe its behavior, have also seen significant advancements. Modern tools offer capabilities like automated sandboxing and advanced code instrumentation, allowing for real-time monitoring of system and network activities and analysis of the malware's actions. Examples of these tools include Cuckoo Sandbox, Any. Run, and Hybrid Analysis.

Behavioral analysis is another critical area, focusing on how malware behaves upon execution. By using advanced techniques to detect abnormal behavior patterns, such as process injection and privilege escalation, these tools can identify malicious actions, enabling the detection of previously unknown malware. Memory forensics tools have become increasingly sophisticated, with tools like Volatility enabling analysts to extract and analyze information from a system's RAM. These tools are crucial for uncovering hidden processes, rootkits, and other memory-resident malware. YARA rules have gained popularity for creating custom patterns to identify specific malware characteristics. These rules, defined by security professionals, are instrumental in both static and dynamic analysis phases. In response to malware authors developing sandbox-evading techniques, analysts have improved sandbox environments to mimic real systems more closely and developed methods to detect sandbox detection techniques. Blockchain technology is also being utilized to create immutable and transparent threat intelligence databases. This innovation aids in the secure sharing and distribution of malware indicators, facilitating quicker detection and response to emerging threats.

Deep learning, including techniques like convolutional and recurrent neural networks, is increasingly applied in malware analysis. These methods are capable of learning intricate patterns and behaviors, enhancing the accuracy of identifying malicious code and activities. Zero-Day Vulnerability Scanners are evolving to identify vulnerabilities that might be exploited by malware. These tools employ a range of techniques, such as static analysis and fuzzing, to detect 0-day vulnerabilities. Finally, with the rise of IoT devices, specialized tools and techniques are emerging for IoT-specific malware analysis. These tools focus on the unique characteristics and communication patterns of IoT devices, aiding in the detection and analysis of potential threats in this growing domain. Figure 4 illustrates the categories of recent tools in malware analysis and detection.

6.1 Behavioral analysis tools

Behavioral analysis is a crucial approach in malware analysis and detection, focusing on how malware behaves when executed in a controlled environment [56–59]. This technique observes the dynamic actions and interactions of malware with the host system and network, allowing for the detection of malicious behavior that may not be evident through static analysis alone. The core of behavioral analysis involves creating a dynamic execution environment, commonly known as a sandbox, where malware samples can be safely executed. This environment replicates the target system, enabling the malware to run without causing harm to the actual host.

Within this setting, various tools monitor different aspects of the malware's behavior, including file system interactions, registry modifications, process creation, and network communication. These tools log system calls, API functions, and other activities to meticulously track the sequence of events. Behavioral analysis tools also establish behavioral signatures that define what constitutes normal system behavior. By comparing the observed actions of the malware against these signatures, abnormal and potentially malicious behavior can be identified, such as attempts to encrypt files or establish unauthorized network connections. Additionally, heuristic algorithms and anomaly detection techniques are applied to the collected data. These algorithms search for patterns that deviate from the expected norm, flagging activities that indicate malicious intent. This approach is particularly effective in detecting previously unknown malware. Once the analysis is complete, these tools generate comprehensive reports detailing the malware's behavior, its impact on the system, and indicators of compromise (IOCs).

6.2 Threat intelligence platforms

Threat Intelligence Platforms (TIPs) are crucial for enhancing malware analysis and detection. They offer a structured framework for collecting, aggregating, analyzing, and disseminating threat intelligence. These platforms are invaluable for cybersecurity professionals and organizations as they provide critical insights to proactively defend against emerging threats [60–63]. A key function of TIPs is aggregating data from diverse sources, including feeds, internal logs, open-source intelligence, and proprietary databases. This data includes indicators of compromise (IOCs) such as malware signatures, IP addresses, domain names, and file hashes, thus providing a comprehensive view of potential threats. TIPs also play a vital role in normalizing and enriching this raw threat data, ensuring consistency and actionability. They standardize different data formats and add contextual information, such as source reputation and known malware families, thereby enhancing the quality and relevance of the intelligence. Moreover, TIPs employ sophisticated algorithms to correlate and analyze the collected data. This process involves identifying patterns, trends, and anomalies that may indicate malware infections or other malicious activities. Advanced techniques like machine learning and data analytics are often utilized to uncover previously unknown threats.

In terms of incident response, TIPs provide real-time alerts and playbooks for security teams. They enable automated actions based on received intelligence, such as blocking malicious IP addresses or isolating infected devices, thereby facilitating swift and effective responses to threats. Furthermore, these platforms promote the sharing of threat intelligence within trusted networks and information-sharing communities. This collaboration allows organizations to benefit from collective insights and strengthen their overall security posture. Standards like STIX/TAXII are often employed to facilitate the exchange of information. TIPs offer a high degree of customization and can be tailored to meet the specific needs of an organization. They often integrate with existing security tools and infrastructures, such as SIEMs, firewalls, and endpoint protection systems, to provide automated responses and adaptability to the ever-evolving threat landscape.

Moreover, the storage of historical threat data by TIPs is crucial for trend analysis and retrospective investigations. This historical perspective aids in identifying long-term patterns and understanding how cybercriminal tactics evolve over time. TIPs assist organizations in compliance reporting by maintaining detailed records of threat intelligence and incident response activities. These records are essential for meeting regulatory requirements and facilitating audits, thus playing a critical role in organizational compliance strategies.

6.3 Deception tools

Deception tools, a relatively new but increasingly vital component in the cybersecurity landscape, have demonstrated remarkable effectiveness in malware analysis and detection. These tools are designed to create a deceptive environment within a network with the aim of misleading, confusing, and ultimately trapping malicious actors and malware. They employ various strategies and technologies to achieve this goal [64–66]. A common technique used in deception tools is the deployment of honeypots and honeynets, which essentially act as decoy systems created to mimic real assets within a network. These systems are designed to be enticing to attackers, drawing them in and piquing their interest. Honeypots can range in complexity, from low-interaction versions that emulate services and applications at a basic level, to high-interaction variants that closely simulate real systems, thus enticing attackers further into the deception.

In addition to these decoy systems, deception tools also generate counterfeit data and services. This includes forged documents, credentials, and network shares that appear legitimate and attractive to attackers. When attackers engage with this deceptive content, the tools capture their actions, enabling detailed analysis of their tactics, techniques, and procedures (TTPs). Advanced simulation techniques are another facet of deception tools, going beyond simple emulation. These tools can mimic actual network behavior, including simulating user actions, generating realistic traffic patterns, and replicating the unique "personality" of a network. This level of sophistication makes the deceptive environment more convincing and effective.

One of the key advantages of deception tools is their ability to provide early detection of potential threats. When attackers or malware interact with these deceptive elements, they inadvertently trigger alerts, notifying security teams of the presence of a threat. This enables rapid investigation and response, helping to mitigate risks more efficiently. Beyond mere detection, these tools play a critical role in attribution and analysis. By examining how attackers interact with the decoys, security teams can gain valuable insights into their methodologies. This understanding is crucial for developing more effective countermeasures against future attacks. Deception tools are also adept at luring and containing malware. They can create simulated vulnerabilities or backdoors specifically designed to be exploited by malware, allowing for the isolation and detailed analysis of the malicious code. This capability is particularly useful for studying malware behavior and developing strategies to neutralize it.

Another significant advantage of deception tools is their ability to minimize false positives. By focusing on interactions with the deceptive elements, these tools reduce the volume of irrelevant alerts, streamlining the workload of security teams and enhancing the overall efficiency of malware detection. The most advanced deception tools are adaptive, capable of evolving over time based on observed attacker behavior. They continuously refine and update their deceptive elements to make them even more convincing, ensuring they remain effective against evolving threats and sophisticated attackers. This adaptive nature underscores the dynamic and proactive approach of deception tools in the ongoing battle against cyber threats.

6.4 Memory forensics tools

Memory forensics tools are essential in the field of malware analysis and detection, providing cybersecurity experts with the means to examine a computer's volatile memory (RAM) for indications of malicious activities [67–69]. In the overall process of forensic analysis, memory analysis plays a crucial role since malware often resides in memory to avoid detection and maintain its presence. These tools facilitate the retrieval of memory dumps from live systems or capture memory images from forensic images, which include active processes, data structures, and code present at the time of acquisition. They enable detailed analysis of these memory dumps, focusing on identifying running processes, their memory footprints, and associated threads. This analytical process is crucial for identifying suspicious or unauthorized applications that may be operating covertly.

A significant function of memory forensics tools is malware detection. They are skilled at scanning memory for known malicious signatures, patterns, or behaviors. This includes identifying injected or obfuscated code, rootkits, and other forms of memory-resident malware that are notoriously difficult to detect through conventional means. Furthermore, these tools are instrumental in uncovering evidence of API hooking and function call redirection, tactics commonly employed by malware to intercept and manipulate system calls. This capability is crucial for understanding the extent of the malware's control over a system. Detection of rootkits is another vital aspect of memory analysis, as rootkits are designed to be invisible to traditional file-based forensics. Memory forensics tools can reveal hidden processes, files, and network connections by exploring memory structures.

Advanced memory forensics tools even extend their capabilities to the examination of kernel memory, a critical area where essential system data and structures reside. Analyzing this segment can provide deep insights into the inner workings of the operating system and any potential manipulations by malware. These tools can also analyze memory dumps to extract information about active network connections and related data, assisting in the identification of malicious network communications. This analysis of network activity is crucial for understanding how malware communicates and potentially exfiltrates data. Timeline reconstruction is another crucial feature offered by memory forensics tools. By analyzing memory dumps over a period of time, analysts can piece together a timeline of events, revealing the sequence in which processes were initiated and actions were executed. This is particularly helpful in understanding the development and spread of a malware infection within a system. The extensibility of many memory forensics tools through plugin support enhances their utility significantly. Analysts can utilize custom scripts or leverage pre-built plugins to automate and refine the analysis process, making these tools even more powerful in combating sophisticated malware threats.

6.5 Sandboxing with threat intelligence integration

Sandboxing with integrated threat intelligence represents a sophisticated and effective approach to malware analysis and detection by combining isolated environments with up-to-date threat intelligence [70–72]. This method offers a comprehensive and dynamic means of understanding and identifying malicious software. At its core, sandboxing involves executing potentially malicious code within a controlled and isolated environment, such as a virtual machine or container. This setup closely mimics a real system, encouraging malware to demonstrate its full functionality and intentions. During the sandboxing process, malware is allowed to run freely, enabling real-time capture of its behavior, including file system interactions, registry changes, network communications, and process activities. This dynamic analysis provides valuable insights into the malware's execution flow, evasion tactics, and persistence mechanisms.

The integration of threat intelligence feeds is a crucial aspect of this approach. These feeds are constantly updated sources of information, providing the latest data on known threats, indicators of compromise (IOCs), malware signatures, and other relevant security details. By incorporating these feeds into the sandboxing environment, it becomes possible to retrieve up-to-date threat information instantly. Consequently, the behavior of the analyzed code can be cross-referenced with this data, facilitating the detection of matches with known malware. The sandbox also plays a vital role in IOC detection by scrutinizing the behaviors and attributes of the executed code against the IOCs from threat intelligence feeds. A match suggests that the code under analysis exhibits characteristics typical of known malicious software. Upon detecting a match or suspicious behavior, the sandbox generates alerts and detailed reports. These reports provide valuable information about the malware's actions, potential impact, and IOCs, which are crucial for further investigation and mitigation efforts. Additionally, integrating sandboxing with incident response tools can trigger automated responses, such as isolating affected systems, blocking malicious domains, or generating tickets for human analysts to investigate further.

Another advantage of this integrated approach is its support for historical analysis. The threat intelligence within the sandbox allows for checking previously analyzed samples for known threats, aiding in the identification of recurring attack patterns and related malware families. By combining sandboxing with threat intelligence integration, organizations gain a proactive, responsive, and insightful method for conducting malware analysis and detection. This approach not only aids in identifying and mitigating known threats but also plays a crucial role in addressing emerging threats by leveraging the latest intelligence data in real-time while closely monitoring the behavior of suspicious code in a secure and controlled environment.

7 Open challenges

In the field of malware detection using deep learning, there are several challenges that need to be addressed and promising avenues for future research [23, 73–85]. Figure 5 illustrates the open challenges associated with the deep learning-powered malware detection in cyberspace. One of the main challenges is the need to enhance the resilience of deep learning models against adversarial attacks, which are increasingly employed by malware authors to evade detection. Additionally, it is crucial to develop interpretable models that shed light on the decision-making processes of these models. Real-time detection, particularly in streaming environments, is becoming imperative to swiftly identify and counteract malware propagation. As the volume of malware samples and feature spaces continues to expand, scalability concerns must be addressed [44, 86–98].

Another important area for exploration is the development of techniques for few-shot and zero-shot learning, which can facilitate the detection of new and previously unseen malware strains. This capability is crucial in the ever-evolving threat landscape. The fusion of data from multiple sources, privacy-preserving methods for sharing labeled malware samples, and ethical considerations are also significant areas for research [14, 75, 98–110]. Improving malware detection accuracy can be achieved through efficiency in model architectures, seamless integration with existing security systems, cross-domain transfer learning, hybrid models that combine different deep learning architectures, and automated feature engineering methods. User education and awareness also play a pivotal role in reducing inadvertent installation or interaction with malware [21, 111–117]. Finally, collaborative threat intelligence platforms that enable information sharing among organizations represent a promising approach to collectively strengthen defenses against malware. Figure 3 illustrates the open challenges in deep learning-powered malware detection in cyberspace.

The field of deep learning-powered malware detection encompasses various challenges and solutions [118–123]. One significant challenge is handling 0-day attacks, as deep learning models traditionally rely on historical data and struggle against novel, unseen threats [124–128]. To address this, techniques such as transfer learning and anomaly detection should be employed to enhance the models' ability to detect new threats. Another area of concern is the collection, standardization, benchmarking, and reproducibility of malware datasets. The lack of standardized datasets and evaluation metrics hinders fair comparisons between different deep learning models. Overcoming this challenge requires the establishment of standardized benchmarks and datasets for malware analysis, as well as promoting open data sharing and collaboration within the research community.

The mathematical provability and interpretability of deep learning-powered models also pose challenges. These models, especially neural networks, are often considered "black boxes," making their decision-making processes opaque. It is essential to develop interpretable models or techniques that explain the predictions of deep learning models to ensure transparency and trust in malware detection systems. Additionally, class imbalance and distribution bias in training and testing datasets can significantly impact model performance. Imbalanced datasets tend to bias models towards the majority class, resulting in poor performance on minority classes that are often crucial in malware detection. Techniques like oversampling, undersampling, or synthetic data generation, along with tailored evaluation metrics, are vital for addressing this issue. Adapting to real-world settings and maintaining context awareness is another hurdle.

Deep learning models may struggle to adapt to rapidly changing environments, leading to potential obsolescence [129–136]. Developing dynamic models capable of continuous learning from new data and adapting to evolving threat landscapes is a solution to this problem. The lack of benchmark platforms for deep learning-powered malware detection research also hampers progress and collaboration in the field. Establishing such benchmark platforms and encouraging competitions can foster innovation and the development of more effective malware detection solutions. The aging problem of malware detection and classification tools is an ongoing challenge. As attackers evolve their tactics and techniques, malware detection tools often become less effective over time. To address this, continuous research and development are necessary to keep these tools updated and capable of identifying new attack vectors.

8 Future research directions

Deep learning techniques are revolutionizing malware detection, offering innovative approaches to tackle the complexity and sophistication of modern cyber threats. Graph Neural Networks (GNNs) excel in comprehending intricate relationships within graph-structured data, enabling a deeper understanding of malware behavior patterns often missed by traditional models. Transformer-based architectures, renowned for their success in natural language processing, hold promise in capturing temporal dependencies within sequences of system or API calls, potentially enhancing the comprehension of malware behavior.

The emergence of meta-learning techniques empowers models to swiftly adapt to new malware variants or unseen attack patterns, bolstering the adaptability and generalization of detection systems. Self-supervised learning, by training models on unlabeled data, unveils latent features and anomalies within malware, potentially improving identification accuracy. Federated learning, a collaborative approach, allows multiple devices or organizations to jointly train models without compromising data privacy, leading to more robust and accurate malware detection systems. A prime example of the effectiveness of Federated Learning is its application in improving predictive text and autocorrect features on smartphones. This technology involves training an algorithm across multiple decentralized devices (or servers) holding local data samples, without exchanging them. This method is used by major tech companies to enhance their keyboard applications. In this scenario, each smartphone has a local model that learns from the user's typing behavior. Instead of sending individual data points (like the words typed) back to a central server, the smartphone computes an update to the model based on the local data and only sends this model update back to the server. This way, the central model gets trained over time with the aggregated updates from millions of users, without ever having access to specific examples from any individual's data. This preserves privacy while still benefiting from the collective learning of all users.

Adversarial robustness techniques aim to fortify models against attacks, ensuring the reliability of malware detection systems in the face of adversarial threats. Continual learning techniques enable models to evolve with changing environments, incorporating new malware behaviors while retaining the ability to detect historical attack patterns. Finally, Explainable AI (XAI) techniques enhance the interpretability of models, fostering trust and aiding cybersecurity experts in comprehending model decisions. These emerging deep learning techniques collectively promise to elevate the efficacy and resilience of malware detection systems, offering a more comprehensive defense against evolving cyber threats.

The advancement of technology will significantly contribute to the progress of research in malware detection using deep learning models [137–140]. In Explainable Artificial Intelligence, the focus should be on enhancing the interpretability and transparency of deep learning models for cybersecurity experts. This entails developing neural network architectures that are easier to understand and techniques for generating explanations of model predictions in a human-readable format. Additionally, in Generative Artificial Intelligence, there is a need to explore how generative models like GANs and VAEs can be utilized to generate synthetic malware samples. These samples can be used to train deep learning models, allowing them to mimic the creativity of malware authors and enabling more robust model training and testing. Moreover, in the context of the Internet of Everything (IoE), deep learning models can be applied to analyze and secure interconnected devices and networks. It is crucial to address the unique challenges and vulnerabilities that arise in malware detection within the IoE ecosystem.

The limitations inherent in singular deep learning models for malware detection necessitate exploration of more advanced methodologies. Hybrid and ensemble techniques present promising avenues for enhanced threat coverage and resilience. These approaches can synergistically combine the strengths of deep learning architectures, such as convolutional neural networks (CNNs) and long short-term memory networks (LSTMs), with established methods like rule-based pre-filtering and feature engineering. For instance, domain knowledge may be leveraged to extract salient features from code and network traffic, which can then be fed into CNNs for automated learning of complex representations. Subsequently, LSTMs can analyze the remaining data for intricate temporal sequences indicative of novel malware, reducing computational burden and focusing resources on potential threats. Ensemble techniques further diversify the defensive landscape by combining diverse deep learning models trained on disparate data representations. Meta-learning algorithms can then orchestrate the collective predictions of these models, resulting in enhanced generalizability and improved resilience against evasion attempts.

However, the dynamic nature of the malware landscape demands agile solutions. Continuous learning techniques empower models to dynamically update their knowledge base with incoming data and emerging threats, obviating the need for complete retraining. Incremental learning approaches, such as online learning with memory replay, enable models to continuously learn from new data points while retaining past knowledge, mitigating the risk of catastrophic forgetting. Curriculum learning further facilitates this process by gradually exposing the model to more complex malware samples, building a robust foundation for accurate real-world detection. Additionally, meta-learning techniques can equip models with the ability to learn how to learn quickly on new tasks, enabling rapid adaptation to novel malware variants.

9 Conclusion

This article delves into the realm of deep learning models for malware detection in cyberspace, highlighting their significance and contributions to the field of cybersecurity. Deep learning models have emerged as powerful tools in combating malware, offering unparalleled potential in automatically learning features from vast datasets. However, it is crucial to acknowledge the limitations of current deep learning techniques in malware detection. These limitations include the vulnerability of deep learning models to adversarial attacks and the necessity of large, labeled datasets for effective training. Future directions in this field could involve exploring federated learning techniques to enhance privacy and reduce reliance on centralized data collection. Additionally, combining multiple deep learning approaches, such as ensemble models, could further enhance detection capabilities, particularly against evolving and sophisticated malware threats. The impact of deep learning on malware detection in cyberspace has been substantial. These models have revolutionized the field by providing accurate and efficient means of categorizing malware into distinct families or types. They empower security researchers and practitioners to swiftly identify and counter emerging threats, ultimately strengthening cybersecurity practices. The diversity of deep learning architectures, including Recurrent Neural Networks (RNNs) and Deep Convolutional Neural Networks (DCNNs), has expanded the range of applications in malware detection, making them a critical tool in the ongoing battle against evolving cyber threats. As cybersecurity concerns continue to grow, deep learning emerges as a viable option for advancing the state of the art in malware identification and analysis.

Author contributions

AR: Data curation, Formal Analysis, Investigation, Writing–original draft, Writing–review and editing. PC: Data curation, Formal Analysis, Investigation, Writing–original draft, Writing–review and editing. KS: Conceptualization, Data curation, Investigation, Methodology, Software, Supervision, Visualization, Writing–original draft, Writing–review and editing. TD: Funding acquisition, Project administration, Validation, Writing–review and editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

The author(s) declared that they were an editorial board member of Frontiers, at the time of submission. This had no impact on the peer review process and the final decision.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Kwon I, Im EG. Extracting the representative API call patterns of malware families using recurrent neural network. In: Proceedings of the Proceedings of the International Conference on Research in Adaptive and Convergent Systems; September 20-23, 2017; ACM: Krakow Poland (2017). p. 202–7.

2. Amin M, Tanveer TA, Tehseen M, Khan M, Khan FA, Anwar S. Static malware detection and attribution in android byte-code through an end-to-end deep system. *Future generation Comput Syst* (2020) 102:112–26. doi:10.1016/j.future.2019.07.070

3. Nobakht M, Javidan R, Pourebrahimi A. DEMD-IoT: a deep ensemble model for IoT malware detection using CNNs and network traffic. *Evolving Syst* (2023) 14(3): 461–77. doi:10.1007/s12530-022-09471-z

4. Imamverdiyev Y, Abdullayeva F. Deep learning method for denial of service attack detection based on restricted Boltzmann machine. *Big data* (2018) 6(2):159–69. doi:10.1089/big.2018.0023

5. Eckhart M, Ekelhart A. Digital twins for cyber-physical systems security: state of the art and outlook. In: Biffl S, Eckhart M, Lüder A, Weippl E, editors. *Security and quality in cyber-physical systems engineering*. Cham: Springer International Publishing (2019). p. 383–412.

6. Souri A, Hosseini R. A state-of-the-art survey of malware detection approaches using data mining techniques. *Hum Cent Comput Inf Sci* (2018) 8:3. doi:10.1186/s13673-018-0125-x

7. Malik MI, Ibrahim A, Hannay P, Sikos LF. Developing resilient cyber-physical systems: a review of state-of-the-art malware detection approaches, gaps, and future directions. *Computers* (2023) 12:79. doi:10.3390/computers12040079

8. Razaulla S, Fachkha C, Markarian C, Gawanmeh A, Mansoor W, Fung BCM, et al. The age of ransomware: a survey on the evolution, taxonomy, and research directions. *IEEE Access* (2023) 11:40698–723. doi:10.1109/ACCESS.2023.3268535

9. Deldar F, Abadi M. Deep learning for zero-day malware detection and classification: a survey. *ACM Comput Surv* (2023) 56(2):1–37. doi:10.1145/3605775

10. Ali M, Hassen HR, Lones MA, Zantout H. An in-depth review of machine learning based Android malware detection. *Comput Security* (2022) 121:102833. doi:10.1016/j.cose.2022.102833

11. Tayyab U-e.-H, Khan FB, Durad MH, Khan A, Lee YS. A survey of the recent trends in deep learning based malware detection. *J Cybersecur Priv* (2022) 2:800–29. doi:10.3390/jcp2040041

12. Gibert D, Mateu C, Planes J. The rise of machine learning for detection and classification of malware: research developments, trends and challenges. *J Netw Comp Appl* (2020) 153:102526. doi:10.1016/j.jnca.2019.102526

13. Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, et al. The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *BMJ* (2021) 372:n71. doi:10.1136/bmj.n71

14. Subrahmanyam SSB, Goutham P, Ambati VKR, Bijitha CV, Nath HV. A hybrid method for analysis and detection of malicious executables in IoT network. *Comput Security* (2023) 132:103339. doi:10.1016/j.cose.2023.103339

15. Jain M, Andreopoulos W, Stamp M. Convolutional neural networks and extreme learning machines for malware classification. *J Comp Virol Hacking Tech* (2020) 16: 229–44. doi:10.1007/s11416-020-00354-y

16. GulatasKilinc HH, Zaim AH, Aydin MA. Malware threat on edge/fog computing environments from Internet of Things devices perspective. *IEEE Access* (2023) 11: 33584–606. doi:10.1109/ACCESS.2023.3262614

17. Zhang N, Xue J, Ma Y, Zhang R, Liang T, Tan YA. Hybrid sequence-based Android malware detection using natural language processing. *Int J Intell Syst* (2021) 36(10):5770–84. doi:10.1002/int.22529

18. Chen X. Power system malware detection based on deep belief network classifier. In: 2022 6th International Conference on Green Energy and Applications (ICGEA); 4th to 6th March 2022; Singapore (2022). p. 245–9.

19. He N, Wang T, Chen P, Yan H, Jin Z. An android malware detection method based on deep autoencoder. In: Proceedings of the Proceedings of the 2018 artificial intelligence and cloud computing conference; July 2 2018 to July 7 2018; San Francisco, CA, USA (2018). p. 88–93.

20. Reilly C, O Shaughnessy S, Thorpe C. Robustness of image-based malware classification models trained with generative adversarial networks. In: Proceedings of the 2023 European Interdisciplinary Cybersecurity Conference; June 14 - 15, 2023; Stavanger, Norway (2023). p. 92–9.

21. Shu L, Dong S, Su H, Huang J. Android malware detection methods based on convolutional neural network: a survey. *IEEE Trans Emerging Top Comput Intelligence* (2023) 7:1330–50. doi:10.1109/tetci.2023.3281833

22. Daniel A, Deebalakshmi R, Thilagavathy R, Kohilakanagalakshmi T, Janakiraman S, Balusamy B. Optimal feature selection for malware detection in cyber physical systems using graph convolutional network. *Comput Electr Eng* (2023) 108:108689. doi:10.1016/j.compeleceng.2023.108689

23. Almaleh A, Almushabb R, Ogran R. Malware API calls detection using hybrid logistic regression and RNN model. *Appl Sci* (2023) 13(9):5439. doi:10.3390/app13095439

24. Rezvy S, Petridis M, Lasebae A, Zebin T. Intrusion detection and classification with autoencoded deep neural network. In: Proceedings of the Innovative Security Solutions for Information Technology and Communications: 11th International Conference, SecITC 2018; November 8–9, 2018; Bucharest, Romania (2019). p. 142–56.

25. D'Angelo G, Ficco M, Palmieri F. Malware detection in mobile environments based on Autoencoders and API-images. *Comput.* (2020) 137:26–33. doi:10.1016/j.jpdc.2019.11.001

26. Alotaibi A. Identifying malicious software using deep residual long-short term memory. *IEEE Access* (2019) 7:163128–37. doi:10.1109/ACCESS.2019.2951751

27. Liu J, Feng Y, Liu X, Zhao J, Liu Q. MRm-DLDet: a memory-resident malware detection framework based on memory forensics and deep neural network. *Cybersecurity* (2023) 6(1):21. doi:10.1186/s42400-023-00157-w

28. Saxe J, Berlin K. Deep neural network based malware detection using two dimensional binary program features. In: Proceedings of the 2015 10th International Conference on Malicious and Unwanted Software (MALWARE); Oct. 20 2015 to Oct. 22 2015; Fajardo, PR, USA (2015). p. 11–20.

29. Li D, Wang Z, Xue Y. Deepdetector: android malware detection using deep neural network. In: Proceedings of the 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE); IEEE; 22-23 June 2018; Paris, France (2018). p. 184–8.

30. Mercaldo F, Santone A. Deep learning for image-based mobile malware detection. *J Comp Virol Hacking Tech* (2020) 16(2):157–71. doi:10.1007/s11416-019-00346-7

31. Alqahtani A, Azzony S, Alsharafi L, Alaseri M. Web-based malware detection system using convolutional neural network. *Digital* (2023) 3(3):273–85. doi:10.3390/digital3030017

32. Chaganti R, Ravi V, Pham TD. Image-based malware representation approach with EfficientNet convolutional neural networks for effective malware classification. *J Inf Security Appl* (2022) 69:103306. doi:10.1016/j.jisa.2022.103306

33. Sl SD, Jaidhar CD. Windows malware detector using convolutional neural network based on visualization images. *IEEE Trans Emerging Top Comput* (2019) 9(2):1057–69. doi:10.1109/TETC.2019.2910086

34. Kim JY, Cho SB. Obfuscated malware detection using deep generative model based on global/local features. *Comput Security* (2022) 112:102501. doi:10.1016/j.cose.2021.102501

35. Yang J, Li T, Liang G, He W, Zhao Y. A simple recurrent unit model based intrusion detection system with DCGAN. *IEEE Access* (2019) 7:83286–96. doi:10.1109/access.2019.2922692

36. Won DO, Jang YN, Lee SW. PlausMal-GAN: plausible malware training based on generative adversarial networks for analogous zero-day malware detection. *IEEE Trans Emerging Top Comput* (2022) 11(1):82–94. doi:10.1109/tetc.2022.3170544

37. Cui Z, Zhao Y, Cao Y, Cai X, Zhang W, Chen J. Malicious code detection under 5G HetNets based on a multi-objective RBM model. *IEEE Netw* (2021) 35(2):82–7. doi:10.1109/mnet.011.2000331

38. Liu Z, Wang R, Japkowicz N, Tang D, Zhang W, Zhao J. Research on unsupervised feature learning for android malware detection based on restricted Boltzmann machines. *Future Generation Comp Syst* (2021) 120:91–108. doi:10.1016/j.future.2021.02.015

39. Jayashree R. Enhanced classification using restricted Boltzmann machine method in deep learning for COVID-19. *Understanding COVID-19: role Comput intelligence* (2022) 425–46. doi:10.1007/978-3-030-74761-9_19

40. Pandey S, Kumar N, Handa A, Shukla SK. Evading malware classifiers using RL agent with action-mask. *Int J Inf Security* (2023) 22(6):1743–63. doi:10.1007/s10207-023-00715-w

41. Kim S, Yoon S, Lim H. Deep reinforcement learning-based traffic sampling for multiple traffic analyzers on software-defined networks. *IEEE Access* (2021) 9:47815–27. doi:10.1109/access.2021.3068459

42. Jahromi AN, Hashemi S, Dehghantanha A, Choo KKR, Karimipour H, Newton DE, et al. An improved two-hidden-layer extreme learning machine for malware hunting. *Comput Security* (2020) 89:101655. doi:10.1016/j.cose.2019.101655

43. Aldehim G, Arasi MA, Khalid M, Aljameel SS, Marzouk R, Mohsen H, et al. Gauss-mapping black Widow optimization with deep extreme learning machine for android malware classification model. *IEEE Access* (2023) 11:87062–70. doi:10.1109/access.2023.3285289

44. Roy KS, Ahmed T, Udas PB, Karim ME, Majumdar S. MalHyStack: a hybrid stacked ensemble learning framework with feature engineering schemes for obfuscated malware analysis. *Intell Syst Appl* (2023) 20:200283. doi:10.1016/j.iswa.2023.200283

45. He Y, Kang X, Yan Q, Li E. ResNeXt+: attention mechanisms based on ResNeXt for malware detection and classification. *IEEE Trans Inf Forensics Security* (2023) 19:1142–55. doi:10.1109/tifs.2023.3328431

46. Choi S, Bae J, Lee C, Kim Y, Kim J. Attention-based automated feature extraction for malware analysis. *Sensors* (2020) 20(10):2893. doi:10.3390/s20102893

47. Agrawal R, Stokes JW, Selvaraj K, Marinescu M. Attention in recurrent neural networks for ransomware detection. In: ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP); May 12-17, 2019; Brighton, UK (2019). p. 3222–6.

48. Alkahtani H, Aldhyani THH. Artificial intelligence algorithms for malware detection in android-operated mobile devices. *Sensors* (2022) 22:2268. doi:10.3390/s22062268

49. Krzysztoń M, Bok B, Lew M, Sikora A. Lightweight on-device detection of android malware based on the koodous platform and machine learning. *Sensors* (2022) 22:6562. doi:10.3390/s22176562

50. Lu K, Cheng J, Yan A. Malware detection based on the feature selection of a correlation information decision matrix. *Mathematics* (2023) 11:961. doi:10.3390/math11040961

51. Lee J, Jang H, Ha S, Yoon Y. Android malware detection using machine learning with feature selection based on the genetic algorithm. *Mathematics* (2021) 9:2813. doi:10.3390/math9212813

52. Cañadas AM, Mendez OM, Vega JDC. Algebraic structures induced by the insertion and detection of malware. *Computation* (2023) 11:140. doi:10.3390/computation11070140

53. Singh AK, Taterh S, Mitra U. An efficient tactic for analysis and evaluation of malware dump file using the volatility tool. *SN COMPUT SCI* (2023) 4:457. doi:10.1007/s42979-023-01844-8

54. Amira A, Derhab A, Karbab EB, Omar N. A survey of malware analysis using community detection algorithms. *ACM Comput Surv* (2023) 56(2):1–29. doi:10.1145/3610223

55. Pereberina A, Kostyushko A, Tormasov A. An algorithm for scheduling of threads for system and application code split approach in dynamic malware analysis. *J Comput Virol Hack Tech* (2023) 19:459–68. doi:10.1007/s11416-023-00473-2

56. Hashida Haidros Rahima Manzil S. Detection approaches for android malware: taxonomy and review analysis. *Expert Syst Appl* (2024) 238(Part F):122255. doi:10.1016/j.eswa.2023.122255

57. Kara I. Fileless malware threats: recent advances, analysis approach through memory forensics and research challenges. *Expert Syst Appl* (2023) 214:119133. doi:10.1016/j.eswa.2022.119133

58. Celdrán AH, Sánchez PMS, Castillo MA, Bovet G, Pérez GM, Stiller B. Intelligent and behavioral-based detection of malware in IoT spectrum sensors. *Int J Inf Secur* (2023) 22:541–61. doi:10.1007/s10207-022-00602-w

59. Bhat P, Behal S, Dutta K. A system call-based android malware detection approach with homogeneous and heterogeneous ensemble machine learning. *Comput Security* (2023) 130:103277. doi:10.1016/j.cose.2023.103277

60. Sun N, Ding M, Jiang J, Xu W, Mo X, Tai Y, et al. Cyber threat intelligence mining for proactive cybersecurity defense: a survey and new perspectives. *IEEE Commun Surv Tutorials* (2023) 25(3):1748–74. doi:10.1109/COMST.2023.3273282

61. Turner A, McCombie S, Uhlmann A. Ransomware-bitcoin threat intelligence sharing using structured threat information expression. *IEEE Security and Privacy* (2023) 21(03):47–57. doi:10.1109/MSEC.2022.3166282

62. Sai Charan PV, Ratnakaram G, Chunduri H, Mohan Anand P, Kumar Shukla S. DKaaS: DARK-KERNEL as a service for active cyber threat intelligence. *Comput Security* (2023) 132:103329. doi:10.1016/j.cose.2023.103329

63. Lin P-C, Hsu W-H, Lin Y-D, Hwang R-H, Wu H-K, Lai Y-C, et al. Correlation of cyber threat intelligence with sightings for intelligence assessment and augmentation. *Computer Networks* (2023) 228:109736. doi:10.1016/j.comnet.2023.109736

64. Sajid MSI, Wei J, Al-Shaer E, Qi D, Abdeen B, Khan L. SymbSODA: configurable and verifiable orchestration automation for active malware deception. *ACM Trans Priv Secur* (2023) 26(4):1–36. doi:10.1145/3624568

65. El-Kosairy A, Abdelbaki N. Deception as a service: intrusion and ransomware detection system for cloud computing (IRDS4C). *Adv Comp Int* (2023) 3:9. doi:10.1007/s43674-023-00056-0

66. Ganfure GO, Wu C -F, Chang Y -H, Shih W -K. RTrap: trapping and containing ransomware with machine learning. *IEEE Trans Inf Forensics Security* (2023) 18:1433–48. doi:10.1109/TIFS.2023.3240025

67. Liu J, Feng Y, Liu X, Zhao J, Liu Q. MRm-DLDet: a memory-resident malware detection framework based on memory forensics and deep neural network. *Cybersecurity* (2023) 6:21. doi:10.1186/s42400-023-00157-w

68. Daghmehchi Firoozjaei M, Samet S, Ghorbani AA. Parent process termination: an adversarial technique for persistent malware. *J Cyber Security Tech* (2023) 1–26. doi:10.1080/23742917.2023.2246229

69. Naeem H, Dong S, Falana OJ, Ullah F. Development of a deep stacked ensemble with process based volatile memory forensics for platform independent malware detection and classification. *Expert Syst Appl* (2023) 223:119952. doi:10.1016/j.eswa.2023.119952

70. Chen T, Zeng H, Lv M, Zhu T. CTIMD: cyber threat intelligence enhanced malware detection using API call sequences with parameters. *Comput Security* (2024) 136:103518. doi:10.1016/j.cose.2023.103518

71. Ilca LF, Lucian OP, Balan TC. Enhancing cyber-resilience for small and medium-sized organizations with prescriptive malware analysis, detection and response. *Sensors* (2023) 23:6757. doi:10.3390/s23156757

72. Geng JX, Wang J, Fang Z, Zhou Y, Wu D, Ge W. A Survey of strategy-driven evasion methods for PE malware: transformation, concealment, and attack. *Comput Security* (2023) 137:103595. doi:10.1016/j.cose.2023.103595

73. Ilca LF, Lucian OP, Balan TC. Enhancing cyber-resilience for small and medium-sized organizations with prescriptive malware analysis, detection and response. *Sensors* (2023) 23(15):6757. doi:10.3390/s23156757

74. Vasani V, Bairwa AK, Joshi S, Pljonkin A, Kaur M, Amoon M. Comprehensive analysis of advanced techniques and vital tools for detecting malware intrusion. *Electronics* (2023) 12(20):4299. doi:10.3390/electronics12204299

75. Singh A, Ikuesan RA, Venter H. MalFe—malware feature engineering generation platform. *Computers* (2023) 12(10):201. doi:10.3390/computers12100201

76. Zhang S, Wu J, Zhang M, Yang W. Dynamic malware analysis based on API sequence semantic fusion. *Appl Sci* (2023) 13(11):6526. doi:10.3390/app13116526

77. Taher F, AlFandi O, Al-kfairy M, Al Hamadi H, Alrabaee S. DroidDetectMW: a hybrid intelligent model for android malware detection. *Appl Sci* (2023) 13(13):7720. doi:10.3390/app13137720

78. Akhtar MS, Feng T. Evaluation of machine learning algorithms for malware detection. *Sensors* (2023) 23(2):946. doi:10.3390/s23020946

79. Taher F, Al Fandi O, Al Kfairy M, Al Hamadi H, Alrabaee S. A proposed artificial intelligence model for android-malware detection. *Informatics* (2023) 10(3):67. doi:10.3390/informatics10030067

80. Alhashmi AA, Darem AA, Alashjaee AM, Alanazi SM, Alkhaldi TM, Ebad SA, et al. Similarity-based hybrid malware detection model using API calls. *Mathematics* (2023) 11(13):2944. doi:10.3390/math11132944

81. Herrera-Silva JA, Hernández-Álvarez M. Dynamic feature dataset for ransomware detection using machine learning algorithms. *Sensors* (2023) 23(3):1053. doi:10.3390/s23031053

82. Lockett A, Chalkias I, Yucel C, Henriksen-Bulmer J, Katos V. Investigating IPTV malware in the wild. *Future Internet* (2023) 15(10):325. doi:10.3390/fi15100325

83. Nachaat Mohamed. Current trends in AI and ML for cybersecurity: a state-of-the-art survey. *Cogent Engineering* (2023) 10:2. doi:10.1080/23311916.2023.2272358

84. Sun H, Shu H, Kang F, Guang Y. ModDiff: modularity similarity-based malware homologation detection. *Electronics* (2023) 12(10):2258. doi:10.3390/electronics12102258

85. Fedorchenko E, Novikova E, Fedorchenko A, Verevkin S. An analytical review of the source code models for exploit analysis. *Information* (2023) 14(9):497. doi:10.3390/info14090497

86. Buriro A, Buriro AB, Ahmad T, Buriro S, Ullah S. MalwD&C: a quick and accurate machine learning-based approach for malware detection and categorization. *Appl Sci* (2023) 13(4):2508. doi:10.3390/app13042508

87. Djenna A, Bouridane A, Rubab S, Marou IM. Artificial intelligence-based malware detection, analysis, and mitigation. *Symmetry* (2023) 15(3):677. doi:10.3390/sym15030677

88. Cha HJ, Yang HK, Song YJ, Kang AR. Intelligent anomaly detection system through malware image augmentation in IIoT environment based on digital twin. *Appl Sci* (2023) 13(18):10196. doi:10.3390/app131810196

89. Babbar H, Rani S, Sah DK, AlQahtani SA, Kashif Bashir A. Detection of android malware in the Internet of Things through the K-nearest neighbor algorithm. *Sensors* (2023) 23(16):7256. doi:10.3390/s23167256

90. Gazzan M, Sheldon FT. Opportunities for early detection and prediction of ransomware attacks against industrial control systems. *Future Internet* (2023) 15(4):144. doi:10.3390/fi15040144

91. Khalid O, Ullah S, Ahmad T, Saeed S, Alabbad DA, Aslam M, et al. An insight into the machine-learning-based fileless malware detection. *Sensors* (2023) 23(2):612. doi:10.3390/s23020612

92. Ba'abbad I, Batarfi O. Proactive ransomware detection using extremely fast decision tree (efdt) algorithm: a case study. *Computers* (2023) 12(6):121. doi:10.3390/computers12060121

93. Zhang S, Hu C, Wang L, Mihaljevic MJ, Xu S, Lan T. A malware detection approach based on deep learning and memory forensics. *Symmetry* (2023) 15(3):758. doi:10.3390/sym15030758

94. Saridou B, Moulas I, Shiaeles S, Papadopoulos B. Image-based malware detection using α-cuts and binary visualisation. *Appl Sci* (2023) 13(7):4624. doi:10.3390/app13074624

95. Alabrah A. A novel neural network architecture using automated correlated feature layer to detect android malware applications. *Mathematics* (2023) 11(20):4242. doi:10.3390/math11204242

96. Lu J, Ren X, Zhang J, Wang T. CPL-net: a malware detection network based on parallel CNN and LSTM feature fusion. *Electronics* (2023) 12(19):4025. doi:10.3390/electronics12194025

97. Aboaoja FA, Zainal A, Ali AM, Ghaleb FA, Alsolami FJ, Rassam MA. Dynamic extraction of initial behavior for evasive malware detection. *Mathematics* (2023) 11(2):416. doi:10.3390/math11020416

98. Deng L, Wen H, Xin M, Li H, Pan Z, Sun L. Enimanal: augmented cross-architecture IoT malware analysis using graph neural networks. *Comput Security* (2023) 132:103323. doi:10.1016/j.cose.2023.103323

99. Kumar EP, Priyanka S. A comprehensive survey on hardware-assisted malware analysis and primitive techniques. *Comp Networks* (2023) 235:109967. doi:10.1016/j.comnet.2023.109967

100. Vashishtha LK, Chatterjee K, Rout SS. An Ensemble approach for advance malware memory analysis using Image classification techniques. *J Inf Security Appl* (2023) 77:103561. doi:10.1016/j.jisa.2023.103561

101. Lv M, Zeng H, Chen T, Zhu T. CTIMD: cyber threat intelligence enhanced malware detection using API call sequences with parameters. *Comput Security* (2023) 136:103518. doi:10.1016/j.cose.2023.103518

102. Khan SH, Alahmadi TJ, Ullah W, Iqbal J, Rahim A, Alkahtani HK, et al. A new deep boosted CNN and ensemble learning based IoT malware detection. *Comput Security* (2023) 133:103385. doi:10.1016/j.cose.2023.103385

103. Kara I. Fileless malware threats: recent advances, analysis approach through memory forensics and research challenges. *Expert Syst Appl* (2023) 214:119133. doi:10.1016/j.eswa.2022.119133

104. Liu C, Lu J, Feng W, Du E, Di L, Song Z. MOBIPCR: efficient, accurate, and strict ML-based mobile malware detection. *Future Generation Comp Syst* (2023) 144:140–50. doi:10.1016/j.future.2023.02.014

105. Kumar S, Panda K. SDIF-CNN: stacking deep image features using fine-tuned convolution neural network models for real-world malware detection and classification. *Appl Soft Comput* (2023) 146:110676. doi:10.1016/j.asoc.2023.110676

106. Zhu H, Wei H, Wang L, Xu Z, Sheng VS. An effective end-to-end android malware detection method. *Expert Syst Appl* (2023) 218:119593. doi:10.1016/j.eswa.2023.119593

107. Kishore P, Barisal SK, Mohapatra DP, Mall R. An efficient two-stage pipeline model with filtering algorithm for mislabeled malware detection. *Comput Security* (2023) 135:103499. doi:10.1016/j.cose.2023.103499

108. Bhat P, Behal S, Dutta K. A system call-based android malware detection approach with homogeneous and heterogeneous ensemble machine learning. *Comput Security* (2023) 130:103277. doi:10.1016/j.cose.2023.103277

109. Banik A, Singh JP. Android malware detection by correlated real permission couples using FP growth algorithm and neural networks. *IEEE Access* (2023) 11:124996–5010. doi:10.1109/access.2023.3323845

110. Perez AJ, Zeadally S, Tan DK. Detecting mobile malware associated with global pandemics. *IEEE Pervasive Comput* (2023) 22:45–54. doi:10.1109/mprv.2023.3321218

111. Chen YH, Lin SC, Huang SC, Lei CL, Huang CY. Guided malware sample analysis based on graph neural networks. *IEEE Trans Inf Forensics Security* (2023) 18:4128–43. doi:10.1109/tifs.2023.3283913

112. Lee H, Kim S, Baek D, Kim D, Hwang D. Robust IoT malware detection and classification using opcode category features on machine learning. *IEEE Access* (2023) 11:18855–67. doi:10.1109/access.2023.3247344

113. Al-Andoli MN, Sim KS, Tan SC, Goh PY, Lim CP. An ensemble-based parallel deep learning classifier with PSO-BP optimization for malware detection. *IEEE Access* (2023) 11:76330–46. doi:10.1109/access.2023.3296789

114. Manthena H, Kimmel JC, Abdelsalam M, Gupta M. Analyzing and explaining black-box models for online malware detection. *IEEE Access* (2023) 11:25237–52. doi:10.1109/access.2023.3255176

115. Abdelwahed MF, Kamal MM, Sayed SG. Detecting malware activities with MalpMiner: a dynamic analysis approach. *IEEE Access* (2023) 11:84772–84. doi:10.1109/access.2023.3266562

116. Lee S, Lee S, Park J, Kim K, Lee K. Hiding in the crowd: ransomware protection by adopting camouflage and hiding strategy with the link file. *IEEE Access* (2023) 11:92693–704. doi:10.1109/access.2023.3309879

117. Shin K, Lee Y, Lim J, Kang H, Lee S. System API vectorization for malware detection. *IEEE Access* (2023) 11:53788–805. doi:10.1109/access.2023.3276902

118. Niu W, Wang Y, Liu X, Yan R, Li X, Zhang X. GCDroid: android malware detection based on graph compression with reachability relationship extraction for IoT devices. *IEEE Internet Things J* (2023) 10:11343–56. doi:10.1109/jiot.2023.3241697

119. Yu Z, Li S, Bai Y, Han W, Wu X, Tian Z. REMSF: a robust ensemble model of malware detection based on semantic feature fusion. *IEEE Internet Things J* (2023) 10:16134–43. doi:10.1109/jiot.2023.3267337

120. Odat E, Yaseen QM. A novel machine learning approach for android malware detection based on the Co-existence of features. *IEEE Access* (2023) 11:15471–84. doi:10.1109/access.2023.3244656

121. Thummapudi K, Lama P, Boppana RV. Detection of ransomware attacks using processor and disk usage data. *IEEE Access* (2023) 11:51395–407. doi:10.1109/access.2023.3279819

122. Kim C, Chang SY, Kim J, Lee D, Kim J. Automated, reliable zero-day malware detection based on autoencoding architecture. *IEEE Trans Netw Serv Manag* (2023) 20:3900–14. doi:10.1109/tnsm.2023.3251282

123. Jin B, Choi J, Hong JB, Kim H. On the effectiveness of perturbations in generating evasive malware variants. *IEEE Access* (2023) 11:31062–74. doi:10.1109/access.2023.3262265

124. Kural OE, Kiliç E, Aksaç C. Apk2Audio4AndMal: audio based malware family detection framework. *IEEE Access* (2023) 11:27527–35. doi:10.1109/access.2023.3258377

125. Yonamine S, Taenaka Y, Kadobayashi Y, Miyamoto D. Design and implementation of a sandbox for facilitating and automating IoT malware analysis with techniques to elicit malicious behavior: case studies of functionalities for dissecting IoT malware. *J Comp Virol Hacking Tech* (2023) 19(2):149–63. doi:10.1007/s11416-023-00478-x

126. Masid AG, Higuera JB, Higuera JRB, Montalvo JAS. Application of the SAMA methodology to Ryuk malware. *J Comp Virol Hacking Tech* (2023) 19(2):165–98. doi:10.1007/s11416-022-00434-1

127. Singh AK, Taterh S, Mitra U. An efficient tactic for analysis and evaluation of malware dump file using the volatility tool. *SN Comp Sci* (2023) 4(5):457. doi:10.1007/s42979-023-01844-8

128. de Lima SM, Souza DM, Pinheiro RP, Silva SH, Lopes PG, de Lima RD, et al. Next-generation antivirus for JavaScript malware detection based on dynamic features. *Knowledge Inf Syst* (2023) 66:1337–70. doi:10.1007/s10115-023-01978-4

129. Sharma A, Gupta BB, Singh AK, Saraswat VK. A novel approach for detection of APT malware using multi-dimensional hybrid Bayesian belief network. *Int J Inf Security* (2023) 22(1):119–35. doi:10.1007/s10207-022-00631-5

130. Pereberina A, Kostyushko A, Tormasov A. An algorithm for scheduling of threads for system and application code split approach in dynamic malware analysis. *J Comp Virol Hacking Tech* (2023) 19:459–68. doi:10.1007/s11416-023-00473-2

131. Seyfari Y, Meimandi A. A new approach to android malware detection using fuzzy logic-based simulated annealing and feature selection. *Multimedia Tools Appl* (2023) 83:10525–49. doi:10.1007/s11042-023-16035-z

132. Alzubi OA, Alzubi JA, Alzubi TM, Singh A. Quantum Mayfly optimization with encoder-decoder driven LSTM networks for malware detection and classification model. *Mobile Networks Appl* (2023) 28:795–807. doi:10.1007/s11036-023-02105-x

133. Ullah F, Ullah S, Srivastava G, Lin JCW, Zhao Y. NMal-Droid: network-based android malware detection system using transfer learning and CNN-BiGRU ensemble. *Wireless Networks* (2023) 1–22. doi:10.1007/s11276-023-03414-5

134. Deng X, Cen M, Jiang M, Lu M. Ransomware early detection using deep reinforcement learning on portable executable header. *Cluster Comput* (2023) 1–15. doi:10.1007/s10586-023-04043-5

135. Balikcioglu PG, Sirlanci M, A. Kucuk O, Ulukapi B, Turkmen RK, Acarturk C. Malicious code detection in android: the role of sequence characteristics and disassembling methods. *Int J Inf Security* (2023) 22(1):107–18. doi:10.1007/s10207-022-00626-2

136. Gao C, Cai M, Yin S, Huang G, Li H, Yuan W, et al. Obfuscation-resilient android malware analysis based on complementary features. *IEEE Trans Inf Forensics Security* (2023) 18:5056–68. doi:10.1109/TIFS.2023.3302509

137. Gopinath M, Sethuraman SC. A comprehensive survey on deep learning based malware detection techniques. *Comp Sci Rev* (2023) 47:100529. doi:10.1016/j.cosrev.2022.100529

138. Zhu H-juan, Gu W, Wang L-min, Xu Z-cheng, Sheng VS. Android malware detection based on multi-head squeeze-and-excitation residual network. *Expert Syst Appl* (2023) 212:118705. doi:10.1016/j.eswa.2022.118705

139. Kumar R, Zhang X, Khan RU, Sharif A. Research on data mining of permission-induced risk for android IoT devices. *Appl Sci* (2019) 9:277. doi:10.3390/app9020277

140. Mustafa Majid A-A, Alshaibi AJ, Kostyuchenko E, Shelupanov A. A review of artificial intelligence based malware detection using deep learning. *Mater Today Proc* (2023) 80(3):2678–83. doi:10.1016/j.matpr.2021.07.012

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 04 April 2024
DOI 10.3389/fphy.2024.1366488

OPEN ACCESS

EDITED BY
Yuanyuan Huang,
Chengdu University of Information Technology, China

REVIEWED BY
Shuangqiang Liu,
Sun Yat-sen University, China
Jie Li,
Tianjin University, China

*CORRESPONDENCE
Jian Shen,
✉ shenjian@hainanu.edu.cn

†These authors have contributed equally to this work

RECEIVED 06 January 2024
ACCEPTED 18 March 2024
PUBLISHED 04 April 2024

CITATION
Zeng Y, Zhang P, Li Z, Shen J and Li C (2024), Fabry-perot interferometers with resin scaffolders for high sensitivity temperature sensing.
Front. Phys. 12:1366488.
doi: 10.3389/fphy.2024.1366488

Fabry-perot interferometers with resin scaffolders for high sensitivity temperature sensing

Yu Zeng[1,2†], Pengyu Zhang[1,2†], Zhiqi Li[1,2], Jian Shen[1,2*] and Chaoyang Li[1,2]

[1]School of Information and Communication Engineering, Hainan University, Haikou, China, [2]State Key Laboratory of Marine Resource Utilization in South China Sea, Hainan University, Haikou, China

This study explores the development of an innovative Fabry-Perot Interferometer (FPI) designed for temperature sensing and environmental monitoring. The device is constructed by embedding optical fibers within a 3D-printed resin scaffold, forming a structure with an open Fabry-Perot cavity. Intended as an integral component of Cyber-Physical-Social Systems (CPSS), this FPI structure aims to enhance the system's capacity to sense changes in external environmental conditions. Within the CPSS context, the FPI offers several advantages, including simple manufacturing processes, low production costs, and high sensitivity. These benefits contribute to providing precise environmental feedback to the system, which is essential in implementing effective security and privacy protection strategies. Experimental evaluations have shown that the FPI exhibits a high linear sensitivity of 14.330 nm/°C within a temperature range of 34.9°C–38.5°C, confirming its potential for application in CPSS for temperature monitoring and environmental sensing.

KEYWORDS

3D print, resin scaffold, open fabry-perot resonator, sensor, environmental monitoring

1 Introduction

Recently, Fabry-Perot temperature sensors attracts more and more attention due to the advantage of small size, high sensitivity, electromagnetic interference resistance, and corrosion resistance [1–7]. In many fields such as medical care, environmental monitoring, and food safety [8], It is very sensitive to changes in temperature, so improving the sensitivity of temperature sensors has always been a concern. As with the Fabry-Perot interferometer (FPI) [9–11], various types of fiber optic temperature sensors have been proposed and demonstrated, such as Mach-Zehnder interferometers [12], distributed sensors [13], fiber Bragg grating, and so on [14–16]. Among them, the FPI has attracted widespread attention due to its advantages of low manufacturing difficulty, low cost, and simple structure.

For a typical FPI sensor, its structure consists mainly of two reflective surfaces, when a beam of light is reflected by the two reflective surfaces multiple times, multiple beams of different phases will be generated, and the light of different phases will interfere after entering the optical fiber. In general, the temperature change of the external environment will cause the phase change of the interference light for two reasons: the thermos-optical coefficient (TOC) and the coefficient of thermal expansion (TEC). This is the principle of FPI sensor temperature sensing. Due to the small TOC and TEC of silica, the sensitivity of traditional all-fiber structure sensors is limited. So high TOC or high TEC materials can be combined with FPI sensing structures to improve temperature sensitivity [17].

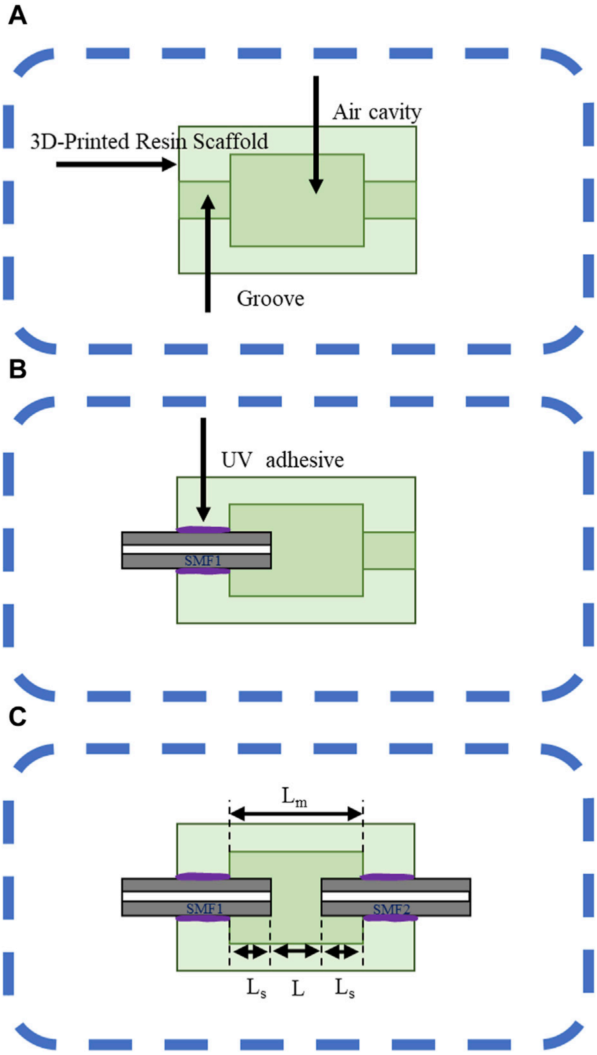


FIGURE 1
Schematic diagram of device fabrication. **(A)** Fabrication of the resin scaffold. **(B, C)** The process of mounting and fixing optical fibers.

In recent years, a method has been proposed to use heat-sensitive materials as auxiliary materials to improve the temperature sensitivity of FPI sensors. B. Sun et al. proposed a novel polymer cap FPI with a temperature response of 249 pm/°C [18]. The sensor is inexpensive to manufacture and the process is very simple, but the polymer cap interference cavity is directly exposed to the air, resulting in insufficient structural strength and easy contamination by the external environment to affect the performance of the sensor. To solve this problem, the polymer was filled into the FPI [10, 19, 20]. In 2018, M.Q. Chen et al. filled the FPI with polydimethylsiloxane (PDMS), which achieved a temperature sensitivity of 2.7035 nm/°C [21]. D. Fu et al. [22] and C. Lang et al. [23] proposed that PDMS and dimethicone oil were injected into the FPI in segments to form a multi-segment air cavity structure. In 2022, H.T. Gao et al., filled a capillary with UV glue, and its temperature sensitivity reached 1226.64 pm/°C [24], but the filling process was complex, and it was difficult to manufacture a thick cavity, even if it had a high TEC, it could not respond to small temperature changes, and the sensitivity was greatly limited. In addition, many materials were combined with FPI sensing structures, such as polymethyl methacrylate (PMMA) [25], Resin [26], Nafion [27], etc.

In this article, we introduce a high-sensitivity temperature sensor designed for intelligent environmental monitoring within Cyber-Physical-Social Systems (CPSS). The sensor's effectiveness has been validated empirically. Engineered to deliver precise temperature readings, our sensor comprises a 3D-printed resin scaffold that embeds dual single-mode optical fibers (SMFs). This configuration facilitates seamless integration with CPSS physical layers. Utilizing a FPI structure, the sensor demonstrated exceptional temperature sensitivity, achieving 14.330 nm/°C, and maintained a linearity of 99.9% across a temperature range from 34.9°C to 38.5°C during testing. Such performance is particularly advantageous in CPSS applications that necessitate stringent temperature regulation and surveillance. Furthermore, attributes such as the sensor's heightened sensitivity, manufacturability, and

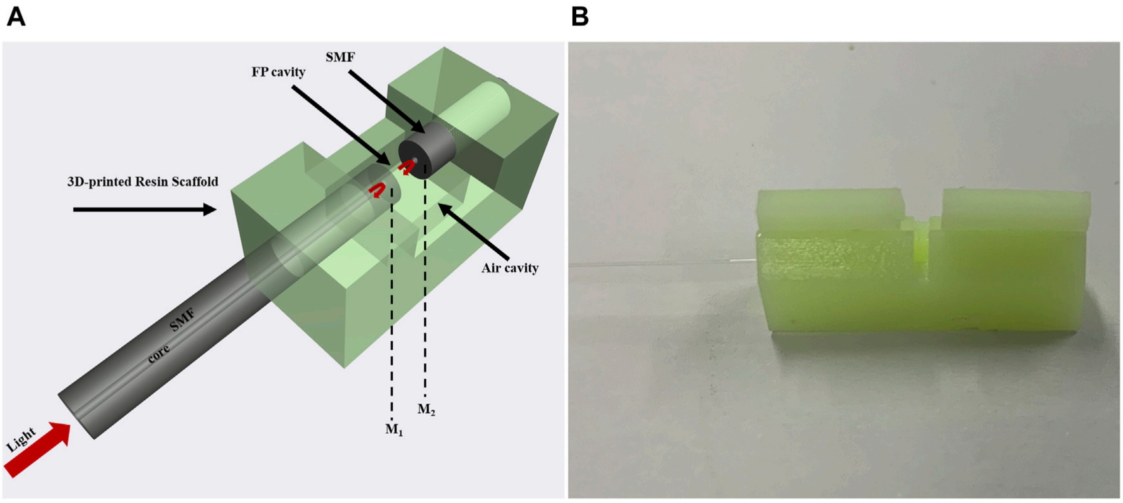


FIGURE 2
(A) Schematic diagram of FPI structure. **(B)** Physical diagram of FPI structure.

cost-effectiveness render it a quintessential component within CPSS infrastructures. The sensor also provides significant insights that advance sensor technology and environmental monitoring research.

2 Structure fabrication and sensing principle

The fabrication process of the FPI structure can be divided into three steps: firstly, a scaffold for embedding optical fibers is 3D printed with resin materials as shown in Figure 1A. Then an optical fiber with a smooth end face is embedded at one end of the resin scaffold and fixed with UV glue, as shown in Figure 1 b). Finally, an optical fiber with a smooth end face is embedded in the same way on the other side of the resin scaffold, so that a Fabry-Perot interferometer is constructed as shown in Figure 1C.

The 3D structure is shown in Figure 2A: two light-reflecting surfaces are constructed on the left and right optical fiber end faces (M_1 and M_2) to form a Fabry-Perot interferometer. The light entering the fiber from the left side propagates in the SMF, partially reflects through M_1, and the remaining light is transmitted in the air cavity and then reflected again in M_2. Eventually, these reflected beams form an interference in the left SMF. The wavelength of the inclination of the reflection spectrum can be defined as:

$$\lambda_m = \frac{4nL}{2m+1}, \tag{1}$$

where m is the order of the interference fringes (m is an integer), n is the effective refractive index of air and L is the F-P air cavity length. The distance between λ_m and λ_{m+1} is called free spectral range (FSR) and can be expressed as:

$$FSR = \frac{\lambda_m^2}{2nL}, \tag{2}$$

as can be seen from Eq. 2, when the refractive index is constant, a change in the length of the cavity will inevitably lead to a change in its FSR. The optical fiber coupling platform is used to control the horizontal movement of the optical fiber to change the cavity length, and the microscope photograph and the corresponding reflectance spectrum are shown in Figure 3. It can be seen that increasing the length of the cavity will lead to a corresponding decrease in the FSR, which is consistent with Eq. 2.

For wavelength demodulation, a change in the total length of the FPI cavity results in a shift in the peak of the interference spectrum. By taking the derivative of Eq. 1, the temperature sensitivity can be expressed as:

$$S_T = \frac{\partial \lambda_m}{\partial T} = \lambda_m \left(\frac{1}{n}\frac{dn}{dT} + \frac{1}{L}\frac{dL}{dT} \right) = \lambda_m (\alpha + \xi), \tag{3}$$

where α refers to the TOC of the cavity medium (air), with a sensitivity less than $-0.86 \times 10^{-6}/^{\circ}C$. ξ refers to the temperature sensitivity of the FPI cavity, which can be expressed as:

$$\frac{TEC_m \times L_m - TEC_s \times L_s}{L}, \tag{4}$$

among them, TEC_m and TEC_s refer to the thermal expansion coefficient of resin material and silicon dioxide, respectively. L_m and L_s refer to the length of resin material and optical fiber, respectively. And L is the length of the FP cavity. The TEC of the resin material is about $1 \times 10^{-4}/^{\circ}C$, and we can see that the TEC of the resin material is two orders of magnitude higher than the TEC of silicon dioxide (about $\times 0.510^{-6}/^{\circ}C$), so the influence of fiber and air in the test can be ignored. Due to the large TEC of the resin material, L_m rapidly elongates or shortens under temperature changes, causing the corresponding elongation or shortening of L, which will cause a wavelength shift in the inclination angle of the interference light according to Eq. (1). For this purpose, we fabricated an FPI sensor with a L_m of about 1068 μm and an L of about 121 μm for testing (Figure 4).

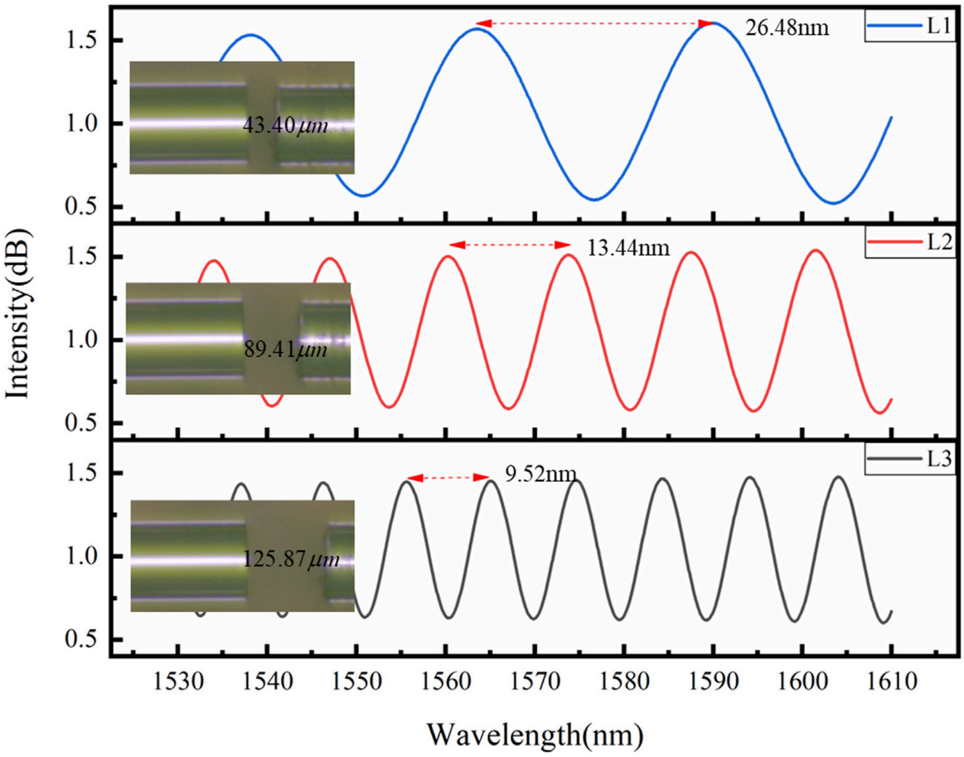


FIGURE 3
Microscope images and Reflectance spectra of sensor samples of different lengths.

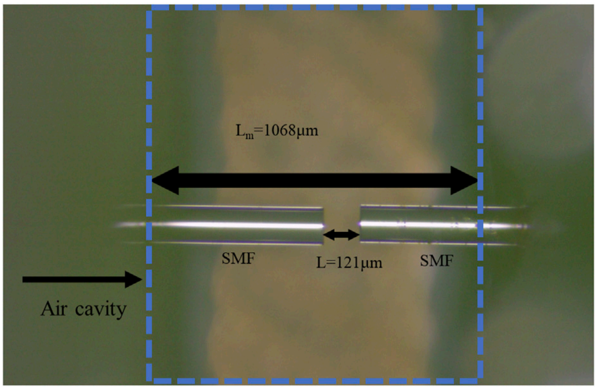


FIGURE 4
Microscope image of the FPI sensor.

3 Experiment and results discussion

The experimental test schematic diagram is shown in Figure 5. The instruments in the system include a fiber optic circulator, a heating platform (0.1°C accuracy), a temperature sensor (0.1°C accuracy), a broadband light source ASE (wavelength range from 1530 nm to 1610 nm), and a spectrum analyzer (OSA, MS9740A with a resolution of 0.1 nm). A broadband light source is emitted from ASE, and the light is fed to the proposed FPI structure via a fiber optic circulator, and the reflected light is collected by a spectrum analyzer. The FPI structure is fixed on the heating platform, and the temperature is controlled by the heating platform. The FPI structure and the temperature sensor are covered with a plastic Petri dish to ensure the uniformity of the temperature in the space above the heating platform, and the real-time temperature in the Petri dish is monitored with a temperature sensor to correct the temperature error. In the experiment, the heating platform provided temperatures from 35 °C to 39 °C, and the actual recorded temperatures under the correction of the temperature sensor were 34.9°C, 35.8°C, 36.6°C, 37.5°C and 38.4 °C.

Figure 6 show the relationship between wavelength and temperature of the FPI structure, with the temperature increasing, the peak wavelength exhibits a redshift, with the temperature decreasing, the peak wavelength exhibits a blue shift. The trough around 1538 nm is selected for tracking, and its temperature sensitivity reaches 14.330 nm/°C and 13.772 nm/°C when the temperature increases and decreases, respectively.

Stability is an important indicator of the sensor, and to test the stability of this FPI structure, we set the temperature to 36.6 °C and recorded the reflectance spectrum every 10 minutes for 1 h. Select four different wavelengths (1543 nm, 1562 nm, 1581 nm, 1600 nm) to track and record its wavelength shift, as shown in Figure 7, the wavelength of each trough is slightly moved back and forth between long wavelength and short wavelength, and the maximum wavelength shift of different troughs were 1.76 nm, 1.84 nm, 1.76 nm and 1.76 nm, and the temperature offset corresponding to wavelength changes is 0.12°C, 0.13°C, 0.12°C and 0.12 °C. Considering the regularity of wavelength back-and-forth shifts,

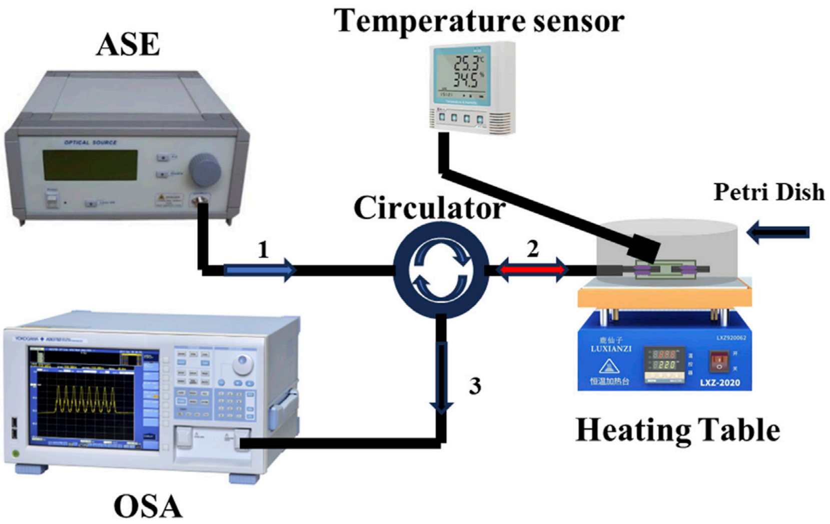


FIGURE 5
Experimental establishment of the temperature sensing system.

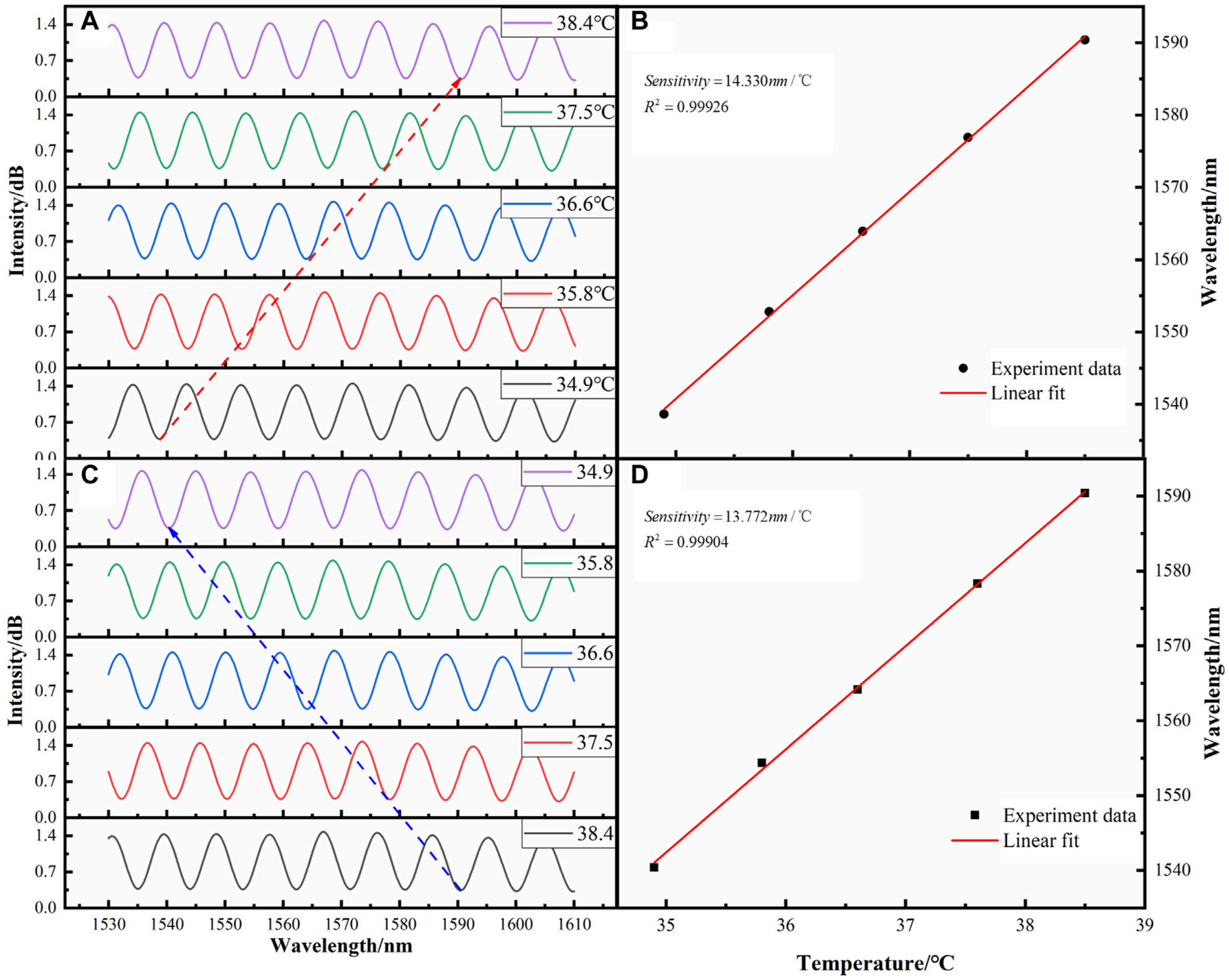


FIGURE 6
(A) The wavelength of the sensor is shifted to the right with increasing temperature. **(B)** Temperature sensitivity of wavelength demodulation. **(C)** The wavelength of the sensor is shifted to the left with decreasing temperature. **(D)** Temperature sensitivity of wavelength demodulation.

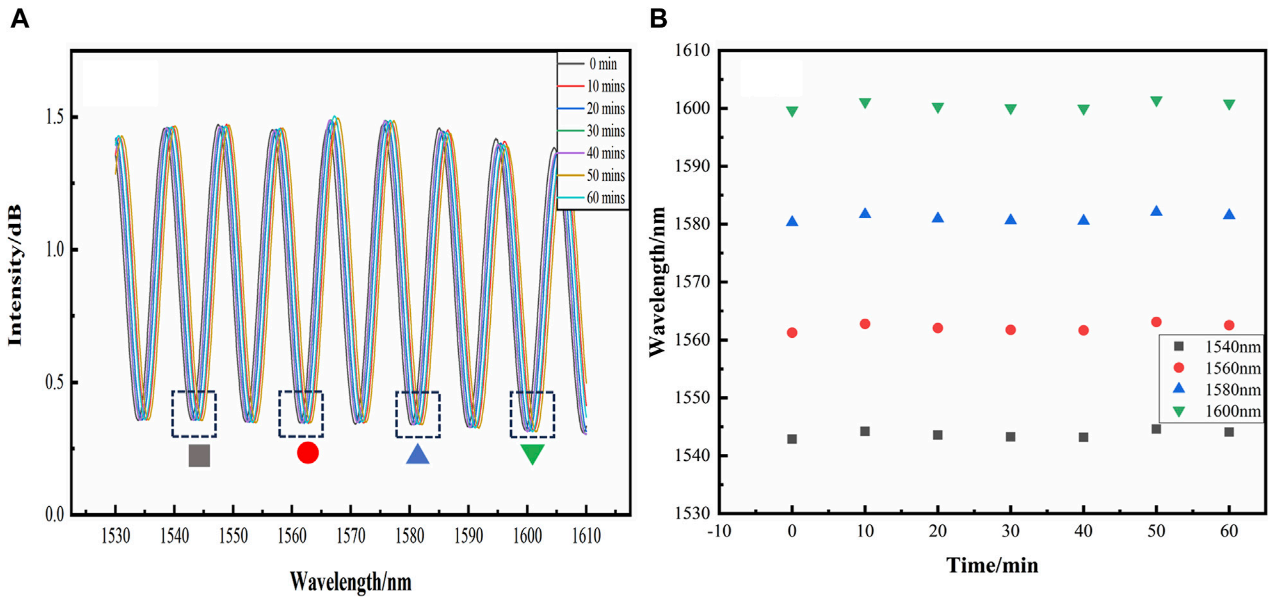


FIGURE 7
Test results of the stability of the sensor at 36.6°C. **(A)** Reflectance spectra recorded at 10-minute intervals. **(B)** Movement of the reflectance spectrum trough.

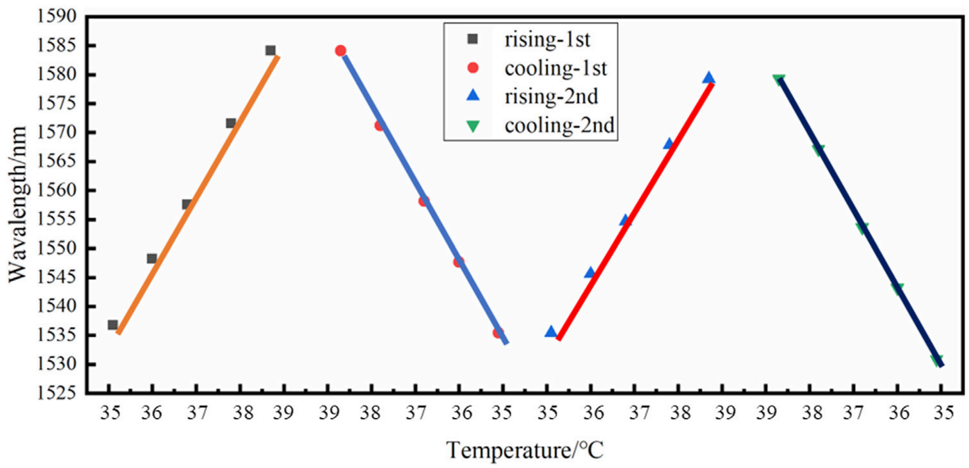


FIGURE 8
Repeatable experimental results of the sensor.

it can be speculated that this phenomenon may be caused by ambient temperature fluctuations.

Repeatability is also an important indicator of the sensor, and we have also performed repeatability experiments on this FPI structure. As shown in Figure 8, we first warmed the heating table from 35.1°C to 38.7°C, then cooled it down to 35.1°C, and then repeated once to get two cycles. From the figure, we can see that after a continuous heating and cooling cycle, the peak finally stays at a shorter wavelength, with a cycle lag of about 5.84 nm, the temperature sensitivity changes are about 0.889 nm/°C and 0.008 nm/°C, respectively.

The proposed temperature sensor is compared with the recently proposed temperature sensors as shown in Table 1. Compared with [17], the sensitivity of the temperature sensor we proposed is greatly improved. In [21], there are multiple reflective surfaces in the sensor, resulting in multiple light interference, which brings great difficulties to the demodulation. The sensing structure in [22] has a higher temperature sensitivity, but the liquid is not stable enough, and the cavity length is difficult to control. In our sensors, the reflective surface is a fiber end face, which is structurally stable, and the cavity length can be adjusted with a fiber coupling platform. In [23], filling the capillary with UV polymer greatly improves the strength and sensitivity of the sensor

TABLE 1 Sensing performance comparison of the proposed sensor with reported sensor.

Sensor structure	Range (°C)	Sensitivity	Ref.
UV polymer-capped	40~90	249 pm/°C	[17]
Capillary/PDMS	20~120	2.62 nm/°C	[21]
Liquid-Air Cavities	34.3~36.1	39.21 nm/°C	[22]
Capillary/UV polymer	39~54	1226.64 pm/°C	[23]
Resin-Fiber Structure	34.9–38.5	14.330 nm/°C	This Work

while controlling the cavity length, but the sensitivity is limited due to the difficulty of making a longer cavity length.

Nevertheless, the proposed sensor still has some shortcomings, due to the F-P open cavity, the reflective surfaces are susceptible to dust contamination in the air. At the same time, compared with the traditional fiber optic sensor, the addition of the resin bracket greatly increases the volume of the sensor, but this problem can be improved by improving the structure of the resin stent.

4 Conclusion

In this study, we address the demand for high-precision environmental monitoring within CPSS by successfully developing and empirically validating a novel high-sensitivity temperature sensor. Uniquely designed to function efficiently within the CPSS framework, this sensor exhibits remarkable sensitivity, with a resolution of 14.330 nm/°C in the critical range of 35–39°C. It is constructed with two coaxial fibers embedded in a resin support, which is fabricated using advanced 3D printing technology, enabling ease of production and demodulation. Moreover, the sensor responds swiftly to real-time environmental changes, enhancing its applicability within CPSS.

Empirical tests have confirmed the sensor's attributes: heightened sensitivity, superb linearity, and steadfast stability within the temperature range of 34.9°C–38.4°C, thereby meeting and exceeding the stringent requirements for accuracy and reliability in CPSS applications.

The significance of this study transcends its technological innovation; it also makes a substantial contribution to the enhancement of smarter, more interconnected CPSS environments. The findings offer an efficacious technical solution for environmental monitoring that has potential implications for smart cities, intelligent transportation systems, and advanced building automation.

In summary, our research introduces an innovative approach to high-precision temperature monitoring and environmental sensing within CPSS, establishing a foundational platform for continued scientific inquiry and practical innovation in the domain. We anticipate that this technology will spur further advancements in CPSS research and contribute substantially to the development of an intelligence-driven, efficient, and secure societal infrastructure.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

YZ: Conceptualization, Methodology, Project administration, Writing–original draft, Writing–review and editing. PZ: Conceptualization, Methodology. ZL: Formal Analysis, Investigation, Writing–review and editing. JS: Methodology, Supervision, Writing–review and editing. CL: Project administration, Resources, Writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work was supported by the Finance Science and Technology Project of Hainan Province (ZDKJ2020009).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Gao H, Jiang Y, Zhang L, Cui Y, Jiang J, Jiang L, et al. Antiresonant mechanism based self-temperature-calibrated fiber optic Fabry–Perot gas pressure sensors. *Opt Express* (2019) 27:22181–9. doi:10.1364/oe.27.022181

2. Li Z, Zhang Y-X, Zhang W-G, Kong L-X, Yan T-Y, Geng P-C, et al. High-sensitivity gas pressure fabry–perot fiber probe with micro-channel based on vernier effect. *J Light Technol* (2019) 37:3444–51. doi:10.1109/jlt.2019.2917062

3. Xu F, Ren D, Shi X, Li C, Lu W, Lu L, et al. High-sensitivity Fabry–Perot interferometric pressure sensor based on a nanothick silver diaphragm. *Opt Lett* (2012) 37:133–5. doi:10.1364/ol.37.000133

4. Quan M, Tian J, Yao Y. Ultra-high sensitivity Fabry-Perot interferometer gas refractive index fiber sensor based on photonic crystal fiber and Vernier effect. *Opt Lett* (2015) 40:4891–4. doi:10.1364/ol.40.004891

5. Ffu HY, Tam HY, Shao L, Dong X, Wai A, Lu C, et al. Pressure sensor realized with polarization-maintaining photonic crystal fiber-based Sagnac interferometer. *Appl Opt* (2008) 47:2835–9. doi:10.1364/ao.47.002835

6. Martynkien T, Statkiewicz-Barabach G, Olszewski J, Wojcik J, Mergo P, Geernaert T, et al. Highly birefringent microstructured fibers with enhanced sensitivity to hydrostatic pressure. *Opt Express* (2010) 18:15113–21. doi:10.1364/oe.18.015113

7. Lin C, Qiu C, Jiang H, Zou L. A deep neural network based on prior-driven and structural preserving for SAR image despeckling. *IEEE J Selected Top Appl Earth Observations Remote Sensing* (2023) 16:6372–92. doi:10.1109/jstars.2023.3292325

8. Roriz P, Silva S, Frazao O, Novais S. Optical fiber temperature sensors and their biomedical applications. *Sensors* (2020) 20(7):2113. doi:10.3390/s20072113

9. Vargas-Rodriguez E, Guzman-Chavez AD, Baeza-Serrato R, Garcia-Ramirez MA. Optical fiber FP sensor for simultaneous measurement of refractive index and temperature based on the empirical mode decomposition algorithm. *Sensors* (2020) 20(3):664. doi:10.3390/s20030664

10. Li M, Liu Y, Gao R, Li Y, Zhao X, Qu S. Ultracompact fiber sensor tip based on liquid polymer-filled Fabry-Perot cavity with high-temperature sensitivity. *Sensors Actuators B: Chem* (2016) 233:496–501. doi:10.1016/j.snb.2016.04.121

11. Kong LX, Zhang YX, Zhang WG, Zhang YS, Yu L, Wang S, et al. High-sensitivity and fast-response fiber-optic micro-thermometer based on a plano-concave Fabry-Pérot cavity filled with PDMS. *Sensors Actuators A: Phys* (2018) 281:236–42. doi:10.1016/j.sna.2018.09.004

12. Liu Z, Xiao H, Liao M, Han X, Chen W, Zhao T, et al. PDMS-assisted microfiber MZ interferometer with a knot resonator for temperature sensing. *IEEE Photon Technol. Lett.* (2019) 31(5):337–40. doi:10.1109/lpt.2019.2892781

13. Mizuno Y, Theodosiou A, Kalli K, Liehr S, Lee H, Nakamura K. Distributed polymer optical fiber sensors: a review and outlook. *Photon Res* (2021) 9(9):1719–33. doi:10.1364/prj.435143

14. Xiong Y, Xu F. Multifunctional integration on optical fiber tips: challenges and opportunities. *Adv Photon* (2020) 2(06):64001. doi:10.1117/1.ap.2.6.064001

15. Liu Y, Yang D, Wang Y, Zhang T, Shao M, Yu D, et al. Fabrication of dual-parameter fiber-optic sensor by cascading FBG with FPI for simultaneous measurement of temperature and gas pressure. *Opt Commun* (2019) 443:166–71. doi:10.1016/j.optcom.2019.03.034

16. Urrutia A, Goicoechea J, Ricchiuti AL, Barrera D, Sales S, Arregui FJ. Simultaneous measurement of humidity and temperature based on a partially coated optical fiber long-period grating. *Sensors Actuators B: Chem* (2016) 227:135–41. doi:10.1016/j.snb.2015.12.031

17. Uyor UO, Popoola API, Popoola OM, Aigbodion VS. Polymeric cladding materials under high temperature from optical fibre perspective: a review. *Polym Bull* (2020) 77(4):2155–77. doi:10.1007/s00289-019-02830-y

18. Sun B, Wang Y, Qu J, Liao C, Yin G, He J, et al. Simultaneous measurement of pressure and temperature by employing Fabry-Perot interferometer based on pendant polymer droplet. *Opt Express* (2015) 23(3):1906–11. doi:10.1364/oe.23.001906

19. Cao K, Liu Y, Qu S. Compact fiber biocompatible temperature sensor based on a hermetically-sealed liquid-filling structure. *Opt Express* (2017) 25(24):29597–604. doi:10.1364/oe.25.029597

20. Wang C, Zhou B, Jiang H, He S. Agarose filled Fabry–Perot cavity for temperature self-calibration humidity sensing. *IEEE Photon Technol. Lett.* (2016) 28(19):2027–30. doi:10.1109/lpt.2016.2581990

21. Chen M, Zhao Y, Xia F, Peng Y, Tong R. High sensitivity temperature sensor based on fiber air-microbubble Fabry-Perot interferometer with PDMS-filled hollow-core fiber. *Sensors Actuators A: Phys* (2018) 275:60–6. doi:10.1016/j.sna.2018.03.044

22. Fu D, Liu X, Shang J, Sun W, Liu Y. A simple, highly sensitive fiber sensor for simultaneous measurement of pressure and temperature. *IEEE Photon Technol. Lett.* (2020) 32(13):747–50. doi:10.1109/lpt.2020.2993836

23. Lang C, Liu Y, Liao Y, Li J, Qu S. Ultra-sensitive fiber-optic temperature sensor consisting of cascaded liquid-air cavities based on Vernier effect. *IEEE Sens J* (2020) 20(10):5286–91. doi:10.1109/jsen.2020.2970431

24. Gao H, Xu D, Ye Y, Zhang Y, Shen J, Li C. Fiber-tip polymer filled probe for high-sensitivity temperature sensing and polymer refractometers. *Opt Express* (2022) 30(5):8104–14. doi:10.1364/oe.449852

25. Salunkhe TT, Choi HW, Park SJ, Kim JH, Kim IT. High sensitivity temperature sensor based on Fresnel reflection with thermosensitive polymer: control of morphology and coating thickness. *Jpn J Appl Phys* (2020) 59(SG):SGGG06. doi:10.7567/1347-4065/ab5c7d

26. Dominguez-Flores CE, Monzon-Hernandez D, Moreno-Basulto JI, Rodriguez-Quiroz O, Minkovich VP, Lopez-Cortes D, et al. Real-time temperature sensor based on in-fiber fabry-perot interferometer embedded in a resin. *J Lightwave Technol* (2019) 37(4):1084–90. doi:10.1109/jlt.2018.2886134

27. Liu S, Ji Y, Yang J, Sun W, Li H. Nafion film temperature/humidity sensing based on optical fiber Fabry-Perot interference. *Sensors Actuators A: Phys* (2018) 269:313–21. doi:10.1016/j.sna.2017.11.034

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 05 April 2024
DOI 10.3389/fphy.2024.1336795

OPEN ACCESS

EDITED BY
Xin Lu,
De Montfort University, United Kingdom

REVIEWED BY
E. Zhang,
University of Leicester, United Kingdom
Yinong Chen,
Arizona State University, United States
Amin Ul Haq,
University of Electronic Science and Technology of China, China

*CORRESPONDENCE
Qianhua Cai,
caiqianhua@m.scnu.edu.cn

RECEIVED 11 November 2023
ACCEPTED 11 March 2024
PUBLISHED 05 April 2024

CITATION
Huang Y, Dai A, Cao S, Kuang Q, Zhao H and Cai Q (2024), A dual contrastive learning-based graph convolutional network with syntax label enhancement for aspect-based sentiment classification.
Front. Phys. 12:1336795.
doi: 10.3389/fphy.2024.1336795

A dual contrastive learning-based graph convolutional network with syntax label enhancement for aspect-based sentiment classification

Yuyan Huang[1,2], Anan Dai[3], Sha Cao[4], Qiuhua Kuang[1,5], Hongya Zhao[6] and Qianhua Cai[1]*

[1]Department of Electronics and Information Engineering, South China Normal University, Foshan, China, [2]Datastory, Guangzhou, China, [3]China Merchants Bank Foshan Branch, Foshan, China, [4]Department of Financial Mathematics, University of Chicago, Chicago, IL, United States, [5]Guangzhou Qizhi Information Technology Co., Ltd., Guangzhou, China, [6]Industrial Centre School of Undergraduate Education, Shenzhen Polytechnic University, Shenzhen, China

Introduction: Aspect-based sentiment classification is a fine-grained sentiment classification task. State-of-the-art approaches in this field leverage graph neural networks to integrate sentence syntax dependency. However, current methods fail to exploit the data augmentation in encoding and ignore the syntactic relation in sentiment delivery.

Methods: In this work, we propose a novel graph neural network-based architecture with dual contrastive learning and syntax label enhancement. Specifically, a contrastive learning-based contextual encoder is designed, integrating sentiment information for semantics learning. Moreover, a weighted label-enhanced syntactic graph neural network is established to use both the syntactic relation and syntax dependency, which optimizes the syntactic weight between words. A syntactic triplet between words is generated. A syntax label-based contrastive learning scheme is developed to map the triplets into a unified feature space for syntactic information learning.

Results: Experiments on five publicly available datasets show that our model substantially outperforms the baseline methods.

Discussion: As such, the proposed method shows its effectiveness in aspect-based sentiment classification tasks.

KEYWORDS

aspect-based sentiment classification, graph convolutional networks, dual contrastive learning, syntax label enhancement, bidirectional encoder representations from transformers (BERT)

1 Introduction

Aspect-based sentiment classification (ABSC) is a fundamental task in sentiment analysis [1]; [2], which aims to infer the sentiment of a specific aspect in sentences [3]. Generally, the sentiment of each aspect is classified according to a predefined set of sentiment polarities, i.e., positive, neutral, or negative. For example, in the comment “the price is reasonable, although the service is poor,” the sentiment toward aspects “price” and “service” is positive and negative, respectively.

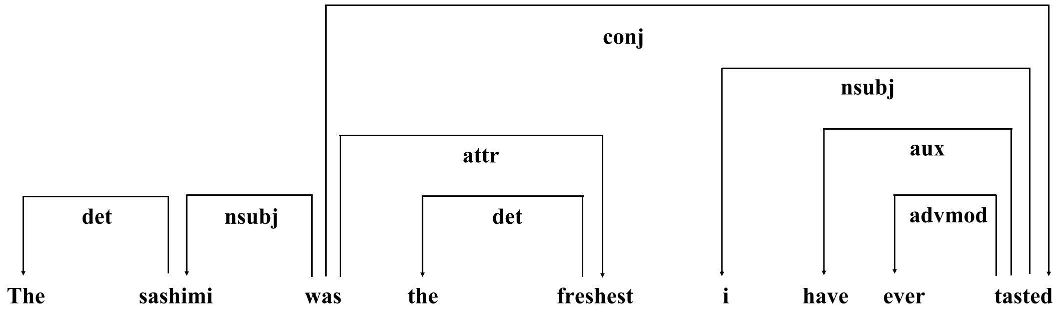


FIGURE 1
Example of syntax dependency parsing.

In general, an ABSC process involves two steps: the identification of sentiment information toward the aspect from the context and the classification of the expressed sentiment from predefined sentiment polarities [4]. Comprehensively, the first step contains key contextual information learning and aspect–context word relation establishment. To capture important contextual words and prevent redundant information, recent publications reveal that encoders and attention networks are taken to encode the sequential information and determine the attentive weights of contexts, respectively [5]. Typically, these deep learning methods are trained via a large amount of textual data to improve their working performance. Notwithstanding, the existing manually annotated data resources are still limited, which causes issues such as model overfitting. As a result, the precise capturing of key contextual words remains challenging. More recently, contrastive learning shows its superiority under the condition of limited training samples. Based on data augmentation, both positive and negative samples are generated. By setting contrastive loss of training models, the representations of positive samples are brought closer, while those of negative samples are pushed apart. In line with the contrastive learning, the model training can be improved, which paves a way for key contextual information learning in ABSC tasks.

On the basis of key contextual information, the aspect–opinion word relation mainly lies in syntax dependency of the sentence [6]. With the parsing of syntax dependency, the relation between the aspect and context words is built. Ongoing studies substantially focus solely on the distance of words while neglecting the syntax label of specific context words toward the aspect. That is, all syntactic relations are interpreted as the same. Figure 1 shows the syntax structure of a given sentence. The establishment of the subject–predicate syntactic relation (nsubj) and adjective modifier syntactic relation (attr) plays a dominate role in sentiment classification, especially compared with other syntactic information. Moreover, the syntax label is also the foundation of textual logical reasoning due to its effects in distinguishing the importance among syntactic relations. So much is the significance of the syntax label that it can be further applied to the aspect–opinion word relation establishment in ABSC.

To address the above issues in ABSC, we propose a graph convolutional network (GCN) based on dual contrastive learning and syntax label enhancement (i.e., DCL-GCN). First, a contrastive learning-based encoder is devised, which brings the context representations of the same sentiment closer and pushes those of different sentiments apart. Furthermore, a weighted label-enhanced syntactic GCN is put forward, dealing with not only the syntactic relation but also the syntax dependencies among words. Lastly, a contrastive learning scheme that focuses on the sentence syntax label is developed. A syntactic triplet between words is constructed. The same syntax label-based triplets are given similar semantic representations, while different syntax label-based triplets are distinguished. Thereby, the syntax and semantics are integrated, which contributes to the sentiment classification.

The contribution of our work is three-fold and given as follows:

- A GCN-based ABSC method is proposed with the integration of dual contrastive learning and syntax label enhancement. Specifically, the sentence is encoded using contrastive learning to bring the context representations of the same sentiment closer and push those of different sentiments apart.
- A weighted label-enhanced syntactic GCN and a contrastive learning scheme are established to tackle the sentence syntax. A syntactic triplet between words can be generated. The same syntax label-based triplets are given similar semantic representations to facilitate the ABSC.
- Experiments conducted on five benchmark datasets demonstrate that our model achieves state-of-the-art results. The proposed method significantly improves the working performance compared to competitive baselines in the ABSC field.

2 Related work

Owing to the advancement of deep learning networks, current methods with various structures are widely developed, aiming to identify their superiority in ABSC tasks [7]. ABSC models are devised to deal with either semantics [8], syntax [9], or both [10] from the given text. In this section, these two major issues in the field of ABSC are presented. In order to achieve better working performance, previous work and their findings about these two focuses are dedicatedly investigated and depicted.

2.1 Contextual information learning

One bottleneck in ABSC comes from capturing key contextual words, which considerably affects the aspect–opinion word relation modeling. Much recent work uses neural networks, attention networks, or both to concentrate on useful contextual information [11]; [12]. Tang

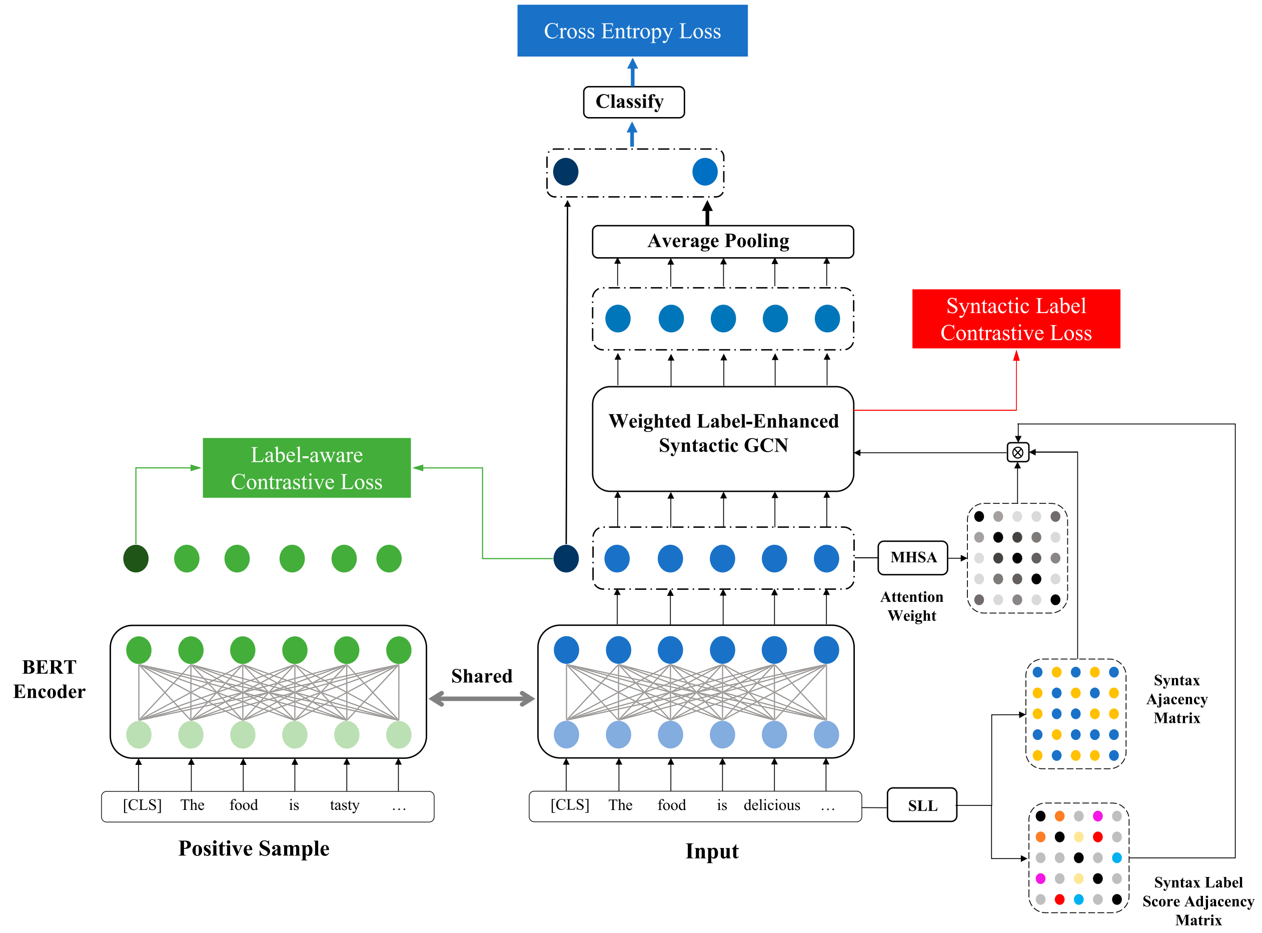


FIGURE 2
Model architecture.

et al. focused on different contextual parts based on LSTM, targeting at obtaining valuable information [13]. In addition, attention-based neural networks are proposed to discriminate more relevant features toward the aspect [14]; [15]. Sun et al. used a BERT-based model to capture semantic features from contexts via fine-tuning, which significantly improves the working performance [16]. Text encoders are widely applied to various tasks [17,18]. Encouragingly, advances in contrastive learning hold great potential in natural language processing (NLP) tasks. Suresh et al. integrated contrastive learning strategy into the pre-training of Bidirectional Encoder Representations from Transformers (BERT) to improve the model efficacy [19]. A contrastive loss among different input categories is introduced, while a weight network refines the differences between each sample pair. In our work, contrastive learning can be taken to distinguish the contextual representations during sentence encoding.

2.2 Syntax dependency parsing

The parsing of syntax dependency plays a pivotal role in the field of ABSC due to its relation establishment between the aspect and contextual words. Previous work primarily tackles the syntactic relation of either single or multiple word pairs. In recent years, the application of a GCN in NLP gave rise to new opportunities in a number of fields [20]; [21]. Regarding sentence syntax parsing, Sun et al. transformed the syntax dependency into an adjacency matrix and propagated the syntactic information using the GCN [22]. Furthermore, Zhang et al. incorporated the aspect-oriented attention mechanism to benefit the contextual information extraction toward a specific aspect [23]. To extract both aspect-focused and inter-aspect sentiment information, an interactive graph convolutional network (InterGCN) is built to leverage the sentiment dependencies of the context [24]. Wang et al. reconstructed the syntax dependency tree rooted at an aspect. A relational graph attention network (R-GAT) is then proposed to encode the aspect-oriented dependency tree and to establish the syntactic relation between the aspect and its opinion words [25].

3 Methodology

A dual contrastive learning GCN (DCL-GCN) is devised on the task of ABSC. Figure 2 shows the framework of the proposed model. A pretrained BERT model is used as the sentence encoder. A contrastive learning scheme is incorporated into contextual encoding during model training, which enhances the semantic information via sentiment labels

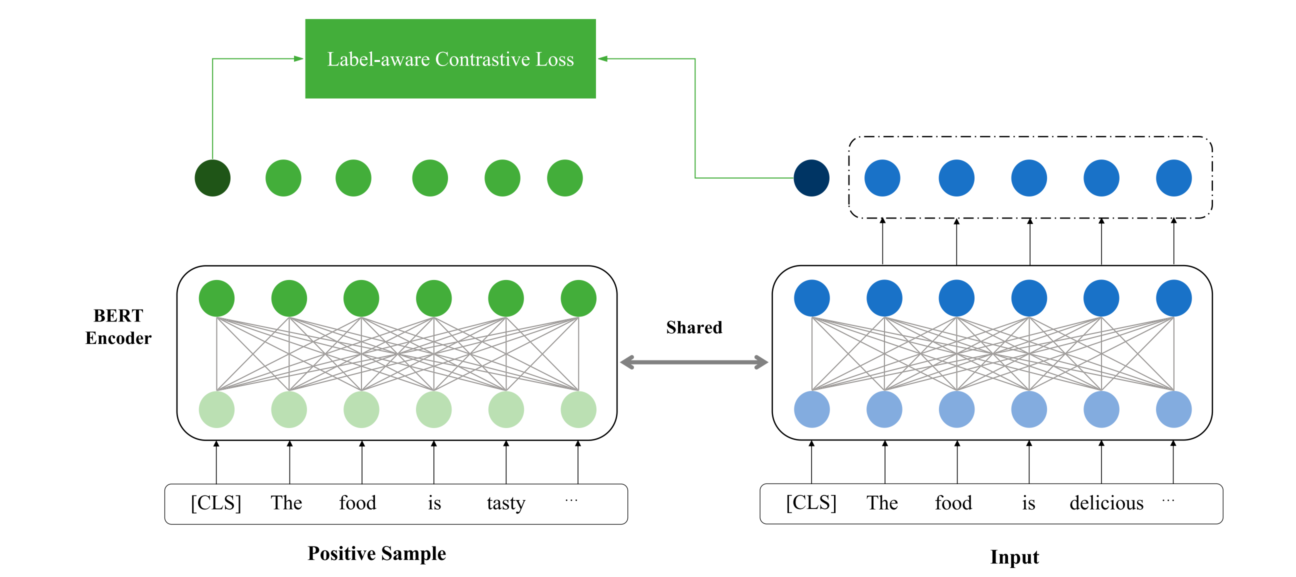

Label-aware Contrastive Loss
BERT Encoder
Shared
[CLS] The food is tasty ···
Positive Sample
[CLS] The food is delicious ···
Input

FIGURE 3
Contrastive learning-based contextual encoder.

to obtain differentiated contextual representations. Then, both the semantic and syntactic features are integrated within a weighted label GCN, aiming at addressing the syntactic relation of context words with the aspect. In line with contrastive learning, the syntax labels of words are used for learning the sentence syntax at a higher level. The sentiment polarity is predicted by sending the final sentence representation into a sentiment classifier. More details of the proposed model are given as follows:

3.1 Contextual encoder with contrastive learning

The architecture of the contrastive learning-based contextual encoder is shown in Figure 3. Let $X = [w_1, \ldots, w_a, \ldots, w_{a+m-1}, \ldots, w_n]$ be a sentence of n words and $A = [w_a, \ldots, w_{a+m-1}]$ be the aspect of m words within S. The contrastive learning scheme during sentence encoding is implemented via data augmentation, feature extraction, and contrastive loss construction. Inspired by the data augmentation in image recognition [26]; [27], positive samples of the same polarities are generated using synonym substitution and random noise injection. Specifically, synonym substitution refers to randomly replacing words within the sentence with their synonyms from WordNet, while noise injection indicates introducing more aspect words and neutral sentiment words to the sentence. The sentiment is enhanced in (1):

$$X^E = Enchance(X). \tag{1}$$

The original sentence X and the data-enhanced sentence X^E are mapped to word vectors within the same feature space. We use the BERT model obtained through large-scale corpus training by Kenton et al to enhance the semantics of word representations. We then train the BERT model in a fine-tuned manner by freezing part of its parameters, which is written in (2):

$$h^{CLS}, H^X, H^A = BERT(CLS, X, SEP, A, SEP), \tag{2}$$

where CLS and SEP are BERT tokens representing the overall representation and the separation of the sentence, respectively. We thus obtain the sentence-level feature representation h^{CLS}, the word-level feature representation H^X, and the aspect feature representation H^A. Assuming that a batch consisting of k sentences is the model input for training, the sentence set composed of the original and the enhanced sentences is $X^{all} = [X^{batch}, (X^E)^{batch}] = [X_1, X_2, \ldots, X_{2k}]\}$, with the corresponding sentiment polarity set denoted as $Y^{all} = [Y_1, Y_2, \ldots, Y_{2k}]$. We also have the index set of all sentences as $I = [1, 2, \ldots, 2k]$. For each sentence in X^{all}, a set of contrastive learning-based sentences with the same sentiment polarity is generated, i.e., $P^{all} = [P_1, P_2 \ldots, P_{2k}]$, where $P_i = \{p: p \in I\ (Y_p = Y_i) \wedge (p \neq i)\}$. The contrastive learning loss of the contextual encoder is defined in (3):

$$\mathcal{L}_{ECL} = \sum_{i=1}^{2k} \frac{-1}{|P_i|} \sum_{p\in P_i} log \frac{\exp\left(h_i^{CLS} \star h_p^{CLS} / \tau\right)}{\sum_{k\in I/i} \exp\left(h_i^{CLS} \star h_k^{CLS} / \tau\right)}, \tag{3}$$

where τ is a hyperparameter, indicating the temperature coefficient of contrastive learning. The higher the temperature coefficient is, the smaller the sum of the loss reaches. The parameter h_i^{CLS} stands for the representation of the ith sentence in X^{all} after BERT coding. In such a manner, the context representations of the same sentiment can be brought closer, and those of different sentiments are separated, improving the use of contextual and sentiment labels. Based on contrastive learning, abundant semantic information is integrated into the encoder, targeting at deriving context representation with key information.

3.2 Weighted label-enhanced syntactic GCN

The framework of the syntactic GCN via weighted label enhancement is presented in Figure 4. The syntax dependency of the input sentence is derived using the spaCy toolkit. Specifically, the sentence syntax dependency is characterized by a triplet, i.e., $(w_i, w_j, r_{i,j})$, where words w_i and w_j are of the relation

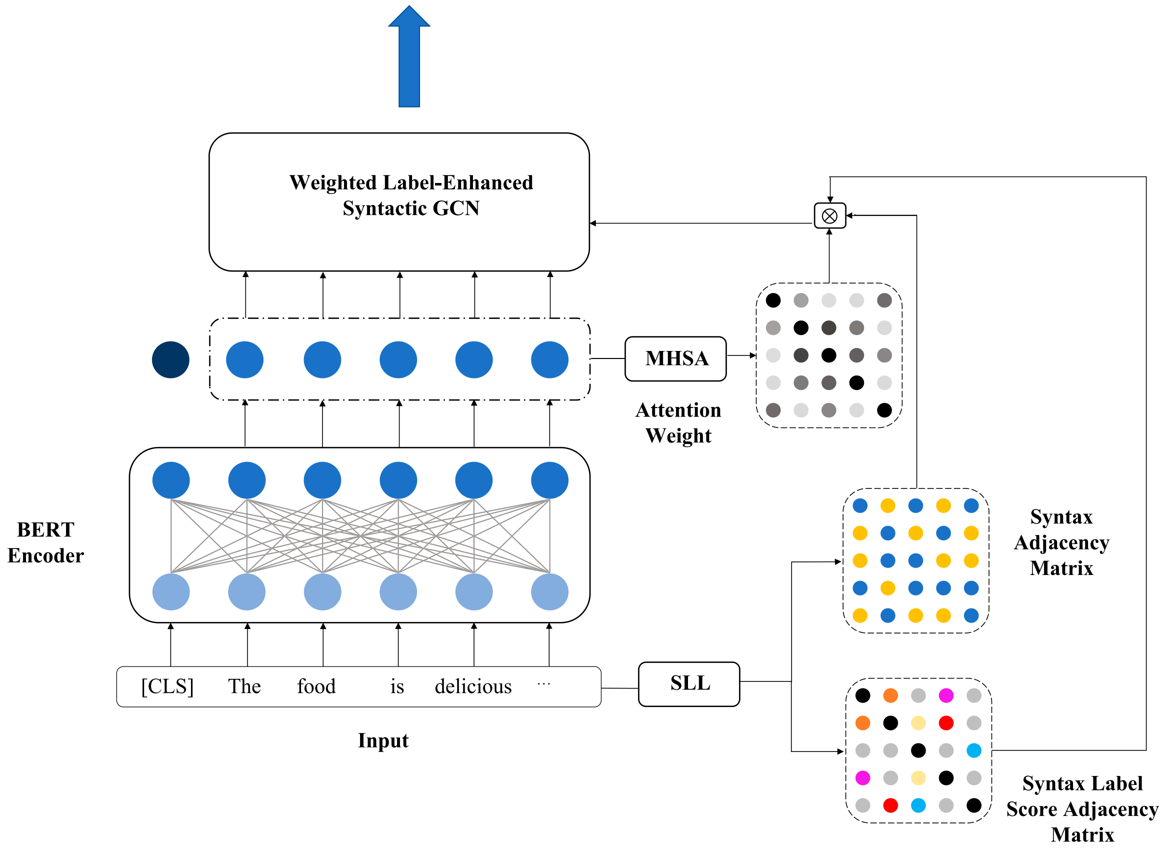


FIGURE 4
Framework of the weighted label-enhanced syntactic graph convolutional network (GCN).

$r_{i,j}$. In line with the sentence syntax, we construct a syntax adjacency matrix $A^S \in \mathbb{R}^{n \times n}$ that denotes the connecting edges of the syntax dependency tree.

To address the effects of various syntactic labels in the sentiment classification, a syntax label learning (SLL) unit is built. The main purpose of the SLL unit is to transform the syntax label matrix to a learnable syntax label score matrix.

A lexicon $R = \{relation1{:}\ 1,\ relation2{:}\ 2,\ \ldots,\ relationt{:}\ t\}$ that consists of all syntactic relations from the corpus is constructed, from which each syntactic label is mapped to an index number. For each input sentence, all index numbers denote the syntactic relations consisting of a syntax label matrix. Then, the syntax label adjacency matrix $A^L \in \mathbb{R}^{n \times n}$ is built based on both the syntax dependency tree and the lexicon R. All syntax labels can be mapped into a unified feature space. The weighted score of each syntactic relation is thus resolved in (4), which is written as a syntax label score adjacency matrix $A^{LS} \in \mathbb{R}^{n \times n}$:

$$A^{LS} = Emb\left(A^L\right)^{*} W_L{}^{*} W_S, \tag{4}$$

where $Emb\,(\cdot)$ represents transforming the syntax label matrix into a learnable matrix for syntax label characterization, $W^L \in \mathbb{R}^{d_L \times d_S}$ and $W^S \in \mathbb{R}^{d_S \times 1}$ are learnable parameter matrices, and d_L and d_S are dimensions of A^L and A^S, respectively. We also have $Emb\left(A^L\right) \in \mathbb{R}^{n \times n \times d_{LS}}$, with d_{LS} standing for the dimension of the syntax label score space.

Likewise, the same syntactic relation type can have different degrees of importance within different semantic contexts. For this reason, the semantics among words are also integrated into the computation of the syntax label score. We take the multi-head self-attention (MHSA) mechanism to learn the semantic features and to revise the syntax label scores based on attentive weights. Notably, the elements in A^{LS} represent all the syntactic relation scores, which are not zero. To preserve the original syntax dependencies and remove irrelevant syntactic information, the basic syntax adjacency matrix is also used. The weighted syntax label adjacency matrix can be computed in (5):

$$A_{ij}^{WL} = A_{ij}^{S} * A_{ij}^{W} * A_{ij}^{LS} \tag{5}$$

with (6, 7, and 8)

$$A^W = Norm\Big(Concat\left(head_1, head_2, \ldots, head_h\right) \cdot W^{head}\Big), \tag{6}$$

$$head_p = \frac{QW_p^Q \times \left(KW_p^K\right)^T}{\sqrt{d_{head}}}, \tag{7}$$

$$Q = W = H^X, \tag{8}$$

where A_{ij}^S is the syntax adjacency matrix of w_i and w_j parsed from the syntax dependency tree; A_{ij}^{LS} is the syntax label score adjacency matrix; A_{ij}^W stands for the semantic weight adjacency matrix derived from the MHSA mechanism; *Concat* represents the vector

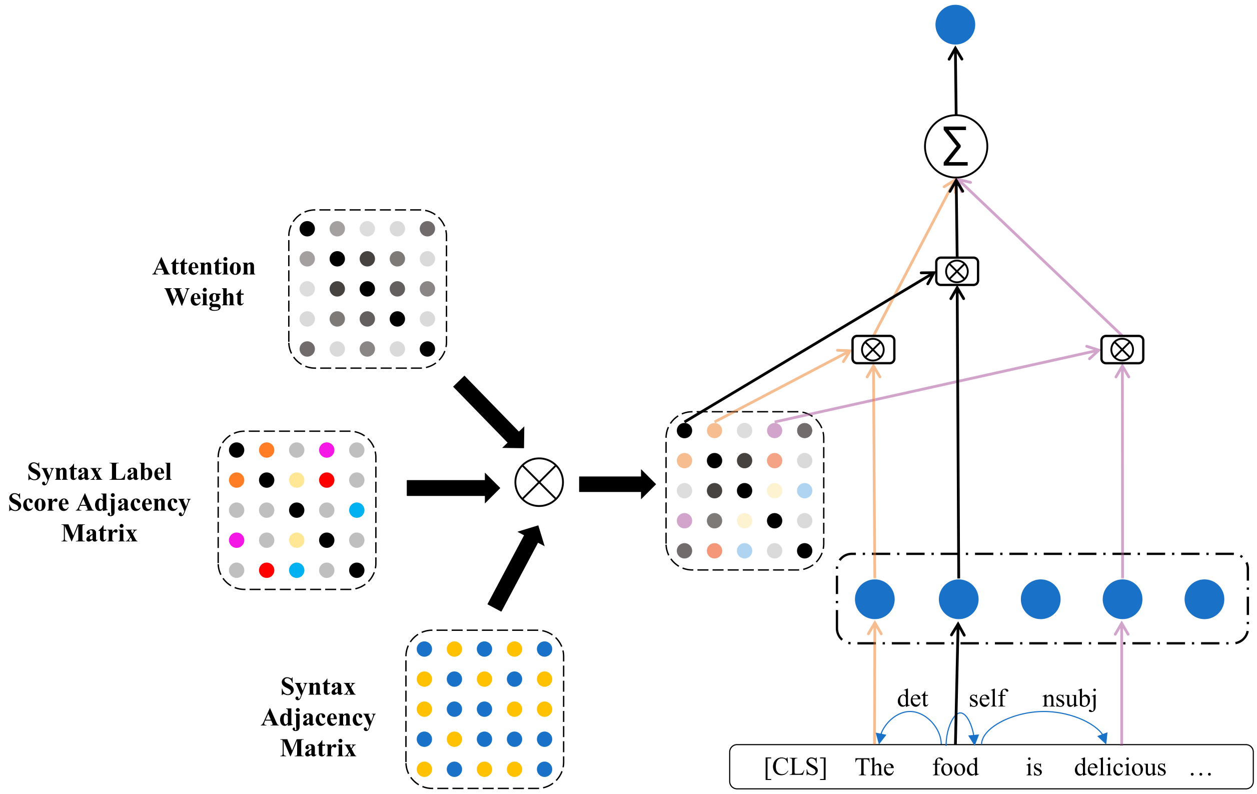

Attention Weight
Syntax Label Score Adjacency Matrix
Syntax Adjacency Matrix
Σ
det
self
nsubj
[CLS] The food is delicious …

FIGURE 5
Working principle of the weighted label GCN.

concatenation; W^{head} is the parameter matrix during concatenation; $Norm\ (\cdot)$ is the normalization operation on the attentive weight matrix; W_p^Q and W_p^K are parameter matrices of the pth attention head in MHSA; d_{head} denotes the vector dimensions of each head; and h is a hyperparameter indicating the attention head number.

The working principle of the weighted label-enhanced syntactic GCN is shown in Figure 5. The input of the GCN is the weighted syntax label adjacency matrix A^{WL} and the feature representation H^X from BERT. The learning of syntactic information is derived in (9):

$$h_i^l = \sigma\left(\sum_{j=l}^{n} A_{ij}^{WL} W_S^l h_j^l + b_S^l\right), \tag{9}$$

where $H^0 = H^X = [h_1^X, h_2^X, \ldots, h_n^X]$, h_j^lrefers to the word vector of the jth word in the lth layer of the GCN, with l as an integer and $l \in [0, F]$, F is the layer number of the GCN, W_S^l is the learnable parametric matrix of the lth layer, b_S^l is the bias vector, and σ is an activation function. The output of the weighted label-enhanced syntactic GCN is the output of the last layer, i.e., $H^{out} = [h_1^F, h_2^F, \ldots, h_n^F]$.

3.3 Syntax label-based contrastive learning scheme

Considering the effect of syntactic information in ABSC, the node pairs with the same syntax label indicate similar syntactic features, and those with different syntax labels have differentiated features. As such, a contrastive learning scheme using syntax labels is proposed, aiming to enhance the learning of syntactic features at a higher level.

Assuming that K' triplets are of syntax dependencies within all the K sentences, the node-pair set of these triplets is $X' = [X_1', X_2', \ldots, X_{K'}']$. The syntax label set of these node pairs is $R' = [r_1, r_2, \ldots, r_{K'}]$ with the index set $I' = [1, 2, \ldots, K']$. Moreover, for each node pair in X', a set of node pairs with the same syntax label for contrastive learning is constructed, i.e., $P_{m'}' = \{p' \colon p' \in I', (r_{p'} = r_{p'}) \wedge (p' \neq m')\}$. The syntax label-based contrastive learning loss is defined in (10):

$$\mathcal{L}_{LCL} = \sum_{m'=1}^{K'} \frac{-1}{|P_{m'}'|} \sum_{p' \in P_{m'}'} log \frac{\exp\left(g_{m'} {}^{*} g_{p'} / \tau'\right)}{\sum_{t \in I' / m'} \exp\left(g_{m'} {}^{*} g_t / \tau'\right)}, \tag{10}$$

together with (11)

$$g_{m'} = \left(X_{m'}' [1] W_1^{cl} + X_{m'}' [2] W_2^{cl}\right) W_3^{cl} + b^{cl}, \tag{11}$$

where τ' is the temperature coefficient for contrastive learning and $g_{m'}$ represents the semantic feature representation by mapping the node-pair representations from the syntax dependency triplet and is normalized before the contrastive learning loss computation. We define $X_{m'}' [1]$ as the feature representation of the first node in the m'th node pair in X' and $X_{m'}' [2]$ as the feature representation of the second node in the m'th node pair. Both $X_{m'}' [1]$ and $X_{m'}' [2]$ are obtained from the BERT encoder, which convey semantic information. In addition,

W_1^{cl}, W_2^{cl}, and W_3^{cl} are learnable parameter matrices, and b^{cl} is a bias vector.

3.4 Feature fusion

Average pooling is performed on H^{out} to obtain the syntactic information-enhanced feature representation H^{out} in (12), which is further concatenated with h^{CLS} derived from the BERT encoder. The final sentence representation $\tilde{H}$ is given in (13).

$$h^{out} = avgpool\left(h^{out}\right), \tag{12}$$

$$\tilde{H} = h^{out} \oplus h^{CLS}, \tag{13}$$

where ⊕ denotes the concatenation operation. The final sentence representation $\tilde{H}$ is sent to a Softmax classifier to obtain the sentiment polarity in (14):

$$y = softmax\left(W_o\tilde{H} + b_o\right). \tag{14}$$

The pseudocode of the proposed model is given as follows:

```
Input: data $D$, $batch\_size$ $N$.
Output: Sentiment polarity $y$
for $i = 0$ to $n$ by $N$ do
  $batch \leftarrow D\,[i : i + N]$
    for $j$ in $[i,\ i + batch\_size)$ do
      $h_j^{CLS}, H_j^X = BERT(X_j)$
      $A_S,\ A_W \leftarrow SLL\ (X_j)$
      $A^W \leftarrow MHSA(H_j^X)$
      $A^{WL} = A^S{*}A^W{*}A^{LS}$
      $H_j^{out} \leftarrow$ Weighted_Label_Enhanced_GCN($A^{WL}, H_j^X$)
      $\tilde{H}_j \leftarrow$ Concatenate_Features($H_j^{out}, h_j^{CLS}$)
      $y_j =$ Softmax($\tilde{H}_j$)
    end for
    $\mathcal{L}_{total} = \mathcal{L}_{CE} + \alpha\mathcal{L}_{ECL} + (1-\alpha)\mathcal{L}_{LCL}$
    Update network by combined loss $\mathcal{L}_{total}$
end for
```

Algorithm 1. Dual contrastive learning-based GCN forward propagation algorithm.

3.5 Model training

Model training is implemented using cross-entropy and regularization as the loss function in (15):

$$\mathcal{L}_{CE} = -\sum_{(x,a)\in D}\sum_{c\in C} y_{(x,a)}^c log\hat{y}_{(x,a)}^c + \lambda\|\theta\|^2, \tag{15}$$

where $(x,\ a)$ represents the vector of a sentence–aspect pair; C refers to the set of sentiment classes; $y_{(x,a)}^c$ is the ground-truth sentiment distribution of $(x,\ a)$ with sentiment C, and $\hat{y}_{(x,a)}^c$ is the predicted one; and λ is the coefficient of $\mathcal{L}_2$ regularization.

On account of the training of contrastive learning in our model, the total loss function $\mathcal{L}_{total}$ is composed of the contrastive learning loss $\mathcal{L}_{ECL}$ from the contextual encoder, and the contrastive learning loss L_{LCL} based on the syntax label and the cross-entropy loss L_{CE} is shown in (16):

TABLE 1 Statistics of datasets.

Dataset	Positive		Neural		Negative	
	Train	Test	Train	Test	Train	Test
Twitter	1,561	173	3,127	346	1,560	173
Laptop 14	994	341	464	169	870	128
Restaurant 14	2,164	728	637	196	807	196
Restaurant 15	912	326	36	34	256	182
Restaurant 16	1,240	469	69	30	439	117

$$\mathcal{L}_{total} = \mathcal{L}_{CE} + \alpha\mathcal{L}_{ECL} + (1-\alpha)\mathcal{L}_{LCL}, \tag{16}$$

where α is a learnable coefficient to adjust the weights of contrastive learning losses in loss function.

4 Experiment

4.1 Experimental setup

The working performance of the DCL-GCN is evaluated on five benchmark datasets, which are Restaurant 14, Restaurant 15, Restaurant 16, and Laptop 14 from SemEval [28]; [29,30], and Twitter [31]. The sentiment of each aspect from the datasets is labeled as positive, neutral, or negative.

Following the idea of [15], the sentences labeled as conflicting sentiment or without explicit aspects from Restaurant 15 and Restaurant 16 are removed. Details of each dataset are given in Table 1.

In this experiment, the lexicon size of the BERT model is set to 30,522, the word embedding dimension is 768, and the layer number of the transformer is 12. The head number of the MHSA is 8, and the learning rate is 0.00001. The layer number of the weighted label-enhanced syntactic GCN is 2. Both τ and τ' in contrastive learning schemes are set to 0.02. The $\mathcal{L}_2$ regularization coefficient is 0.00001. An Adam optimizer is adopted during training with a data batch size of 32. All the hyperparameters used in the experiment are given in Table 2.

4.2 Baseline

In order to verify the effectiveness of the DCL-GCN in ABSC, five state-of-the-art methods are taken for comparison:

- **BERT** [32]: The basic BERT model is established based on the bidirectional transformer. With the concatenation of sentences and the corresponding aspect, BERT can be applied to ABSC.
- **BERT4GCN** [33]: The BERT model and GCN are integrated, which exploits sequential features and positional information to augment the model learning.
- **R-GAT + BERT** [25]: The pre-trained BERT is integrated with the R-GAT, where BERT is used for sentence encoding.

TABLE 2 Parameter settings.

Parameter	Value
BERT model lexicon size	30,522
Word embedding dimension	768
Transformer layers	12
Multi-head self-attention (MHSA) heads	8
Learning rate	0.00001
Weighted label-enhanced syntactic graph convolutional network (GCN) layers	2
T	0.02
τ'	0.02
$\mathcal{L}_2$ regularization coefficient	0.00001
Batch size	32
Optimizer	Adam

- **DGEDT + BERT** [34]: The pre-trained BERT is integrated with DGEDT, where BERT is used for sentence encoding.
- **TGCN + BERT** [35]: The dependency type is identified with type-aware graph convolutional networks, while the relation is distinguished with an attention mechanism. The pre-trained BERT is used for sentence encoding.

All results are expressed in percentage values. "-" denotes that the results are not reported in the published research article. The best performance achieved is marked in bold.

4.3 Result analysis

We take two metrics, i.e., accuracy and Macro-F1, to evaluate the working performance of the proposed model. Table 3 shows the results of six different methods on the task of ABSC. One can observe that our model achieves the best and most consistent result among all the evaluation settings. It is clear that the DCL-GCN result is more remarkable than a range of competitive baselines on all five benchmark datasets. In line with these results, the following observations are made.

First, our model achieves the best and most consistent result among all the evaluation settings. The minimum performance gaps between the DCL-GCN and the baselines are 1.33% (against the R-GAT) on Restaurant 14, 1.62% (against the T-GCN) on Restaurant 15, 1.33% (against the T-GCN) on Restaurant 16, 1.54% (against DGEDT) on Laptop 14, and 0.22 (against DGEDT) on Twitter. In addition, the F1 values on Restaurant 15 and Restaurant 16 are 3.64% (against the T-GCN) and 5.04% (against DGEDT), respectively, higher than the best-performing baseline method, which are significant.

Second, the syntax-dependent-method (BERT4GCN) performs worse than models integrated with both syntax dependency and syntactic relations (R-GCT and T-GCN). The main reason is that the deeper-level syntactic information can be neglected by solely exploiting the dependencies among words. By contrast, the syntactic relation encoded in our model benefits the sentiment comprehending to a large extent. The highest accuracy of our model reaches 93.65 on Restaurant 16, indicating the importance of syntax dependency and syntactic relations in ABSC.

Third, compared with other baselines, the basic BERT model has its own distinctiveness in tackling sentence semantic information. By incorporating BERT into state-of-the-art methods, the working performance is substantially improved, which is the outcome of our model. Notably, the proposed model significantly outperforms the baselines, demonstrating that the contextual semantics take full advantage in line with the BERT-based contrastive learning scheme.

It is worth noting that the DCL-GCN gives rise to the enhancement in both syntax and semantics learning. With the application of the dual contrastive learning scheme, it is reasonable to expect better working performance in ABSC, as it is the case.

4.4 Ablation study

The impact of different components in our model is investigated by conducting an ablation study (Table 4). *w/o* L_{ECL} specifies that the contrastive learning scheme of the contextual encoder is removed; *w/o* L_{LCL} specifies that the syntax label-based contrastive learning scheme is removed; and *w/o* WL-GCN indicates that the weighted label-enhanced syntactic GCN is ablated.

As presented in Table 4, the most significant module in our model is the weighted label-enhanced syntactic GCN. The exploiting of syntactic information shows its effectiveness in word sentiment learning. With the sole utilization of semantics, even with a contrastive learning strategy, the working performance is inferior to the syntactic-based methods in all evaluation settings. Clearly, the integration of semantics and syntax has superiority in ABSC tasks. Moreover, the removal of the contrastive learning scheme from the contextual encoder leads to a substantial decrease on all five datasets. The performance decreases of the accuracy and F1 score on Twitter are 2.76% and 2.25%, respectively. As a result, the contrastive learning scheme in the BERT encoder effectively promotes semantic information learning. By contrast, the syntax label-based contrastive learning scheme makes a relatively small contribution to the model. We can infer that the application of

TABLE 3 Experimental results on five public datasets.

Model	Twitter		Laptop 14		Restaurant 14		Restaurant 15		Restaurant 16	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
BERT [32]	75.00	72.53	78.68	74.64	84.55	77.34	83.40	65.28	89.54	70.47
BERT4GCN [33]	74.73	73.76	77.49	73.01	84.75	77.11	-	-	-	-
R-GAT + BERT [25]	76.15	74.88	78.21	74.07	86.60	81.35	-	-	-	-
DGEDT + BERT [34]	77.90	75.40	79.80	75.60	86.30	80.00	84.00	71.00	91.90	79.00
TGCN + BERT [35]	76.45	75.25	80.88	77.03	86.16	79.95	85.26	71.69	92.32	77.29
Our DCL-GCN + BERT	**78.12**	**76.37**	**82.42**	**79.20**	**87.93**	**82.53**	**86.88**	**75.35**	**93.65**	**84.04**

The bold values represent the best performance achieved among the different models or methods compared in the table. Specifically, the bold values indicate the highest accuracy and Macro-F1 scores obtained for each dataset (Twitter, Laptop14, Restaurant14, Restaurant15, Restaurant16) in the aspect-based sentiment classification (ABSC) task. These bold values highlight the superior results of the model we proposed compared to the baseline methods, showcasing its effectiveness in sentiment classification across different datasets.

TABLE 4 Results of the ablation study.

Model	Twitter		Laptop 14		Restaurant 14		Restaurant 15		Restaurant 16	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
w/o L_{ECL}	75.36	74.12	81.28	77.49	85.94	80.18	84.63	72.78	91.44	81.64
w/o L_{LCL}	76.12	74.89	81.13	77.30	86.12	80.67	85.24	73.68	92.17	82.26
w/o WL-GCN	75.13	73.85	80.62	76.83	85.20	79.47	84.27	72.14	91.23	81.08
Full model	**78.12**	**76.37**	**82.42**	**79.20**	**87.93**	**82.53**	**86.88**	**75.35**	**93.65**	**84.04**

The bold values represent the best performance achieved among the different models or variations compared in the table. Specifically, the bold values indicate the highest accuracy and Macro-F1 scores obtained for each dataset (Twitter, Laptop14, Restaurant14, Restaurant15, Restaurant16) in the aspect-based sentiment classification (ABSC) task when specific components or modules of the proposed model are included.

syntax labels also enhances the use of syntactic information and, thus, contributes to the sentiment classification.

4.5 Impact of hyperparameters

An experiment is carried out to analyze the effect of the self-attention head number on model working performance. The head number of the self-attention network is set to [1, 2, 3, . . ., 8]. The model accuracy with different head numbers is presented in Figure 6.

Apparently, the DCL-GCN achieves the highest accuracy with a head number of 5 on Laptopt 14 and Restaurant 15 and a head number of 6 on Twitter, Restaurant 14, and Restaurant 16. In line with the multi-head self-attention mechanism, the attention head stands for the vector representation in feature spaces via different mapping methods. When the number of attention heads is reduced, the self-attention mechanism operates within a smaller space with correspondingly fewer semantic features. Accordingly, the proposed model fails to capture sufficient semantic information. On the other hand, when the head number exceeds 6, the model parameter size significantly increases, resulting in overfitting issues during training. In this way, a test accuracy decrease is inevitable.

4.6 Case study

Two samples are selected to visualize the working performance, in order to further validate the distinctiveness of DCL-GCN. Specifically, the representations of the sentence and the words are maintained. We shall define a parameter φ as the contribution of each word for sentiment delivery in the sentence, which is defined in Eq. 17:

$$\varphi(X, w_i) = \frac{|\tilde{H} - \tilde{H}_{\frac{X}{w_i}}|}{\sum_{j=1}^{n} |\tilde{H} - \tilde{H}_{\frac{X}{w_j}}|}. \tag{17}$$

The sentiment contribution of each word is shown in Figure 7. For the sample given in Figure 7A, the contextual words "professional," "courteous," and "attentive" make the largest contribution toward the aspect "waiters." Our model is capable of extracting the most informative words for sentiment

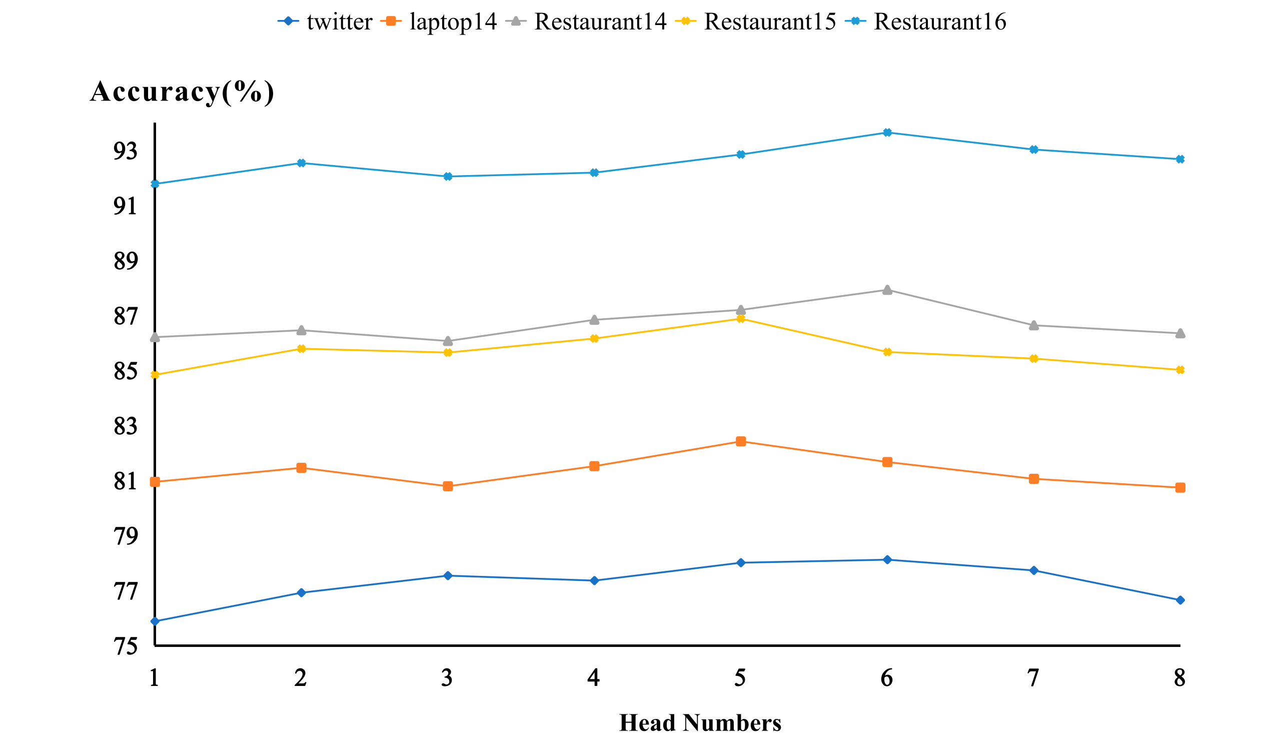
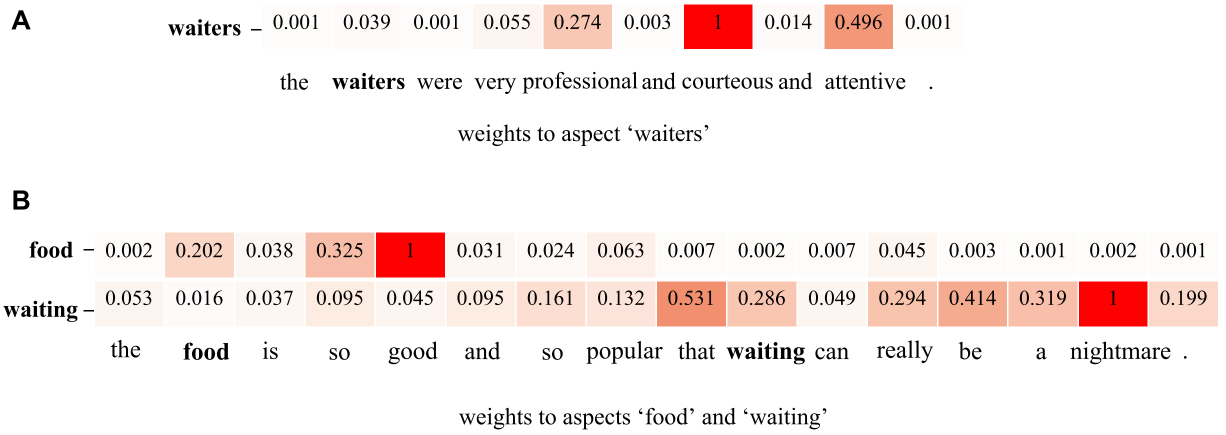


FIGURE 6
Accuracy of different head numbers.



waiters	0.001	0.039	0.001	0.055	0.274	0.003	1	0.014	0.496	0.001
	the	**waiters**	were	very	professional	and	courteous	and	attentive	.

weights to aspect ‘waiters’

food	0.002	0.202	0.038	0.325	1	0.031	0.024	0.063	0.007	0.002	0.007	0.045	0.003	0.001	0.002	0.001
waiting	0.053	0.016	0.037	0.095	0.045	0.095	0.161	0.132	0.531	0.286	0.049	0.294	0.414	0.319	1	0.199
	the	**food**	is	so	good	and	so	popular	that	**waiting**	can	really	be	a	nightmare	.

weights to aspects ‘food’ and ‘waiting’

FIGURE 7
Word sentiment contribution. **(A)** Weights to aspect “waiters” **(B)** and weights to aspects “food” and “waiting.”

expressing. The sentence in Figure 7B contains two aspects, i.e., “food” and “waiting.” For the aspect word “food,” the proposed model accurately identifies the top two highest sentiment contribution words as “good” and “so.” Regarding “waiting,” not only is the the sentiment word “nightmare” captured but also the syntactic relation words “so. . .that. . .” for resultative adverbial clause establishment. Both semantics and syntax are used for sentiment classification.

In our model, the use of contrastive learning enhances the learning of sentence semantics, and the build of the weighted label-enhanced syntactic GCN fully exploits the syntactic information. The integration of semantic information and syntactic information leads to a competitive manner in ABSC.

5 Conclusion

In this work, we propose a GCN based on dual contrastive learning and syntax label enhancement for ABSC tasks. To obtain sentiment information, a contrastive learning scheme is integrated to a BERT encoder to enhance the learning of semantic-related contextual information. Then, our model exploits both the syntax dependency and syntactic relation, based on which a weighted label-enhanced syntactic GCN is established. In addition, the learning of the syntax label is enhanced using contrastive learning. A syntactic triplet between words is mapped into a unified feature space for syntax and semantic integration. The rxperimental results reveal that the proposed model achieves state-of-the-art performance on five benchmark datasets. The ablation study, the hyperparameter analysis experiment, and the case study also obtain superior working performance.

Future work will focus on introducing more information for further improving the accuracy of ABSC and other sentiment analysis tasks, such as background knowledge and part-of-speech information. In addition, the integration of different categories of information into the model is also considered.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary Material; further inquiries can be directed to the corresponding author.

Author contributions

YH: conceptualization, methodology, and writing–original draft. AD: conceptualization, methodology, and writing–original draft. SC: formal analysis, methodology, and writing–original draft. QK: conceptualization, formal analysis, and writing–original draft. HZ: funding acquisition, supervision, and writing–review and editing. QC: supervision and writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. This work was supported in part by the Guangdong Basic and Applied Basic Research Foundation (no. 2023A1515011370), the National Natural Science Foundation of China (no. 32371114), and the Characteristic Innovation Projects of Guangdong Colleges and Universities (no. 2018KTSCX049).

Conflict of interest

Author YH was employed by Datastory, author AD was employed by China Merchants Bank, and author QK was employed by Guangzhou Qizhi Information Technology Co., Ltd.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors, and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Pang B, Lee L. Opinion mining and sentiment analysis. *Foundations Trends® in information retrieval.* Now Publishers Inc. (2008) 2 (1–2):1–135. doi:10.1561/1500000011

2. Bing L. *Sentiment analysis and opinion mining (synthesis lectures on human language technologies).* Chicago, IL, USA: University of Illinois (2012).

3. Zheng Y, Zhang R, Mensah S, Mao Y. Replicate, walk, and stop on syntax: an effective neural network model for aspect-level sentiment classification. *Proc AAAI Conf Artif intelligence* (2020) 34:9685–92. doi:10.1609/aaai.v34i05.6517

4. Tsytsarau M, Palpanas T. Survey on mining subjective data on the web. *Data Mining Knowledge Discov* (2012) 24:478–514. doi:10.1007/s10618-011-0238-6

5. Lin T, Joe I. An adaptive masked attention mechanism to act on the local text in a global context for aspect-based sentiment analysis. *IEEE Access* (2023) 11:43055–66. doi:10.1109/access.2023.3270927

6. Žunić A, Corcoran P, Spasić I. Aspect-based sentiment analysis with graph convolution over syntactic dependencies. *Artif Intelligence Med* (2021) 119:102138. doi:10.1016/j.artmed.2021.102138

7. Yusuf KK, Ogbuju E, Abiodun T, Oladipo F. A technical review of the state-of-the-art methods in aspect-based sentiment analysis. *J Comput Theories Appl* (2024) 2:67–78. doi:10.62411/jcta.9999

8. He Y, Huang X, Zou S, Zhang C. Psan: prompt semantic augmented network for aspect-based sentiment analysis. *Expert Syst Appl* (2024) 238:121632. doi:10.1016/j.eswa.2023.121632

9. Huang X, Li J, Wu J, Chang J, Liu D, Zhu K. Flexibly utilizing syntactic knowledge in aspect-based sentiment analysis. *Inf Process Manag* (2024) 61:103630. doi:10.1016/j.ipm.2023.103630

10. Wang P, Tao L, Tang M, Wang L, Xu Y, Zhao M. Incorporating syntax and semantics with dual graph neural networks for aspect-level sentiment analysis. *Eng Appl Artif Intelligence* (2024) 133:108101. doi:10.1016/j.engappai.2024.108101

11. Nazir A, Rao Y, Wu L, Sun L. Iaf-lg: an interactive attention fusion network with local and global perspective for aspect-based sentiment analysis. *IEEE Trans Affective Comput* (2022) 13:1730–42. doi:10.1109/taffc.2022.3208216

12. Gou J, Sun L, Yu B, Wan S, Ou W, Yi Z. Multilevel attention-based sample correlations for knowledge distillation. *IEEE Trans Ind Inform* (2022) 19:7099–109. doi:10.1109/tii.2022.3209672

13. Tang D, Qin B, Feng X, Liu T. Effective lstms for target-dependent sentiment classification (2015). Available at: https://arxiv.org/abs/1512.01100 (Accessed December 3, 2015).

14. Wang Y, Huang M, Zhu X, Zhao L. Attention-based lstm for aspect-level sentiment classification. In: Proceedings of the 2016 conference on empirical methods in natural language processing; November, 2016; Austin, Texas (2016). p. 606–15.

15. Chen P, Sun Z, Bing L, Yang W. Recurrent attention network on memory for aspect sentiment analysis. In: Proceedings of the 2017 conference on empirical methods in natural language processing; September, 2017; Copenhagen, Denmark (2017). p. 452–61.

16. Sun C, Huang L, Qiu X. Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence (2019). Available at: https://arxiv.org/abs/1903.09588 (Accessed March 22, 2019).

17. Gou J, Yuan X, Yu B, Yu J, Yi Z. Intra-and inter-class induced discriminative deep dictionary learning for visual recognition. *IEEE Trans Multimedia* (2023) 25:1575–83. doi:10.1109/tmm.2023.3258141

18. Gou J, Xie N, Liu J, Yu B, Ou W, Yi Z, et al. Hierarchical graph augmented stacked autoencoders for multi-view representation learning. *Inf Fusion* (2024) 102:102068. doi:10.1016/j.inffus.2023.102068

19. Suresh V, Ong DC. Not all negatives are equal: label-aware contrastive loss for fine-grained text classification (2021). Available at: https://arxiv.org/abs/2109.05427 (Accessed September 12, 2021).

20. Wang S-H, Govindaraj VV, Górriz JM, Zhang X, Zhang Y-D. Covid-19 classification by fgcnet with deep feature fusion from graph convolutional network and convolutional neural network. *Inf Fusion* (2021) 67:208–29. doi:10.1016/j.inffus.2020.10.004

21. Zhang Y-D, Satapathy SC, Guttery DS, Górriz JM, Wang S-H. Improved breast cancer classification through combining graph convolutional network and convolutional neural network. *Inf Process Manag* (2021) 58:102439. doi:10.1016/j.ipm.2020.102439

22. Sun K, Zhang R, Mensah S, Mao Y, Liu X. Aspect-level sentiment analysis via convolution over dependency tree. In: Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP); November, 2019; Hong Kong, China (2019). p. 5679–88.

23. Zhang C, Li Q, Song D. Aspect-based sentiment classification with aspect-specific graph convolutional networks (2019). Available at: https://arxiv.org/abs/1909.03477 (Accessed September 8, 2019).

24. Liang B, Yin R, Gui L, Du J, Xu R. Jointly learning aspect-focused and inter-aspect relations with graph convolutional networks for aspect sentiment analysis. In: Proceedings of the 28th international conference on computational linguistics; December, 2020; Barcelona, Spain (Online) (2020). p. 150–61.

25. Wang K, Shen W, Yang Y, Quan X, Wang R. Relational graph attention network for aspect-based sentiment analysis (2020). Available at: https://arxiv.org/abs/2004.12362 (Accessed April 26, 2020).

26. Chen T, Kornblith S, Norouzi M, Hinton G. A simple framework for contrastive learning of visual representations. In: International conference on machine learning (PMLR); July, 2020; Virtual (2020). p. 1597–607.

27. Khosla P, Teterwak P, Wang C, Sarna A, Tian Y, Isola P, et al. Supervised contrastive learning. *Adv Neural Inf Process Syst* (2020) 33:18661–73. doi:10.48550/arXiv.2004.11362

28. Kirange D, Deshmukh RR, Kirange M. Aspect based sentiment analysis semeval- 2014 task 4. *Asian J Comp Sci Inf Tech (Ajcsit)* (2014) 4. doi:10.15520/ajcsit.v4i8.9

29. Pontiki M, Galanis D, Papageorgiou H, Manandhar S, Androutsopoulos I. Semeval-2015 task 12: aspect based sentiment analysis. In: Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015); June, 2015; Denver, Colorado (2015). p. 486–95.

30. Pontiki M, Galanis D, Papageorgiou H, Androutsopoulos I, Manandhar S, Al-Smadi M, et al. Semeval-2016 task 5: aspect based sentiment analysis. In: ProWorkshop on Semantic Evaluation (SemEval-2016) (Association for Computational Linguistics); June, 2016; San Diego, California, USA (2016). p. 19–30.

31. Dong L, Wei F, Tan C, Tang D, Zhou M, Xu K. Adaptive recursive neural network for target-dependent twitter sentiment classification. In: Proceedings of the 52nd annual meeting of the association for computational linguistics; June, 2014; Baltimore, Maryland (2014). p. 49–54.

32. Devlin J, Chang M-W, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding (2018). Available at: https://arxiv.org/abs/1810.04805 (Accessed October 11, 2018).

33. Xiao Z, Wu J, Chen Q, Deng C. Bert4gcn: using bert intermediate layers to augment gcn for aspect-based sentiment classification (2021). Available at: https://arxiv.org/abs/2110.00171 (Accessed October 1, 2021).

34. Tang H, Ji D, Li C, Zhou Q. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In: Proceedings of the 58th annual meeting of the association for computational linguistics; July, 2020; Online (2020). p. 6578–88.

35. Tian Y, Chen G, Song Y. Aspect-based sentiment analysis with type-aware graph convolutional networks and layer ensemble. In: Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies; June, 2021 (2021). p. 2910–22.

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 18 April 2024
DOI 10.3389/fphy.2024.1387338

Check for updates

OPEN ACCESS

EDITED BY
Yuanyuan Huang,
Chengdu University of Information Technology,
China

REVIEWED BY
Li Chaorong,
Yibin University, China
Yufei Wang,
Jilin Engineering Normal University, China
Fei Wang,
Nanjing University of Aeronautics and
Astronautics, China

*CORRESPONDENCE
Quan Wen,
✉ wenquan@jlu.edu.cn

RECEIVED 17 February 2024
ACCEPTED 02 April 2024
PUBLISHED 18 April 2024

CITATION
Luan X, Wen Q and Hang B (2024), Driver emotion recognition based on attentional convolutional network.
Front. Phys. 12:1387338.
doi: 10.3389/fphy.2024.1387338

Driver emotion recognition based on attentional convolutional network

Xing Luan[1], Quan Wen[1]* and Bo Hang[2]

[1]College of Communication Engineering, Jilin University, Changchun, China, [2]Hubei University of Arts and Science, Xiangyang, China

Unstable emotions, particularly anger, have been identified as significant contributors to traffic accidents. To address this issue, driver emotion recognition emerges as a promising solution within the realm of cyber-physical-social systems (CPSS). In this paper, we introduce SVGG, an emotion recognition model that leverages the attention mechanism. We validate our approach through comprehensive experiments on two distinct datasets, assessing the model's performance using a range of evaluation metrics. The results suggest that the proposed model exhibits improved performance across both datasets.

KEYWORDS

road rage detection, driver emotion recognition, facial expression recognition, attention mechanism, deep learning

1 Introduction

The driver emotion recognition has garnered substantial scholarly attention as a consequential application within cyber-physical-social systems. A driver emotion recognition system is structured into three pivotal layers: perception, cognition and decision, and interaction [1]. Within this framework, perception involves the deployment of sensors in the cockpit to meticulously acquire data pertaining to the driver's emotional state. Cognition and decision denote the integration of emotion recognition models with real-time data to analyze the driver's emotions. The interaction layer includes a vehicle warning system to detect and alert on the driver's emotional instability or fatigue. This paper endeavors to explore an approach to driver emotion recognition with a specific focus on the cognitive and decision layers. The emphasis lies in the sophisticated integration of emotion recognition models with dynamic data streams, facilitating a nuanced real-time analysis of the driver's emotional states.

A number of methods for driver emotion recognition by facial expression have emerged due to the low price of vision sensors and their easy installation and realization in the driving environment [2]. After acquiring the data, it is also critical to extract the emotional characteristics from the data. The attention mechanism [3] in deep learning is a way to mimic human vision, allowing neural networks to focus more on top of important information and improve the effectiveness of their models. Jiyong Xue proposed a deep convolutional model based on multi-head self-attention that fuses utterance-level acoustic features and frame-level acoustic features [4]. Wei Tao proposes an attention-based convolutional recurrent neural network (ACRNN) which extracted more discriminative features in EEG (electroencephalogram) signals through the attention mechanism [5]. These methods have yielded excellent results.

In the field of emotion recognition, researchers often choose the emotion model (sad, happy, fear, disgust, surprise, and angry) proposed by Ekman [6] as the starting point of research. From the perspective of preventing traffic accidents, some emotions are somewhat redundant. Including an excessive variety of emotions as the subject of study amplifies model redundancy and diminishes the recognition rate. Therefore, it is necessary to consider several emotions that are most relevant to drivers for the study. Research with drivers of varying ages has found that anger is associated with speeding, fear with stronger braking, lower speeds, and poorer lateral vehicle control [7]. Additionally, anger and happiness were found to be associated with more driving errors than fear or a neutral emotional state [8]. Henceforth, within this dissertation, we elect to investigate the emotional domains of happiness, anger, fear, and sadness as our primary research subjects. Employing the attention convolutional network, our objective is to discern and analyze drivers' facial expressions with precision and relevance.

2 Related work

2.1 Contact method and contactless method

According to the different information obtained, the types of driver emotion recognition can be divided into two types: contact method and contactless method [9]. The contact method uses special equipment to measure the drivers' physiological signals [10, 11], such as body temperature [12], electrocardiographic signals [13], skin electrical signal [14] et al. While the contactless method analyzes the drivers facial [15, 16] or voice information [17] through cameras or microphones. Using the contact method for emotion recognition has high accuracy and high real-time performance. However, the effect in actual use is often not satisfactory. This is not only because the drivers' physiological signals are inconvenient to obtain and the identification device is difficult to wear, but also because the device will cause psychological stress to the driver, which makes it impossible for drivers to drive vehicles in a relaxed environment [18].

Within the realm of contactless methods investigations, facial expressions predominantly serve as indicators of the driver's emotional state [19]. Through the scrutiny of the driver's facial image information, it becomes feasible to intuitively ascertain the driver's ongoing emotional state. Employing this methodology not only avoids causing any disturbance to the driver but also enables the continuous monitoring of the driver.

2.2 Face emotion recognition

Miyajia [20] uses Kohonen neural network as a classification algorithm to recognize the drivers' facial emotion and proposes an early warning method of the driver's angry state. However, the KNN solely concentrates on the emotion of anger and overlooks other emotions in drivers that might possibly lead to a collision. Alessandro [21] used the VGG (Visual Geometry Group) model to detect the drivers anger and recognized the continuous image by sliding the window. Both frontal and non-frontal facial expressions have been explored in literature, however, their precision falls short when compared to contemporary methods. Geesung [22] used a variety of CNN (Convolutional Neural Networks) models to test the driver's facial expression and acquired the driver's skin electric signal to judge the driver's emotion synthetically. The DRER model proposed in the literature has an accuracy of 88.6%, but it only identifies emotional states for a short period. However, emotions are continuous, and the model requires improvement to identify emotional states in real-time. H. Varun Chand [16] presented a multi-layer drowsiness detection system based on CNN and emotion analysis, which achieved an accuracy rate of 93%. The spatial transformer network [23] adjusts the image by learning spatial transformations and applying them to the original image. This approach significantly reduces the interference of environmental factors in the extraction of emotional features. However, real-time monitoring of the driver's mood has not yet been attained. In this work, benefiting from attention convolutional neural networks, we obtain higher recognition accuracy while maintaining recognition speed.

3 Methods

3.1 Preprocessing

The preprocessing stage mainly includes face detection and segmentation. When the camera captures an image of the driver, the first step is to extract the facial information in the image. In this paper, we use OpenCV (Open-Source Computer Vision Library) to detect face Haar features [24] by loading the pre-trained classifiers. The advantage of this method is that the recognition speed is fast, it can be used for real-time detection, and it has a high recognition rate. At the same time, the model is small and can be run on an embedded platform, which is more suitable for the scene of driver face detection.

3.2 The proposed framework

Figure 1 illustrates the structural framework of SVGG, a driver emotion recognition model based on attention mechanisms. When the driver's face image information is acquired, the face image is first input into the STN model for face alignment to reduce the interference of environmental factors. Then the processed image is inputted into the improved VGG network model for feature extraction of the image in depth, in which the convolution adopts the Ghost Module to reduce the parameters of the model and speed up the inference speed of the network, and the activation function adopts the Mish function to speed up the convergence speed of the model. After obtaining the feature maps with complete feature extraction, the channel attention model ECA-Net will redistribute the weights of the feature maps in the channel, and finally the feature maps of multiple channels will be passed through a fully connected layer for emotion classification and recognition.

3.2.1 Spatial Transformer Networks

Spatial Transformer Networks (STNs) [23] is a neural network model as well as a spatial attention mechanism. Among the ways of spatial transformations are translation, rotation or scaling. The

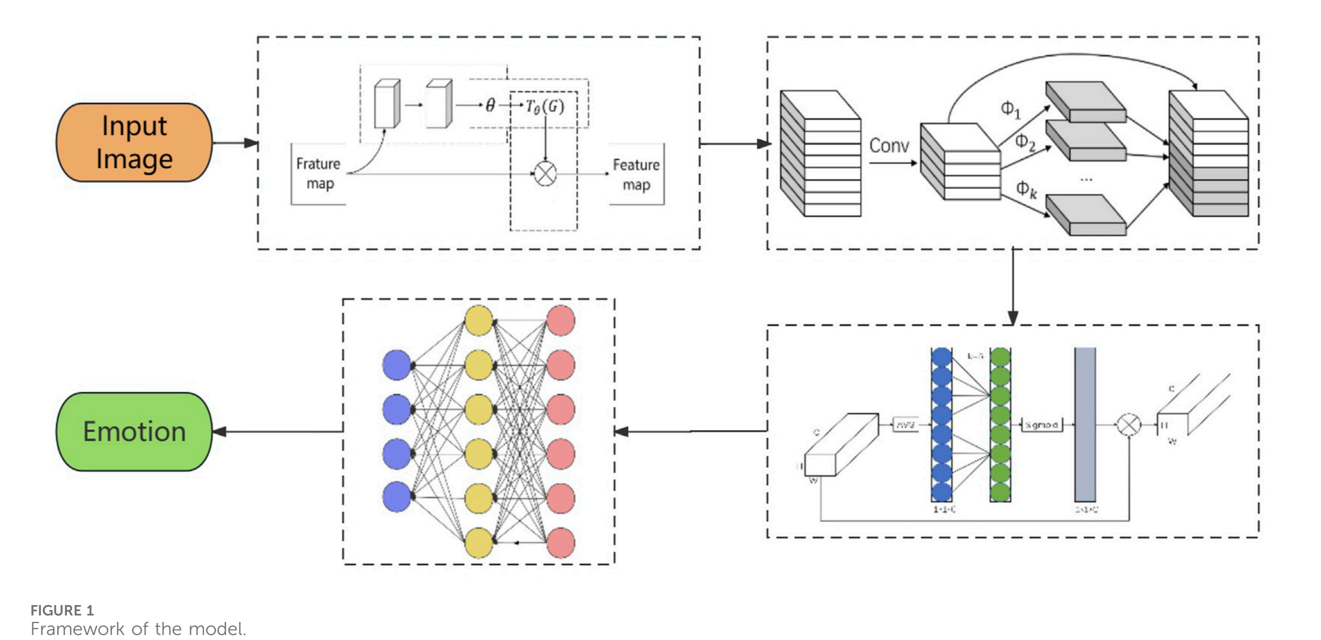


FIGURE 1
Framework of the model.

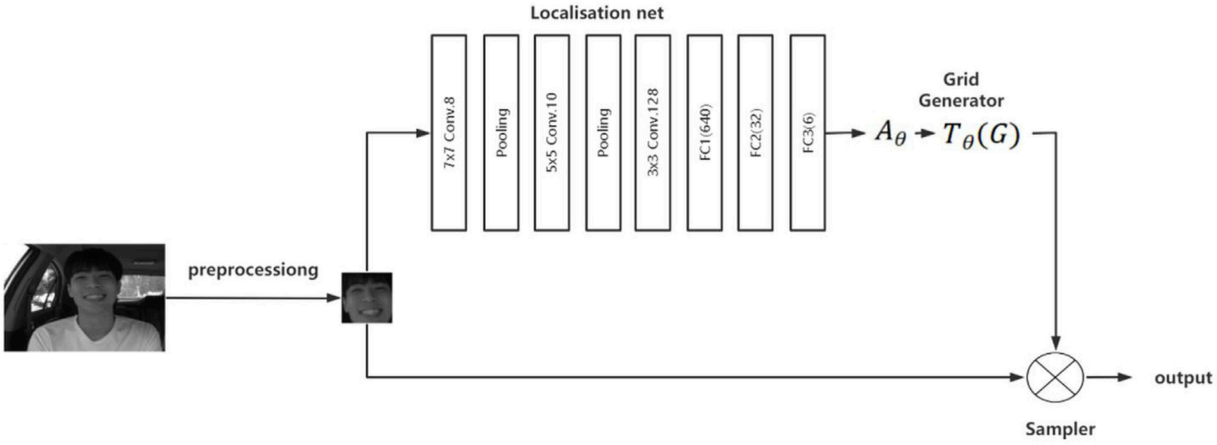


FIGURE 2
Spatial transform network. Reprinted with permission from "Driver's Facial Expression Recognition in Real-Time for Safe Driving" by Mira Jeong, ByoungChul Ko. The license for this content can be found at https://cvpr.kmu.ac.kr/KMU-FED.htm.

TABLE 1 Comparison of model parameter sizes.

Model	No. of parameters (in million)
VGG19	20.1
MobileNet	3.2
LiveEmoNet [30]	1.3
CNN [31]	1.3
SVGG	10

advantage of the spatial transformation network is that the network can autonomously learn certain key changes in the natural image without human labeling, and thus adjust the image so that the network focuses on these changes. Theoretically, the spatial transform network can be added to any layer of the convolutional neural network, but in practice, most researchers add the network to the convolutional neural network before preprocessing the original image, and then input the image to the convolutional layer for feature extraction, so as to ensure the integrity of the original input data. The spatial transform network only adjusts the pixel positions of the original image, and does not adjust the size of the original image, i.e., the input and output images of the spatial transform network have the same size.

The spatial transformation network model was introduced in 2015, which contains three models: Localization net, Grid generator, and Sampler. The model structure diagram is shown in Figure 2. When we get the driver's face image after preprocessing, the image

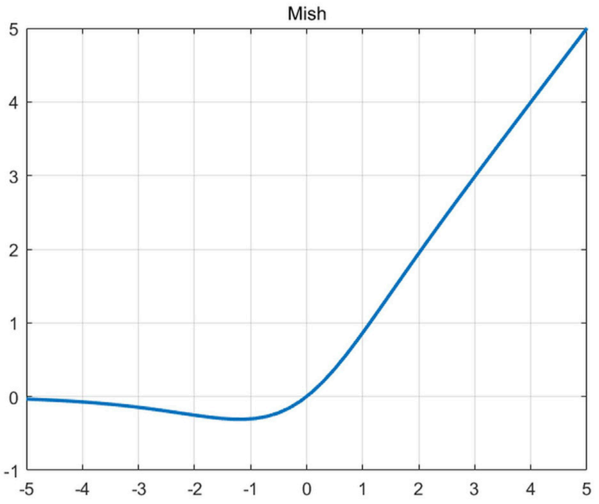


FIGURE 3
Mish activation function.

TABLE 2 Comparison of experimental results on the KMU-FED dataset.

Model	Accuracy (%)	Recognition speed (ms/frame)
VGG19	94.3	140
SVGG	96.6	80

will go through the spatial transformation network to extract the key regions of the face. First through the Localization net to generate affine transformation A_θ. The Grid generator used affine transformation A_θ to create a sampling grid. At last, the drivers face image and the sampling grid are taken as inputs to the Sampler. And we can finally get the most relevant parts of a face image from Sampler.

Localization net is a small convolutional neural network used to generate affine transformation parameters. The input image shape is 48*48*1 and the output is A_θ, the affine transformation A_θ has 6 parameters, as Eq. 1. These parameters are constantly optimized during the training process to identify the most relevant face regions in an image.

$$\boldsymbol{A_\theta} = \begin{bmatrix} \boldsymbol{\theta_{11},\theta_{12},\theta_{13}} \\ \boldsymbol{\theta_{21},\theta_{22},\theta_{23}} \end{bmatrix} \tag{1}$$

Grid generator uses the affine transformation matrix A_θ to create. Assume that pixel in the input image is (x_i^s, y_i^s), the pixel in the output image is (x_t^i, y_t^i), and the corresponding relationship between an input image pixel and output image pixel is as shown in Eq. 2. It can obtain the values of the output image pixel values by taking the inverse.

$$\begin{pmatrix} \boldsymbol{x_i^s} \\ \boldsymbol{y_i^s} \end{pmatrix} = \boldsymbol{T_\theta}(\boldsymbol{G_i}) = \boldsymbol{A_\theta} \begin{pmatrix} \boldsymbol{x_i^t} \\ \boldsymbol{y_i^t} \\ \boldsymbol{1} \end{pmatrix} = \begin{bmatrix} \boldsymbol{\theta_{11}} & \boldsymbol{\theta_{12}} & \boldsymbol{\theta_{13}} \\ \boldsymbol{\theta_{21}} & \boldsymbol{\theta_{22}} & \boldsymbol{\theta_{23}} \end{bmatrix} \begin{pmatrix} \boldsymbol{x_i^t} \\ \boldsymbol{y_i^t} \\ \boldsymbol{1} \end{pmatrix} \tag{2}$$

The sampler module is employed for the execution of spatial transformations. It applies the previous sampling grid to the input image, to produce the final image which is the most relevant parts of a face image.

TABLE 3 Performance in KMU-FED by 5-fold cross-validation.

Model	Accuracy (%)
SqueezeNet	89.7
Modified SqueezeNet	95.8
MobileNetV2	93.8
MobileNetV3	94.9
LMRF	95.1
SVGG	96.6

3.2.2 VGG network

The VGG network [23] is employed for recognizing emotions in driver face images that are processed by the Spatial Transform Network. The VGG19 [23] network contains five convolutional groups and 16 convolutional layers, each of which contains a large number of redundant computations. For the special environment of driving, in order to make the VGG network have a more efficient recognition effect, this paper is influenced by the idea of “Cheap Operations” in Ghost Net, and adopts the convolutional method of Ghost Module to improve the convolutional layers in the VGG network. In an ordinary convolution operation, assuming the input feature map

$$\boldsymbol{X} \in \boldsymbol{R}^{\boldsymbol{h^*w^*c}} \tag{3}$$

where h is the height of the input image, w is the width of the feature map, and c is the number of channels of the feature map, and the output feature map

$$\boldsymbol{Y} \in \boldsymbol{R}^{\boldsymbol{h'^*w'^*n}} \tag{4}$$

where, h' and w' are the height and width of the feature map, respectively, and n is the number of channels of the feature map, and the convolution kernel is, and k is the size of the kernel, and n is the number of the kernels of the convolution, the computation of this convolution operation can be expressed in Eq. 5:

$$\boldsymbol{F = h'^*w'^*n^*c^*k^*k} \tag{5}$$

In convolutional operations, the number of n and c is high, so the feature maps generated by ordinary convolution consume a high amount of computation and have a large redundancy. Ghost Module, on the other hand, utilizes the redundancy of convolutional operations and uses simple convolutional and linear operations to obtain the same feature maps as normal convolution, which is called “Cheap Operations”. Specifically, Ghost Module first uses the regular convolution to generate the eigenfeature map Y', which contains m feature maps, the number of m is less than n. The amount of computation F_1 needed after omitting the bias term in the convolution operation can be expressed as Eq. 6:

$$\boldsymbol{F_1 h'^*w'^*m^*c^*k^*k} \tag{6}$$

After that, the Ghost Module performs linear operations on the obtained eigenfeature maps Y' to generate phantom feature maps, and the process can be expressed as Eq. 7:

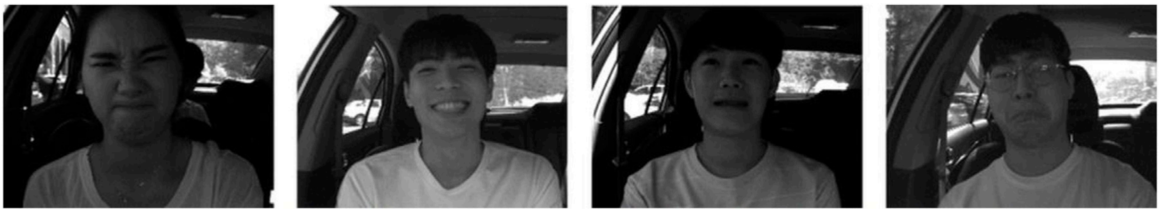


FIGURE 4
The four expression images contained in the KMU-FED dataset. Reprinted with permission from "Driver's Facial Expression Recognition in Real-Time for Safe Driving" by Mira Jeong, ByoungChul Ko. The license for this content can be found at https://cvpr.kmu.ac.kr/KMU-FED.htm.

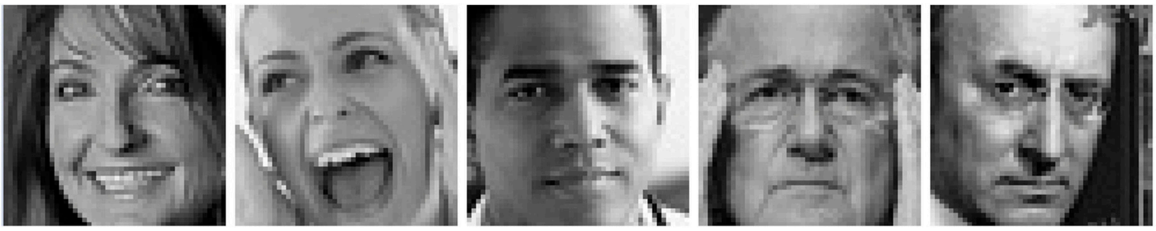


FIGURE 5
The five expression images contained in the FER-2013 dataset. Reprinted with permission from "Challenges in Representation Learning: Facial Expression Recognition Challenge" by Dumitru, Ian Goodfellow, Will Cukierski, Yoshua Bengio. https://kaggle.com/competitions/challenges-in-representation-learning-facial-expression-recognition-challenge.

$$y_{ij} = \varnothing_{i,j}\left(y_i'\right), \forall i = 1, 2, ..., m, j = 1, 2, ..., s \tag{7}$$

Comparing the convolution formed by Ghost Module with ordinary convolution, it can be seen that the computation of Ghost Module is only 1/s of ordinary convolution, as shown in Eq. 8. In this paper, s is fixed to 2, i.e., compared with the ordinary convolution in the VGG network, the computation of the convolution layer in the improved VGG will be reduced to 1/2. The total number of parameters for SVGG is 10,067,914 compared to the literature as shown in Table 1.

$$r_s = \frac{n * h' * w' * c * k * k}{\frac{n}{s} * h' * w' * c * k * k + (s-1)\frac{n}{s} * h' * w' * d * d} \approx s \tag{8}$$

The activation function in VGG networks is the ReLU function, which is a simple and effective nonlinear activation function and has been widely used in many neural network models. However, it has an obvious drawback: the "dead neuron" problem. In the ReLU function, when the value of the input is less than 0, the output value will always be 0, which means the neuron is "dead". As the number of neurons increases, the number of dead neurons will also increase, resulting in some neurons cannot be effectively used in the backpropagation. Also, this function does not solve the problem of vanishing gradients. This problem can be solved by using the Mish activation function, whose functional formula is shown in Eq. 9 and the graph is shown in Figure 3.

$$f(x) = x^* \tanh(\ln(1 + e^x)) \tag{9}$$

3.2.3 ECA-Net

ECA-Net [25] represents a lightweight and efficient channel attention model proficient in capturing inter-channel feature map information with the introduction of a minimal number of parameters. Unlike the SENet (Squeeze-and-Excitation Networks) structure, ECA-Net does not use the fully connected layer in SENet, but chooses to use one-dimensional convolution to dynamically adjust the size of the kernel, and establishes the relationship between the feature channels and the size of the convolution kernel using an approximate linear mapping to achieve the ability of focusing on channel convolution information exchange while avoiding dimensionality degradation.

4 Experiment

4.1 Datasets

The KMU-FED dataset [26] was selected as the primary dataset to assess the proposed approach. In order to evaluate the performance of the model more thoroughly, the FER-2013 dataset was also selected for testing (Figure 4).

KMU-FED dataset is a real-world driver's facial image dataset collected by the CVPR laboratory of Keimyung University, contains 1,106 emotional images about drivers, the picture pixel is 1,600*1,200. It is a dataset of driver emotions in real environments captured by infrared cameras placed on the steering wheel or dashboard and contains emotional pictures of multiple drivers under different lighting conditions. It contains six emotions: anger, surprise, happiness, fear, disgust, and sad.

The FER-2013 dataset [32], compiled by Google in 2013, comprises 35,887 grayscale images stored in a CSV file with dimensions of 48 × 48 pixels. These images are categorized into seven emotions: anger, disgust, fear, happiness, neutral, sadness, and surprise. Despite being

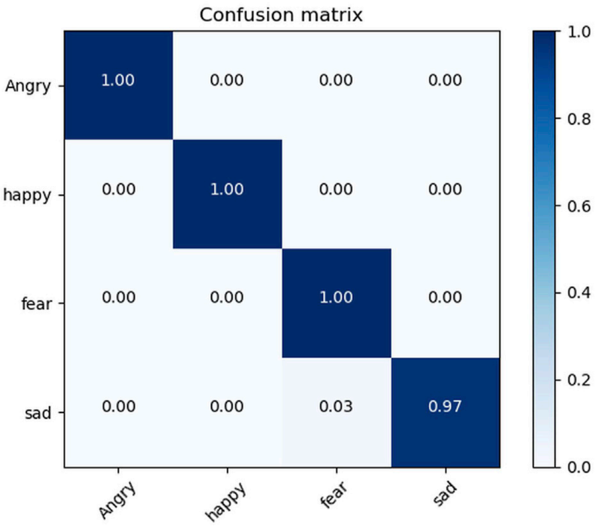


FIGURE 6
Confusion matrix in KMU-FED.

TABLE 4 Effect of different modules in the model on experimental results.

STN	Convolution	Activation	ECA-net	Accuracy (%)
-	-	-	-	70.4
√	-	-	-	72.0
√	√	-	-	71.5
√	√	√	-	71.8
√	√	√	√	72.4

TABLE 5 Comparison of results of different models.

Model	Accuracy (%)
MobileNetV2	68.3
SqueezeNet	64.5
LiveEmoNet [30]	69.0
CNN(31)	65.0
SVGG	72.4

originally designed for general in-the-wild conditions and including animated characters displaying diverse emotions, this dataset is not explicitly tailored for driver-centric scenarios (Figure 5).

For our analysis, we specifically focus on four emotions, resulting in a subset of 26,217 images. This subset includes 4,953 images depicting anger, 8,989 images displaying happiness, 5,121 images conveying fear, and 6,077 images depicting sadness. Notably, since all images in this dataset represent facial expressions, no additional image preprocessing steps were applied.

4.2 Training procedures and evaluation criteria

Prior to discussing model performance, we will provide a brief overview of the training procedures and evaluation criteria used in this study. Training epochs of the model is set to 300, and an early stopping strategy is used to avoid overfitting. When the validation accuracy did not improve in 30 iterations, the training would stop. The batch size is set to 128, and an Adam algorithm [27], a useful optimizer, is set to optimize the model parameters. Adam optimizers learning rate is set to 0.001.

In order to comprehensively validate the effectiveness of the Ghost Net-SSD algorithm, it is necessary to comprehensively evaluate both the average precision and the recognition speed of the algorithm recognition. Among them, the average precision (AP) is calculated by combining the recognition accuracy (Precision) and the recall rate (Recall).

Recognition Precision is the probability that a face is correctly detected among all detected samples, assuming that the number of samples in which a face is detected is TP and the number of samples in which a non-face is detected is FP, then the formula for the Recognition Precision (Precision) can be expressed as Eq. 10.

Recall is the probability of correctly detected faces among all faces that should be detected. Assuming that the number of incorrectly detected non-face samples is FN, the formula for recall (Recall) can be expressed as Eq. 11.

The P-R curve can be established with the recognition precision rate as the vertical coordinate and the recall rate as the horizontal coordinate, then the area of the curve combined with the axes is the Average Precision (AP, Average Precision), which is calculated as shown in Eq. 12.

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FP}} \tag{10}$$

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FN}} \tag{11}$$

$$\boldsymbol{AP} = \int_{0}^{1} \boldsymbol{P}(\boldsymbol{R})\boldsymbol{dR} \tag{12}$$

The recognition speed is calculated by synthesizing the recognition speed of multiple images by the model on the experimental platform to determine whether the algorithm meets the real-time requirements.

4.3 Results and analysis

4.3.1 KMU-FED dataset

In order to verify the effectiveness of emotion recognition in real driving environments, the SVGG model and the VGG19 model are compared and experimented on the KMU-FED dataset, and the experimental results are shown in Table 2. As evidenced by the data presented in Table 2, the SVGG model achieves a recognition accuracy of 96.6% on the KMU-FED dataset, surpassing the accuracy of the VGG19 model at 94.3%. This signifies an improvement of 2.3% relative to the VGG19 model. Regarding recognition speed, the SVGG model operates at 80 m/frame, while the VGG19 model exhibits a recognition speed of 140 m/frame. This

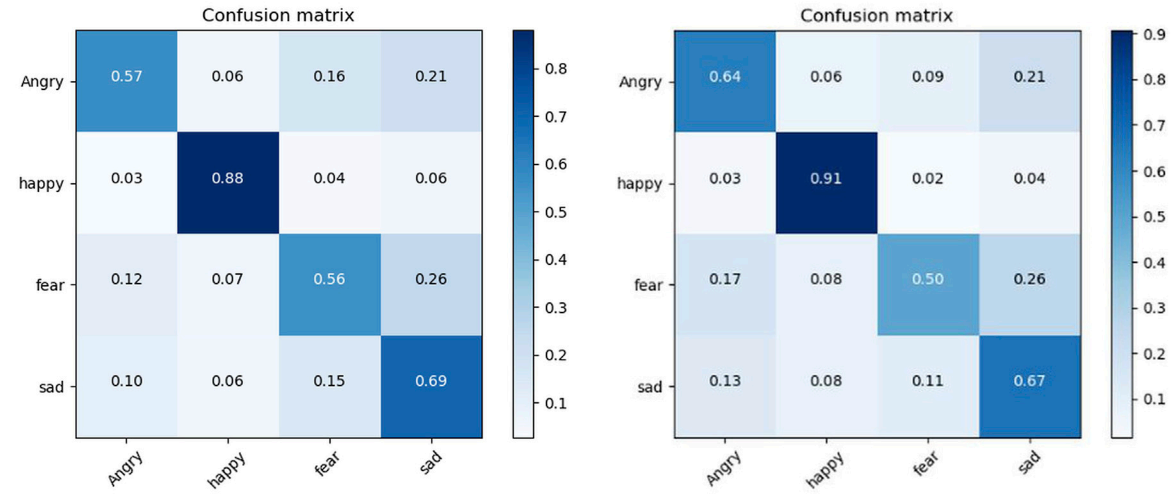


FIGURE 7
Confusion matrix in FER-2013 used VGG left and our method right.

represents a significant improvement of 43% in comparison to the VGG19 network model.

Table 3 demonstrates the results of comparing the recognition accuracy of different methods on the KMU-FED dataset, from the data in the table, it can be seen that the SVGG model is higher than the classical network model in terms of recognition accuracy. Specifically, compared to the Squeeze Net and MobileNetV3 models, the recognition accuracy of the SVGG model has improved by 6.9% and 1.7%. Meanwhile, comparing the new network models proposed in recent years, such as LMRF [28], Modified Squeeze Net [29], *etc.*, the SVGG model also has a more obvious advantage in terms of recognition accuracy, in which the recognition accuracy of the SVGG model is improved by 1.5% compared with the LMRF model, and compared with Modified Squeeze Net. SVGG model has improved the recognition accuracy by 0.8%.

Figure 6 shows the confusion matrix of the model in the case of using the KMU-FED dataset. The horizontal axis of the confusion matrix represents the predicted emotion categorization, the vertical axis represents the true emotion categorization, and the diagonal of the matrix represents the correct recognition rate of emotions. In the figure, the model demonstrates precise recognition of anger, happiness, and fear. Notably, only a minimal number of instances portraying sad expressions are misclassified as fear.

4.3.2 FER-2013 dataset

The experiments conducted in the preceding subsection confirmed the effectiveness of the SVGG model in real-world driving scenarios. In this subsection, additional experiments will be carried out on the FER-2013 dataset to assess the model's generalization capabilities First of all, this paper does is experiments on the improvement effect of different modules on the overall model. The experimental results are shown in Table 4. Where, the mark "√" indicates that the module is used in the model, and the mark "-" indicates that the module is not used in the model.

From the results of the experiment in Table 4, it can be seen that the recognition accuracy of the original model is 70.4%, and the face alignment implemented by the STN module has the most obvious effect on the model recognition accuracy improvement, which can improve the recognition accuracy by 1.6%. The improvement to the convolution module will make the number of parameters of the model decrease significantly and improve the recognition speed of the model, but from the results of this experiment, the improvement to the convolution module will make the model recognition accuracy decrease slightly. The activation function module and the ECA-Net module both have positive enhancement effects for the model. The recognition accuracy of the SVGG model on the FER-2013 dataset is 72.4%, which is an improvement of 2.0% compared to the original model, which verifies the validity of the model proposed in this paper.

To more thoroughly evaluate the efficacy of the proposed SVGG model, it is essential to conduct comparative experiments with other emotion recognition models. In this paper, experimental comparisons are conducted on the FER-2013 dataset. The experimental results are shown in Table 5.

From the results of the comparison experiments in Table 5, it can be seen that in terms of recognition accuracy, the SVGG model has a significantly higher recognition accuracy than some lightweight network models. For example, compared with MobileNetV2, the average accuracy of SVGG model is improved by 4.1%, and compared with Squeeze Net, the average accuracy of SVGG model is improved by 7.9%. Also, the recognition accuracy of SVGG model is better than network models optimized for emotion recognition in recent years, e.g., compared with LiveEmoNet, the recognition accuracy of SVGG model is improved by 3.4%. Compared with CNN, the recognition accuracy of SVGG model is improved by 7.4%.

Figure 7 shows the accuracy of the two models under the confusion matrix model. Contrasting the anger that causes road rage, it can be seen from the figure that our model's recognition rate of anger is 7% higher than that of the VGG model. At the same time, it can be seen that the recognition rates of fear and sadness are not high for both two models, and it is easy to misjudge these two emotions. This is due to the relatively small amount of data for these two emotions compared to the other emotions in the dataset.

5 Conclusion

In this paper, an emotion recognition model SVGG based on the attention mechanism is proposed. The SVGG model addresses picture jitter in driving environments through a spatial transformation network for facial alignment. To enhance recognition speed, it employs the Ghost Module's convolution method and the Mish activation function for accelerated convergence. Addressing low accuracy, the SVGG model utilizes ECA-Net to redistribute output channel weights. The experimental results demonstrate that our model achieves a 43% improvement in processing speed, with a rate of 80 milliseconds per frame, compared to the VGG model. Furthermore, it attains an accuracy of 96.6% on the KMU-FED dataset and 72.4% on the FER-2013 dataset, suggesting its high potential for practical applications.

This paper focuses on analyzing frontal or partially frontal face images of drivers, emphasizing specific camera placement requirements within the driving environment. Future research aims to explore emotion recognition from the driver's side face, potentially overcoming camera placement limitations. Additionally, we consider integrating the perception and interaction layers to form a more holistic Cyber-Physical Social System, enhancing the system's overall functionality in driver-assistance technologies.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found here: https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data https://cvpr.kmu.ac.kr/KMU-FED.htm.

Author contributions

XL: Data curation, Methodology, Software, Writing–original draft. QW: Conceptualization, Funding acquisition, Methodology, Resources, Writing–review and editing. BH: Formal Analysis, Investigation, Supervision, Writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. Jilin Science and Technology Development Program Project: Research on Intelligent Recognition and Prevention of Driver's Emotions under Multimodality, Project No. 20230203032SF.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Li W, Wu L, Wang C, Xue J, Hu W, Li S, et al. Intelligent cockpit for intelligent vehicle in metaverse: a case study of empathetic auditory regulation of human emotion. *IEEE Trans Syst Man, Cybernetics: Syst* (2023) 53(4):2173–87. doi:10.1109/tsmc.2022.3229021

2. Yang L, Yang H, Hu BB, Wang Y, Lv C. A Robust driver emotion recognition method based on high-purity feature separation. *IEEE Trans Intell Transportation Syst* (2023) 24(12):15092–104. doi:10.1109/tits.2023.3304128

3. Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. *arXiv* (2016). Available from: http://arxiv.org/abs/1409.0473. doi:10.48550/arXiv.1409.0473

4. Xue J, Li W, Zhang Y, Xiao H, Tan R, Xing Y, et al. Driver's speech emotion recognition for smart cockpit based on a self-attention deep learning framework. In: *2021 5th CAA international conference on vehicular control and intelligence (CVCI)* (2021). p. 1–5. Available from: https://ieeexplore.ieee.org/document/9661268.

5. Tao W, Li C, Song R, Cheng J, Liu Y, Wan F, et al. EEG-based emotion recognition via channel-wise attention and self attention. *IEEE Trans Affective Comput* (2023) 14(1):382–93. doi:10.1109/taffc.2020.3025777

6. Ekman P. Facial expression and emotion (1993). Available from: https://www.semanticscholar.org/paper/Facial-Expression-and-Emotion-Ekman/b0153a91c7124644f8515625e3a0e41193b2fc23.

7. Roidl E, Frehse B, Höger R. Emotional states of drivers and the impact on speed, acceleration and traffic violations—a simulator study. *Accid Anal Prev* (2014) 70:282–92. doi:10.1016/j.aap.2014.04.010

8. Jeon M, Walker BN, Yim JB. Effects of specific emotions on subjective judgment, driving performance, and perceived workload. *Transportation Res F: Traffic Psychol Behav* (2014) 24:197–209. doi:10.1016/j.trf.2014.04.003

9. Oh G, Jeong E, Kim RC, Yang JH, Hwang S, Lee S, et al. Multimodal data collection system for driver emotion recognition based on self-reporting in real-world driving. *Sensors* (2022) 22(12):4402. doi:10.3390/s22124402

10. Singh RR, Conjeti S, Banerjee R. Biosignal based on-road stress monitoring for automotive drivers. In: 2012 National Conference on Communications (NCC); 03-05 February 2012; Kharagpur, India (2012). p. 1–5. Available from: https://ieeexplore.ieee.org/document/6176845.

11. Singh RR, Conjeti S, Banerjee R. An approach for real-time stress-trend detection using physiological signals in wearable computing systems for automotive drivers. In: 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC); 05-07 October 2011; Washington, DC, USA (2011). p. 1477–82. Available from: https://ieeexplore.ieee.org/document/6082900.

12. Muhammad G, Hossain MS. Light deep models for cognitive computing in intelligent transportation systems. *IEEE Trans Intell Transportation Syst* (2023) 24(1):1144–52. doi:10.1109/tits.2022.3171913

13. Prasolenko O, Lobashov O, Bugayov I, Gyulyev N, Filina-Dawidowicz L. Designing the conditions of road traffic in the cities taking into account the human factor. In: 2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS); 05-07 June 2019; Cracow, Poland (2019). p. 1–8. Available from: https://ieeexplore.ieee.org/document/8883381.

14. Lingelbach K, Bui M, Diederichs F, Vukelić M. Exploring conventional, automated and deep machine learning for electrodermal activity-based drivers' stress recognition. In: 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC); 17-20 October 2021; Melbourne, Australia (2021). p. 1339–44. Available from: https://ieeexplore.ieee.org/document/9658662.

15. Ujir H, Jee EM, Farhaan Iqbal M, Mun QK, Hipiny I. Real-time driver's monitoring mobile application through head pose, drowsiness and angry detection. In: 2021 8th International Conference on Computer and Communication Engineering (ICCCE); 22-23 June 2021; Kuala Lumpur, Malaysia (2021). p. 1–6. Available from: https://ieeexplore.ieee.org/document/9467232.

16. Chand V, Karthikeyan J. CNN based driver drowsiness detection system using emotion analysis. *Intell Automation Soft Comput* (2022) 31:717–28. doi:10.32604/IASC.2022.020008

17. Du G, Wang Z, Gao B, Mumtaz S, Abualnaja KM, Du C. A convolution bidirectional long short-term memory neural network for driver emotion recognition. *IEEE Trans Intell Transportation Syst* (2021) 22(7):4570–8. doi:10.1109/tits.2020.3007357

18. Li W, Cui Y, Ma Y, Chen X, Li G, Zeng G, et al. A spontaneous driver emotion facial expression (DEFE) dataset for intelligent vehicles: emotions triggered by video-audio clips in driving scenarios. *IEEE Trans Affective Comput* (2023) 14(1):747–60. doi:10.1109/taffc.2021.3063387

19. Luo J, Yoshimoto H, Okaniwa Y, Hiramatsu Y, Ito A, Hasegawa M. Emotion monitoring sensor network using a drive recorder. In: 2023 IEEE 15th International Symposium on Autonomous Decentralized System (ISADS); 15-17 March 2023; Mexico City, Mexico (2023). p. 1–8. Available from: https://ieeexplore.ieee.org/document/10092139

20. Miyajia M. Driver's anger state identification by using facial expression in cooperation with artificial intelligence. *J Fundam Appl Sci* (2017) 9(7S):87–97. doi:10.4314/JFAS.V9I7S.9

21. Leone A, Caroppo A, Manni A, Siciliano P. Vision-based road rage detection framework in automotive safety applications. *Sensors* (2021) 21(9):2942. doi:10.3390/s21092942

22. Oh G, Ryu J, Jeong E, Yang JH, Hwang S, Lee S, et al. DRER: deep learning–based driver's real emotion recognizer. *Sensors* (2021) 21(6):2166. doi:10.3390/s21062166

23. Jaderberg M, Simonyan K, Zisserman A, Kavukcuoglu K. Spatial transformer networks. *arXiv* (2016). Available from: http://arxiv.org/abs/1506.02025. doi:10.48550/arXiv.1506.02025

24. Viola P, Jones MJ. *Robust real-time face detection*.

25. Wang Q, Wu B, Zhu P, Li P, Zuo W, Hu Q. ECA-net: efficient Channel Attention for deep convolutional neural networks. In: *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (2020). p. 11531–9. Available from: https://ieeexplore.ieee.org/document/9156697.

26. Jeong M, Ko BC. Driver's facial expression recognition in real-time for safe driving. *Sensors* (2018) 18(12):4270. doi:10.3390/s18124270

27. Kingma DP, Ba J. Adam: a method for stochastic optimization. *arXiv* (2017). Available from: http://arxiv.org/abs/1412.6980. doi:10.48550/arXiv.1412.6980

28. Jeong M, Nam J, Ko BC. Lightweight multilayer random forests for monitoring driver emotional status. *IEEE Access* (2020) 8:60344–54. doi:10.1109/access.2020.2983202

29. Sahoo GK, Das SK, Singh P. Deep learning-based facial emotion recognition for driver healthcare. In: 2022 National Conference on Communications (NCC).; 24-27 May 2022; Mumbai, India (2022). p. 154–9. Available from: https://ieeexplore.ieee.org/document/9806751.

30. Podder T, Bhattacharya D, Majumdar A. Time efficient real time facial expression recognition with CNN and transfer learning. *Sādhanā* (2022) 47(3):177. doi:10.1007/s12046-022-01943-x

31. Kaviya P, Arumugaprakash T. Group facial emotion analysis system using convolutional neural network. In: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184); 15-17 June 2020 (2020). p. 643–7. Available from: https://ieeexplore.ieee.org/document/9143037.

32. Dumitru Goodfellow L, Cukierski W, Bengio Y. Challenges in Representation Learning: Facial Expression Recognition Challenge. *Kaggle* (2013). Available online at: https://kaggle.com/competitions/challenges-in-representation-learning-facial-expression-recognition-challenge.

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 22 April 2024
DOI 10.3389/fphy.2024.1350117

OPEN ACCESS

EDITED BY
Jianping Gou,
Southwest University, China

REVIEWED BY
Peilin He,
University of Pittsburgh, United States
Xiaolei Liu,
China Academy of Engineering Physics, China

*CORRESPONDENCE
Weisha Zhang,
✉ zhangweisha@uestc.edu.cn

RECEIVED 05 December 2023
ACCEPTED 20 March 2024
PUBLISHED 22 April 2024

CITATION
Zhang W, Liu J, Peng J, Liu Q and Yu K (2024), Malware traffic detection based on type II fuzzy recognition.
Front. Phys. 12:1350117.
doi: 10.3389/fphy.2024.1350117

Malware traffic detection based on type II fuzzy recognition

Weisha Zhang[1]*, Jiajia Liu[2], Jimin Peng[2], Qiang Liu[2] and Kun Yu[2]

[1]School of Foreign Languages, University of Electronic Science and Technology of China, Chengdu, China, [2]Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu, China

In recent years, a surge in malicious network incidents and instances of network information theft has taken place, with malware identified as the primary culprit. The primary objective of malware is to disrupt the normal functioning of computers and networks, all the while surreptitiously gathering users' private and sensitive information. The formidable concealment and latency capabilities of malware pose significant challenges to its detection. In light of the operational characteristics of malware, this paper conducts an initial analysis of prevailing malware detection schemes. Subsequently, it extracts fuzzy features based on the distinct characteristics of malware traffic. The approach then integrates traffic detection techniques with Type II fuzzy recognition theory to effectively monitor malware-related traffic. Finally, the paper classifies the identified malware instances according to fuzzy association rules. Experimental results showcase that the proposed method achieves a detection accuracy exceeding 90%, with a remarkably low false alarm rate of approximately 5%. This method adeptly addresses the challenges associated with malware detection, thereby making a meaningful contribution to enhancing our country's cybersecurity.

KEYWORDS

traffic detection, Malware, fuzzy recognition, feature extraction, association rules

1 Introduction

The Internet hosts various forms of malware, including botnets, network worms, and malicious phishing websites. This category of malware exhibits distinct characteristics of malicious network behaviors, such as spam dissemination, the presence of malicious crawlers, dos attack, and port scanning. These activities have a detrimental impact on the network data security of both users and enterprises, posing a significant threat to the network information security of society and the country. Malware has the capability to establish a persistent, malicious controlling network topology that is highly contagious. Port-scanning malware conducts polling attacks on the ports of target computers, particularly targeting commonly used 80. Once the port is attacked and occupied by malware, it significantly disrupts the normal operation of web pages and hampers users' regular Internet activities.

Computer users frequently navigate through a substantial number of web pages during their internet browsing activities. Consequently, numerous malware instances are deployed across a plethora of websites. This situation exposes users to considerable risks as they traverse the web, increasing the likelihood of falling into traps that can result in the compromise of their information, privacy, and property. This phenomenon poses a significant threat to both social and national internet security, eroding the trust of the majority of internet users in the national internet security system.

Simultaneously, the structure of internet cybersecurity is intricate, making it challenging to precisely characterize the evolving features of data traffic in certain research fields due to the inherent uncertainty in research outcomes. Model mathematics emerges as a solution to address this challenge. The primary role of fuzzy mathematics is to effectively blur the boundaries of dichotomous problems. Consequently, this paper employs fuzzy recognition methodology for the detection of malware traffic in network communication.

The article has two innovative points:

Firstly, it involves conducting a statistical analysis of malware traffic through the application of fuzzy recognition theory. Utilizing fuzzy membership functions, a substantial volume of traffic is assessed, and the ambiguity characteristics are employed to extract the value range denoting the maliciousness of each traffic.

Secondly, the malware detection technique, supported by the innovative fuzzy mathematics theory, is aptly tailored for the demands of the current Internet era. Furthermore, tools for malware identification based on fuzzy recognition are anticipated to gain widespread adoption among a diverse range of cybersecurity enterprises and professionals, fueled by continuous innovation and improvement.

The remainder of this article is structured as follows. Section 2 is the related works of this paper and introduces the theoretical foundation and methodology of this paper. Section 3 describes Malware Feature Extraction. Section 4 describes Malware identification and classification. Section 5 discusses the experiments and experimental results. Section 6 concludes the paper with some final remarks and future research directions.

2 Related works

With the rapid proliferation of malware, traditional static analysis techniques are no longer sufficient to meet the demands of detection. Consequently, the adoption of fuzzy recognition theory for classifying and detecting malware has become increasingly prominent. An illustrative example of this is the application of fuzzy recognition to analyze network patterns and behavioral styles. Fuzzy mathematics [1] represents a contemporary branch of mathematics that emerged in the 20th century. It relies on fuzzy concepts to enable the estimation and computation of subjects that are not readily addressed by classical mathematics.

In some foreign countries, current practices involve employing two-way traffic analysis [2] and sensory inspection of network data packets [3] to detect specific states of malware, such as inward scanning, exploits, egg downloading, outward parallel sessions, *etc.* When these particular states align with predefined rules, they are identified as malicious traffic.

In certain security domains in China, the analysis of malware traffic predominantly relies on the WinPcap [4] function library, supplemented by external dependent software and applications. Enterprises leverage their internal functions. An application designed for monitoring network traffic or a user-friendly desktop application is developed in alignment with the specified software system functionality and the assessment of malicious traffic [5]. Subsequently, the proposed scheme outlines the specific system structure and optimization process diagram for each module.

In general, numerous cybersecurity projects, both domestically and internationally, have proposed effective solutions to mitigate excessive reliance on source IP, target IP, and the number of host ports during traffic monitoring. This particular scheme involves analyzing whether the uplink effective load and total downlink load of Internet traffic contain distinctive signatures or markers associated with known malicious programs or software for traffic classification [6]. It subsequently calculates fuzzy feature values, thereby achieving a high level of accuracy to some extent. Despite its accuracy, this solution entails a high analytical cost and demands significant effort. To alleviate resource consumption in terms of cost, time, and space, it can be synergistically employed with already analyzed and low-cost monitoring methods. This way, it can efficiently filter out straightforward and easily analyzable traffic in the initial stages.

In addressing the aforementioned challenges, this article employs malware detection tools grounded in fuzzy mathematics as the theoretical foundation, with fuzzy recognition theory serving as the detection method. Through extensive experimentation involving data statistics and analysis of data traffic packet captures, the study aims to offer practical assistance in enhancing the security of personal computers or enterprise extranets. The objective is to furnish efficient tools for inspecting malware traffic and analyzing data packets. By scrutinizing traffic characteristics, along with the expansive range of malware traffic, and employing fuzzy membership function calculations, the proposed approach aims to effectively and efficiently identify malware within IoT or personal computer network cards.

3 Malware feature extraction

3.1 Feature classification

Cluster analysis serves as a pivotal method in fuzzy feature classification. Cluster analysis serves as a valuable tool in identifying fuzzy patterns and similarities within data, providing a means to navigate the inherent ambiguity and uncertainty associated with the classification of such fuzzy features. Additionally, cluster analysis aids in the exploration of the intrinsic structure and patterns embedded in the data. This facet is particularly crucial for the classification of fuzzy features, as they may be inherent in the data's structure and can be better comprehended and recognized through the clustering process.

Moreover, the method's applicability to large-scale datasets further enhances its significance in the realm of fuzzy feature classification. These attributes collectively render cluster analysis advantageous and highly applicable in addressing the challenges posed by fuzzy features in classification tasks.

In this experiment, the classification of traffic features is categorized into five intervals based on the similarity of features and their close relationships: malicious traffic, approximate

TABLE 1 Fuzzy feature range of traffic.

	1 (%)	2	3	4 (%)	5 (%)
all_pkts	94.3–100	66.8%–84.4%	44.1%–47.1%	18.5–32.7	0–4.2
up_pkts	218–230	122.8%–160%	75.9%–97.2%	25.1–58.2	0–22.2
dw_pkts	97.2–99.9	92.9%	35.7%–42.4%	27–25.2	0–8.5
up_pl_pkts	84.7–100	66.7%–71%	38.5%–52.8%	22.3–36.3	0–9.7
dw_pl_pkts	94.3–100	66.8%–84.4%	44.1%–47.1%	18.5–32.7	0–4.2
up_pl_byte	218–230	122.8%–160%	75.9%–97.2%	25.1–58.2	0–22.2
dw_pl_byte	97.2–99.9	92.9%	35.7%–42.4%	27–25.2	0–8.5
duration	84.7–100	66.7%–71%	38.5%–52.8%	22.3–36.3	0–9.7
up_avg_plsize	94.3–100	66.8%–84.4%	44.1%–47.1%	18.5–32.7	0–4.2
dw_avg_plsize	218–230	122.8%–160%	75.9%–97.2%	25.1–58.2	0–22.2
up_min_plsize	97.2–99.9	92.9%	35.7%–42.4%	27–25.2	0–8.5
dw_min_plsize	84.7–100	66.7%–71%	38.5%–52.8%	22.3–36.3	0–9.7
up_max_plsize	218–230	122.8%–160%	75.9%–97.2%	25.1–58.2	0–22.2
dw_max_plsize	97.2–99.9	92.9%	35.7%–42.4%	27–25.2	0–8.5
up_stdev_size	84.7–100	66.7%–71%	38.5%–52.8%	22.3–36.3	0–9.7
dw_stdev_size	94.3–100	66.8%–84.4%	44.1%–47.1%	18.5–32.7	0–4.2
up_avg_ipt	218–230	122.8%–160%	75.9%–97.2%	25.1–58.2	0–22.2
dw_avg_ipt	97.2–99.9	92.9%	35.7%–42.4%	27–25.2	0–8.5
up_min_ipt	84.7–100	66.7%–71%	38.5%–52.8%	22.3–36.3	0–9.7
dw_min_ipt	84.7–100	66.7%–71%	38.5%–52.8%	22.3–36.3	0–9.7
up_max_ipt	94.3–100	66.8%–84.4%	44.1%–47.1%	18.5–32.7	0–4.2
dw_max_ipt	218–230	122.8%–160%	75.9%–97.2%	25.1–58.2	0–22.2
up_stdev_ipt	97.2–99.9	92.9%	35.7%–42.4%	27–25.2	0–8.5
dw_stdev_ipt	84.7–100	66.7%–71%	38.5%–52.8%	22.3–36.3	0–9.7

malicious traffic, no obvious features, approximate normal traffic, and normal traffic. Striking a balance is crucial; excessive intervals (>5) can diminish recognition accuracy, leading to frequent results spanning two intervals simultaneously, thereby introducing ambiguity. Conversely, few classification intervals (0–5) can also elevate recognition ambiguity, making it challenging to clearly discern the malicious nature of the traffic. Achieving an optimal number of intervals is key to ensuring accurate and unambiguous traffic feature classification results.

KM fuzzy clustering [7] classifies the feature classification of malicious traffic into five intervals. This method groups the values recorded under the same quantitative feature into the target dataset. For each interval, it extracts the maximum and minimum values, using the minimum value as the closed left endpoint and the maximum value as the closed right endpoint of the interval. A comprehensive analysis and summary of numerous sets of malicious data have been conducted, as illustrated in Table 1. The data characteristics represented are: total number of data packets, number of uplink data packets, number of downlink data packets, number of uplink loads, number of downlink loads, total uplink load, total downlink load, flow duration, uplink data Avg, downlink data Avg, Uplink minimum load, downlink minimum load, uplink maximum load, downlink maximum load, uplink load variance, downlink load variance, uplink load Avg, downlink load Avg, uplink minimum data, downlink minimum data, uplink maximum data, downlink maximum data, uplink data Variance, downward data variance.

3.2 Feature extraction

3.2.1 Feature calculation

Feature extraction represents a pivotal step in fuzzy recognition. To extract fuzzy features, the initial task involves determining the weight of each traffic feature. Subsequently, an extensive examination and analysis of the range of each feature across all malicious traffic instances are conducted using big data. The testing data for this experiment is sourced from a malware simulator, which generates malicious traffic. This traffic is then combined with normal traffic. The membership function is pre-established based on the characteristics of traffic emitted by known malware, resulting in a unique function. Following this, the dataset is utilized for testing, aiming to identify the number of malicious traffic instances, analyze the type of malware, and ultimately calculate the proportion and false alarm rate.

In the experimental section, we scrutinize the features of captured malicious network samples and public datasets, extracting distinct characteristics of malware traffic. These characteristics encompass the five-tuple [8], packet size, port number, DNS response time, and data packet load. Each type of malware exhibits its unique traits. During the statistical analysis, we filter out all malicious traffic instances, focusing on extracting abnormal characteristics from malicious traffic and HTTP network traffic.

This experiment primarily employs the method of fuzzy cluster analysis for malware identification. In traffic clustering, we utilize common clustering algorithms such as database scanning, memory sharing, K-Means, and design pattern [9]. Particularly, when handling substantial data with high concurrency, the KM algorithm proves effective in revealing the actual distribution and transmission of traffic.

As a result, the membership function will be translated into program code, and the likelihood of malicious traffic will be calculated by executing the program.

TABLE 2 Traffic feature range and weight.

	Weights (%)	Upper range (%)	Lower bound of range (%)	Optimal number of classifications
all_pkts	84.4	44.1–47.1	18.5–32.7	4
up_pkts	31	75.9–97.2	25.1–58.2	3
dw_pkts	92.9	35.7–42.4	27–25.2	5
up_pl_pkts	71.3	38.5–52.8	22.3–36.3	4
dw_pl_pkts	84.4	44.1–47.1	18.5–32.7	5
up_pl_byte	45	75.9–97.2	25.1–58.2	3
dw_pl_byte	92.9	35.7–42.4	27–25.2	3
duration	71	38.5–52.8	22.3–36.3	4
up_avg_plsize	84.4	44.1–47.1	18.5–32.7	5
dw_avg_plsize	21.1	75.9–97.2	25.1–58.2	5
up_min_plsize	92.9	35.7–42.4	27–25.2	3
dw_min_plsize	71	38.5–52.8	22.3–36.3	3
up_max_plsize	36.2	75.9–97.2	25.1–58.2	4
dw_max_plsize	92.9	35.7–42.4	27–25.2	5
up_stdev_plsize	1	38.5–52.8	22.3–36.3	4
dw_stdev_plsize	3.3	44.1–47.1	18.5–32.7	5
up_avg_ipt	9.1	75.9–97.2	25.1–58.2	3
dw_avg_ipt	92.9	35.7–42.4	27–25.2	4
up_min_ipt	71	38.5–52.8	22.3–36.3	5
dw_min_ipt	71	38.5–52.8	22.3–36.3	3
up_max_ipt	84.4	44.1–47.1	18.5–32.7	3
dw_max_ipt	7.4	75.9–97.2	25.1–58.2	5
up_stdev_ipt	92.9	35.7–42.4	27–25.2	3
dw_stdev_ipt	1	38.5–52.8	22.3–36.3	4

We can summarize that there are three types of fuzzy membership functions with a normal distribution:

$$A(x)\begin{cases} 1 & x \le a \\ e^{\frac{-(x-a)}{\delta}} & x > b \end{cases} \tag{1}$$

$$A(x)\begin{cases} 1 & x \le a \\ e^{\frac{-(x-a)}{\delta}} & x > a \end{cases} \tag{2}$$

$$A(x)\begin{cases} 1 & x \le a \\ e^{\frac{-(x-a)}{\delta}} & x > b \end{cases} \tag{3}$$

In this experiment, the network is modeled, analyzed, and detected based on the ecological characteristics of malicious traffic. These ecological characteristics encompass the utilization of commands to control communication traffic, which is generated during the propagation of the network and when the malware reaches a certain scale.

During the generation and dissemination of malicious traffic by malicious application software, fuzzy mathematics establishes its own models and conducts extensive calculations and statistical analysis on the fuzzy characteristics of network data structures. By considering the fuzzy characteristics specific to malicious network traffic, we ultimately perform clustering on such traffic. Leveraging big data investigation methods, we monitor malicious network traffic to detect common malicious software and applications. The fuzzy recognition scheme, based on cluster analysis, utilizes fuzzy clustering to analyze malware and identify the technical core of the operating system. Employing high-speed mirroring [10] for saving malicious network traffic, it acts as a snapshot, subsequently stored on the computer's hard disk. This traffic is then input into a malicious network identification system based on cluster analysis. The system filters, monitors, and analyzes the traffic, extracts features, and finally conducts cluster analysis to determine the accuracy or success rate of all enterprise or personal traffic received.

3.2.2 Flow characteristics

The optimal number of classifications is a crucial concept in data feature segmentation within fuzzy recognition theory. Given the intricate nature of traffic features, different characteristics exhibit variations in their thresholds after the application of the fuzzy clustering analysis algorithm. The optimal number of classifications is defined as the number at which the threshold is maximized. At this point, the membership function is most accurate in determining the feature, resulting in the highest fuzzy recognition rate. Subsequently, we calculate the similarity of each feature after determining the optimal number of classifications for individual features. Ultimately, the optimal number of classifications for the entire set of traffic features is determined using the maximum number algorithm, resulting in five intervals.

To align with the feature function described in fuzzy mathematics, this experiment aims to automatically identify fuzzy features, learning their range and weight, as illustrated in Table 2.

3.3 Feature membership function

Since malicious traffic generated by different malware exhibits significant variations in the range of fuzzy features, the fuzzy features' membership function is fine-tuned with the aid of artificial learning. Leveraging an extensive analysis of massive malicious traffic techniques, we scrutinize and compare identical features to ascertain the average value and range of each fuzzy data level. Subsequently, we incorporate this information into the standard fuzzy membership function. Through program recognition and the application of a series of mathematical algorithms, we amalgamate normal distribution characteristics with the membership function of fuzzy features. Each traffic encompasses dozens of fuzzy features, and distinct types of traffic are associated with unique fuzzy features. To illustrate, consider the following fuzzy feature along with its corresponding membership function:

up_pkts:

$$W(x)\begin{cases} 0 & 0 \le x \le 23.7 \\ 1 - e\left(-\dfrac{x-5}{10}\right) & 23.7 \le x \le 41.2 \\ 1 - e^{\left(-\frac{x}{8}\right)} & 41.2 \le x \le 60 \\ e^{\left(-\frac{x-5}{6}\right)} & 60 \le x \le 81 \\ 1 & 81 \le x \le 100 \end{cases} \tag{4}$$

dw_pkts:

$$W(x)\begin{cases} 0 & 0 \le x \le 23.7 \\ 1 - e\left(-\dfrac{x-5}{10}\right) & 23.7 \le x \le 41.2 \\ 1 - e^{\left(-\frac{x}{8}\right)} & 41.2 \le x \le 60 \\ e^{\left(-\frac{x-5}{6}\right)} & 60 < x \le 81 \\ 1 & 81 \le x \le 100 \end{cases} \tag{5}$$

up_pl_pkts:

$$W(x)\begin{cases} 0 & 0 \le x \le 23.7 \\ 1 - e\left(-\dfrac{x-5}{10}\right) & 23.7 \le x \le 41.2 \\ 1 - e^{\left(-\frac{x}{8}\right)} & 41.2 \le x \le 60 \\ e^{\left(-\frac{x-5}{6}\right)} & 60 < x \le 81 \\ 1 & 81 \le x \le 100 \end{cases} \tag{6}$$

dw_pl_pkts:

$$W(x)\begin{cases} 0 & 0 \le x \le 23.7 \\ 1 - e\left(-\dfrac{x-5}{10}\right) & 23.7 \le x \le 41.2 \\ 1 - e^{\left(-\frac{x}{8}\right)} & 41.2 \le x \le 60 \\ e^{\left(-\frac{x-5}{6}\right)} & 60 < x \le 81 \\ 1 & 81 \le x \le 100 \end{cases} \tag{7}$$

Upstream Payload Variance:

$$W(x)\begin{cases} 0 & 0 \le x \le 23.7 \\ 1 - e\left(-\dfrac{x-5}{10}\right) & 23.7 \le x \le 41.2 \\ 1 - e^{\left(-\frac{x}{8}\right)} & 41.2 \le x \le 60 \\ e^{\left(-\frac{x-5}{6}\right)} & 60 < x \le 81 \\ 1 & 81 \le x \le 100 \end{cases} \tag{8}$$

Downstream Payload Variance:

$$W(x)\begin{cases} 0 & 0 \le x \le 23.7 \\ 1 - e\left(-\dfrac{x-5}{10}\right) & 23.7 \le x \le 41.2 \\ 1 - e^{\left(-\frac{x}{8}\right)} & 41.2 \le x \le 60 \\ e^{\left(-\frac{x-5}{6}\right)} & 60 < x \le 81 \\ 1 & 81 \le x \le 100 \end{cases} \tag{9}$$

In the traffic analysis of certain software, the time interval proves to be a crucial feature. Therefore, it is essential to examine the fuzzy feature of time intervals. This involves calculating the minimum, maximum, average, and variance of the uplink and downlink time intervals. Subsequently, the weight of the time interval in the overall fuzzy recognition feature is determined through computation:

Upstream Mean Time Interval:

$$W(x)\begin{cases} 0 & 0 \le x \le 23.7 \\ 1 - e\left(-\dfrac{x-5}{10}\right) & 23.7 \le x \le 41.2 \\ 1 - e^{\left(-\frac{x}{8}\right)} & 41.2 \le x \le 60 \\ e^{\left(-\frac{x-5}{6}\right)} & 60 < x \le 81 \\ 1 & 81 \le x \le 100 \end{cases} \tag{10}$$

Downstream Mean Time Interval:

$$W(x)\begin{cases} 0 & 0 \le x \le 23.7 \\ 1 - e\left(-\dfrac{x-5}{10}\right) & 23.7 \le x \le 41.2 \\ 1 - e^{\left(-\frac{x}{8}\right)} & 41.2 \le x \le 60 \\ e^{\left(-\frac{x-5}{6}\right)} & 60 < x \le 81 \\ e^{\left(-\frac{x-5}{6}\right)} & 81 < x \le 100 \end{cases} \tag{11}$$

4 Malware identification and classification

4.1 Fuzzy recognition process

Firstly, in fuzzy recognition theory, the identification process begins with defining the object to be recognized. Subsequently, the fuzzy characteristics of the target object are analyzed. Finally, ambiguity is calculated through functions, and the result interval is determined to achieve the recognition and classification of malware. For the experiment, the identification process is divided into the following steps:

1. Identifying the target object involves the initial step of capturing and filtering data packets that satisfy specific identification criteria. Subsequently, the traffic is regarded as the object of identification in accordance with the five-tuple principle.
2. Conducting fuzzy feature calculation and classification involves the determination of fuzzy features within the data flow. Subsequently, cluster analysis is applied to ascertain the optimal number of interval classifications for each identified feature. Ultimately, this process culminates in the establishment of classification intervals for the entirety of the fuzzy recognition procedure.
3. Following the establishment of classification intervals, the evaluation of fuzzy features involves assessing the degree of variation using a normal distribution. Employing the fuzzy membership functions derived from the normal distribution, determine the membership functions for each feature within the data flow.
4. Upon completing all prerequisites, execute fuzzy recognition computations within the program. Integrate the interval values of data flow features and the pre-determined membership functions of features into the program to achieve comprehensive fuzzy software recognition.
5. Ultimately, leveraging the fluctuation range of recognized data flow features, proceed with the identification and classification of malicious software.

4.2 Type I fuzzy recognition

Fuzzy recognition theory posits that the attributes of the object undergoing recognition exhibit fuzziness throughout the recognition process. In other words, the standard fuzzy model is inherently fuzzy. Type I fuzzy recognition theory involves the manual determination of the variable range of data characteristic ambiguity. This is achieved through human learning and experiential judgment, leading to the subdivision of intervals for data characteristics. By iteratively tuning and calculating, we identify the maximum threshold through cluster analysis, facilitating subsequent stages of identification.

Type I fuzzy recognition theory focuses on the proximity of data features. The proximity of fuzzy sets is inversely proportional to the size of the outer product: the closer the fuzzy set, the smaller the outer product. Conversely, the larger the inner product, the closer the fuzzy set. Hence, closeness serves as a metric to depict the similarity between two fuzzy sets.

The algorithmic principles guiding the design of the first type of fuzzy recognition theory include:

1. Maximum Membership Principle;
2. Threshold Principle; This fuzzy algorithm employs a fuzzy decision-making method to prescribe a specific design plan, addressing issues that may arise in the current or future selection of the optimal plan. The objective of fuzzy decision-making [11] is to rank objects in the domain by considering their superiority and inferiority, or to choose a satisfactory plan from the domain using a predefined method. Ultimately, the application of fuzzy decision-making is specifically manifested in the realms of scientific technology and economic management.

4.3 Type II fuzzy recognition

In practical datasets, instances often emerge where data display ambiguity and uncertainty. Traditional binary classification methods may fall short in effectively addressing such complexities. The Type II fuzzy recognition theory excels in handling issues related to fuzziness and uncertainty, providing a robust framework for the classification and recognition of data characterized by fuzziness.

In intricate scenarios, data features can become highly complex, posing a challenge for traditional classification methods to adapt effectively. The Type II fuzzy recognition theory demonstrates notable prowess in managing large volumes of complex data, enabling the classification and recognition of extensive datasets. This capability significantly enhances the accuracy and efficiency of data processing in such complex situations.

Type II fuzzy recognition theory differs from Type I in that it mandates that the feature set to be recognized possesses attributes that either completely belong or do not belong at all. In other words, there is a stringent [0,1] closed interval constraint between feature elements and fuzzy sets within the fuzzy model lib. Unlike Type I recognition, Type II recognition dispenses with the need for fuzzy cluster analysis, as it eschews artificial feature interval classification. Instead, it directly determines the membership function based on the established fuzzy standard model lib, thereby facilitating the identification process.

The design of Type II fuzzy recognition theory adheres to the following principles:

1. The Proximity Principle.
2. Multi-characteristic Proximity Principle.

We leverage the fuzzy association rules within Type II fuzzy recognition theory for the classification of fuzzy features. The primary objective involves parsing and calculating the entire dataset of malware traffic using big data techniques. Subsequently, we determine the degree of membership for each fuzzy feature and assess fuzzy association rules based on support and trust criteria.

Compared to Type I fuzzy recognition, Type II fuzzy recognition can adjust recognition methods according to specific situations, better adapting to different scenarios and requirements, thereby improving the flexibility and applicability of recognition. Moreover, Type II fuzzy recognition methods have relatively lower requirements on hardware devices and software running memory, making them better suited to meet the needs in resource-limited environments.

Type II fuzzy recognition introduces a novel concept known as closeness [12]. In contrast to Type I's fuzzy recognition theory, Type II involves the comparison of two fuzzy sets: the model fuzzy set and the standard fuzzy set. The process entails identifying the affiliation between these sets, establishing fuzzy subsets, and determining the closeness between subsets and supersets.

4.4 Malware classification

Following the completion of fuzzy identification on the traffic data collection, a subsequent analysis is essential to extract malicious traffic and ascertain the type of malware. In this experiment, the calculation of the fuzzy degree of trust is conducted based on the fuzzy association rules outlined in Section 3. Ultimately, the degree of trust, represented by FConf, is employed for the classification of malware:

Dos attack: The predominant fuzzy feature, given its significant weight, is the time interval of the traffic. Additionally, the port number serves as a robust criterion for determining its nature. This type of malware is classified as C1.

Web crawlers: These malware entities engage in the unauthorized retrieval of users' or enterprises' data through the transmission of malicious crawler data. Consequently, this type of malware falls under the classification C2.

Mail interception: This category involves the interception or camouflage of Internet mail on the designated network card. Hence, this type of malware is classified as C3.

Phishing websites: This category involves the deployment of phishing advertisements or the malicious download of phishing software on specific websites. The primary goal is to illicitly obtain users' information, primarily targeting individual users. This type of malware is classified as C4.

Port scanning: This type of malware engages in extensive port scanning processes on corporate extranets to identify available ports. It subsequently floods idle ports with numerous malicious and invalid data packets, creating confusion in corporate network data and disrupting the analysis of network traffic. Consequently, this type of malware is classified as C5.

In addition to the aforementioned malware, there are numerous comprehensive threats, including Pajio and Ofred, among others. These comprehensive malware variants use a diverse range of malicious attack methods. Consequently, a thorough traffic analysis of this type of malware necessitates multiple iterations for comprehensive understanding.

5 Experiments

5.1 Data sets

In this experiment, the dataset comprises network data packets, with the traffic data collection predominantly categorized into two segments: malware traffic data and normal traffic data, as outlined in Table 3 and Table 4:

In this experiment, the malware traffic was generated by the malicious traffic simulator, directing phishing website traffic to the network. Simultaneously, the Doser packet sender executed a simulated Dos attack on a designated port. Furthermore, the MBlocker mail [13] blocking simulation tool intercepted mail on the experimental machines. The normal traffic, on the other hand, involved regular Internet access by the experimental computer network card, encompassing both the sending and receiving processes of network data packets. Various protocols, such as HTTP, FTP, SMTP, RIP, DNS, ARP, were employed in the traffic packets. Once the normal traffic reached a specified volume, it was deliberately mixed with malware traffic packets at an appropriate ratio. Subsequently, the network simulated the reception of malware attacks based on this proportion, thus forming the dataset utilized in this experiment.

5.2 Environment and operation

For environmental configuration, Wireshark was employed in this experiment to capture data packets for testing purposes. The data to be tested originated from traffic transmitted through the SMTP or HTTP protocol. Malicious traffic was generated through the testing of phishing websites and malware, and subsequently captured on the experimental machine, as illustrated in Figure 1.

The equipment used in this experiment comprises an experimental computer equipped with 8.00 GB of memory and a 64-bit operating system, featuring an x64 processor. In addition to Wireshark, the primary software tools include Qt Creator and Visual Studio 2019 as programming tools. The characteristics of the traffics are saved in Excel tables, and the results are documented in Word text files.

The initial phase of this experimental program involves parsing Pcap data packets as the primary input. Specifically, it parses all data packets within files designated with the. pcap suffix and subsequently classifies the traffic. The analysis process is as shown below:

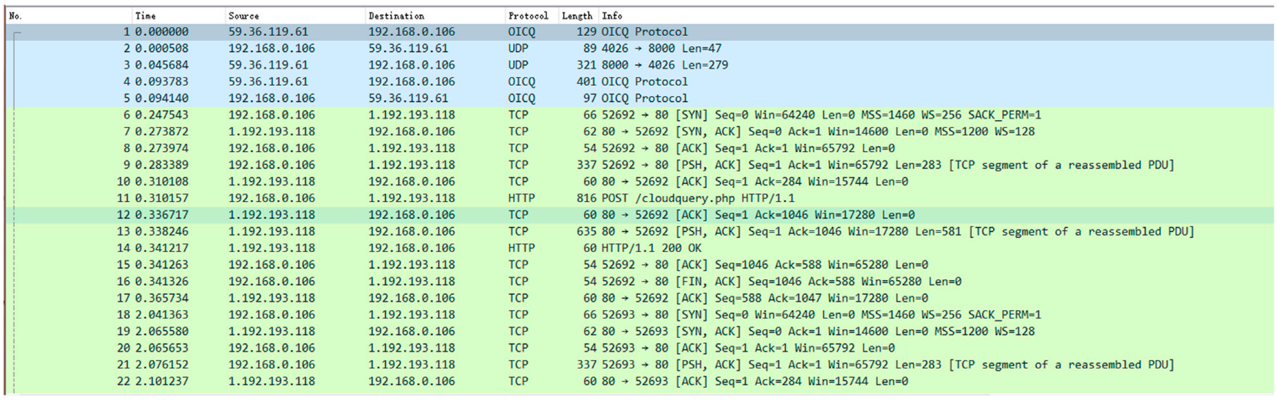
```
Serial Number:90.
89:725906(Len:60) (capLen:60).
5254 00 12 35 02 08 00 27 e6 9f 5f 08 00 45 00.
0028 00 82 40 00 80 06 2b a2 0a 00 02 0f 1f aa
a2 f3 04 1a 00 50 84 31 f7 72 00 03 f9 c8 50 10
fa f0 6c 5d 00 00 00 00 00 00 00 00.
Serial Number:91.
89:725981(Len:60) (cpLen60).
52 54 00 12 35 02 08 00 27 e6 9f 5f 08 00 45 00.
0028 00 83 40 00 80 06 2b a1 0a 00 02 0f 1f aa
a2 f3 04 1a 00 50 84 31 f7 72 00 04 04 e0 50 10
fa f0 61 45 00 00 00 00 00 00 00 00.
Serial Number:92.
89:726024(Len:60) (capLen:60).
52 54 00 12 35 02 08 00 27 e6 9f 5f 08 00 45 00.
00 28 00 82 40 00 80 06 2b a2 0a 00 02 0f 1f aa
a2 f3 04 1a 00 50 84 31 f7 72 00 04 0a 6c 50 10
f5 64 61 45 00 00 00 00 00 00 00 00.
```

TABLE 3 Malicious traffic dataset.

	Total malicious flows (Entries)	Packet size (MB)	Traffic percentage (%)
Dos Attack	1,387	765	1.15
Mail Interception	2,456	874	2.04
Malicious Crawlers	3,399	3,240	2.81
Phishing Websites	2,365	2,310	1.96
Port Scanning	1777	965	1.48
Comprehensive Malware	8,365	1,028	6.97

TABLE 4 Normal traffic dataset.

	Total malicious flows (Entries)	Packet size (MB)	Traffic percentage (%)
HTTP Requests	45630	54420	38
SMTP Mail Requests	14721	12351	12.2
FTP File Requests	11023	9,897	9.19
DNS Domain Requests	21100	7,684	17.5
Telnet	350	5568	0.29
SNMP	765	6,327	0.63



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	59.36.119.61	192.168.0.106	OICQ	129	OICQ Protocol
2	0.000508	192.168.0.106	59.36.119.61	UDP	89	4026 → 8000 Len=47
3	0.045684	59.36.119.61	192.168.0.106	UDP	321	8000 → 4026 Len=279
4	0.093783	59.36.119.61	192.168.0.106	OICQ	401	OICQ Protocol
5	0.094140	192.168.0.106	59.36.119.61	OICQ	97	OICQ Protocol
6	0.247543	192.168.0.106	1.192.193.118	TCP	66	52692 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
7	0.273872	1.192.193.118	192.168.0.106	TCP	62	80 → 52692 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1200 WS=128
8	0.273974	192.168.0.106	1.192.193.118	TCP	54	52692 → 80 [ACK] Seq=1 Ack=1 Win=65792 Len=0
9	0.283389	192.168.0.106	1.192.193.118	TCP	337	52692 → 80 [PSH, ACK] Seq=1 Ack=1 Win=65792 Len=283 [TCP segment of a reassembled PDU]
10	0.310108	1.192.193.118	192.168.0.106	TCP	60	80 → 52692 [ACK] Seq=1 Ack=284 Win=15744 Len=0
11	0.310157	192.168.0.106	1.192.193.118	HTTP	816	POST /cloudquery.php HTTP/1.1
12	0.336717	1.192.193.118	192.168.0.106	TCP	60	80 → 52692 [ACK] Seq=1 Ack=1046 Win=17280 Len=0
13	0.338246	1.192.193.118	192.168.0.106	TCP	635	80 → 52692 [PSH, ACK] Seq=1 Ack=1046 Win=17280 Len=581 [TCP segment of a reassembled PDU]
14	0.341217	1.192.193.118	192.168.0.106	HTTP	60	HTTP/1.1 200 OK
15	0.341263	192.168.0.106	1.192.193.118	TCP	54	52692 → 80 [ACK] Seq=1046 Ack=588 Win=65280 Len=0
16	0.341326	192.168.0.106	1.192.193.118	TCP	54	52692 → 80 [FIN, ACK] Seq=1046 Ack=588 Win=65280 Len=0
17	0.365734	1.192.193.118	192.168.0.106	TCP	60	80 → 52692 [ACK] Seq=588 Ack=1047 Win=17280 Len=0
18	2.041363	192.168.0.106	1.192.193.118	TCP	66	52693 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
19	2.065580	1.192.193.118	192.168.0.106	TCP	62	80 → 52693 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1200 WS=128
20	2.065653	192.168.0.106	1.192.193.118	TCP	54	52693 → 80 [ACK] Seq=1 Ack=1 Win=65792 Len=0
21	2.076152	192.168.0.106	1.192.193.118	TCP	337	52693 → 80 [PSH, ACK] Seq=1 Ack=1 Win=65792 Len=283 [TCP segment of a reassembled PDU]
22	2.101237	1.192.193.118	192.168.0.106	TCP	60	80 → 52693 [ACK] Seq=1 Ack=284 Win=15744 Len=0

FIGURE 1
Packet capture diagram.

5.3 Experimental results

Due to constraints in the program operating environment and limited CPU memory, this experiment will selectively choose a subset of traffic from the dataset as samples for testing and analysis. The goal is to make comprehensive assessments through multiple analysis and comparison processes, as depicted in Table 5 and Table 6.

The analysis results can be accessed and reviewed from the Word document. Each analysis produces a distinct Word document, as illustrated in Table 7.

The malware traffic statistics are presented in Table 8.

The outcomes of the malware identification process are displayed in Table 9.

In addition to checking through the result Word log document, the final parsing results can also be directly obtained from the program's execution results, as shown in Figure 2. The final analysis result of the data in the figure is malware. The total number of data flows is 1,384, the total number of malicious data flows is 11, and the current percentage of malicious data flows is 0.008%, of which 0.0022% is email interception, 0.00122% is Dos attack, 0.00138% is malicious crawler, and 0.00122% is phishing website traffic. 0.00122% is a port scan, the identification success rate is about 96.00%, and the false positive rate is 1%.

In contrast to specific technologies employing approaches focused on monitoring malware traffic, particularly those dependent on the fuzzy characteristics of network data to differentiate between diverse network applications—whether benign or malicious—with the ultimate goal of identifying traffic using fuzzy mathematics. Although most of

TABLE 5 Malicious data flow samples.

	Total malicious flows (Entries)	Packet size (MB)	Traffic percentage (%)
Dos Attack	121	45.4	1.15
Malicious Crawlers	54	36.7	2.04
Phishing Websites	79	79.5	2.81
Port Scanning	83	31.4	1.96
Mail Interception	34	42.6	1.48

TABLE 6 Normal data flow samples.

	Total malicious flows (Entries)	Packet size (MB)	Traffic percentage (%)
HTTP Requests	1,235	32.4	38
SMTP Mail Requests	897	19.9	12.2
FTP File Requests	2,548	32.2	9.19
DNS Domain Requests	1,241	49.1	17.5
Telnet	1,090	14.1	0.29
SNMP	765	22.3	0.63

TABLE 7 Data flow recognition results table.

	Malicious traffic	Approximate malicious traffic	No clear characteristics	Approximate normal traffic	Normal traffic
Fuzziness Level Classification	92.193%	71.020%	51.531%	20.389%	0.896%
Malware Determination	Yes	Yes	Pending	No	No
Number of Data Flows	135	452	654	124	781

TABLE 8 Malware traffic statistics table.

Malware status	Yes
Total Number of Data Flows (Entries)	1,389
Total Number of Malicious Data Flows (Entries)	16
Current Packet Malicious Flow Percentage (%)	0.012%
Recognition Success Rate (%)	96.000%
False Positive Rate (%)	3.000%

TABLE 9 Malware identification results table.

Mail interception	2.67
Dos Attack	4.01
Malicious Crawlers	1.90
Phishing Website Traffic	2.10
Port Scanning	1.88
Trojan Virus	3.00

these methods showcase a low false alarm rate coupled with high accuracy, the primary challenge resides in establishing an appropriate classification basis for the categorization of network traffic.

Our approach demonstrates a feature balance, referring to a rational weighing and adjustment of different features in malicious traffic detection. This ensures that the model comprehensively considers the contribution of each feature, preventing any single feature from becoming overly prominent or dominant, thereby affecting the overall accuracy and stability of detection.

We use fuzzy recognition theory for malicious traffic monitoring. Through the analysis and extraction of malicious traffic features, we determine the importance and weights of different features, maintaining a balance among them. Additionally, we utilize fuzzy feature extraction methods to identify malicious traffic. During feature extraction, it is essential to assess and balance the weights of each feature to ensure the model comprehensively considers their contributions. Finally, we use clustering analysis methods to categorize malicious traffic features into different intervals. Through reasonable classification and interval assignment, we maintain a balance among features, preventing any single feature from becoming overly prominent or dominant.

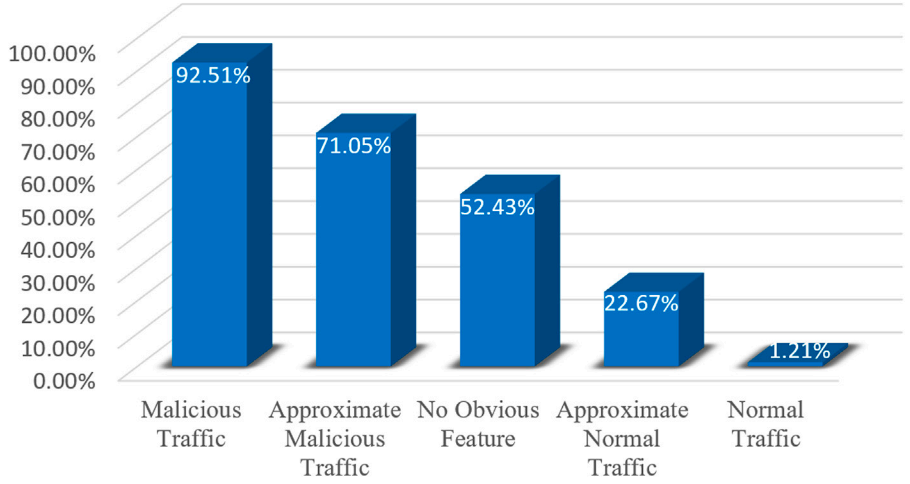


FIGURE 2
Program result graph.

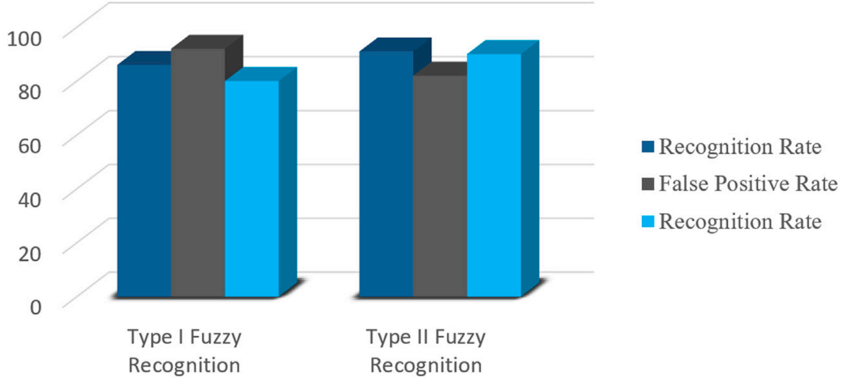


FIGURE 3
Identification scheme comparison chart.

This paper predominantly revolves around the application of the Type II fuzzy recognition theory in the context of a malware traffic detection method. The targeted scenario involves addressing the inherent uncertainty and fuzziness associated with data features. Leveraging fuzzy set theory and its conceptual and operational aspects, the paper employs fuzzy sets to describe and handle the intricacies of the problem.

While both Type I fuzzy recognition and Type II fuzzy recognition fall under the umbrella of fuzzy theory, they diverge in their approaches to problem-solving. Despite these differences, both are dedicated to addressing similar challenges. Therefore, this paper is primarily dedicated to a comparative analysis between Type I fuzzy recognition and Type II fuzzy recognition.

Type II fuzzy recognition methods adeptly handle the volatility of fuzzy features, presenting results within a range interval. Even with minor fluctuations in the results, they do not exert a significant influence on the overall judgment of malicious traffic, thus maintaining a higher level of accuracy.

As the application fields of Type II fuzzy set theory continue to expand with ongoing development, there is a need to delve into the nature and measurement methods of uncertainty within Type II fuzzy sets. Building upon an examination of the uncertainty characteristics and fuzzy entropy of Type II fuzzy sets, we propose the definition of discrete Type II fuzzy set entropy by extending the conventional fuzzy entropy definition. This endeavor opens up novel perspectives and methodologies for the application of Type II fuzzy sets in uncertain environments, as depicted in Figure 3.

It is evident that opting for Type I fuzzy recognition is more rational when dealing with messy data and intricate traffic types. On the other hand, the selection of Type II fuzzy recognition becomes more accurate in scenarios with a substantial volume of data but simpler traffic software types. Tailoring the recognition method to specific circumstances significantly impacts identification accuracy.

Deep learning techniques present a versatile approach to handling situations characterized by vast datasets and intricate data flows. However, in the context outlined in this paper, the application of deep learning typically necessitates a substantial volume of labeled data for effective training. In the domain of cybersecurity, acquiring large-scale labeled data poses challenges, particularly when dealing with labeled data pertaining to malicious traffic.

Furthermore, in the field of network security, the prompt detection of malicious traffic demands real-time responsiveness. The training and inference processes of deep learning models often entail a significant time investment, rendering them unsuitable for meeting real-time requirements. As a result, the constraints related to data availability and real-time processing pose practical challenges to the widespread application of deep learning in the described cybersecurity scenario.

The approach grounded in fuzzy recognition theory proves adept at addressing uncertainty and fuzzy patterns within the realm of network security. It demonstrates adaptability to the intricate characteristics and dynamic changes inherent in malicious traffic. The findings affirm that opting for a method based on fuzzy recognition theory is more fitting within the specific domain and scenario delineated in this paper.

6 Conclusion

In this paper, we introduce a malware traffic detection approach grounded in fuzzy-theory recognition, leveraging fuzzy mathematics as its theoretical foundation. Acknowledging the limitations of the certainty inherent in classical sets, we leverage the ambiguity offered by fuzzy sets to establish the variable range of characteristics for the research object, thereby extending the scope of fuzzy recognition theory. Ultimately, we use membership functions to compute the ambiguity of fuzzy features, providing an effective basis for malware detection.

This method effectively addresses uncertainty and ambiguity within the realm of cybersecurity, showcasing adaptability to the intricate characteristics and dynamic changes inherent in malicious traffic. It automatically recognizes ambiguous features, learning their ranges and weights to accommodate various types and sizes of malicious traffic. The utilization of the maximum number algorithm enhances the precision of classification results, ensuring greater accuracy.

However, it is essential to note that the method's computational process can be complex, particularly when handling large-scale datasets. This complexity may lead to longer processing times and increased demands on computational resources. Fuzzy theoretical models typically involve parameter selection and tuning, and determining the optimal fuzzy set and affiliation function necessitates thorough validation and experimentation.

Based on the above analysis, the next steps in research should focus on the following issues:

1. A versatile and efficient method for collecting multi-source data in network environments, coupled with the rapid evolution of malware targeting backbone networks.
2. A malicious traffic detection technique grounded in the temporal and spatial characteristics of behavior has been devised, endowing it with broader applications and higher efficacy. This technology relies on behavioral patterns over time and space for effective identification of malicious network traffic.
3. The development of three-level hierarchical models encompassing traffic analysis, fuzzy feature recognition, and collaborative decision-making.
4. A collaborative-capable malicious traffic detection system has been created, providing support for multi-party cooperation, thereby comprehensively safeguarding network security. This system is designed to facilitate collaboration among various entities in order to bolster defenses against potential threats.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

WZ: Writing–original draft. JiL: Writing–review and editing. JP: Writing–review and editing. QL: Writing–review and editing. KY: Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. We acknowledge the financial support received for this research from the National Social Science Fund of China under grant number 20CXW016 for the project titled "Computational Communication Strategy for Chinese Technology Image on Twitter", The authors also extend their appreciation to the support of The National Natural Science Foundation of China (Grant No. 62102049). The National Key R&D Program of China (Grant No. 2017YFB0802300). The Natural Science Foundation of Sichuan Province (Grant No. 2022NSFSC0557). The National Key Research and Development Plan of China, Key Project of Cyberspace Security Governance (No. 2022YFB3103103). The Key Research and Development Project of Chengdu (No. 2023-XT00-00002-GX). The Key Research and Development Project of Sichuan Province (No. 2022YFS0571, No. 2021YFSY0012, No. 2021YFG0332, No. 2020YFG0307).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Weiyong Y, Peng L, Jinsuo L, Yibin H. Research on data protection technologies against emerging network threats. *Electric Power* (2014) 12(5):14–5.

2. Harnish R. Cybersecurity in the world of social engineering. *Cybersecurity in Our Digital Lives* (2015) 12(5):20–1.

3. Di W, Xiang C, QiXu L, FangJiao Z. Research on Ubiquitous Botnet. *Inf Netw Security* (2017) 18(7):16–28.

4. Shuning W, Xingru C, Qiang C. Application research of AR-OSELM algorithm in network intrusion detection. *Inf Netw Security* (2017) 17(6):56–7. doi:10.3969/j.issn.1671-1122.2018.06.001

5. Lu J, Chen K, Zhuo Z, Zhang X. A temporal correlation and traffic analysis approach for APT attacks detection. *Cluster Comput* (2019) 22(Suppl 3):7347–7358. doi:10.1007/s10586-017-1256-y

6. Lu J, Lan J, Huang Y, Song M, Liu X. Anti-attack intrusion detection model based on MPNN and traffic spatiotemporal characteristics. *J Grid Computing* (2023) 21:60. doi:10.1007/s10723-023-09703-9

7. Peng L, Wuping W, Shiyong Z. Hybrid network monitoring system based on active networking technology. *Comput Eng Des* (2014) 25(9):1427–31.

8. Jun C. Network traffic management implementation via SNMP protocol. *Coal Technol* (2019) 28(8):162–5.

9. Jun L, Liang X. Distributed network traffic monitoring. *Traffic Manage* (2017) 17(7):56–8.

10. Rosenberg I, Shabtai A, Rokach L, Elovici Y. Generic black-box end-to-end attack against state classifiers. *Intrusions* (2018) 490–510. doi:10.48550/arXiv.1707.05970

11. Wang Q, Guo W, Zhang K, Ororbia A, Xing X, Liu X, et al. Adversary resistant deep neural networks with an applicatn to malware detection. In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining* (2017). p. 1145–53. doi:10.1145/3097983.3098158

12. Kim JY, Bu SJ, Cho SB. Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders. *Inf Sci* (2018) 460:83–102. doi:10.1016/j.ins.2018.04.092

13. Raff E, Barker J, Sylvester J, Brandon R, Catanzaro B, Nicolas C. *Malware detection by eating whole exe*. Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence (2018). p. 531–3. doi:10.48550/arXiv.1710.09435

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 24 April 2024
DOI 10.3389/fphy.2024.1391266

Check for updates

OPEN ACCESS

EDITED BY
Yuanyuan Huang,
Chengdu University of Information Technology,
China

REVIEWED BY
Shaowei Wang,
Guangzhou University, China
Sheng Chen,
Zhejiang Lab, China

*CORRESPONDENCE
Teng Hu,
✉ mailhuteng@gmail.com

RECEIVED 25 February 2024
ACCEPTED 04 April 2024
PUBLISHED 24 April 2024

CITATION
Chang Y, Hu T, Lou F, Zeng T, Yin M and Yang S (2024), Anomalous process detection for Internet of Things based on K-Core.
Front. Phys. 12:1391266.
doi: 10.3389/fphy.2024.1391266

Anomalous process detection for Internet of Things based on K-Core

Yue Chang, Teng Hu*, Fang Lou, Tao Zeng, Mingyong Yin and Siqi Yang

Institute of Computer Application, China Academy of Engineering Physics, Mianyang, China

In recent years, Internet of Things security incidents occur frequently, which is often accompanied by malicious events. Therefore, anomaly detection is an important part of Internet of Things security defense. In this paper, we create a process whitelist based on the K-Core decomposition method for detecting anomalous processes in IoT devices. The method first constructs an IoT process network according to the relationships between processes and IoT devices. Subsequently, it creates a whitelist and detect anomalous processes. Our work innovatively transforms process data into a network framework, employing K-Core analysis to identify core processes that signify high popularity. Then, a threshold-based filtering mechanism is applied to formulate the process whitelist. Experimental results show that the unsupervised method proposed in this paper can accurately detect anomalous processes on real-world datasets. Therefore, we believe our algorithm can be widely applied to anomaly process detection, ultimately enhancing the overall security of the IoT.

KEYWORDS

Internet of Things, process network, white list, anomaly detection, process rank

1 Introduction

Due to the limitations of IoT devices, such as low power consumption, small size and low cost, the security performance of IoT is poor, so these IoT devices are easy to be attacked by hackers. It remains challenging to protect against hacker attacks. There are many factors that can be a threat to server security, such as IoT device vulnerability, virus, malicious procedure, etc. Common attack contains worm, botnet, Trojan horse and DDOS attack (distributed denial of service). Most of these attacks invade the IoT devices by using the malicious process and then hackers implement the further attack. In the network security field, server security occupies an important position. While someone runs the vicious procedure, it always starts some anomalous processes in the IoT devices. If we can detect anomalous process as early as possible, then we will solve the problems in the early stage and avoid heavy loss.

At present, the main technology of anomalous process detection is firewall and intrusion detection technology. As a cordon between the internal network and the public network, the firewall blocks most malicious attacks. However, the effect of the firewall is limited because its defense strategy is static and can only block attacks from the outside network. Intrusion detection technology effectively remedies the short-comings of the firewall. Intrusion Detection System can monitor the real-time status of the IoT devices and detect the anomalous action. Intrusion Detection Systems detect the process mainly by process behavior [1], but there is a problem that we cannot find the anomalous process in time. Besides, Intrusion Detection Systems heavily depend on rules and expert experience. Some researchers proposed to detect anomalous processes based on the system call sequence [2] and this method required kernel process data.

In this work, we detect anomalous processes in IoT devices from a different perspective. The principle of anomalous processes detection is that anomalous processes detection only works in a few IoT devices Therefore, it is necessary to calculate the popularity of Internet of Things devices, and the higher the popularity is, the less suspicious process is. Considering the relationship between the processes and servers, we propose a new approach that can efficiently detect the malicious process in a short time. The proposed approach is as follows. Firstly, we build process white list by using a graph algorithm called K-Core, then we detect the anomalous processes in the servers based on this white list. The strength of this proposed method is that we build the anomaly detection model by constructing the process network. Moreover, our method is unsupervised and we can find out anomalous processes in servers quickly.

The rest of this paper is organized as follows. We briefly review related works in Section 2. Section 3 introduces the proposed method to build weighted process networks. Detailed explanation of detecting anomalous processes is presented in Section 3.3, and the evaluation of the proposed method is discussed in Section 4 and finally we draw a conclusion and discuss future work in Section 5.

2 Related work

The process can be defined as a basic unit of system dynamic execution operation and it is a dynamic concept. Processes in servers are not only the dynamic implementation of programs but also the resource scheduling and allocation. There is a big difference between the process and the program. Programs are static instructions or code sets, while processes are the dynamic execution of programs. Process monitoring is an important part of network security technology. Most intrusion detection systems and anti-virus software have the process monitoring module in the servers.

Considerable research efforts have been devoted to monitoring the process in the servers. In 1996, Stephanie Forrest et al. firstly proposed the delay-embedded sequence model, and analyzed the process behavior based on the system call sequence [2]. In 1998, Hofmeyr proposed an N-gram anomaly detection model [3], which is used to monitor the process behavior. Much work so far has focused on detecting anomalous processes at home and abroad and made outstanding achievements, such as the Fuzzy ART neural network algorithm, the method based on the hidden Markov model, the method based on frequency statistics and method of data mining. The main focus of this research is to extract the characteristics of subsequences for the process. These detection methods ignore the global characteristics of the process with a shortage of poor timeliness.

Besides, much work has focused on anomaly intrusion detection. The system detects anomaly behavior by using statistical profiles such as IDES [4–6], and inductive pattern generation, as in TIM [7]. These methods require an audit trail of actions to all users in servers. Moreover, these detection methods will perform terribly if we change the model of user behavior in a new environment. Levitt et al. proposed the method to define normal behavior for privileged processes [8, 9]. Sezgin proposed the intrusion detection instrument called AID4I and achieved better accuracy than traditional intrusion detection methods in experiments on public datasets [10].

In addition to the detection based on rule matching, there are also researches that use machine learning to detect anomalies. For example, Yang et al. constructed LM-BP algorithm from the characteristics of the Internet of Things [11]. Zhang et al. designed an intrusion detection system based on genetic algorithm and deep confidence network, which can adaptively change the network structure to adapt to different types of attacks in the Internet of Things [12]. Bhatt et al. designed a hybrid machine learning detection system called HADS to detect anomalies in time series with different characteristics generated by Internet of Things devices [13]. Weinger B et al. use supervised deep learning to achieve higher detection accuracy through special processing of data features [14]. Alaiz-Moreton et al. realized anomaly detection by multi-classifying the traffic of IoT devices [15]. Nagarajan proposed a new hybrid deep learning in Industrial Control Systems, which can be used to detect unknown attacks [16]. Al-Wesabi proposed an optimization algorithm based on federated learning and reached great results in IoT attack detection [17].

In this paper, we proposed an unusual method to detect anomalous processes. Firstly, we build the weighted process network based on the relationship between processes and IoT devices. Then creating a process white list by using the K-Core decomposition method. Finally, we can detect anomalous process by comparing with the white list.

3 Materials and methods

Processes running in IoT devices can be divided into four categories: server system process, the third-party application process, user-initiated process, and anomalous process. The first three kinds of processes are common processes, and they are widely distributed in devices. The last kind of process is running in devices by hackers or other attackers. The distribution density in the devices is different among these processes. Then we can establish the process network topology with these distribution characteristics of processes.

In this section, we build a graph of processes based on the relationship between processes and IoT devices. And we will introduce some definitions used in this paper.

3.1 Establishing the process network topology

We mainly collect the following two types of process logs, the real-time status logs of processes in the operating system kernel and snapshots of processes in the server. We monitor the whole life cycle of all processes and find out the popular processes that could be considered as normal processes by analyzing the process graph.

In this paper, an independent process p_i is regarded as a node i. When process p_i and process p_j run simultaneously on more than

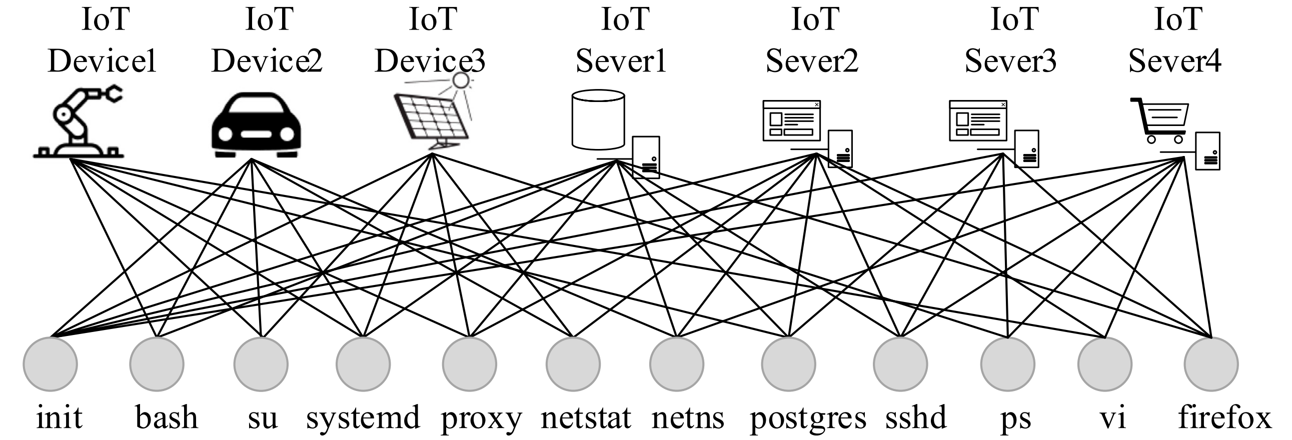


FIGURE 1
Processes in different servers.

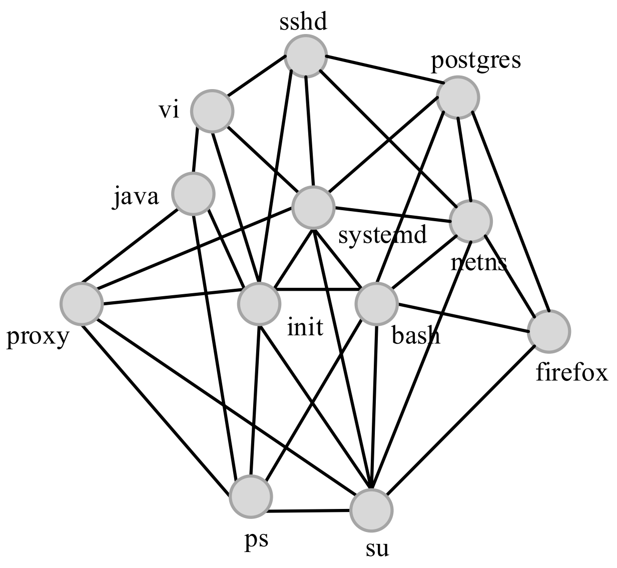


FIGURE 2
Process network.

one server, there is an edge e (i, j) between the corresponding nodes i and j, then we can establish a process network graph. As shown in Figure 1, a process named "init" runs on the database server, and another process named "sshd" runs on the same database server, so there is an edge between two corresponding nodes in the process network graph. By collecting and analyzing all processes data on multiple devices, we can draw the process network topology graph like Figure 2.

System processes such as system daemon processes, system log management processes and kernel processes are very popular in devices. The execution of these processes guarantees the basic functions of devices. We can find these processes in mainstream operating systems such as Ubuntu, CentOS, Red Hat, and Kali. The second variety of processes is mainly third-party processes such as Apache and Nginx applications which are very popular in many devices. The third kind of process is user-initiated process. In enterprises or huge server clusters, users will startup similar processes such as the enterprises' OA system processes. All three of these processes are popular in the devices. And in the process network topology graph, the corresponding nodes play an important role in the whole network graph.

The more popular processes are in the devices, the greater the core degree of the corresponding nodes are, and the more influential they are in the process network topology graph. Therefore, the problem of finding popular processes in devices is transformed into the problem of finding core nodes in the network graph.

3.2 Building the weighted process graph

Graph density is an important index to measure the edge density of a network graph [18]. The specific definition of graph density is the ratio of the actual number of edges to the maximum potential number of edges in the network. After constructing the process network graph on the IoT devices, we found that the value of the process network graph density is too high. And when the number of IoT devices increases, the graph density changes little. Such a process network graph structure is not an ideal model to analyze processes.

We make some improvements to the undirected process network graph shown in Figure 2. The approach is to represent the progress network with a weighted graph. If two independent processes p_i and p_j run on the same n_{ij} $(n_{ij} > 0)$ server, then there is an edge e (p_i, p_j) between two corresponding nodes and the weight of this edge is w_{ij}. The larger the value of n_{ij} is, the larger the corresponding w_{ij} is. The specific weight calculation equation is defined as follows:

$$\boldsymbol{w_{ij}} = \frac{|\boldsymbol{U_i} \cap \boldsymbol{U_j}|}{|\boldsymbol{U_i} \cup \boldsymbol{U_j}|} \tag{1}$$

Where, U_i is the set of devices which process p_i has ever been running in, and U_j is the set of devices which process p_j has ever been running in. $| U_i \cap U_j |$ is the module of intersection and $| U_i \cup U_j |$ is the module of the union. Here, we normalize all the weights to ensure that the correlation degree between two processes is in a reasonable range.

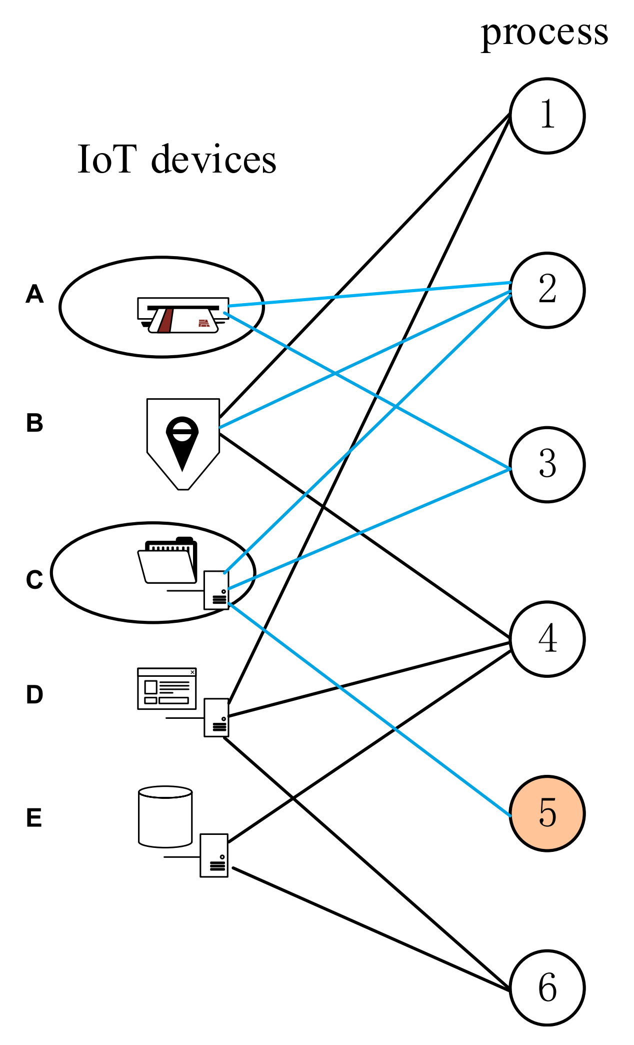


FIGURE 3
Processes in different IoT servers.

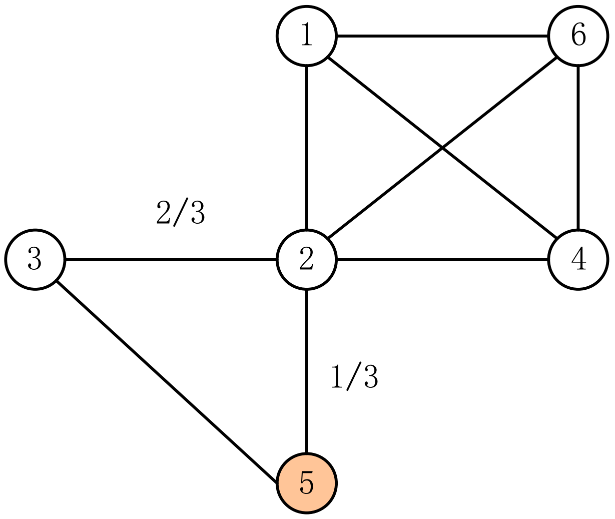


FIGURE 4
Weighted process network.

As shown in Figure 3, process 2 runs in the devices {a, b, c}, process 3 runs in the devices {a, c} and process 5 runs in the devices {c}, then we perform the following steps. First, we gather the set of devices which one process has ever been running in, such that U_2 is equivalent to {a, b, c} . Next, we calculate w_{ij} by using Eq. 1. And the calculation result of w_{23} is $\frac{2}{3}$. After calculating all w_{ij}, we can finally build the weighted graph like Figure 4.

To form the process white list, we impose the following conditions: 1) Malicious processes run on a small number of servers; 2) Processes running on a large number of devices are more trustworthy; 3) The strength of association between two processes is related to the number of common servers they share. Based on the premise, we can further deduce that the number of servers a process resides on has a significant impact on its suspicion level. For example, when a suspicious process and a common process run on the same device, the association between the two processes increases the node degree of the suspicious process, which is not as expected. To model more realistically, it is necessary to eliminate this bias. Therefore, we introduce a weighting factor C, when the number of devices on which a process resides is small, its weighting factor C is small. When the number of devices is large, the weighting factor is large, and it is less influenced by the number of devices. Furthermore, we can define C as following:

$$\boldsymbol{C} = \frac{\mathbf{1}}{\mathbf{1} + \boldsymbol{e}^{\gamma - \boldsymbol{min}\left(|U_i|,|U_j|\right)}} \tag{2}$$

Where, U_i and U_j are the sets of devices, and $\gamma\,(\gamma = 0, 1, \ldots)$ is the correction factor. In different detection scenarios, the value of γ varies. For example, in a certain IoT environment, specific normal processes independently run on 4 optical sensor devices. To increase the confidence in these processes, γ can be set to 2. From the characteristics of the function curve, the weighting factors of these specific processes are significantly greater than those of anomalous processes. According to the definition of the weighting factor C, when the number of devices on which two associated processes reside is large, their weighting factor is large. When the number of devices for one of the processes is small, the calculated weighting factor is small, indicating a higher suspicion level for that process. Therefore, in the current definition, the weighting factor C addresses the issue of the absolute value of device count affecting suspicion levels. Then the new edge weight of the process network is redefined as following by using C in Eq. 2:

$$\boldsymbol{w}_{\boldsymbol{ij}}' = \boldsymbol{w}_{\boldsymbol{ij}} * \boldsymbol{C} \tag{3}$$

With the above processing, we can construct the weighted undirected graph based on the relationship between processes and devices. After analyzing the coreness of all nodes in the graph, we can obtain the widely popular processes in the devices. These popular processes are legitimate and can be used to create process white list.

3.3 Detecting anomalous processes

In this section, K-Core algorithm is proposed to help create a process white list. According to previous classification, the first three

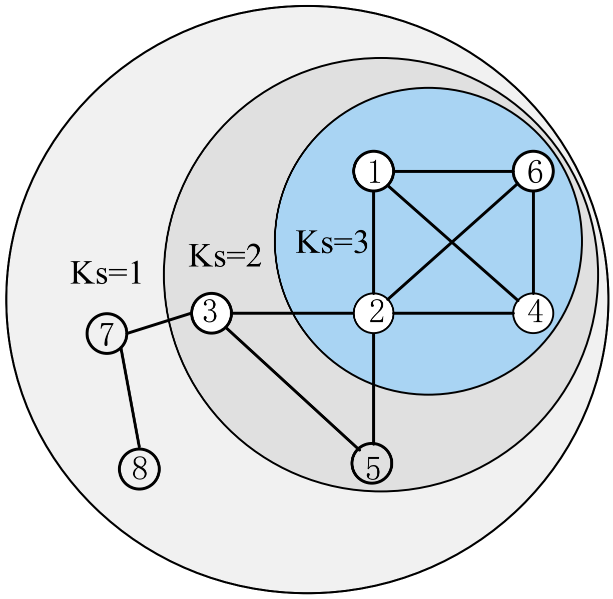


FIGURE 5
Illustration of K-Core concept.

kinds of processes are very popular and their nodes are core nodes. If we find out these core nodes based on K-Core algorithm, then we can find out the popular processes and write them to the white list.

The node importance of complex networks, including the influence, status, the popularity of nodes and the synthesis of these elements, was first raised by social relationship scientists [19, 20]. With the development of society, the research value of this study has been gradually discovered, and it plays an important role in the communication network, social network, and engineering practice. There are many indicators or methods to judge the importance of nodes, such as degree, betweenness, coreness, degree centrality and ranking of node importance based on random walk model [21, 22], the ranking of node importance based on propagation dynamics [23, 24]. Kitsak first proposed that the node importance depends on its location in the network [25], and then used the K-Core decomposition algorithm to rank the importance of nodes. Among these indicators and methods, the coreness can obtain a more accurate ranking of node importance than others such as betweenness. And it is easy to realize with the K-Core algorithm. The time complexity of the K-Core algorithm is O (N), which is suitable for huge complex networks.

3.3.1 Algorithm description

K-Core decomposition method is classical in graph theory which can be used to analyze the importance of nodes in the network [26]. The main idea is to iteratively generate different kinds of node groups with different k-values (k_s). In this work, a k-core is a maximal group of processes, all of which are connected to at least k other processes in the group. K-Core is a measure that can help identify small interlinked core areas on a network.

As shown in Figure 5, there is the K-Core decomposition of a network. The nodes of the outer layer compose shell 1 ($k_s = 1$), while the nodes within the central ring compose shell 3 ($k_s = 3$). We can see that a group is the k-core if it contains all nodes that are connected to at least k other nodes within the group. Besides, the k-core contains the (k + 1)-core. When k increases, the core sizes decrease while the cores become more interlinked and the nodes are more popular. Algorithm 1 provides the pseudocode for finding the k-values of networks.

```
Input: Network graph data.
Output: The list of different k-values
$k_s \leftarrow 0$, $\mathbf{D} \leftarrow [\Phi, \ldots, \Phi]$, $\mathbf{d} \leftarrow \mathbf{0}$, $\mathbf{S} \leftarrow \mathbf{0}$.
for all $i \leftarrow 0$ to n do
    $\mathbf{d}[i] \leftarrow$ degree($i$)
    $\mathbf{D}[\mathbf{d}[i]]$.add($i$)
end for
for $k$ to $degree(max)$ do
    $k_s = max\{k_s, k\}$
    while $\mathbf{D}[k]$ not empty do
        $\mathbf{D}[k]$.remove($vertex$)
        $\mathbf{S}[vertex] \leftarrow k_s$
    end while
    $k_s \leftarrow k_s + 1$
end for
return $\mathbf{S}$
```

Algorithm 1. K-Core decomposition.

In Algorithm 1, we define three arrays, D which contains a list of the nodes with a different degree, d which holds the degree of each node and S holds the k-values of all nodes. Firstly, we initialize three arrays and k_s. Then we increase the value of k from minimum degree to maximum degree of the graph. In this cycle process, we delete the nodes from D whose degree is equivalent to the present value of k_s and add these nodes to S. Finally, the algorithm returns S which holds different k-values of all nodes.

3.3.2 Creating process white list based on K-Core

The K-Core decomposition method is used to analyze unweighted undirected graphs. And this is a major limitation of k-core decomposition method. However, most real networks are weighted in practice, and the weight property describes the model's important features. In order to overcome these limitations, Garas and Schweitzer proposed a weighted K-Core decomposition method [27]. The basic idea is to redefine the weighted degree of a node, they considered both the degree of a node and the weights of its links. And the new degree is the multiplication of two types.

In this work, we define a weighted degree of one node which is the sum over all its link weights. Then we use K-Core decomposition method to partition a network into sub-structures that are directly linked to centrality [28]. After finishing one decomposition, we calculate all nodes' weighted degree again to prepare the next decomposition. This procedure is repeated iteratively until all nodes are removed from the network.

A more detailed description of creating process white list based on K-Core is as follows:

1) Using the approach in Section 3 to build a weighted process network;
2) For process p_i in the process network, there are m links $\{e(p_i, p_1),\ e(p_i, p_2)\ldots,\ e(p_i, p_m)\}$ in total. And the link weights are $\{w_{i1}, w_{i2}, \ldots, w_{im}\}$ by using Eq. 3, we calculate the new degree of the weighted process network as follows,

TABLE 1 Statistical properties of process networks.

Number of servers	Number of nodes	Number of common processes	Number of edges	Average degree
3	674	359	13,516	40
5	926	513	16,593	36
10	1,048	576	17,842	34
20	1,351	738	20,275	30
30	1,587	852	22,354	28

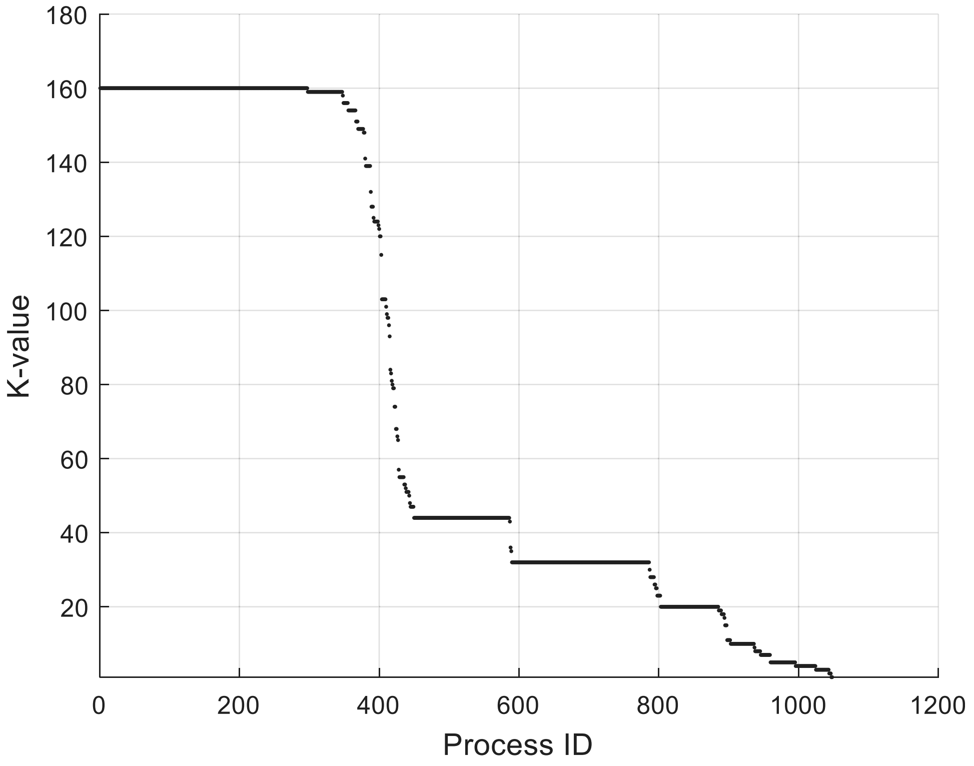


FIGURE 6
K-values of processes.

where, $k(i)_w$ is the weighted degree of node i, and j is node who has the same link with node I;

$$\boldsymbol{k}(\boldsymbol{i})_{\boldsymbol{w}} = \sum_{\boldsymbol{j}=\boldsymbol{1}}^{\boldsymbol{m}} \boldsymbol{w}_{\boldsymbol{ij}} \tag{4}$$

3) Using Eq. 4 and the K-Core decomposition method in Algorithm 1 to calculate all nodes' k-values. And then we can obtain a group of different k-values;
4) Choosing a threshold k-value then filter out the processes whose k-values are less than the threshold value;
5) After filtering, we can obtain the process white list.

3.3.3 Detecting anomalous process

We introduce the method to create a process white list including building process networks. However, the final goal of our work is to detect anomalous processes. Therefore, we detect anomalous processes by using the white list. Users can integrate the proposed method to some systems. For example, users can use directly detect anomalous process by comparing with the process white list. Engineering implementation is involved in detecting anomalous processes based on white list. And different user scenarios entail different software architectures. Since our work involves real-time detection of large volumes of IoT data, we utilize Kafka, Spark, etc. We transmit process data by Kafka and detect anomalous processes in Spark. The spark system can help us deal with huge process data in time.

4 Results

In this section, we test our proposed method on the real-world process data. All of our process data have been generated on IoT

TABLE 2 Process white list.

Rank	Process	Rank	Process
1	init	26	netns
2	sh	27	ksmd
3	top	28	kaluad
4	kthreadd	29	nfit
5	crond	30	jbd2
6	grep	31	rpciod
7	celery	32	postgres
8	migration	33	dbus-launch
9	events	34	redis-server
10	uwsgi	35	sshd
11	java	36	mysql
12	bash	37	python manage.py
13	kworker	38	cut
14	netstat	39	anacron
15	hald	40	kblockd
16	crypto	41	rpm
17	sleep	42	ata_sff
18	deferwq	43	md_misc
19	bioset	44	aio
20	ksoftirqd	45	ext4-dio-unwrit
21	rcu_bh	46	su
22	rcu_sched	47	scp
23	systemd	48	ipv6_addrconf
24	md	49	kintegrityd
25	writeback	50	chown

servers. And we gather the snapshots of these servers' processes at different time intervals.

4.1 The process network

We build the process network topology structure on different data sets. As shown in Table 1, we provide some detailed statistical properties of process networks. In this table, the number of processes is equivalent to the number of nodes. We obtain the dense network on processes data of three servers for which the number of edges is about 20 times higher than the number of nodes. This is supported by the conclusion that there is a close relationship between processes and servers. The number of edges is about 14 times higher than the number of nodes when the number of servers is equivalent to 30. While the number of servers increase, we can obtain the sparse process networks.

4.2 The process whitelist

We use the K-Core decomposition method to partition the process network into sub-structures on ten servers. And Figure 6 shows K-values of all processes. There are 1,048 processes which run on 10 servers. We can see that about 300 processes are in the group of 160-core which is the biggest k-value. These processes are very popular in servers. Moreover, 87% of these processes are server system processes. This is supported by the appearance that there are some system processes which are very popular. K-values of the rest of processes are less than the k-value of system processes. We can see that there are several obvious steps from this diagram.

There are some processes whose k-values are less than 10. And part of them is user-initiated processes such as "python test_one.py" which runs on only one server. Besides, there are many processes whose k-values are in the area around the average degree. Most of these processes are third party processes such as "firefox" and "nginx." These processes are popular in some special servers. Some processes such as "ps–ef," "ls–al" and "tailf" are system processes, but not always run on the servers. Therefore, the k-values of these processes are less than expected.

In our work, we gather process data on 30 servers and obtain 1,587 processes. The servers contain IoT database servers, web servers, spark clustering servers and so on. Finally, we obtain the process white list based on K-Core decomposition method. And Table 2 shows part of the white list.

Ranking refers to the popularity ranking of processes, where a higher process ranking indicates greater popularity among IoT devices and lower suspicion. By predefining thresholds, highly ranked processes are written into the white list, which can then be used for anomalous process detection. The result in Table 2 shows that the most popular processes are mainly system processes such as "init" and "crond."

4.3 Anomaly detection of real-world data

In Section 3, we discussed how to detect anomalous processes. The important part is to create a process white list and then we compare process data with the white list to detect. In this procedure, we use real-world process data that conclude process data in database servers and process data in Hadoop clustering servers. Surprisingly, we find out several anomalous processes in Hadoop clustering servers which are viciously used to produce bitcoin. The anomalous processes are "sustes," "sh mr.sh" and "sh i.sh" etc. We find that hackers make use of vulnerability to invade our servers and download malicious procedure such as "mr.sh" and "i.sh" from proxy servers in a foreign country. We didn't know this problem in our Hadoop servers until detecting anomalous processes based on the process white list. The proposed anomaly detection method helps us find the threat in clustering servers and avoid huge damage. Table 3 shows the malicious processes and vicious proxy IPs.

This real intrusion was mainly from Canada and the malicious procedure is used to produce bitcoin. The IPs which hackers used in Table 3 are the experimental servers.

TABLE 3 Information of malicious processes.

Malicious process	Proxy IPs	Country
sustes, sh mr.sh, sh i.sh, sh cr.sh	158.69.133.18, 192.99.142.226, 192.99.142.229	Canada

5 Discussion

We introduced a novel method to create process white list which used to detect anomalous processes. Different from previous approaches, the proposed method transforms the process data into networks based on the relationship between processes and IoT devices. Then we use K-Core decomposition method to partition the process network into sub-structures. It is generally accepted that the K-Cores with the biggest coreness values represent the most popular nodes of the whole network. Therefore, we filter processes by controlling threshold k-value. The popular processes such as system processes are put into the white list. The rest of the work is simple. Just compare process data with the white list and we can detect anomalous processes in devices. The proposed method is unsupervised so we don't need labeled data. From the section of experimental results, we can see that processes at the front of the white list are mainly system processes and they are very popular in devices. This is consistent with the observation we expected. Notably, the proposed method helps us find out the threat in clustering devices and avoid huge damage.

The method proposed in this paper is suitable for detecting less popular anomalous processes. It is difficult to detect widely prevalent malicious processes used for attacks when most devices in the IoT have already been compromised. For example, in DDOS attack scenarios, attackers may use compromised C&C servers to send commands to a large number of IoT devices. Therefore, processes generated by these commands are highly prevalent in the current IoT environment, and such processes may be included in our white list, thereby eliminating their suspicion and causing leaks. One way to address these issues is for security experts to intervene. After generating the process white list, security experts further analyze and remove suspicious processes to improve the credibility of the white list.

The current work does not consider the influence of parent-child process relationships on correlation. Therefore, in future research, we consider introducing the natural correlation between processes to detect anomalous processes. Compared to the undirected graph structure in the current work, the future plan is to use a directed graph to represent parent-child process relationships. The probability that the parent process or child process of a malicious process is malicious is relatively high. Therefore, after introducing the directed graph, it is possible to model and detect malicious process networks more realistically. To address the limitations of the current work, subsequent work will incorporate process features such as resource consumption and file transfer into the attributes of graph nodes. For IoT environments vulnerable to DDoS attacks, data theft, etc., the calculation function of the weight factor will be adjusted to achieve better results.

Data availability statement

The data analyzed in this study is subject to the following licenses/restrictions: The dataset cannot be publicly disclosed due to confidentiality reasons. Requests to access these datasets should be directed to YC, 15667083521@163.com.

Author contributions

YC: Conceptualization, Data curation, Investigation, Methodology, Software, Validation, Writing–original draft. TH: Writing–review and editing. FL: Data curation, Funding acquisition, Project administration, Resources, Writing–review and editing. TZ: Writing–review and editing. MY: Writing–review and editing. SY: Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work was supported by the Presidential Foundation of CAEP (Grant No. YZJJZQ2023026).

Acknowledgments

The authors would like to thank the editors and reviewers for their efforts in supporting the publication of this paper.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Gupta J, Sharma S *Handbook of research on information security and assurance*. Information Science Publishing (2008).

2. Forrest S, Hofmeyr SA, Somayaji A, Longstaff TA A sense of self for unix processes. In: Proc. 1996 IEEE Symp. Security and Privacy; 06-08 May 1996; Oakland, CA, USA (1996). p. 120–8.

3. Hofmeyr SA, Forrest S, Somayaji A Intrusion detection using sequences of system calls. *J Comput Security* (1998) 6:151–80. doi:10.3233/jcs-980109

4. Anderson D, Frivold A T *Valdes Next-generation intrusion detection expert system (NIDES): a summary* (1995).

5. Denning DE An intrusion-detection model. *IEEE Trans Softw Eng* (1987) SE-13(2): 222–32. doi:10.1109/TSE.1987.232894

6. Lunt T, Tamaru A, Gilham F, Jagannathan R, Neumann P, Javitz H, et al. *A real-time intrusion detection expert system (IDES)* (1992).

7. Teng HS, Chen K, Lu SC Security audit trail analysis using inductively generated predictive rules. In: Proceedings of the Sixth Conference on Artificial Intelligence Applications; 05-09 May 1990; Santa Barbara, CA, USA (1990). p. 24–9.

8. Levitt GFK Property-based testing of privileged programs. In: Proceedings of the 10th Annual Computer Security Applications Conference; 05-09 December 1994; Orlando, FL, USA (1994). p. 154–63.

9. Ko C, Levitt GFK Automated detection of vulnerabilities in privileged programs by execution monitoring. In: Proceedings of the 10th Annual Computer Security Applications Conference; 05-09 December 1994; Orlando, FL, USA (1994). p. 134–44.

10. Sezgin A, Boyaci A. AID4I: an intrusion detection framework for industrial Internet of Things using automated machine learning. *Comput Mater Continua* (2023) 76:2121–43. n. pag. doi:10.32604/cmc.2023.040287

11. Yang A, Zhuansun Y, Liu C, Li J, Zhang C Design of intrusion detection system for Internet of Things based on improved BP neural network. *IEEE Access* (2019) 7: 106043–52. doi:10.1109/ACCESS.2019.2929919

12. Zhang Y, Li P, Wang X Intrusion detection for IoT based on improved genetic algorithm and deep belief network. *IEEE Access* (2019) 7:31711–22. doi:10.1109/ACCESS.2019.2903723

13. Bhatt P, Morais A HADS: hybrid anomaly detection system for IoT environments. In: 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC); 20-21 December 2018; Hammamet, Tunisia (2018). p. 191–6. doi:10.1109/IINTEC.2018.8695303

14. Weinger B, Kim J, Sim A, Nakashima M, Moustafa N, Wu KJ Enhancing IoT anomaly detection performance for federated learning. In: 2020 16th International Conference on Mobility, Sensing and Networking (MSN); 17-19 December 2020; Tokyo, Japan (2020). p. 206–13. doi:10.1109/MSN50589.2020.00045

15. Alaiz-Moretón H, Aveleira-Mata J, Ondicol-Garcia J, Castañeda ÁL, García I, Benavides C Multiclass classification procedure for detecting attacks on MQTT-IoT protocol. *Complex* (2019) 2019:1–11. doi:10.1155/2019/6516253

16. Nagarajan S, Kayalvizhi S, Subhashini R, Anitha V Hybrid honey badger-world cup algorithm-based deep learning for malicious intrusion detection in industrial control systems. *Comput Ind Eng* (2023) 180:109166. doi:10.1016/j.cie.2023.109166

17. Al-Wesabi FN, Mengash HA, Marzouk R, Alruwais N, Allafi R, Rana A, et al. Pelican optimization algorithm with federated learning driven attack detection model in Internet of Things environment. *Future Gener Comput Syst* (2023) 148:118–27. doi:10.1016/j.future.2023.05.029

18. Intanagonwiwat C, Estrin D, Govindan R, Heidemann J Impact of network density on data aggregation in wireless sensor networks. In: Proc. 22nd Int'l Conf. Distributed Computing Systems (ICDCS '02); 02-05 July 2002; Vienna, Austria (2002).

19. Centola D The spread of behavior in an online social network experiment. *Science* (2010) 329:1194–7. doi:10.1126/science.1185231

20. Ugander J, Backstrom L, Marlow C, Kleinberg J Structural diversity in social contagion. *Proc Nat Acad Sci* (2012) 109(16):5962–6. doi:10.1073/pnas.1116502109

21. Page L, Brin S, Motwani R, Winograd T *The pagerank citation ranking: bringing order to the web, technical report, computer system laboratory*. Stanford Univ. (1998).

22. Kleinberg JM Authoritative sources in a hyperlinked environment. *J ACM* (1999) 46(5):604–32. doi:10.1145/324133.324140

23. Bond RM, Fariss CJ, Jones JJ, Kramer ADI, Marlow C, Settle JE, et al. A 61-million-person experiment in social influence and political mobilization. *Nature* (2012) 489(7415):295–8. doi:10.1038/nature11421

24. Muchnik L, Aral S, Taylor SJ. Social influence bias: a randomized experiment. *Science* (2013) 341(6146):647–51. doi:10.1126/science.1240466

25. Kitsak M, Gallos LK, Havlin S, Liljeros F, Muchnik L, Stanley H, et al. Identification of influential spreaders in complex networks. *Nat Phys* (2010) 6: 888–93. doi:10.1038/nphys1746

26. García-Algarra J, Pastor JM, Iriondo JM, Galeano J. Ranking of critical species to preserve the functionality of mutualistic networks using the k-core decomposition. *PeerJ* (2017) 5:e3321. doi:10.7717/peerj.3321

27. Garas A, Schweitzer F, Havlin S A k-shell decomposition method for weighted networks. *New J Phys* (2012) 14:083030. doi:10.1088/1367-2630/14/8/083030

28. Batagelj V, Zaver Nik M. Fast algorithms for determining (generalized) core groups in social networks. *Adv Data Anal Classification* (2011) 5:129–45. doi:10.1007/s11634-010-0079-y

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 26 April 2024
DOI 10.3389/fphy.2024.1383276

OPEN ACCESS

EDITED BY
Yuanyuan Huang,
Chengdu University of Information Technology, China

REVIEWED BY
Jiazhong Lu,
Chengdu University of Information Technology, China
Ding Zhe,
Chengdu University of Information Technology, China
Peilin He,
University of Pittsburgh, United States

*CORRESPONDENCE
Yanjun Liu,
✉ yanjunliu1223@163.com

RECEIVED 07 February 2024
ACCEPTED 26 March 2024
PUBLISHED 26 April 2024

CITATION
Liu Y, Li H and Hao M (2024), Personalized and privacy-preserving federated graph neural network.
Front. Phys. 12:1383276.
doi: 10.3389/fphy.2024.1383276

Personalized and privacy-preserving federated graph neural network

Yanjun Liu*, Hongwei Li and Meng Hao

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China

High-performance GNN obtains dependencies within a graph by capturing the mechanism of message passing and aggregation between neighboring nodes in the graph, and successfully updates node embeddings. However, in practical applications, the inherent model structure of the graph is highly susceptible to privacy attacks, and the heterogeneity of external data can lead to a decrease in model performance. Motivated by this challenge, this work proposes a novel framework called Personalized Federated Graph Neural Network for Privacy-Preserving (PFGNN). Specifically, firstly, this work introduces a graph similarity strategy. Based on the principle that clients with similar features exhibit stronger homophily, this work divides all participating clients into multiple clusters for collaborative training. Furthermore, within each group, this work employs an attention mechanism to design a federated aggregation weighting scheme. This scheme is used to construct a global model on the server, which helps mitigate the difficulty of model generalization resulting from data heterogeneity collected from different clients. Lastly, to ensure the privacy of model parameters during the training process and prevent malicious adversaries from stealing them, this work implements privacy-enhancing technology by introducing an optimized function-hiding multi-input function encryption scheme. This ensures the security of both model data and user privacy. Experiments on real datasets show that our scheme outperforms FedAvg in accuracy, and the communication overhead is linearly related to the number of clients. Through this framework, PFGNN can handle all kinds of non-Euclidean structured data, multiple clients collaborate to train high-quality and highly secure global models. This work provides the foundation for designing efficient and privacy-preserving personalized federated graph neural networks.

KEYWORDS

federated learning, graph neural network, privacy preserving, multi-input function encryption, artificial intelligence

1 Introduction

Cyber-physical-social systems (CPSSs) are a new paradigm extended by cyber-physical systems (CPSs), which have attracted widespread attention in the academic community Li et al. [1]. CPSS seamlessly connects networks, physical devices and social spaces through data. CPSS provides a more comprehensive intelligent system for federated graph neural networks, thus promoting the rapid development of artificial intelligence (AI). However, the heterogeneous of graph data in CPSS, coupled with the limitations of mobile devices and communication overhead during data transmission, makes CPSS not only vulnerable to privacy attacks, but also the heterogeneity of external data can lead to model performance degradation Wang et al. [2]. Therefore, the security and privacy of CPSS graph data have become a key research object of artificial intelligence.

The introduction of Graph Neural Networks (GNNs) has successfully applied the concept of deep learning to non-Euclidean space data sets Bronstein et al. [3]. With its powerful spatial graph structure, Graph Neural Network helps various industries deeply explore the value of their own data. GNN obtains dependencies in the graph by capturing the message passing mechanism and aggregation method between adjacent nodes in the graph structure, and converts it into standardized complete node embedding information and rich data information Fu et al. [4], Liu et al. [5].

Graph neural network training requires a large amount of graph data, which is distributed among different data owners. For instance, as described in Zhang et al. [6], the hospital wishes to train a graph neural network model for small cell carcinoma of lung (SCLC), each hospital has its own patient graph network that tracks common diagnoses of SCLC and other diseases. However, due to privacy issues and legal and regulatory considerations, these graph data cannot be shared with others, which leads to data isolation problems. This prompts us to ponder deeply: How to collaboratively train GNNs without leaking the local data of each institution? Federated learning is a distributed machine learning paradigm that not only protects the privacy of local data but is also the most effective way to deal with data isolation McMahan et al. [7]. Federated Learning (FL) with GNNs, where each client trains a GNN model locally and learns the local embedding information, and then the central server collects the gradients or model parameters of each client for federated aggregation Liu et al. [8].

However, an important challenge faced by federated graph neural networks is the privacy leakage issue Hu et al. [9]. Different from Euclidean spatial data such as pictures and texts, graph neural networks incorporate additional information because of their powerful graph structure, such as the information of nodes in the graph. It is this highly descriptive information that makes the GNN model extremely vulnerable to privacy attacks Zhang et al. [10] and even exploited by adversaries, resulting in leakage of attribute and member information He et al. [11] or affecting data set reconstruction Olatunji et al. [12]. Moreover, in a federated graph neural network, the adversary can reversely infer the client's local data through node embedding information, leading to the leakage or even abuse of sensitive data He et al. [13].

Also, a more important challenge is the heterogeneity of graph data Wang et al. [14]. In the collaborative modeling process, the graph data of different clients have varying degrees of heterogeneity in graph structure and node features, so these stored graph data are generally non independent and identically distributed (non-IID) Liu et al. [15]. This kind of graph data heterogeneity may cause the traditional federated averaging algorithm (FedAVG) to seriously diverge, resulting in global model performance degradation Zheng et al. [16]. Therefore, how to design a federated graph neural network framework suitable for non-IID graph data is particularly important.

Motivated by this challenge, this work proposes a novel framework named Personalized Federated Graph Neural Network for Privacy-Preserving (PFGNN), by which a high-quality and highly secure global model is trained collaboratively by multiple clients. The PFGNN framework is built on a set of state-of-the-art training paradigms, including graph similarity strategies, attention-based model aggregation schemes, and implementation of privacy-enhancing techniques to protect the uploading of sensitive model parameters. The processing of PFGNN can be divided into three stages to ensure high quality, high accuracy and high security of graph neural network training. Based on the above description of the PFGNN framework, our contributions are as follows.

- Enhanced the performance of federated learning in processing non-Euclidean spatial data. This work designs a graph similarity estimation strategy that takes stronger homophily among clients with similar characteristics as a clustering reference, while using random graphs as input of the GNN model to measure the similarity between each client and server, and dividing the clients into different clusters.
- Improved the accuracy of the global model. In order to accurately handle model parameters and replace the average mechanism, this work introduces the attention mechanism to design a federated aggregation weighting scheme to build a global model on the server. This global model can alleviate the difficulty of global model generalization caused by the heterogeneity of different client data.
- Realized personalized privacy protection. In order to hide the model parameters during the model training process and prevent malicious adversaries from stealing the model parameters, the privacy enhancement technology is implemented by introducing an optimized Function hiding multi-input function encryption scheme to ensure the privacy security of the model data and users.

2 Preliminaries

2.1 Federated learning

Federated learning is a type of distributed machine learning that can aggregate multiple data sources for collaborative training Lyu et al. [17]. During the model training process, data storage and model training are performed locally, and only model parameters or intermediate results are exchanged with the central server, the central server integrates different terminal parameters to implement a complete model training process. Federated learning effectively helps multiple organizations jointly conduct training and model modeling on the premise that data does not leave the domain and data security is met, thereby improving the effectiveness of artificial intelligence models and mitigating the costs and privacy risks in the traditional machine learning process.

2.2 Graph neural network

Graph neural network is a framework that uses deep learning to learn non-Euclidean spatial data. Its superior performance can help various industries deeply mine data value from complex graph structures. The GNN framework obtains the dependencies in the graph by capturing the message passing mechanism and aggregation method between adjacent nodes in the graph structure, and converts it into standardized and standard complete node embedding information and rich data information. Therefore, GNN has been rapidly developed and achieved good results in downstream tasks such as node classification, link prediction, graph and subgraph generation, etc.

In this work, PFGNN is modeled with the message passing neural network framework (MPNN) Gilmer et al. [18]. The forward passing process of MPNN includes two phases: Message Passing and Readout. In our framework, assume that there are K clients, and the data set of the kth client is $D^{(k)} = (G^{(k)}, Y^{(k)})$, where $G^{(k)} = (V^{(k)}, E^{(k)})$, $V^{(k)}$ is the node set of $G^{(k)}$, $E^{(k)}$is the edge set of $G^{(k)}$, $\{e_{ij}\}_{i,j\in V^{(k)}}$ is the edge feature set.

Phase 1: Message Passing. The function of this phase is to aggregate the node's neighborhood sampling information and update the embedding information of the node itself, as follows:

$$m_i^{(k,l+1)} = \mathrm{AGG}\left(\left\{M_t\left(h_i^{(k,l)}, h_j^{(k,l)}, e_{ij}\right)\right\} j \in N(i)\right) \tag{1}$$

$$h_i^{(k,l+1)} = U_t\left(h_i^{(k,t)}, m_i^{(k,l+1)}\right) \tag{2}$$

where $h_i^{(k,l)} = x_i^{(k,l)}$ is the node feature of the Lth layer of the Kth client. AGG is an aggregate function, and M_t is a message function, U_t is the update function, $N(i)$ represents a group of adjacent nodes of node i.

Phase 2: Readout. The function of this phase is to calculate the feature vector of the node based on the output layer for different downstream tasks.

$$y = Q\left(\left\{h_i^{(k,T)} \mid i \in G_p\right\}\right) \tag{3}$$

where Q is the readout function, which represents the features of the entire graph neural network, and p represents different downstream tasks.

2.3 Functional encryption

Function encryption is a lightweight public key encryption algorithm designed to protect data security. However, function encryption cannot be applied in real distributed scenarios, such as federated learning. Therefore, multi-input function encryption (MIFE) is an enhanced version of function encryption that emerged for application in distributed scenarios Abdalla et al. [19]. In MIFE, n participants are allowed to encrypt their own private data and generate ciphertext $CT = (c_1, c_2 \ldots c_n)$, generate the private key sk_f through the key generation algorithm and jointly perform function operations in the ciphertext state. That is to say, holding the ciphertext $CT = (c_1, c_2 \ldots c_n)$ and the private key sk_f can produce the calculation result $y = f(x_1, x_2 \ldots x_n)$ without revealing any information about the plaintext. This shows that sensitive data can be protected during the computing process while effectively preventing data leakage and privacy violations.

3 Proposed framework

3.1 High-level overview

In this subsection, this work gives a detailed introduction of PFGNN framework, that is, federated graph neural network for privacy-preserving. The goal of PFGNN is to achieve accurate, efficient, low communication cost, privacy-preserving personalized federated graph neural network. Participants of PFGNN include a trusted authority (TA) responsible for public key distribution and private key generation, a central server that coordinates model training and parameter aggregation, and a large number of clients that collaboratively train GNN models. Each client processes its own graph data by training a local graph neural network and uploads model parameters to the central server. Then the central server aggregates the received model parameters and iterates continuously until an excellent global model is trained. The framework diagram of PFGNN is shown in Figure 1.

The PFGNN framework aims to achieve accurate, efficient, low-communication-cost, and privacy-preserving personalized federated graph neural networks. The specific steps of the PFGNN are as follows.

1) Global Initialization and Security Parameter Setup: Initially, a Trusted Authority (TA) conducts global initialization by defining security parameters λ and generating the master public key (mpk) and master private key (msk). and then distributes them to clients.
2) Client Model Training Locally: Clients independently train their graph neural network (GNN) models locally to obtain initial node embeddings.
3) Client Grouping: Clients are organized into different clusters based on the criteria defined by Algorithm 1. This grouping facilitates cooperation and coordination among clients.
4) Attention Proportion Generation: Within each cluster, attention proportions are generated according to the definition provided in Formula 7. These proportions will be used to weight the contributions of different clients.
5) Client Upload of Attention Proportion: Each client uploads their computed attention proportion to a trusted third party (TA).
6) TA distributes cryptographic keys: TA generates the corresponding private key according to the proportion of attention uploaded by the client, and sends it to the server for decryption, which helps to ensure the data security of different clients.
7) Encryption of Model Parameters and Attention Proportion: The client's model parameters and attention proportion are encrypted using an optimized function-hiding MIFE algorithm.
8) Secure Aggregation: Upon receiving the encrypted model parameters and attention proportion, the server performs a secure aggregation operation to combine the client's attention proportion. Then, the server decrypts to obtain the aggregated model parameters, thus completing one round of model training.

3.2 Graph similarity estimation strategy

The heterogeneity of graph data is a major challenge in federated graph neural network optimization. Consider this scenario: Assuming there are three clients, the graph structure

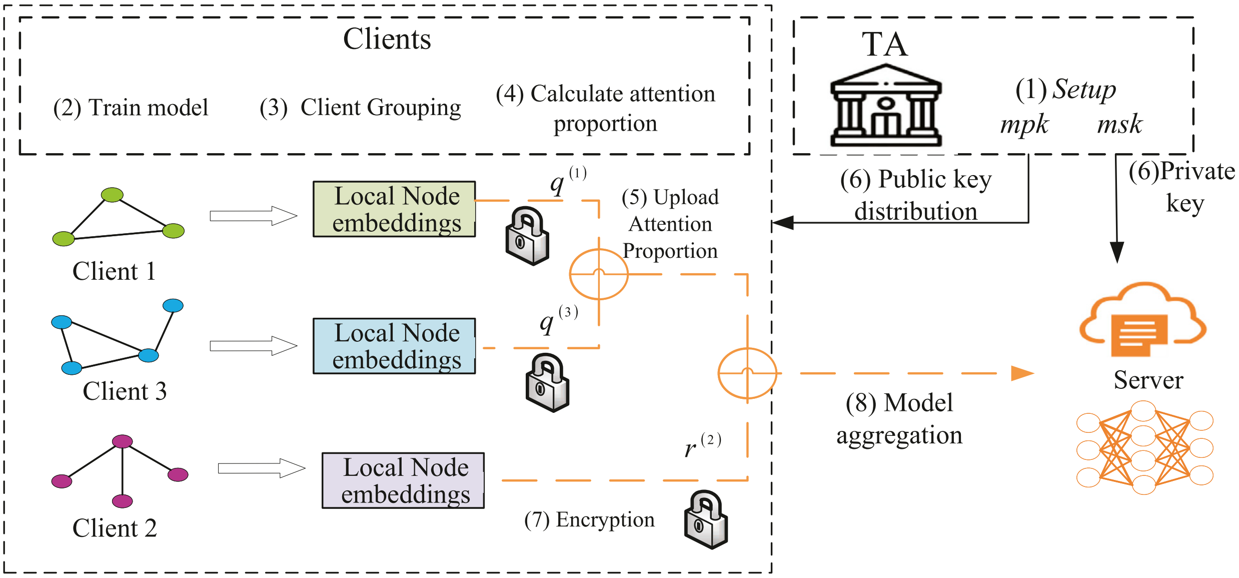


FIGURE 1
High-lever overview of Personalized Federated Graph Neural Network.

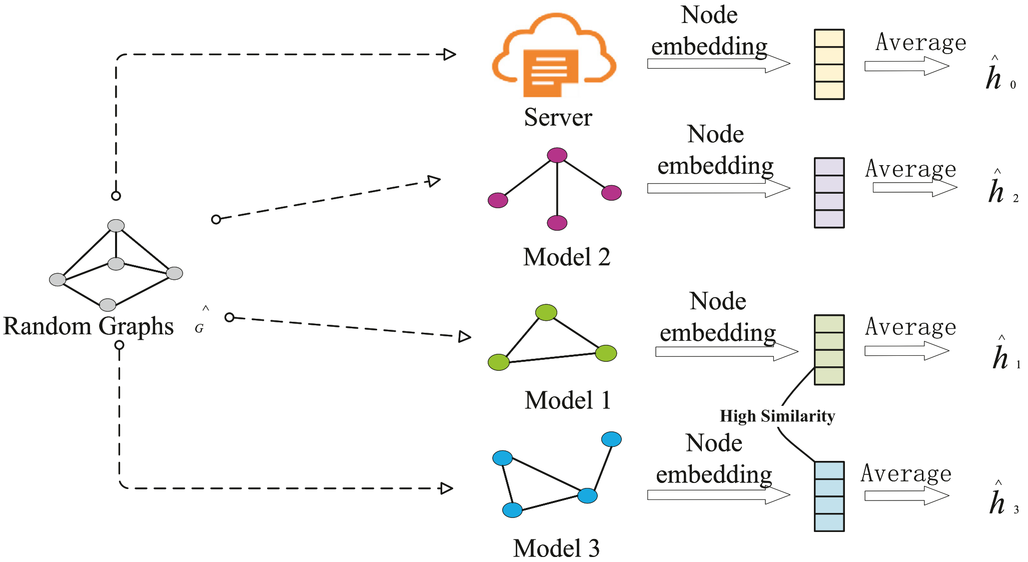


FIGURE 2
The overview of graph similarity estimation strategy.

between client 1 and client 3 is significantly different, and may even exhibit completely opposite properties. At the same time, there may be an overlap of nodes between Client 1 and Client 2. These nodes have similar characteristics and can form a cluster. It is known that clients with similar characteristics have stronger homophily McPherson et al. [20]. In order to capture the data heterogeneity between clients and train an accurate model suitable for most client data, this work can analyze and measure different clients based on the similarity of the client's graph structure, similar clients are grouped into a cluster. Regarding finding similarity in graph structure, usually, everyone will use model parameters or gradients to calculate similarity. In fact, because the dimensionality is too high, the similarity between parameters will continue to grow as the dimensionality of the model increases, so this method has serious flaws.

Inspired by Jeong et al. [21], and making it clear that our purpose is to measure the similarity between client graph structures, this work can provide the same input to all client graph structures (including the server model) and then analyze the similarity of their output results. In other words, consider all graph structure models as a black box function, input the same graph data, analyze and evaluate the output distance to represent the similarity between different graph structures. The specific algorithm is shown in Algorithm 2, where the random graph is initialized by the stochastic block model Baek et al. [22], and this randomness will not bias the model structure of any client. The detailed graph similarity strategy is shown in (Figure 2).

The server uses the similarity function to calculate the similarity between any client and server model. The expression is as follows:

$$S(i) = \frac{\hat{h}_0 \cdot \hat{h}_i}{\left\|\hat{h}_0\right\|\left\|\hat{h}_i\right\|} \tag{4}$$

the server classifies clients whose similarity is higher than a threshold (such as 0.5) into a cluster.

```
• Public Parameters: $N$ is the total number of clients, $C$ is the fraction of client, $U$ is a set of all clients, $B$ is the local mini-batch size, $E$ is the number of local epochs.
• Input: the GNN model $M^{(i)}$ on the client $G^{(i)}$, the GNN model $M$ on the server side
• Output: C clusters.
/* Runs on Server */
 Ensure Server executes:
    for each round $t = 1, 2, \ldots$ do
      $m \leftarrow \max(C \cdot N, 1)$
      $S_n \leftarrow \{u_i \mid u_i \in U\}_1^m$
      Initialize random graph $\hat{G}$
      With $\hat{G}$ on model $M$, compute $\hat{h}_0$
      Send $\hat{G}$ to client $i$
   end for
/* Runs on Client $k$ */
 Ensure Client $k$ executes:
    for  each local epoch $i$ from 1 to $E$ do   do
      for  for batch $b \in B$ do   do
        With $\hat{G}$ on model $M^{(k)}$, compute $\hat{h}_k$
      end for
      Send $\hat{h}_k$ to the server
    end for
/* Runs on Server */
 Ensure Server executes:
      Similarity $S(i)$ calculation with $\hat{h}_0$ and $\hat{h}_k$ based on Eq. 2
      Group into $C$ clusters with $S(i)$
```

Algorithm 1. Graph similarity calculation strategy.

3.3 A function encryption optimization algorithm with attentive aggregation

During the training process of the personalized federated graph neural network, the client trains the GNN model locally, generates local node embeddings, and directly uploads the model parameters or gradients to the server through federated aggregation, malicious adversaries can steal user data through model reconstruction attacks. At the same time, since each client's graph data has differences in graph structure and node features, this heterogeneity causes the traditional federated averaging algorithm to be seriously divergent, so this work needs to train an effective global model. To solve these problems, this work proposes a function encryption optimization algorithm based on attention aggregation, which not only considers the contribution of the client model to the global model, but also encrypts the aggregated model parameters and fusion weights.

3.3.1 A federated graph neural network algorithm with attentive aggregation

The most important part of the federated graph neural network is the server-side federated aggregation. In the traditional federated averaging algorithm, each client is given the same weight. This averaging processing method is rough and cannot well evaluate the advantages and disadvantages of the local model, which will have an adverse impact on the performance of the model. In order to train efficient global models and focus on the importance of client models, this work proposes a federated graph neural network algorithm with attentive aggregation, focusing on using FL with a central server to train GNN models.

The intuition behind federated graph neural network optimization is to find a global model that can improve the generalizability of distributed clients, the attentive aggregation algorithm proposed is a simple reward mechanism that can evaluate the contribution of client model parameters to the global model. Next, this work focuses on the aggregation mode of the client model. Specifically, this work takes the server model parameters as the query and the client model parameters as the key, calculate their similarity, and obtain the attention proportion of each client through the SoftMax function, finally, the model parameters are weighted and summed according to the attention proportion.

Given the lth layer parameter of the server global model as h^l, $h^{(k,l)}$ represents the model parameter of the lth layer of the kth client, and the similarity $p^{(k,l)}$ between h^l and $h^{(k,l)}$ is calculated by the Frobenius norm. Which is denoted as:

$$p^{(k,l)} = \mathrm{att}\left(h^{(k,l)}, h^l\right) = \tau\left\|h^{(k,l)} - h^l\right\|_2^2 \tag{5}$$

In order to further explore the relationship between client model parameters and global model parameters, this work uses hyperparameters τ to adjust the similarity online.

Then, since the similarity may have large differences and needs to be normalized, this work applies SoftMax function to calculate the attention proportion of each layer.

$$q^{(k,l)} = \mathrm{SoftMax}\left(p^{(k,l)}\right) = \frac{\exp\left(p^{(k,l)}\right)}{\sum_{k\in m}\exp\left(p^{(k,l)}\right)} \tag{6}$$

where $q^{(k,l)}$ represents the attention proportion of the lth layer model parameter of the kth client. After the server obtains the attention proportion of each client, it generates a global model based on the proportion of each client.

$$h_{t+1}^{l} = \sum_{k=1}^{m} q_t^{(k,l)} h^{(k,l)} \tag{7}$$

where $q_t^{(k,l)}$ is the proportion of the kth client at time t, and represents the global model parameters at time $t + 1$.

3.3.2 Optimized function encryption algorithm

In the process of federated aggregation, in order to defend against potential adversarial attacks, it is essential to encrypt the model parameters during transmission. This work has adopted the enhanced version of the MIFE algorithm, known as the Function-Hiding Multi-Input Function Encryption (FH-MIFE) scheme Abdalla et al. [19]. Specifically, in addition to safeguarding the uploaded model parameters, this work places particular emphasis on protecting the weights proportion by each client. The traditional single-layer MIFE falls short in adequately securing functions that may contain sensitive information. The FH-MIFE scheme employs a double-layer encryption process on both plaintext and keys, thereby enhancing the overall security of the model.

In some actual distributed scenarios, the decryption key contains a function f, and the function f itself also contains sensitive information, which allows the decryptor to obtain the weight value of each user in the decryption result. This will lead to the leakage of the user's plaintext information, so a single layer of function encryption is not enough to protect the function f with sensitive information. Therefore, we choose the function hiding multi-input function encryption scheme, which adds an extra layer of encryption on the ciphertext and key of the original MIFE. This double-layer encryption can not only ensure the security of the plaintext and model, but also protect the function f security, providing the model with high security and efficiency. In addition, in the process of model training, the client will fail due to network instability or connection problems, thus affecting the secure communication between clients and the server. However, the PFGNN scheme allows some clients to exit and rejoin at any time during the training phase, because the function-hiding multi-input function encryption scheme does not require the order in which clients join, nor does it require resetting keys for disconnected clients. Which $\mathrm{sum}(Y) > \frac{n}{2}$ indicates that TA collects the minimum number of participating clients and then generates the corresponding private key. In order to mitigate inference attacks, the sum of the number of clients participating in aggregation should be greater than or equal to $\frac{n}{2}$ ensure the normal progress of aggregation.

Furthermore, to more effectively apply the function hiding multiple-input function encryption in federated learning, this work has optimized the scheme. This work has introduced a key distribution phase in which the TA distributes unique public keys to each client based on their respective IDs. This allows each client to have their own unique public key for encryption, rather than using a uniform public key. This improvement enhances the security and flexibility of the scheme.

```
• **Public Parameters:** $N$ is the total number of clients, $B$
  is the local mini-batch size, $E$ is the number of local
  epochs, $t$ represents the number of layers of the neural
  network, $h^t(k)$ represents the model parameters of the
  client $k$.
/*Run on TA*/
  Ensure TA executes:
  Initialized with $mpk$, $msk$
  function $query - key(y_k, \varepsilon_{FH-MFH})$
      **if** sum$(Y) > \frac{n}{2}$ **then**
        return sky1y2...‖yk
      **end if**
/* Runs on Client $k$ */
  Ensure Client $k$ executes:
      **for** each local epoch $i$ from 1 to $E$ do **do**
        **for** for batch $b \in B$ do **do**
        obtain exclusive public key based on ID
        function collect-client $(h^t(k), b)$
        $c_k \leftarrow Enc_{pk_k}^{FH-MIFH}(h^t(k))$
      **end for**
      Send $c_k$ to the server
    **end for**
/* Runs on Server */
  Ensure Server executes:
  generate batch indices {1, 2, ..., $B$}
      **for** $b \in B$ **do**
      **for** $k \in K$ **do**
        $C_k \leftarrow$ collect-client $(h^t(k), b)$
        $sk_{y_1 \| y_2 \| \ldots \| y_k} \leftarrow query - key(y_k, \varepsilon_{FH-MIFH})$
        $h^t(k) \leftarrow Dec_{sk_{y_1 \| y_2 \| \ldots \| y_k}}^{FH-MIFH}(\{C_k\}_{k \epsilon K})$
      **end for**
**end for**
```

Algorithm 2. Optimized function encryption algorithm.

4 Security and privacy analysi

The goal of the PFGNN framework is to train a secure and efficient personalized federated graph neural network. This work analyzes the security and privacy of the PFGNN framework in detail.

4.1 Security analysis

Function Encryption is a cryptographic technique designed to protect data privacy, while allowing specific function computations to be performed on encrypted data without decrypting the data. This encryption method strikes a balance between privacy preserving and data processing, and is particularly suitable for scenarios such as federated learning. To prevent gradient inversion attacks in federated learning, PFGNN uses function-hiding multi-input function encryption to prevent collusion between malicious servers and TA, privately trade key parameters, and protect user encryption model gradient, so as to protect user privacy.

The cryptographic security of Function-Hiding MIFE is the top priority of the security of the PFGNN framework. Function-Hiding MIFE is a way to resist malicious adversaries from stealing model parameters and aggregate weights. In this work, to apply function hiding MIFE to federated learning more effectively, this work introduces a key distribution stage, in which a third-party server distributes an exclusive public key based on the ID of each client. This allows each client to obtain its own unique public key for encryption instead of using a unified public key. This improvement does not involve core algorithm processes, such as public key encryption and private key decryption. Therefore, this algorithm has no impact on the security of Function-Hiding MIFE. Function-Hiding MIFE is proven to be many-SEL-wFH-IND-secure, the proof process adopts a hybrid argument method, please refer to Abdalla et al. [19] for detailed understanding.

4.2 Privacy analysis

Function hiding MIFE provides computational privacy guarantees for secure aggregation in the PFGNN framework. Function hiding MIFE provides computational privacy guarantees for secure aggregation in the PFGNN framework. During the model training process, the Function-Hiding MIFE protects the model parameters and client weights from the client to the server, the decrypted result only contains the aggregated results of the model parameters, and the model parameters for any specific client are not available at all. In other words, function hiding MIFE double-encrypts the plaintext and key, effectively protecting the weight information of each client during decryption. This method can prevent malicious adversaries from using the weight of a single client to effectively speculate on the source of a certain attribute, and further prevents the adversary from identifying and leaking the client's identity through understanding the client's background knowledge.

5 Evaluation

In this section, this work evaluates the performance of the PFGNN scheme. This solution is a federated learning framework based on graph neural network, including n clients and a central server. This work mainly studies protocol performance evaluation in the semi-honest condition. In order to verify the effect of the proposed scheme, this work implements a federated learning prototype system based on graph similarity strategy, attentive aggregation scheme and function encryption, and conducts accuracy and efficiency experiments on it.

5.1 Experimental settings

In order to evaluate the performance of this scheme, PFGNN chose to perform the node classification task on three graph structure datasets, namely, Cora, Pubmed and Citeseer. The statistical summary of the datasets is shown in Table 1. And compare it with traditional graph neural network, thus proving the accuracy and versatility of PFGNN in processing non-Euclidean data.

TABLE 1 Dataset statistic.

Dataset	Node	Edge	Feature	Classes
Cora	2708	5429	1433	7
Pubmed	19717	44338	500	3
Citeseer	3327	4732	3703	6

During the process of model training, the client trains the graph neural network locally, taking GraphSAGE as an example, the propagation depth is $L \in \{1, 2, 3, 4, 5\}$, the number of iterations of the client's local model training is set as 10, the training batch size is 60. In this work, the maximum layer of the fully connected neural network is set as 2, and hyperbolic tangent (TanH) is adopted as the activation function of the hidden layer. Parameter drop rate is $d \in \{0.0, 0.5\}$, learning rate $l_r \in \{5e^{-4}, 5e^{-3}, 1e^{-3}, 1e^{-2}\}$. Since the task of the graph neural network in this work is node classification, the loss function adopts cross entropy. In order to prevent the model from overfitting, an additional regular term $L2$ is added $L2 \in \{5e^{-4}, 5e^{-3}, 1e^{-3}, 1e^{-2}, 0.0\}$. All experiments in this work are conducted on a single machine without the Internet to simulate communication in federated learning. The training set of the model is used to train the model, the verification set is used to adjust parameters, and the test set is used to measure the quality of model training. When adjusting parameters, the grid search method is selected to seek the highest accuracy under appropriate parameter settings.

This work implements PFGNN in python. Like the function encryption algorithm in MIFE Abdalla et al. [19], this work employs gmpy2 to implement the Paillier function encryption system.

5.2 Accuracy analysis

5.2.1 Comparison of model accuracy under different labels

To test the accuracy of the model, PFGNN chose to perform the node classification task on three graph-structured datasets, namely, Cora, Pubmed and Citeseer. In order to test the accuracy of models under different labels, this work divides the Cora data set into C_1, C_2 and C_3, according to the types of labels, where C_1 has three label categories with 1,296 nodes, C_2 has two label categories, and finally C_3 has two label categories. Similarly, this paper also divides the Pubmed and Citeseer data sets into three parts.

In order to study the accuracy of model aggregation under different labels, this paper assumes that there are three clients (A, B and C), the data of client A is composed of C_1, the data of client B is composed of C_2, and the data of client C is composed of C_3. In other words, the labels for the three clients are different. Next, comparative experiments were conducted between PFGNN, traditional Centralized machine learning (Centralized ML), and the classic FedAvg algorithm on three data sets. The local model training of the three algorithms is the graph neural network GraphSAGE. PFGNN is trained in the same way as

TABLE 2 Performance comparison on three datasets in terms of accuracy.

Dataset	Centralized ML	FedAvg	PFGNN
Cora	0.8345	0.8924	0.9213
Pubmed	0.8134	0.8812	0.9315
Citeseer	0.7237	0.7723	0.8145
Average	0.7905	0.8486	0.8891

TABLE 3 Performance comparison on different labels and different graphs.

Dataset	FedAvg	PFGNN	Improvement (%)
Cora	0.7546	0.8085	7.14
Pubmed	0.7435	0.7734	6.03
Citeseer	0.7137	0.7623	9.04
Average	0.7078	0.7814	7.38

FedAvg to verify the advantages of PFGNN with attentive aggregation.

To evaluate the performance of PFGNN in classification tasks, this paper examines its average accuracy on three different data sets. As shown in Table 2, PFGNN performs best in average accuracy on these datasets, significantly outperforming the other two models. In particular, compared with the classical FedAvg model, the average accuracy of PFGNN is improved by 5.4%. This result emphasizes the superiority of PFGNN in classification tasks and shows that after the introduction of the attentive aggregation mechanism, it has achieved satisfactory results in handling data aggregation and model updating in distributed learning scenarios.

5.2.2 Comparison of model accuracy under different labels and different graphs

The framework of message passing neural network in this paper is GraphSAGE, which mainly includes two steps: Sample and Aggregate. Sampling is to sample the number of neighbors through fixed-length sampling with replacement, thereby ensuring that each node after sampling has the same number of neighbors. GraphSAGE model training benefits from the transfer of adjacent information. Therefore, in order to study the accuracy of the model under different labels and graphs, this section divides the data set Cora according to the average edges, the samples with edges less than or equal to 3 in Cora are recorded as C_a, and the samples with edges greater than 3 are recorded as C_b. Then, similar to Section 5.2.1, the data of client A comes from the sample number C_{a1} of three label categories in sample C_a, and the data of client B consists of the sample number C_{a2} of two label categories in sample C_a, the data of client C comes from the C_{b3} samples of the two label categories in sample C_b. In the same way, the two data sets of Pubmed and Citeseer can be divided.

In this work, PFGNN model and FedAvg model are trained on three data sets respectively, and their accuracy is compared in Table 3. The results show that under different labels and different graphs, PFGNN model performs better than FedAvg, and the average accuracy rate increases from 5.48% to 7.38%. This shows that the PFGNN frame is suitable for handling different scenarios of label and graph distribution, which further emphasizes the superiority of the PFGNN model on non-IID data.

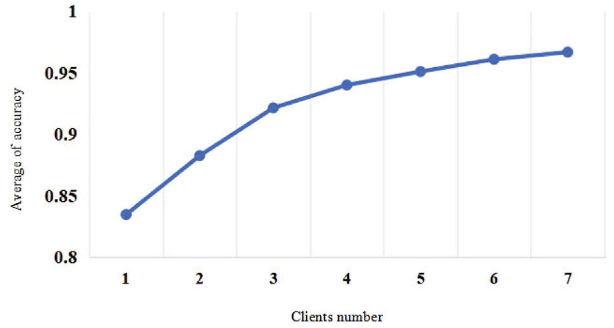


FIGURE 3
Average accuracy comparison of different clients' number with different labels.

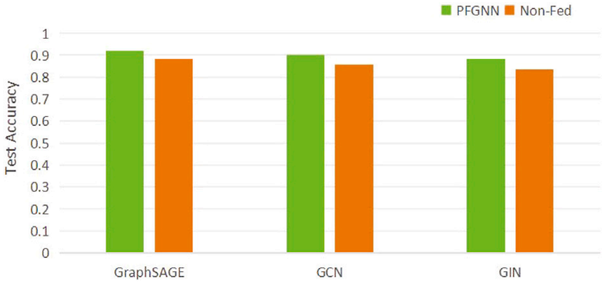


FIGURE 4
The generality of the PFGNN framework.

Centralized ML refers to uploading data to the server during the training process, performing training and inference on the server, and finally returning the results to the user. In this work, traditional machine learning can be regarded as the case where the PFGNN model has only one client. However, this method involves privacy and security risks in data uploading, and can also lead to excessive latency and waste the computing power of the terminal device. To study the impact of the number of clients on model performance, we increase the number of clients on the Cora dataset from 3 to 7, with each client having a different label. The Figure 3 shows that the accuracy of the PFGNN model increases with the number of clients and eventually stabilizes. This shows that as the number of clients increases, the types of tags each client has becomes smaller, but the performance of the overall model is still improved. This finding highlights the advantages of PFGNN in dealing with large-scale data sets, which can effectively utilize attentive aggregation and improve the performance and scalability of the model.

As shown in Figure 4, in order to test whether PFGNN is versatility, PFGNN is applied in different graph neural networks to test the Cora dataset, such as GCN Kipf and Welling [23] and GIN Hard et al. [24]. The green bar in the figure represents GraphSAGE with PFGNN settings, and the orange bar represents pure

TABLE 4 The time overhead of the function encryption scheme.

Clients	Enc (Hybrid)	Dec (Hybrid)	Enc (PFGNN)	Dec (PFGNN)
3	4.145	11.654	1.883	2.034
6	4.121	20.234	2.054	2.956
9	4.077	30.345	2.076	4.956

TABLE 5 Communication per iteration for *n* clients.

Phase	Communication	VFGNN	PFGNN)
Training Process	Secure SGD: clients ↔ CSP	n	n
	Secure SGD: clients ↔ clients	$(n^2 - n)/2$	0
	Secure SGD: TOTAL	$(n^2 + n)/2$	n

GraphSAGE. The accuracy of PFGNN after 100 rounds of communication in the figure is higher than 100 epochs iterative of GraphSAGE, which shows that PFGNN is effective for federated graph neural networks and can processes various non-Euclidean structured data and can be easily embedded into other models.

5.3 Computational overhead analysis

The PFGNN runs all encryption schemes under LAPTOP-OSDQQEMN equipped with lntel(R) Core (TM) i7-8565U CPU. In order to evaluate the computational overhead of function hiding MIFE in PFGNN, this work can set different numbers of clients and compare the encryption time of different schemes.

Table 4 clearly shows that as the number of clients increases, the time required for function encryption and decryption shows completely different trends. Specifically, as the number of clients increases, the encryption time on the client side remains almost constant, while on the server side, the decryption time grows linearly. However, the secure aggregation scheme of federated learning has a computational overhead of $O(N^2)$. In comparison, PFGNN only need $O(N)$. Compared with the scheme proposed in Yin et al. [25], the scheme adopted is not only more efficient, but also keeps the encryption and decryption time within an acceptable range even when the number of parameters reaches millions.

5.4 Communication overhead analysis

This work performs a detailed comparison between the PFGNN framework and VFGNN, especially in terms of communication overhead within one iteration. This solution is a federated learning framework based on graph neural network, including n clients and a central server. The detailed communication overhead is shown in Table 5. During model training, there is no direct communication between clients in the PFGNN scheme. This improvement reduces the total communication overhead from $(n^2 + n)/2$ to n. This means that the communication overhead is linearly related to the number of clients throughout the model training process.

6 Conclusion

This work proposes the Personalized and Privacy-Preserving Federated Graph Neural Network (PFGNN). The PFGNN framework is built on a set of state-of-the-art training paradigms, including graph similarity strategies, attention mechanism-based model aggregation schemes, and optimized function hiding encryption scheme to protect the upload of sensitive model parameters. Experiments on real datasets show that our scheme outperforms FedAvg in accuracy, and the communication overhead is linearly related to the number of clients. Through this framework, PFGNN can handle all kinds of non-Euclidean structured data, multiple clients collaborate to train high-quality and highly secure global models. This work provides the foundation for designing efficient and privacy-preserving personalized federated graph neural networks.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

YL: Conceptualization, Formal Analysis, Investigation, Methodology, Software, Writing–original draft, Writing–review and editing. HL: Writing–review and editing. MH: Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work is supported by the National Key RD Program of China under Grant 2022YFB3103500, the Key-Area Research and Development Program of Guangdong Province under Grant 2020B0101360001, the National Natural Science Foundation of China under Grants 62020106013 and 61972454, the Fundamental Research Funds for Chinese Central Universities under Grant ZYGX2020ZB027.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Li X, Wang P, Jin X, Jiang Q, Zhou W, Yao S. Reinforcement learning architecture for cyber–physical–social ai: state-of-the-art and perspectives. *Artif Intelligence Rev* (2023) 56:12655–88. doi:10.1007/s10462-023-10450-2

2. Wang X, Yang J, Wang Y, Miao Q, Wang F-Y, Zhao A, et al. Steps toward industry 5.0: building "6s" parallel industries with cyber-physical-social intelligence. *IEEE/CAA J Automatica Sinica* (2023) 10:1692–703. doi:10.1109/jas.2023.123753

3. Bronstein MM, Bruna J, LeCun Y, Szlam A, Vandergheynst P. Geometric deep learning: going beyond euclidean data. *IEEE Signal Process. Mag* (2017) 34:18–42. doi:10.1109/msp.2017.2693418

4. Fu X, Zhang B, Dong Y, Chen C, Li J. Federated graph machine learning: a survey of concepts, techniques, and applications. *ACM SIGKDD Explorations Newsl* (2022) 24:32–47. doi:10.1145/3575637.3575644

5. Liu Y, Qian X, Li H, Hao M, Guo S. Fast secure aggregation for privacy-preserving federated learning. In: GLOBECOM 2022-2022 IEEE Global Communications Conference (IEEE); December, 2022; Rio de Janeiro, Brazil (2022). p. 3017–22.

6. Zhang K, Yang C, Li X, Sun L, Yiu SM. Subgraph federated learning with missing neighbor generation. *Adv Neural Inf Process Syst* (2021) 34:6671–82.

7. McMahan B, Moore E, Ramage D, Hampson S, y Arcas BA. Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics (PMLR); April, 2017; Fort Lauderdale, FL, USA (2017). p. 1273–82.

8. Liu Y, Li H, Qian X, Hao M. Esa-fedgnn: efficient secure aggregation for federated graph neural networks. *Peer-to-peer Networking Appl* (2023) 16:1257–69. doi:10.1007/s12083-023-01472-2

9. Hu P, Lin Z, Pan W, Yang Q, Peng X, Ming Z. Privacy-preserving graph convolution network for federated item recommendation. *Artif Intelligence* (2023) 324:103996. doi:10.1016/j.artint.2023.103996

10. Zhang Z, Liu Q, Huang Z, Wang H, Lu C, Liu C, et al. Graphmi: extracting private graph data from graph neural networksarXiv preprint arXiv:2106.02820 (2021). https://arxiv.org/abs/2106.02820.

11. He X, Wen R, Wu Y, Backes M, Shen Y, Zhang Y. Node-level membership inference attacks against graph neural networks [J]arXiv preprint arXiv:2102.05429 (2021). https://arxiv.org/abs/2102.05429.

12. Olatunji IE, Nejdl W, Khosla M. Membership inference attack on graph neural networks. In: 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA); December, 2021; Atlanta, GA, USA (2021). p. 11–20.

13. He X, Jia J, Backes M, Gong NZ, Zhang Y. Stealing links from graph neural networks. In: 30th USENIX Security Symposium (USENIX Security 21); August, 2021; Virtual Event (2021). p. 2669–86.

14. Wang S, Zheng Y, Jia X. Secgnn: privacy-preserving graph neural network training and inference as a cloud service. *IEEE Trans Serv Comput* (2023) 16:2923–38. doi:10.1109/tsc.2023.3241615

15. Liu Y, Zheng Y, Zhang D, Chen H, Peng H, Pan S. Towards unsupervised deep graph structure learning. In: Proceedings of the ACM Web Conference 2022; April, 2022; Virtual Event, Lyon France (2022). p. 1392–403.

16. Zheng X, Wang Z, Chen C, Qian J, Yang Y. Decentralized graph neural network for privacy-preserving recommendation. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management; October, 2023; Birmingham United Kingdom (2023). p. 3494–504.

17. Lyu L, Yu H, Ma X, Chen C, Sun L, Zhao J, et al. Privacy and robustness in federated learning: attacks and defenses. *IEEE Trans Neural networks Learn Syst* (2022) 1–21. doi:10.1109/tnnls.2022.3216981

18. Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE. Neural message passing for quantum chemistry. In: International conference on machine learning (PMLR); August, 2017; Sydney, Australia (2017). p. 1263–72.

19. Abdalla M, Catalano D, Fiore D, Gay R, Ursu B. Multi-input functional encryption for inner products: function-hiding realizations and constructions without pairings. In: Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference; August 19–23, 2018; Santa Barbara, CA, USA (2018). p. 597–627.

20. McPherson M, Smith-Lovin L, Cook JM. Birds of a feather: homophily in social networks. *Annu Rev Sociol* (2001) 27:415–44. doi:10.1146/annurev.soc.27.1.415

21. Jeong W, Lee H, Park G, Hyung E, Baek J, Hwang SJ. Task-adaptive neural network search with meta-contrastive learning. *Adv Neural Inf Process Syst* (2021) 34:21310–24.

22. Baek J, Jeong W, Jin J, Yoon J, Hwang SJ. Personalized subgraph federated learning. In: International Conference on Machine Learning (PMLR); July, 2023; Honolulu, Hawaii, USA (2023). p. 1396–415.

23. Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks [J]arXiv preprint arXiv:1609.02907 (2016). https://arxiv.org/abs/1609.02907.

24. Hard A, Rao K, Mathews R, Ramaswamy S, Beaufays F, Augenstein S, et al. Federated learning for mobile keyboard prediction [J]arXiv preprint arXiv:1811.03604 (2018). https://arxiv.org/abs/1811.03604.

25. Yin L, Feng J, Xun H, Sun Z, Cheng X. A privacy-preserving federated learning for multiparty data sharing in social iots. *IEEE Trans Netw Sci Eng* (2021) 8:2706–18. doi:10.1109/tnse.2021.3074185

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 01 May 2024
DOI 10.3389/fphy.2024.1402326

OPEN ACCESS

EDITED BY
Xin Lu,
De Montfort University, United Kingdom

REVIEWED BY
Saima Qureshi,
University of Novi Sad, Serbia
Muhammad Awais,
University of East Anglia, United Kingdom
Mitar Simic,
University of Novi Sad, Serbia

*CORRESPONDENCE
Xian Qi Lin,
xqlin@uestc.edu.cn

RECEIVED 17 March 2024
ACCEPTED 15 April 2024
PUBLISHED 01 May 2024

CITATION
Khan MT, Lin XQ, Zhe C and Saboor A (2024), Design, analysis and validation of a microstrip patch antenna with enhanced coupling for leaf moisture sensing: an IoT approach.
Front. Phys. 12:1402326.
doi: 10.3389/fphy.2024.1402326

Design, analysis and validation of a microstrip patch antenna with enhanced coupling for leaf moisture sensing: an IoT approach

Muhammad Talha Khan[1], Xian Qi Lin[1]*, Chen Zhe[1] and Abdus Saboor[2]

[1]School of Electronic Science and Engineering, University of Electronic Science and Technology (UESTC), Chengdu, China, [2]School of Computer Science and Engineering, University of Electronic Science and Technology (UESTC), Chengdu, China

An innovative IoT-based system utilizing a modified slotted microstrip patch antenna with enhanced coupling is presented for precise measurement of leaf moisture content. The antenna employs a rectangular slot above the feed point with an advanced coupling technique to enhance sensitivity. The antenna, fabricated on a 0.8 mm F4B substrate, is designed to resonate within the 2.40 to 3.0 GHz range under unloaded conditions. A parametric analysis focusing on leaf permittivity ranging from 20 to 30 is conducted to determine the antennas' sensitivity. Experimental measurements of the reflection coefficient (S11) with respect to resonant frequency shift are performed with leaf samples as the samples under test (SUT). Experimental results reveal that the proposed patch antenna's sensitivity is significantly enhanced, ranging from 0.57 to 1.67 times greater than that of traditional patch antennas for the five leaf samples tested. The antenna exhibits a sensitivity of 0.06 GHz and 0.02 GHz for the modified and enhanced coupling designs, respectively. The mean relative error between predicted and measured moisture content values is low at 0.038. The findings highlight the antenna's increased sensitivity in detecting leaf moisture content and illustrate the potential of the proposed IoT-based system for real-time agricultural monitoring, marking an advancement in precision farming practices. The study validates the microstrip patch antenna's capability as a moisture sensor through detailed sensitivity analysis, frequency shift measurements, and regression modeling.

KEYWORDS

microstrip, patch, antenna, sensor, IoT, regression analysis

1 Introduction

In recent years, research has demonstrated that antennas can also serve effectively as sensors in various industrial applications. This emerging field takes advantage of the fact that antennas interact with their surrounding environment, with factors like nearby objects, humidity, moisture [1,2], temperature, etc. impacting the antenna's impedance and resonance characteristics. By monitoring the variation in antenna parameters, valuable sensor data can be obtained. Implementing antennas as sensors offers notable advantages in industry. Antenna sensors are typically inexpensive compared to traditional sensor modalities, given the mass production of antennas for communication purposes. Additionally, it enables wireless sensing in contexts where wired sensors may be impractical or impossible to implement. The dual use of antennas for both communication and sensing furthermore maximizes efficiency and value. Hence, electromagnetic (EM) resonant sensors demonstrates a type of sensors capable of providing distributed sensing. An example of an electromagnetic (EM) resonant sensor that enables distributed sensing is the microwave patch antenna (MPA) sensor [3,4].

In recent times, there has been immense interest in strengthening the sensing capabilities of microstrip patch antennas. Various ideas and technologies have been employed to design antennas with sensing capabilities. In the design process of a microstrip patch antenna, a broadband electromagnetic (EM) signal is delivered to the radiation patch via a microstrip transmission line. The radiation patch transmits the portion of the signal that resonates along the antenna's resonant frequency, while the remaining signal is reflected back. Consequently, the frequency spectrum of the reflected signal demonstrates considerable losses close to the antenna's resonant frequencies. Therefore, precise determination of a material's permittivity has emerged as a critical factor in microwave antenna design, owing to its straightforwardness and non-destructive nature of application [5].

The need of accurate measurement of permittivity of a material has become an important parameter in the design of microwave antennas due to their simplicity of design and non-destructive applications [5]. Planar resonators designs such as split ring resonators have become increasingly prevalent among various techniques used for determining permittivity. Such strategies are preferred due to their cost-effectiveness, low size, simplistic shape, and relatively simple fabrication [6–8]. In such experiments, the sample under test (SUT) is taken as part of the resonator and the permittivity is determined by analyzing the shift in resonant frequency. Such methods involve addressing the sample under test (SUT) as a component of the resonator and the permittivity is calculated by studying the variation in resonant frequency. Microstrip patch antennas, resembling resonators, have been researched as sensors for measuring the permittivity of liquid or solid substances [9]. In literature, a method is suggested for determining the resonant frequency of a patch antenna clad with a dielectric substrate. The technique relies on the efficient dielectric constant of complete design that is proposed to be determined by use of the variational approach [10].

Subsequently, monitoring the moisture content (MC) is crucial in precision farming and the food industry as an indicator of the quality of [11], assessment of numerous quality parameters of animal food [12], granular materials in containers [13], wheat [14], grain [15], rice [16], corn kernel [17] and various crops [18–21]. Accurate and timely measurement of leaf moisture is essential for optimizing water usage, ensuring plant health, and improving crop yield [22]. While utilizing a coaxial probe is a method that can potentially damage the sample, employing antennas is a preferred non-contact approach for MC monitoring. Moreover, the need for leaf moisture sensing in precision agriculture, environmental monitoring, and sustainable resource management is underscored by its critical role in optimizing water usage, ensuring plant health, and improving crop yield [23]. Leaf wetness sensors, as described in the sources, offer a technological solution to accurately and timely gather data on leaf moisture, which is essential for several reasons. In agricultural production, precise and real-time monitoring of plant physiological data is crucial. One such piece of information is leaf wetness, which is correlated with plant capacitance. Nevertheless, plants are susceptible to damage by the methods used currently to measure leaf capacitance, which would compromise the monitoring's accuracy. As a result, it is advisable to avoid frequency bands where the dielectric constant of water experiences abrupt changes with temperature.

Therefore, the proposed study introduces a study of microstrip patch sensor antenna designed for sensitivity enhancement in permittivity measurement for moisture content. The antenna employs a few rectangular slots that are loaded above the microstrip feed line that has an enhanced coupling. The designed antenna is simulated and fabricated on a F4B substrate with 0.8 mm thickness. The variation in resonant frequency with respect to reflection coefficient (S_{11}) is analyzed to verify the sensing capability of the proposed antenna. The proposed antenna design with enhanced coupling provides higher sensitivity compared to traditional patch antenna designs. Hence, the antenna achieves resonances at lower frequencies while maintaining a compact size. Moreover, the proposed antenna design achieves a sensitivity enhancement ranging from 0.57 to 1.67 times higher than traditional patch antennas for permittivity values between 20 and 30 while maintaining a lower mean relative error (MRE) between the actual and predicted values of moisture content. Furthermore, the proposed integration of the antenna sensor into an IoT-based system for real-time monitoring of leaf moisture content in agricultural applications marks an advancement in precision farming practices. Full-wave simulations using ANSYS EM Suite 2022 are performed and the sensors' performance is verified through mathematical calculations and experimental processes.

2 Design idea

Research indicates that the resonant frequency of a MPA, intended for integrated sensing capabilities, is impacted by the dielectric properties of the substrate material and the dimensions of its radiating component. These radiation characteristics of the microstrip patch antenna are inherently frequency-dependent [24]. The design of such an antenna encompasses three crucial elements: a radiating surface, a microstrip feed line, and a dielectric substrate. The traditional rectangular microstrip patch antenna's structure is depicted in Figure 1A.

The standard microstrip patch antenna (CPA) illustrated in Figure 1A incorporates a patch having a rectangular shape that is connected via 50 Ω microstrip feed line. It is manufactured on an F4B substrate that has a thickness of 0.8 mm and dielectric constant (ε_r) of 2.65. The designed dimensions of the patch are such that h_1 = 19.6 mm and w_1 = 15.5 mm, respectively. The width (w_2) and calculated length (L_1) of the feed line are determined as w_2 = 3.2 mm and L_1 = 8 mm, respectively. The dimensions of the substrate are 28.1 mm in length (*L*) and 39.08 mm in width (*W*).

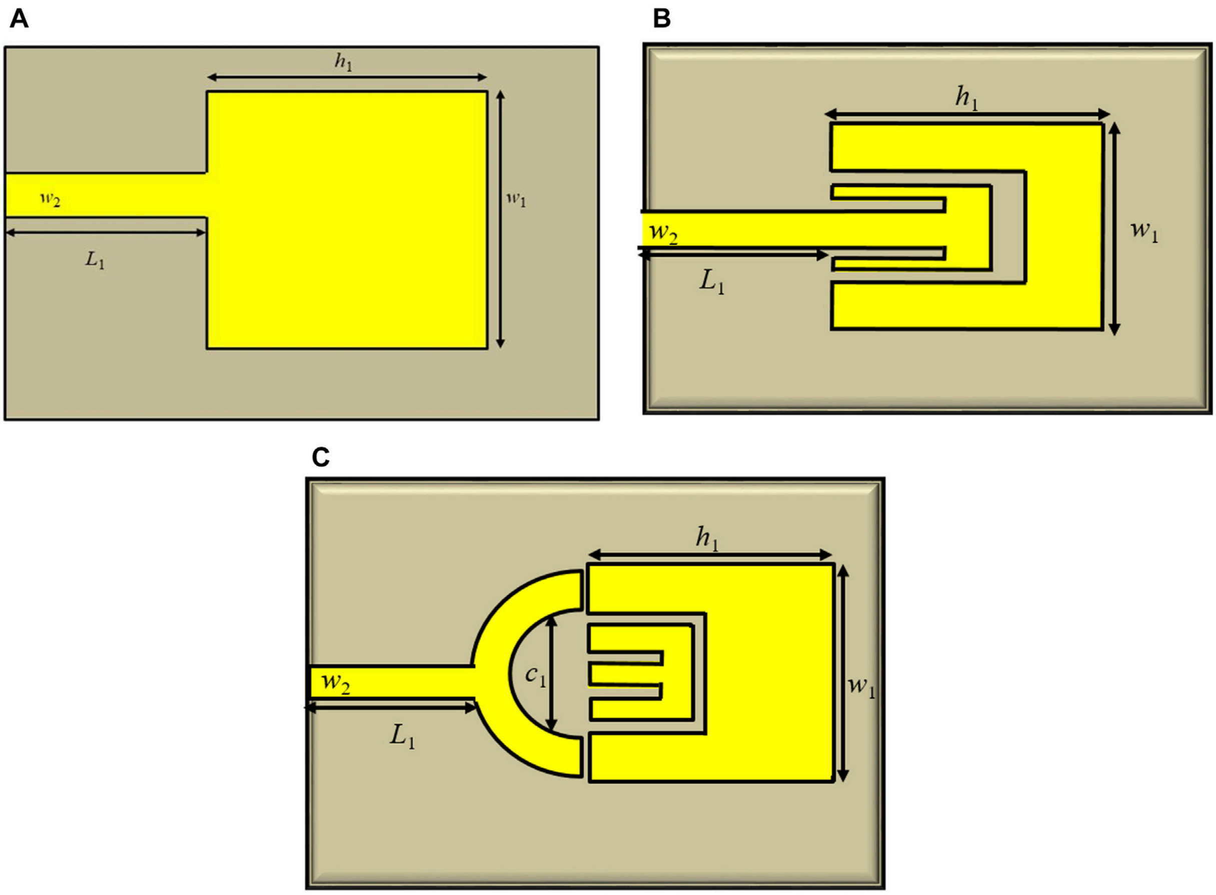


FIGURE 1
Microstrip patch sensor antenna structures: **(A)** standard rectangular patch antenna; **(B)** slotted patch antenna; **(C)** enhanced coupling modified slotted patch antenna.

The dimensions, width (W), length (L), change in length (ΔL) and relative permittivity ($\mathcal{E}_r$) of rectangular microstrip patch are calculated using Eqs 1–4 [25].

$$W = \left(\frac{c}{2f_r}\right)\sqrt{\frac{2}{\mathcal{E}_r + 1}} \tag{1}$$

$$\mathcal{E}_{reff} = \frac{\mathcal{E}_r + 1}{2} + \frac{\mathcal{E}_r + 1}{2}\left(1 + \frac{12h}{W}\right)^{-0.5} \tag{2}$$

$$\Delta L = 0.412{*}h\frac{\left(\mathcal{E}_{reff} + \frac{3}{10}\right).\left(\frac{W}{h} + \frac{2.64}{10}\right)}{\left(\mathcal{E}_{reff} - \frac{2.58}{10}\right).\left(\frac{W}{h} + \frac{8}{10}\right)} \tag{3}$$

$$L = c\left(\frac{1}{2f_r\sqrt{\mathcal{E}_{reff}}}\right) - 2\Delta L \tag{4}$$

where, f_r is desired resonant frequency, h is height of substrate and c is speed of light in free space. In order to achieve size reduction at lower frequencies compared to a design reported in [26], slots have been incorporated into the antenna structure.

It is known from literature that reducing the gap size can significantly increase the gap capacitance [27]. With strong coupling, maximum electric fields are present, making it more sensitive to overlay permittivity variations. Therefore, a patch antenna is often referred to as a resonator. The proposed design utilizes the concept of enhanced coupling at the periphery of a resonator, as shown in Figure 2. By incorporating an enhanced coupling idea, insertion loss is reduced and gap capacitance is considerably increased. Such a method is more frequently utilized for filters and is used in antenna for same purpose. In addition to

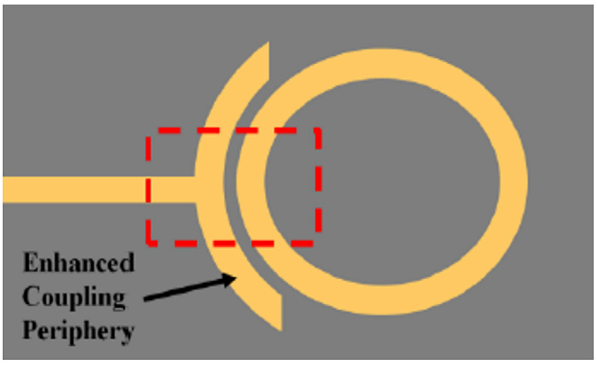


FIGURE 2
Enhanced coupling periphery of a ring resonator.

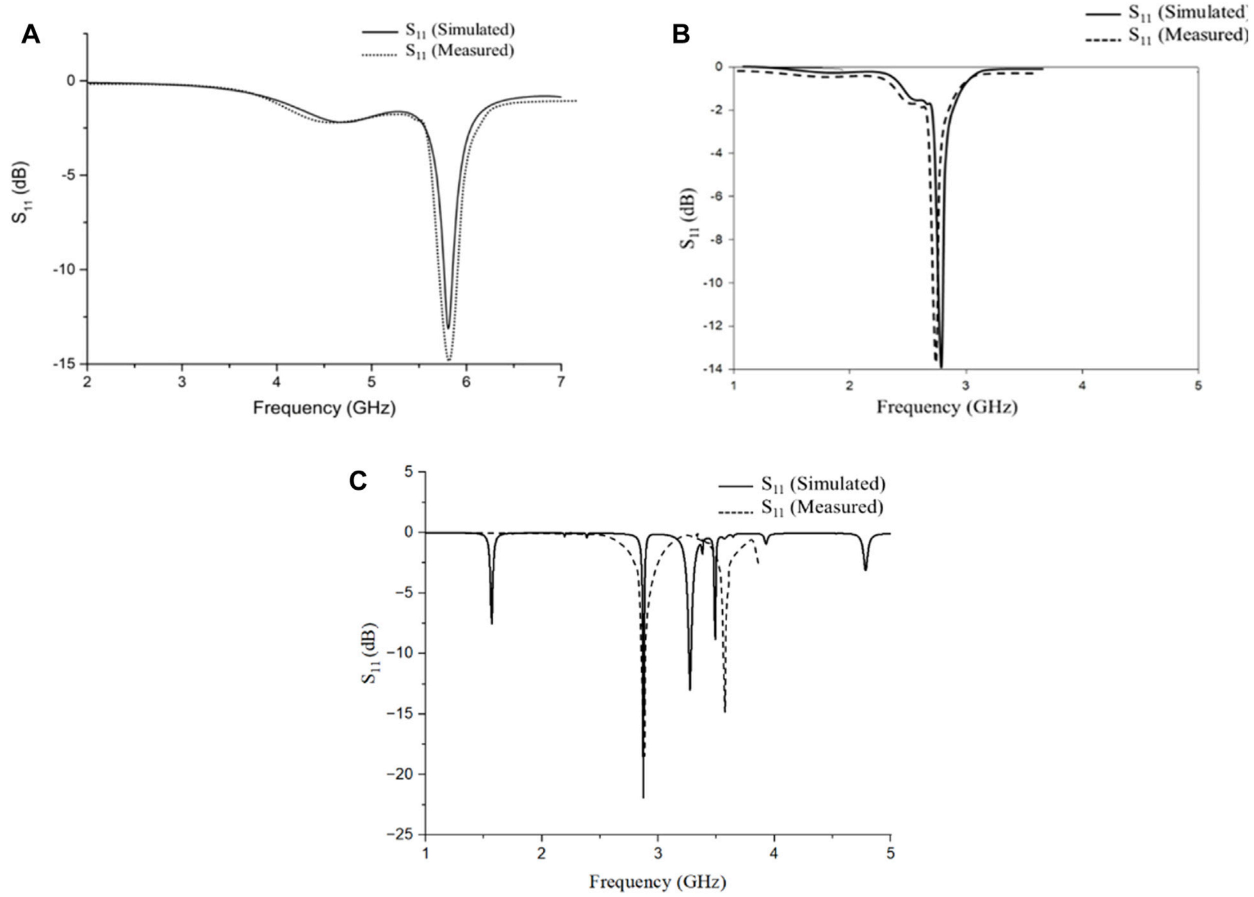


FIGURE 3
Reflection coefficient values (S_{11}) of the designed antennas **(A)** standard microstrip patch antenna (CPA); **(B)** modified slotted microstrip patch antenna; **(C)** enhanced coupling slotted microstrip patch antenna (ECMPA).

coupling capacitance, the capacitance of the ring structure can be further enhanced through mutual coupling. The technique used to increase mutual coupling by adding additional rings is known as concentric circular ring [28]. This technique has shown greater sensitivity in the dielectric characterization of liquid and powder materials.

Accordingly, a thin rectangular slot is etched radiating above the microstrip feed line, as depicted in Figure 1B. The slotted modified patch antenna (MPA) features a rectangular patch with dimensions of "h_1 = 28.3 mm" in length and "w_1 = 25.4 mm" in width, which are connected with the microstrip line. In this configuration, the microstrip line is designed using length, L_1 = 12 mm and width, w_2 = 3.2 mm. Consequently, the proposed enhanced coupling based slotted-patch antenna (ECMPA) sensor is designed on the basis of MPA with an addition of a semi-circle with a radius of 6 mm denoted as C_1 that provides the enhanced coupling to the antenna as shown in Figure 1C.

3 Results and discussion

The designed antennas are simulated using a microwave simulation software, ANSYS EM Suite 2022. Analysis of the S_{11} parameter for the standard inset-fed rectangular microstrip patch antenna (CPA) revealed resonant frequency at 5.807 GHz that has a bandwidth of 400 MHz, as shown in Figure 3A. Subsequently, the fabricated antenna sensor exhibited resonance at 5.82 GHz. In addition, the first slotted rectangular microstrip patch antenna sensor produced a resonant frequency at 2.85 GHz based on simulation results, in addition to a bandwidth of 300 MHz. On the other hand, the fabricated design depicted a similar resonant frequency significantly lower at 2.82 GHz, as indicated in Figure 3B. Finally, the enhanced coupling based microstrip patch antenna (ECMPA) has been simulated utilizing the same parameters and it accomplished dual frequency functioning. The initial resonant frequency, denoted as f_1, is at 2.87 GHz, while the other resonant frequency, f_2, is at 3.27 GHz, as depicted in Figure 3C. However, measurements revealed the first resonant frequency to be slightly lower at f_1 = 2.86 GHz, and the second higher at f_2 = 3.4 GHz. The measurement of resonant frequency of the all three designed antennas, i.e., CPA, MPA and ECMPA is determined by measuring its S_{11} parameter using a vector network analyzer (VNA), which represents the power of the microwave signal reflected by the antenna sensor as a function of frequency. The VNA was calibrated to the end of the coaxial cable so the effect of the microstrip feed is considered to be a part of the antenna sensor.

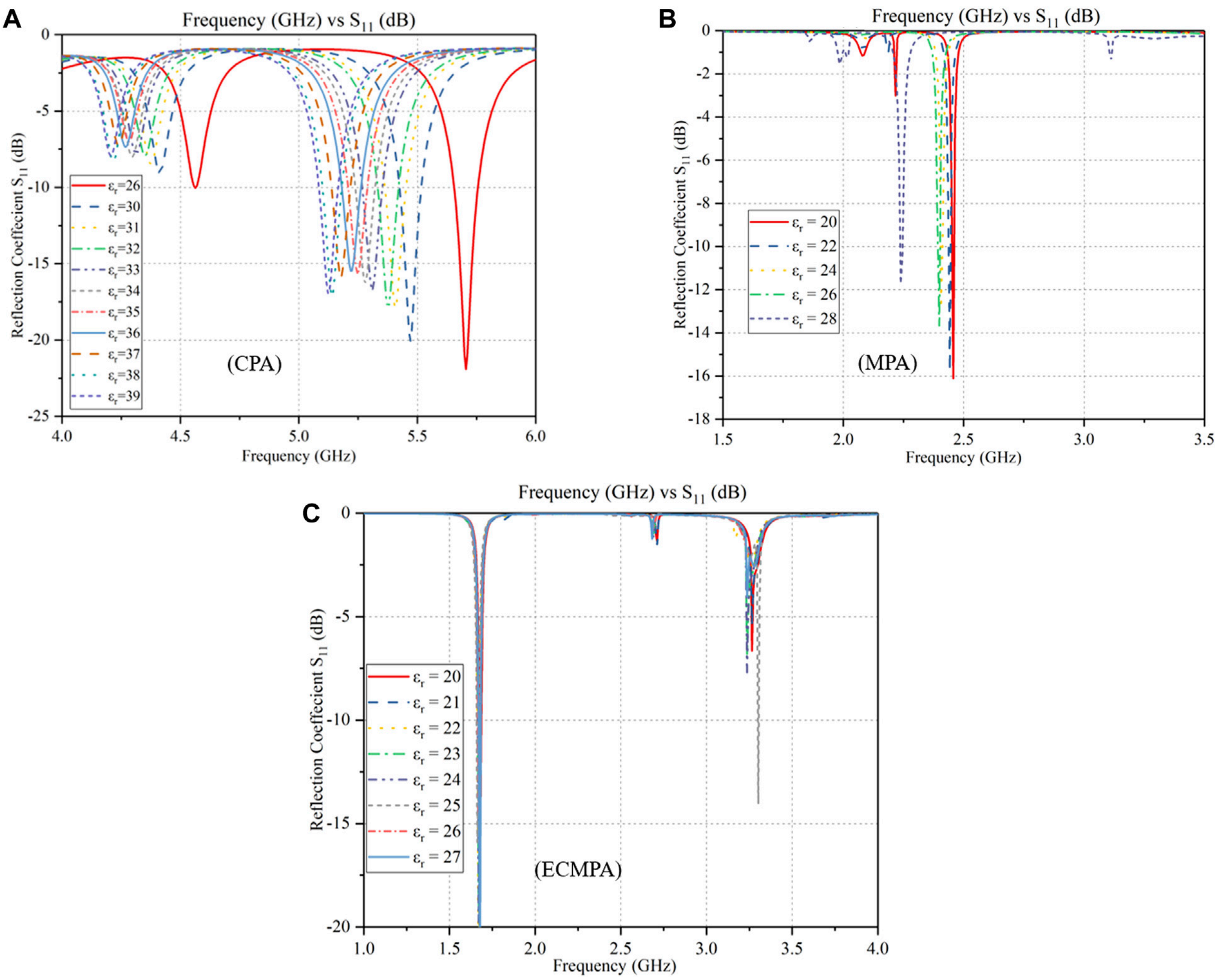


FIGURE 4
S_{11} (dB) vs. frequency (GHz) for varying relative permittivity of the SUT: **(A)** standard rectangular patch antenna; **(B)** slotted patch antenna; **(C)** enhanced coupling modified slotted patch antenna.

4 Sensitivity of the enhanced coupled microstrip patch antenna

The effectiveness of the designed enhanced coupled microstrip patch antenna (ECMPA) for sensing purposes is validated through measurements in variation in the resonant frequency (f) in gigahertz (GHz) compared to S_{11} (dB). The electric field intensity is highest in the area influenced by the loaded slot and the coupling, forming a capacitor. This sensor detects variations in the resonant frequency, reflecting the dielectric properties of leaves. By analyzing experimental results, a connection between the resonant characteristics and leaf moisture content can be established. Consequently, during periods of moisture stress in plants, the sensor can detect significant changes in leaf moisture. For precise sensing, the leaf being analyzed should be positioned close to the area with the strongest electric field. The relationship between frequency and capacitance is described by Eq. 5:

$$f = \frac{1}{2\pi\sqrt{L_s C_s}} \tag{5}$$

where, f is frequency, L_s and C_s are the inductance and capacitance of the designed sensor. In accordance with Eq. 5, the more concentrated the moisture content of the leaf, the larger the permittivity and capacitance. Consequently, the shift in resonant frequency can be exploited to detect leaf moisture. A leaf model also called sample under test (SUT) with a thickness of 0.5 mm and loss tangent of 0.2 is considered in this experiment. The SUT is positioned over the patch and its relative permittivity (ε_r) is diversified in the range of 20–30 with an increment of 1. Such a range is selectd because the relative permittivity (ε_r) of plant-based materials, such as fruits and leaves, typically ranges from around 20 to 30.2 This range encompasses the dielectric properties commonly observed in plant tissues {Doidy, 2019 #128}.

Figure 4 depicts the S_{11} characteristics of the traditional patch antenna and the proposed modified slotted patch antenna. For the traditional patch antenna, it is observed that the resonant frequency is achieved at 5.12 GHz as the test sample dimension having ε_{r1} = 30. Moreover, the frequency response of 5.70 GHz is achieved with value of ε_{r1} = 20.

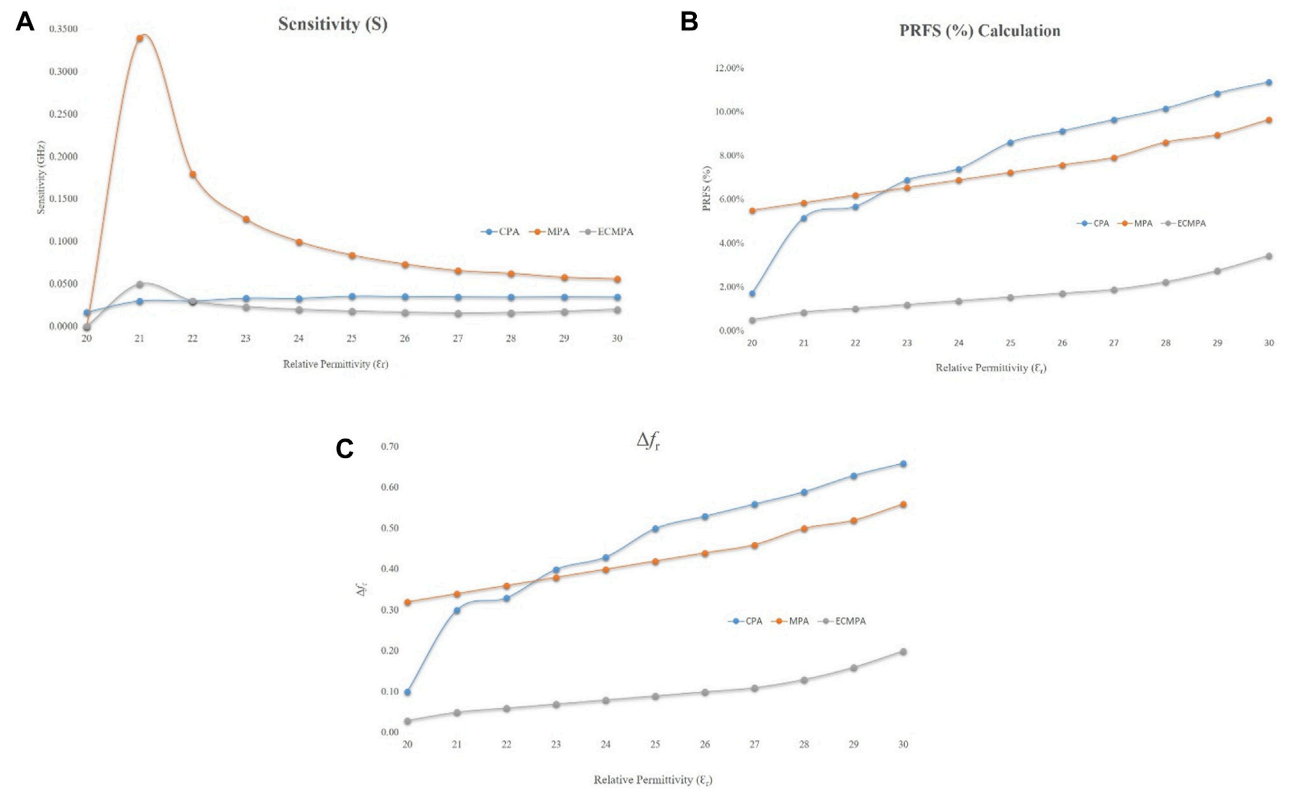


FIGURE 5
Comparative plots for sensitivity performance of traditional microstrip patch antenna and proposed slotted patch antennas: **(A)** sensitivity (S); **(B)** % relative frequency shift (PRFS); **(C)** variable resonant frequency (Δf_r).

Whereas, for the proposed slot-loaded modified antenna (MPA), SUT having a permittivity value of ε_{r1} = 30, the antenna demonstrates its lowest frequency response at 2.24 GHz. Conversely, a higher frequency response of 2.48 GHz is achieved when the test sample's permittivity is set to ε_{r1} = 20. Furthermore, in the case of modified slotted patch antenna with enhanced coupling (ECMPA), it is observed that increasing the relative permittivity (ε_r) from 20 to 30 that led to a modification in the initial resonant frequency (f_{r1}) of S_{11}, moving it from 2.87 GHz to 2.79 GHz. Furthermore, the second resonant frequency (f_{r2}) of the designed slotted patch antenna shifted from 3.27 GHz to 3.13 GHz. Moreover, it has been observed that introducing slot and enhanced coupling provides better frequency responses in MPA and ECMPA sensors while maintaining acceptable radiation characteristics, i.e., realized gain is above 4 dB and VSWR less than 2. The detailed analysis of radiation characteristics has not been provided as it is out of the scope of paper and it is more focused on sensing analysis.

In order to verify the sensitivity enhancement of the proposed slotted patch antenna with that of the standard patch antenna, the frequency change (Δf), relative frequency change in percent (PRFS), enhancement in PRFS (PRFSE), sensitivity (S) and sensitivity enhancement (SE), are calculated and plotted with respect to S_{11} responses using Eqs 6–10 [29]:

$$\Delta f = f_u - f_l \text{ (GHz)} \tag{6}$$

$$PRFS = \frac{\Delta f}{f_u} \times 100 \text{ (\%)} \tag{7}$$

$$PRFSE = \frac{PRFS_{proposed}}{PRFS_{conventional}} \tag{8}$$

$$S = \frac{\Delta f}{\Delta \mathcal{E}} \tag{9}$$

$$SE = \frac{S_{proposed}}{S_{conventional}} \tag{10}$$

where, f_u and f_l represent the resonant frequency under unloaded and loaded (with leaf sample) conditions. The relative position of the sample under test (SUT) just above the patch's surface plays a crucial role in determining the total capacitance and effective relative permittivity of microstrip patch sensor. As a result, the resonant frequency of S_{11} displays as a nonlinear function with regard to the effective relative permittivity [30,31]. indicates that the sensitivity, S, varied as the relative permittivity modified with higher values for lower relative permittivity. In this case, when the permittivity of the SUT is $\mathcal{E}_r$ = 20, the resonant frequency shift, Δf_r, for the traditional patch antenna is 0.10 GHz, compared to 0.32 GHz and 0.03 GHz for the two proposed patch antennas. The percentage resonant frequency shift (PRFS) for the traditional patch antenna is 1.72%, whereas for primary and secondary proposed patch antennas, it reached at 5.52% and 0.51% correspondingly as shown in Figure 5B. Therefore, the percentage resonant frequency shift enhancement (PRFSE) of the proposed patch antennas is 3.2 and 0.3 accordingly.

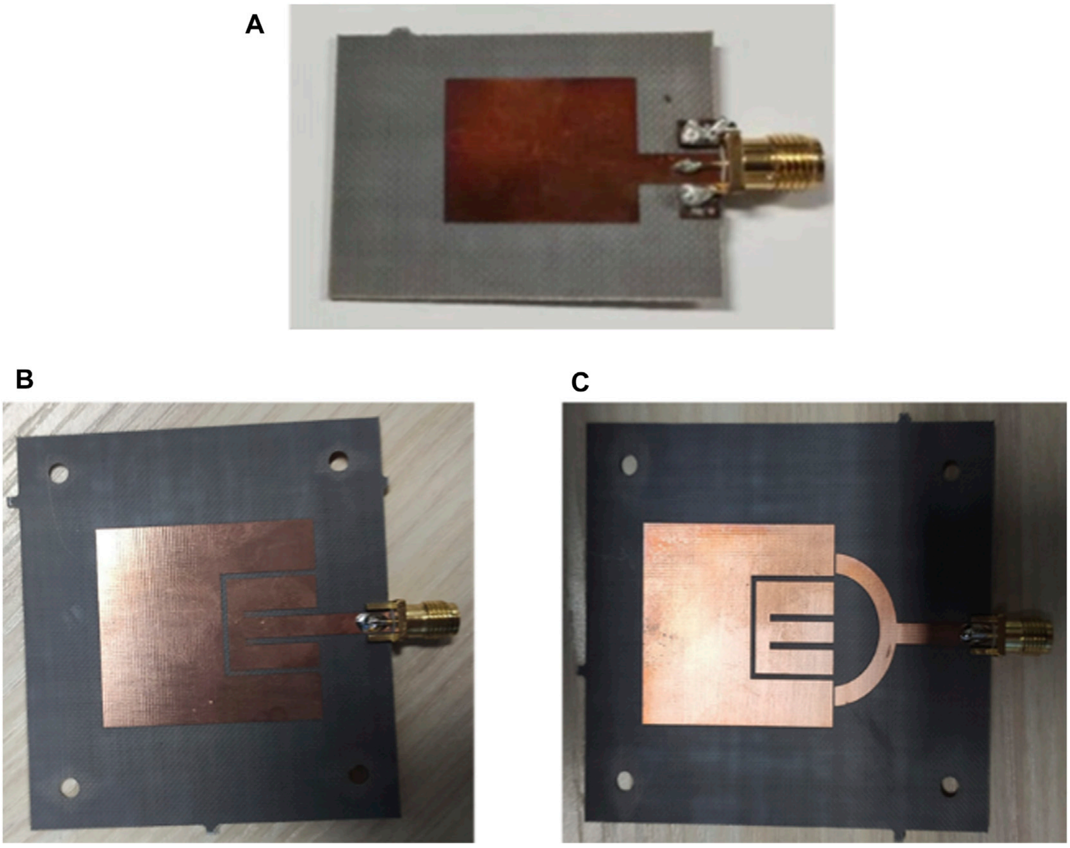


FIGURE 6
The fabricated slotted patch antennas: **(A)** standard microstrip patch antenna; **(B)** modified slotted microstrip patch antenna; **(C)** enhanced coupling slotted microstrip patch antenna.

Moreover, it is evident from Figure 5C that as the permittivity of the SUT is increased to $\varepsilon_r = 30$, the shift in resonant frequency, Δf_r for the typical patch antenna (CPA) reached 0.68 GHz. In comparison, it reached to 0.56 GHz and 0.20 GHz for the proposed patch antennas, MPA and ECMPA respectively as depicted in Figure 5C. The PRFS for the traditional patch antenna is 11.72%. Meanwhile, for the proposed slotted patch antennas, the percentage is 9.66% and 3.44%, respectively. Subsequently, the PRFSE for both the initial (MPA) and second designed slotted patch antennas (ECMPA) is 0.84 and 0.30. Additionally, the S (sensitivity) of the modified patch antenna (MPA) and enhanced coupling slotted patch antenna (ECMPA) is 0.06 GHz and 0.020 GHz, respectively. Consequently, the sensitivity Enhancement (SE) is found out to be 1.61 GHz and 0.575 GHz respectively for the proposed designs. Therefore, it is evident that the sensitivity of the newly designed patch antennas at the resonant frequency of concern surpasses that of conventional patch antennas by a factor ranging from 0.57 to 1.61, within a relative permittivity range of 20–30.

5 Experimental process for characterization of moisture content relative to mass

The S_{11} parameters of the designed fabricated traditional and slotted patch antennas shown in Figures 6A–C, have been analyzed by means of an Agilent N5230A network analyzer as indicated in Figure 7A. The leaf is placed above the designed antenna sensor and the shift in resonant frequency is measured. The leaf that is used as a sample is freshly plucked from the renowned Ginkgo tree. It is important to recognize that sensitivity levels may vary between dead and living leaves, as well as between different types of leaves, due to each plant's unique characteristics in responding to water stress. The experiments are conducted in a laboratory environment filled with various objects to simulate the complex signal propagation challenges encountered in agricultural settings. Measurements were taken under controlled conditions (temperature of 25°C ± 2°C, and 55% RH ±5% humidity) to simulate agricultural environments.

For the sensitivity analysis, the modified slotted patch antenna (ECMPA) is tested under moisture stress. In the moisture content analysis setup, four (4) distinct levels are chosen with water content that constitutes to approximately mass ranging from 4.96 g to 5.30 g. The weighing process is depicted in Figure 7B. Initially, with an empty sample holder and the dry leaf weighing 4.96 g, there is negligible change in the resonance frequency, indicating the baseline measurement scenario. As the leaf's mass increases due to water absorption, observable shifts in the resonant frequency occur, attributable to the varying dielectric properties of the leaf as it transitions from dry to increasingly moist. For the simplicity of discussion, four distant moisture levels, i.e., 0%, 1.6%, 3.0%, 5.0%,

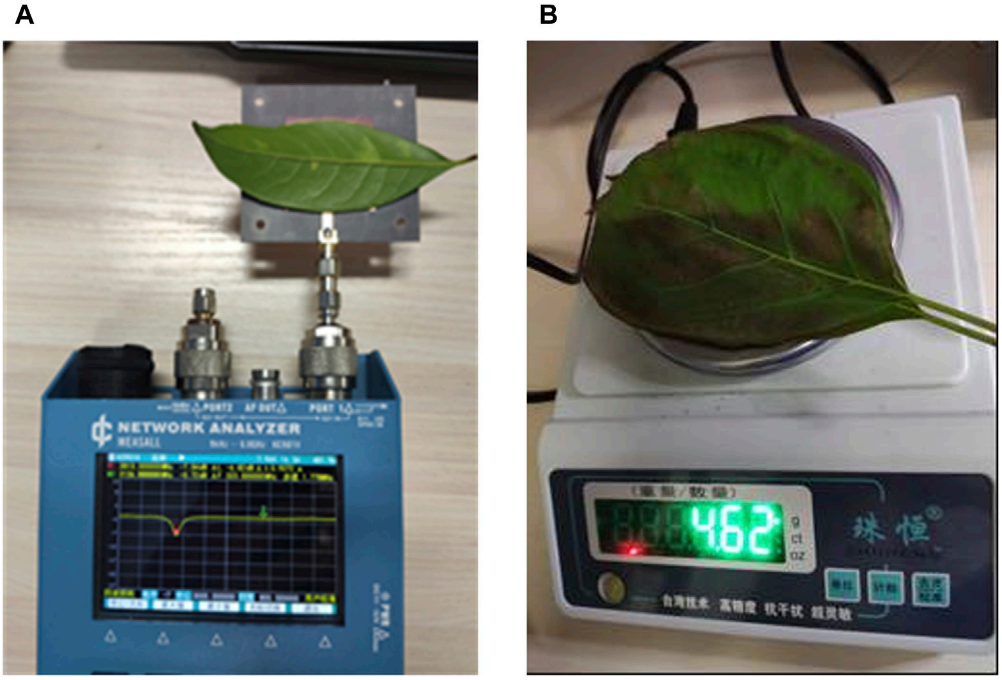


FIGURE 7
(A) Test bed with network analyzer **(B)** weighting process of the leaf after every level of water.

TABLE 1 Mass of wet leaf and moisture content (%age) by adding water.

M_{wet}	M_{dry}	Moisture content (%age) (%)	Added water (mL)
4.96	4.96	0	0
5.04	4.96	1.6	10
5.11	4.96	3.0	20
5.21	4.96	5.0	30

and 6.9% of moisture content are selected. Consequently, the relative moisture content (mc) in leaf samples, in percentage can be calculated from Eq. 11:

$$mc = \frac{m_{water}}{\left(m_{water} + m_{dryleaf}\right)} \times 100\% \tag{11}$$

where m_{water} and $m_{dry\ \mathrm{leaf}}$ are the weight of water and weight of dry leaf sample.

Table 1 summarizes the results from an experiment designed to measure the moisture content in leaves, where moisture is quantified by the volume of water added, denoted as M_{wet} in grams. The experiment employs the designed enhanced coupled microstrip patch antenna sensor to detect changes in the moisture content through shifts in resonant frequency and insertion loss.

The experimental data convey a clear correlation between the moisture content of a leaf and the resonant frequency exhibited by the enhanced coupled patch antenna (ECMPA) sensor. When the sensor is assessing a dry leaf, devoid of additional moisture, the recorded resonant frequency stands at 2.85 GHz. As the leaf's moisture content escalates to 1.6%, there's a discernible dip in the resonant frequency to 2.83 GHz. Progressing to 3.0%, the frequency further descends to 2.80 GHz, and with a 5.0% moisture increment, it plunges to 2.79 GHz. The most substantial moisture level tested, which is 6.9%, correlates with a resonant frequency of 2.75 GHz. This consistent downward shift in frequency with rising moisture levels is attributable to the increased dielectric constant of the leaf material. The addition of moisture elevates the dielectric constant since water possesses a higher dielectric constant relative to air. Consequently, as the moisture content within the sample under test (SUT) surges, the resonant frequency of the antenna correspondingly diminishes, affirming the sensor's capacity to detect and quantify varying levels of leaf hydration through frequency shifts.

6 Regression analysis

Calibration fitting tool is applied to find out the which of the techniques among the relative permittivity measurement and reflection coefficient measurement is best suited for the sensitivity of the proposed sensor with respect to resonant

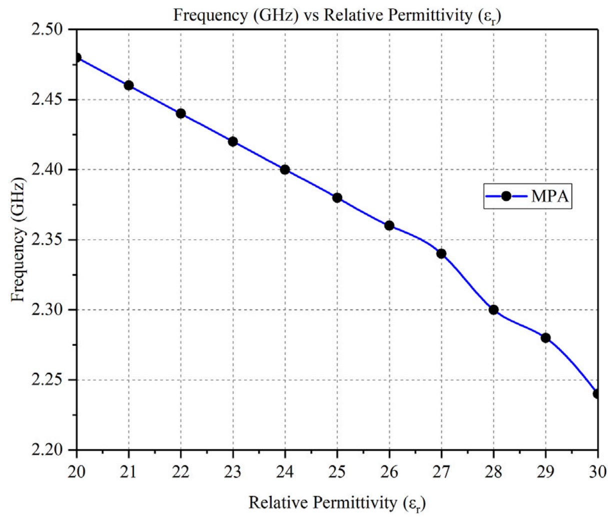


FIGURE 8
Calibration curve for regression analysis related to frequency (GHz) vs. permittivity ($\mathcal{E}_r$).

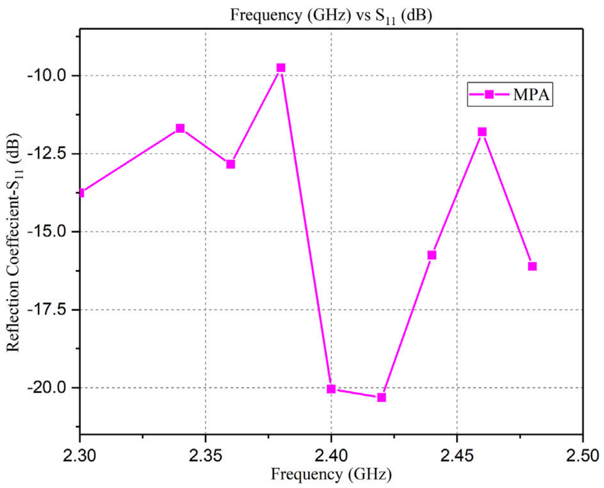


FIGURE 9
Calibration curve for regression analysis related to frequency (GHz) vs. reflection coefficient S_{11} (dB).

frequency. The graph displayed in Figure 8, shows a linear regression analysis showing the relationship concerning the frequency of a slotted patch antenna (MPA) as shown in Figure 1B and the relative permittivity of a leaf, which is an indirect measure of its moisture content. As the relative permittivity increases, the frequency decreases, suggesting that moisture content has a damping effect on the frequency. The calibration equation ($\mathrm{MC} = -0.0345f + 3.323$) shows the variation in moisture content (MC) as a function of frequency (f). The calibration equation is valid over a range of 2.2 GHz–2.5 GHz as it is the optimal range for sensor's performance. The high (R^2) value of 0.9539 indicates a strong linear relationship between the variables.

Figure 9 displays a plot of resonant frequency against reflection coefficient for a slotted patch antenna, indicating moisture content (MC) calibration. As frequency increases from 2.2 to 2.5 GHz, the reflection coefficient becomes less negative, suggesting less energy is reflected back and more is absorbed by the load (leaf in this case). The calibration equation ($\mathrm{MC} = -17.54f + 26.627$) with ($R^2 = 0.1503$) is displayed, suggesting a weak linear relationship between frequency and moisture content based on reflection coefficient values. The low (R^2) value indicates that reflection coefficient alone may not be a strong predictor of moisture content as mentioned in [32].

Moreover, regression analysis technique is performed for frequency with respect to permittivity and frequency with respect to reflection coefficient to find out which approach is much better for sensing purpose. The model summary tabulated in Table 2 indicates a high correlation coefficient (R) of 0.961, suggesting a strong positive relationship between the resonant frequency and moisture content. The R Square value of 0.924 implies that approximately 92.4% of the variability in the moisture content can be explained by the model, which is a significant proportion, indicating a strong model fit. This is further reinforced by the Adjusted R Square value of 0.916, which slightly adjusts the R Square value for the number of predictors in the model, ensuring the model's validity despite the number of terms. The performed ANOVA analysis depicts that the regression model is statistically substantial (p-value <0.001). The coefficients generated reveals the model's equation: Each unit decrease in frequency is associated with a significant increase in moisture content, as shown by the large negative coefficient for frequency. The calculated t-values and corresponding p-values indicate that both the intercept and slope are significantly different from zero, reinforcing the model's predictive.

6.1 Calculation of mean relative error

Additionally, the mathematical validation leads to determining the sensitivity of the experimental data by Eq. 9 whereas the MRE (mean relative error) for the moisture content is found out by Eq. 12:

$$Mean\, Relative\, Error\ (MRE) = \frac{MC_{Actual} - MC_{Predicted}}{MC_{Actual}} \qquad (12)$$

Where, $\mathrm{MC}_{\mathrm{Actual}}$ is the moisture content being measured via experimental analysis depicted in Table 1. The value of predicted moisture content (PMC) is determined by plotting a calibration curve between frequency and permittivity of the SUT as depicted in Figure 8. The calibration curve provided the calibration equation, i.e., $\mathrm{MC} = -0.0345f + 3.323$. It is calculated at second level of actual moisture content (AMC), which is at 3.55%. By using the calibration equation, the PMC is derived at 3.43%. Therefore, the mean relative error (MRE) at this point is determined as 0.0338 which is an indication of a very low error between the predicted and measured values. Moreover, the designed slotted patch antenna sensor depicts better sensitivity, i.e. 2% and lower MRE, i.e. 0.038 as compared to previous research work done in moisture content detection [33].

TABLE 2 Summary of the performed Regression Analysis.

Model	R	R Square	Adjusted R Square	Std. Error of the estimate
Moisture Content	0.961a	0.924	0.916	5.2216

TABLE 3 Comparative Analysis with previous designs.

Parameters	Proposed design	Previous design 1 [24]	Previous design 2 [35]	Previous design 3 [34]
Size	47.7 × 56.3 mm^2	38 × 38 mm^2	75.85 × 100 mm^2	48.3 × 62.1 mm^2
Operating Frequency	2.87 GHz and 3.2 GHz	5.2 GHz and 6.8 GHz	2.45 GHz	8.48 and 10.69 GHz
Sensing Material	Leaf	Rice	Granular	Liquids
Substrate	F4B	FR4	RF Duroid	Polytetrafluoroethylene RT Duroid
Sensor Cost	Cheapest	Relatively High	High	High
R^2 value	0.954	0.411 and 0.379	Not Reported	0.57 and 0.78
Moisture Content (%)	0.86%–22.48%	10.71%–21.87%	0–30	Not Reported
MRE (%)	0.30	0.55	3.54	0.65

7 Comparison with previous research

Table 3 provides comparative analysis of the proposed ECMPA sensor with the three sensor designs used for moisture content detection in different materials. The proposed design of the enhanced coupled microstrip patch antenna (ECMPA) sensor presents several advancements over the previous reported designs [24,34,35] as indicated by various parameters. With dimensions of 47.7 × 56.3 mm^2, it is larger than previous design 1, potentially offering a more extensive surface area for interaction, which can be crucial for sensing performance. It operates at lower frequencies of 2.87 GHz and 3.2 GHz compared to previous designs 1 and 3, which could allow for deeper penetration into materials, beneficial for specific sensing applications. In contrast, its frequencies are somewhat similar to previous design 2, suggesting possible similarities in penetration capabilities. The choice of leaf as the sensing material for the proposed design suggests a specialized or targeted application, possibly in the agricultural or environmental monitoring sectors, differing from the rice, granular, and soil materials of the previous designs. This specificity could leverage the unique properties of leaves in sensing applications. The substrate material, F4B, differs from the FR4 used in previous design 1, which can affect the antenna's performance characteristics such as dielectric properties and mechanical stability. Cost-wise, the proposed design is noted as the cheapest, offering a significant advantage in terms of affordability and accessibility for widespread use. Its performance in terms of accuracy is substantiated by a high R^2 value of 0.954, far surpassing the 0.411 and 0.57 of design 1 and design 3, indicating a reliable and accurate model fitting which is crucial for sensor efficacy. In terms of moisture sensing, the proposed design can detect a range from 0.86% to 22.48%, offering a more precise and suitable range for specific applications compared to range of previous design 1, 2, and 3. Finally, its mean relative error (MRE) at 0.30% is significantly lower than that of the other designs, underscoring its superior accuracy and reliability in sensing applications.

Hence, it can be proved with this method that he resonant frequency of the microstrip patch antenna decreases as the moisture content of the leaf increases due to the variation in relative permittivity as proved in previous research [36]. It is due to the fact that the water molecules in the leaf absorb some of the electromagnetic radiation, which reduces the resonant frequency of the antenna. This method of measuring leaf moisture content is non-destructive and can be used in a variety of applications, such as agriculture, environmental monitoring, and plant physiology research.

8 IOT system for moisture content detection

The architectural framework of any system is pivotal in delineating its structure and elucidating the functionality of its components. In the realm of IoT-based systems, numerous architectures have been suggested. The Service Oriented Architecture (SOA), renowned for its efficacy in smart applications, is adopted for the proposed IOT based moisture content detection system due to its simplicity and clear delineation of components. The IOT system as an architecture, encompasses four layers: sensing, network, service, and interface [37].

8.1 Sensing layer

Central to the IoT system is the sensing layer, focused on the "things"—in our context, a microstrip patch antenna designed for moisture sensing in leaves. The primary challenge is engineering a sensor with the precision to detect moisture content accurately. Utilizing a microstrip patch antenna with slotted enhancements, based on the principle of effective permittivity changes, we targeted the microwave frequency range known for its sensitivity to moisture content. Selecting a 2–3 GHz frequency for its compact size and effective correlation with moisture content, the research embarked on optimizing the antenna's sensitivity through various techniques, including enhanced coupling, to ensure minute variations in moisture content are detectable.

8.2 Network layer

Achieving reliable wireless connectivity for developed sensor to transmit data is crucial. By integrating the sensor within a SOA framework, it ensures modular manageability suited for agricultural settings [38]. After evaluating various communication technologies, Bluetooth Low Energy (BLE) emerged as the optimal choice due to its low energy consumption, compatibility with numerous operating systems, and effective data rate for our application. This BLE-based network facilitates seamless data transmission from the sensor to the gateway, ensuring real-time monitoring of leaf moisture levels.

8.3 Service layer

The service layer, demanding significant computational resources, is hosted on a robust Dell precision tower 3960 workstation, equipped to handle the intensive tasks of data storage, analysis, and reporting. This local-cloud setup not only supports real-time operations but also enables data analytics, essential for the comprehensive IoT-based leaf moisture monitoring system. The system is designed to offer a plethora of services crucial for processing and analyzing the data collected from the field.

8.4 Interface layer

The user interface, developed in LabView®, allows for direct interaction with the system, presenting processed data to users in an accessible manner. This application, coupled with LabView web services, enables both local and remote monitoring of moisture levels, facilitating timely decision-making in agricultural management. Through RESTful API, the system ensures data is shared seamlessly, allowing users to access real-time information and thus optimize irrigation practices based on the moisture content detected by our sophisticated sensor. The flowchart illustration of an IOT-based leaf moisture content monitoring system is depicted in Figure 10.

8.5 IOT based monitoring system

Evaluating the effectiveness of a cutting-edge IoT-based leaf moisture sensing system is the primary aim of the research. An experimental setup is established, as shown in Figure 11, within a laboratory designed to replicate the varying conditions typical of an agricultural setting as proposed by [39]. A microstrip patch antenna sensor, crucial to the research, is strategically placed few meters away from the gateway node to mirror the real-world deployment distance in a field, though for visualization purposes in Figure 11, it is presented next to the gateway.

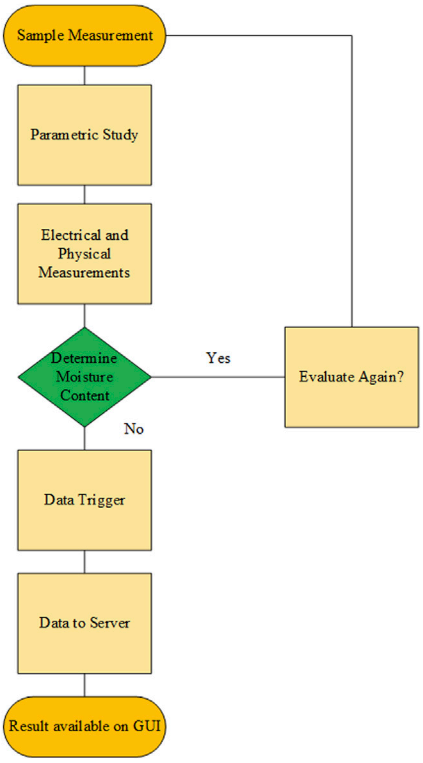


FIGURE 10
Flowchart of an IOT-based leaf moisture content monitoring system.

The selected laboratory environment is intentionally filled with various objects, such as equipment and furniture, to simulate the complex signal propagation challenges encountered in agricultural environments. This configuration is chosen to evaluate the durability of the wireless network against the physical barriers common in farming areas. Furthermore, the laboratory included two WLAN routers positioned at different sites to determine the system's capability to withstand interference.

As detailed in Figure 11, the microstrip patch antenna is connected to a high-precision microwave analyzer, which is equipped with a BLE adapter, emphasizing the system's applicability to IoT scenarios. The calibration of the system is conducted through standardized procedures to ensure the accuracy of measurements, focusing on the S_{11} parameter across a broad frequency spectrum.

Before introducing leaf samples, a baseline performance of the system is recorded to identify the unloaded resonance frequency. Subsequently, leaf samples with precise dimensions are placed onto the sensor, aligning carefully with the antenna's enhanced coupling region for accuracy. Multiple measurements are taken for each sample to verify the consistency of the data. The orientation of the samples is meticulously arranged to facilitate optimal interaction with the electromagnetic fields, a vital factor for precise moisture detection. All experiments are performed under controlled conditions—maintaining a steady temperature of 25°C ± 2°C and humidity levels within 55% RH ± 5%—to closely mimic the operational environment of the IoT-based system in an agricultural context, thus confirming its potential for practical applications. The detailed working of the idea is to be presented as future research.

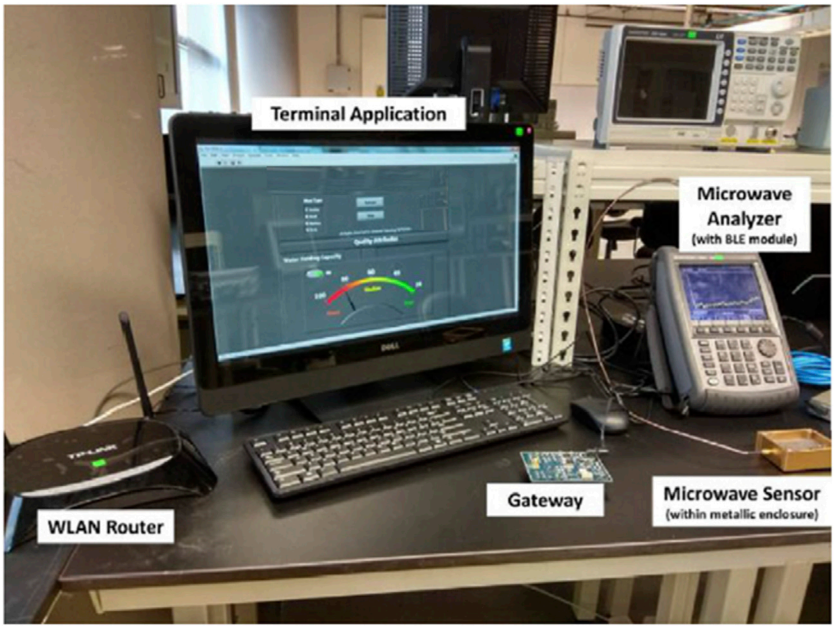

Terminal Application
Microwave Analyzer (with BLE module)
WLAN Router
Gateway
Microwave Sensor (within metallic enclosure)

FIGURE 11
A real-time IOT based monitoring system in lab.

9 Conclusion

The research presented herein introduces an innovative approach to moisture measurement in foliage through the use of modified slotted microstrip patch sensor antennas, showcasing a marked improvement in sensitivity over the conventional patch antenna models. Our findings reveal that the sensitivity of these newly developed slotted patch antennas surpasses that of the traditional designs by a factor ranging from 0.57 to 1.67 when tested against dielectric samples with constants in the 20 to 30 range. Fabricated on a 0.8 mm F4B substrate, these antennas achieve operational resonance frequencies between 2.4 and 3.0 GHz without load. The key to their enhanced performance lies in the strategic integration of slots above the antenna's feed point, coupled with an advanced coupling technique, elevating their sensitivity levels. Moreover, an IOT based implementation for real time monitoring is also presented. Such improvements position the modified slotted patch antennas as highly versatile tools, suitable for a wide array of applications that span from proximity sensing in automated systems to the wireless monitoring of biological materials, the measurement of permittivity in both solid and liquid mediums, and the non-destructive evaluation of moisture content in soils, foods, and various liquids. This versatility, combined with their increased sensitivity, underscores the potential of these antennas to significantly advance the field of IoT implementations by providing more accurate and reliable sensors for a multitude of applications.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

MK: Conceptualization, Data curation, Formal Analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing–original draft, Writing–review and editing. XL: Funding acquisition, Investigation, Methodology, Project administration, Supervision, Validation, Writing–review and editing. CZ: Conceptualization, Investigation, Methodology, Validation, Writing–review and editing. AS: Software, Writing–review and editing.

Funding

The author(s) declare financial support was received for the research, authorship, and/or publication of this article. The research received financial support from the National Natural Science Foundation of China (NSFC) under Grant No. 61571084 and is also partly funded by the Fundamental Research Funds for the Central Universities (FRFCU) under Grant No. ZYGX2019Z022, acknowledging the institutional and financial backing that facilitated this study.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Li J, Hong X, Wang F, Yang L, Yang D, Simultaneous retrieval of corn growth status and soil water content based on one GNSS antenna. *Remote sensing* (2023) 15(7):1738. doi:10.3390/rs15071738

2. Nguyen TPT, Do TG, Dao PT, Le MT, Underground soil moisture sensor based on monopole antenna for precision agriculture. In: *2023 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*. Bali, Indonesia: IEEE (2023). doi:10.1109/IAICT59002.2023.10205648

3. Huang H, Flexible wireless antenna sensor: a review. *IEEE sensors J* (2013) 13(10): 3865–72. doi:10.1109/jsen.2013.2242464

4. Huang H. Antenna sensors in passive wireless sensing systems. *Handbook Antenna Tech* (2016) 2795–838. doi:10.1007/978-981-4560-44-3_86

5. Saeed K, Shafique MF, Byrne MB, Hunter IC, Planar microwave sensors for complex permittivity characterization of materials and their applications. *Appl Meas Syst* (2012) 319–50. doi:10.5772/36302

6. Fratticcioli E, Dionigi M, Sorrentino R, A simple and low-cost measurement system for the complex permittivity characterization of materials. *IEEE Trans Instrumentation Meas* (2004) 53(4):1071–7. doi:10.1109/tim.2004.830753

7. Withayachumnankul W, Tuantranont A, Fumeaux C, Abbott D. Metamaterial-based microfluidic sensor for dielectric characterization. *Sensors Actuators A: Phys* (2013) 189:233–7. doi:10.1016/j.sna.2012.10.027

8. Jirousek EP, *Material characterization using complementary split ring resonators*. Diploma Thesis, Technische Universität Wien (2022). doi:10.34726/hss.2022.93500

9. Herrmann PSd. P, Sydoruk V, Marques Porto FN, Microwave transmittance technique using microstrip patch antennas, as a non-invasive tool to determine soil moisture in rhizoboxes. *Sensors* (2020) 20(4):1166. doi:10.3390/s20041166

10. Bahl I, Bhartia P, Stuchly S, Design of microstrip antennas covered with a dielectric layer. *IEEE Trans antennas propagation* (1982) 30(2):314–8. doi:10.1109/tap.1982.1142766

11. Zambrano MV, Dutta B, Mercer DG, MacLean HL, Touchie MF. Assessment of moisture content measurement methods of dried food products in small-scale operations in developing countries: a review. *Trends Food Sci Tech* (2019) 88: 484–96. doi:10.1016/j.tifs.2019.04.006

12. Bekal A, Balsubramaniam BG, Awaghade V, Ghute S, Application of microwave moisture sensor for DOC and animal feed. *IEEE Sensors J* (2020) 20(24):14809–16. doi:10.1109/jsen.2020.3010574

13. Jiarasuwan S, Chamnongthai K, Kittiamornkul N, A design method for a microwave-based moisture sensing system for granular materials in arbitrarily shaped containers. *IEEE sensors J* (2021) 21(17):19436–52. doi:10.1109/jsen.2021.3087414

14. Bartley PG, Nelson SO, McClendon RW, Trabelsi S, Determining moisture content of wheat with an artificial neural network from microwave transmission measurements. *IEEE Trans Instrumentation Meas* (1998) 47(1):123–6. doi:10.1109/19.728803

15. Kim K-B, Kim J-H, Lee SS, Noh SH. Measurement of grain moisture content using microwave attenuation at 10.5 GHz and moisture density. *IEEE Trans Instrumentation Meas* (2002) 51(1):72–7. doi:10.1109/19.989904

16. Zhang H-L, Ma Q, Fan L-F, Zhao P-F, Wang J-X, Zhang X-D, et al. Nondestructive *in situ* measurement method for kernel moisture content in corn ear. *Sensors* (2016) 16(12):2196. doi:10.3390/s16122196

17. Lin L, He Y, Xiao Z, Zhao K, Dong T, Nie P, Rapid-detection sensor for rice grain moisture based on NIR spectroscopy. *Appl Sci* (2019) 9(8):1654. doi:10.3390/app9081654

18. Gilmore C, Asefi M, Paliwal J, LoVetri J, Industrial scale electromagnetic grain bin monitoring. *Comput Elect Agric* (2017) 136:210–20. doi:10.1016/j.compag.2017.03.005

19. Zhang C, Shi Z, Yang H, Zhou X, Wu Z, Jayas DS, A novel, portable and fast moisture content measuring method for grains based on an ultra-wideband (UWB) radar module and the mode matching method. *Sensors* (2019) 19(19):4224. doi:10.3390/s19194224

20. Fan L, Chai Z, Wang Y, Wang Z, Zhao P, Li J, et al. A novel handheld device for intact corn ear moisture content measurement. *IEEE Trans Instrumentation Meas* (2020) 69(11):9157–69. doi:10.1109/tim.2020.2994603

21. Javanbakht N, Xiao G, Amaya RE, Sangha J, Ruan Y, Compact frequency selective surface antenna for grain moisture content monitoring. In: *2021 IEEE 19th International Symposium on Antenna Technology and Applied Electromagnetics (ANTEM)*. Winnipeg MB, Canada: IEEE (2021), pp. 1-2. doi:10.1109/ANTEM51107.2021.9518859

22. Singh U, Ahmed A, Mukherjee J, Microstrip Patch antenna as sensor based on Meta material integrated structures for non-invasive characterization of biological materials. *IEEE Sensors J* (2024) 24:9970–81. doi:10.1109/jsen.2024.3365526

23. Bansal S, Kaur P, Microstrip and metamaterial embedded patch antenna sensors for determination of moisture in rice, wheat, and pulse grains. *J Electrochem Soc* (2024) 171(1):017504. doi:10.1149/1945-7111/ad1c17

24. Jain S, Mishra PK, Thakare VV, Mishra J, Design of microstrip moisture sensor for determination of moisture content in rice with improved mean relative error. *Microwave Opt Technol Lett* (2019) 61(7):1764–8. doi:10.1002/mop.31763

25. Huang Y, *Antennas: from theory to practice*. John Wiley and Sons (2021).

26. Khan MT, Lin XQ, Chen Z, Xiao F, Yan YH, Memon AK, Design and analysis of a microstrip patch antenna for water content sensing. In: *2019 16th International Computer Conference on Wavelet Active Media Technology and Information Processing*. Chengdu, China: IEEE (2019), pp. 438-442.

27. Chang K, Hsieh L-H, *Microwave ring circuits and related structures*. John Wiley and Sons (2004).

28. Khan MT, Ali SM, A brief review of measuring techniques for characterization of dielectric materials. *Int J Inf Tech Electr Eng* (2012) 1(1):1–5.

29. Salim A, Lim S, Complementary split-ring resonator-loaded microfluidic ethanol chemical sensor. *Sensors* (2016) 16(11):1802. doi:10.3390/s16111802

30. You K, Salleh J, Abbas Z, You L, A rectangular patch antenna technique for the determination of moisture content in soil. *Prog Electromagnetics Res* (2010) C:850–4.

31. Yahaya NZ, Abbas Z, Ali BM, Ismail A, Ansarudin F, Intercomparison of methods for determination of resonant frequency shift of a microstrip patch antenna loaded with hevea rubber latex. *J Sensors* (2014) 2014:1–9. doi:10.1155/2014/656972

32. Jain S, Mishra PK, Thakare VV, Mishra J, Microstrip moisture sensor based on microstrip patch antenna. *Prog Electromagnetics Res M* (2018) 76:177–85. doi:10.2528/pierm18092602

33. Yeo J, Lee J-I, Slot-loaded microstrip patch sensor antenna for high-sensitivity permittivity characterization. *Electronics* (2019) 8(5):502. doi:10.3390/electronics8050502

34. Ghretli MM, Khalid K, Grozescu IV, Sahri MH, Abbas Z, Dual-frequency microwave moisture sensor based on circular microstrip antenna. *IEEE Sensors J* (2007) 7(12):1749–56. doi:10.1109/jsen.2007.908920

35. Limpiti T, Krairiksh M, *In situ* moisture content monitoring sensor detecting mutual coupling magnitude between parallel and perpendicular dipole antennas. *IEEE Trans Instrumentation Meas* (2012) 61(8):2230–41. doi:10.1109/tim.2012.2186656

36. Jilani MT, Wen WP, Rehman MZU, Khan AM, Cheong LY, Microwave sensor for non-destructive dielectric characterization of biological systems. *Int J Appl Electromagnetics Mech* (2016) 50:353–63. doi:10.3233/jae-150114

37. Haq AU, Li JP, Agbley BLY, Khan A, Khan I, Uddin MI, et al. IIMFCBM: intelligent integrated model for feature extraction and classification of brain tumors using MRI clinical imaging data in IoT-healthcare. *IEEE J Biomed Health Inform* (2022) 26(10):5004–12. doi:10.1109/jbhi.2022.3171663

38. Li JP, Haq AU, Din SU, Khan J, Khan A, Saboor A, Heart disease identification method using machine learning classification in e-healthcare. *IEEE access* (2020) 8: 107562–82. doi:10.1109/access.2020.3001149

39. Jilani MT, Rehman MZU, Khan AM, Chughtai O, Abbas MA, Khan MT, An implementation of IoT-based microwave sensing system for the evaluation of tissues moisture. *Microelectronics J* (2019) 88:117–27. doi:10.1016/j.mejo.2018.03.006

frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 19 June 2024
DOI 10.3389/fphy.2024.1379873

Cross-modal retrieval based on multi-dimensional feature fusion hashing

Dongxiao Ren[1]* and Weihua Xu[2]

[1]Department of Data Science, School of Sciene, Zhejiang University of Science and Technology, Hangzhou, China, [2]Department of Digital Finance, Quanzhou Branch of Industrial and Commercial Bank of China, Quanzhou, China

OPEN ACCESS

EDITED BY
Qifei Wang,
University of California, Berkeley, United States

REVIEWED BY
Jianping Gou,
Southwest University, China
Ying Ma,
Harbin Institute of Technology, China
Peilin He,
University of Pittsburgh, United States

*CORRESPONDENCE
Dongxiao Ren,
✉ rendx29@163.com

RECEIVED 31 January 2024
ACCEPTED 08 May 2024
PUBLISHED 19 June 2024

CITATION
Ren D and Xu W (2024), Cross-modal retrieval based on multi-dimensional feature fusion hashing.
Front. Phys. 12:1379873.
doi: 10.3389/fphy.2024.1379873

Along with the continuous breakthrough and popularization of information network technology, multi-modal data, including texts, images, videos, and audio, is growing rapidly. We can retrieve different modal data to meet our needs, so cross-modal retrieval has important theoretical significance and application value. In addition, because the data of different modalities can be mutually retrieved by mapping them to a unified Hamming space, hash codes have been extensively used in the cross-modal retrieval field. However, existing cross-modal hashing models generate hash codes based on single-dimension data features, ignoring the semantic correlation between data features in different dimensions. Therefore, an innovative cross-modal retrieval method using Multi-Dimensional Feature Fusion Hashing (MDFFH) is proposed. To better get the image's multi-dimensional semantic features, a convolutional neural network, and Vision Transformer are combined to construct an image multi-dimensional fusion module. Similarly, we apply the multi-dimensional text fusion module to the text modality to obtain the text's multi-dimensional semantic features. These two modules can effectively integrate the semantic features of data in different dimensions through feature fusion, making the generated hash code more representative and semantic. Extensive experiments and corresponding analysis results on two datasets indicate that MDFFH's performance outdoes other baseline models.

KEYWORDS

information retrieval, cross-modal retrieval, vision transformer, multi-dimensional semantic feature, hashing

1 Introduction

The swift growth of multimedia data has brought a lot of demand for cross-modal retrieval. With the growing scale of data on the Internet, data types are becoming more and more diversified, including text, images, videos, audio, etc. The data modality that users are interested in is no longer single, and the user retrieval shows a development trend from single modality to cross modalities. Data has different modalities and these expression forms are different, while the semantics behind them may be related to each other and good use of different modal data can facilitate our lives to a certain extent. For instance, when you visit the Great Wall of China, you can retrieve the corresponding text and video introduction through the photos of the Great Wall. The information supplement helps you to quickly familiarize yourself with scenic spots for the first time. Besides the field of daily life, cross-modal retrieval has important applications in many domains such as medicine [1], finance [2], and information security [3]. Therefore, it is an interesting and challenging problem to construct an effective cross-modal retrieval system.

Since the data distributions and feature representations of different modal data are different, they cannot be compared directly. Representation learning can effectively deal with this problem. In such methods, the aim is to learn a function that can transform different modalities into a common feature space [4, 5], where we can compare them directly. Due to the quick expansion of the data scale and the decline of data retrieval efficiency, the hashing codes are applied to cross-modal retrieval tasks [6–8]. This type of method maps high-dimensional features to the Hamming space by transforming data into hash binary codes and uses XOR of hash binary codes to calculate the Hamming distance. Hash binary codes with small Hamming distance have similar original data, and *vice versa*.

Through many scholars' research and efforts, cross-modal hashing retrieval has achieved many successes. Specifically, based on artificial features representing the original data, many models [9–14] are proposed, known as traditional cross-modal hashing models. Due to the limitations of handmade features, the retrieval efficiency of such models is hard to further breakthrough. Because of the good performance in feature learning, deep learning has been applied in cross-modal hashing retrieval. For example, deep neural networks can automatically capture the data features and hash functions in Refs. [15–20].

However, existing deep cross-modal hash models usually only pay attention to the single-dimensional semantic features of data and do not fully consider the information complementation between specific features presented by data in different dimensions. Besides, the multi-dimensional fusion of semantic information is more conducive to capturing the semantic correlation of different modal data, thus helping to narrow the semantic gap. So, effective fusing of multi-dimensional semantic features of different modal data is very important in improving cross-modal retrieval. Because of Transformer's excellent performance in the computer vision field in recent years, we try to use it to better learn the images' semantic features in different dimensions. Similarly, we construct a text multi-dimensional fusion module in the text network, which learns the text multi-dimensional semantic features. Based on these, we propose a novel method for cross-modal retrieval, which is called Multi-Dimensional Feature Fusion Hashing (MDFFH). Our method has these three characteristics.

- MDFFH constructs multi-dimensional fusion modules in image networks and text networks to learn multi-dimensional semantic features of data, which can effectively complement the semantic features of data in specific dimensions. It is better in semantic relevance, obtained hash codes are more semantic as well.
- Vision Transformer is integrated with a convolutional neural network to form an image multi-dimensional fusion module in MDFFH so the image's local and global information can be well fused.
- Feature extraction and hash function generation are well integrated into a deep learning framework in MDFFH. Comparative experiments and corresponding analyses on two datasets show that MDFFH is superior to other baseline models.

This paper mainly includes five sections. The related work is introduced in Section 2, MDFFH is given in Section 3, and the experiments and comparative analysis are demonstrated in Section 4. Finally, the conclusion is in Section 5.

2 Related work

Representative cross-modal hashing models: There are two categories in Cross-modal hashing models. If supervised information (such as data tags) needs to be used during model training, this type of model is called an unsupervised model; the other type needs to use supervision information during model training, which is called a supervision model. According to the way they learn features, cross-modal hashing retrieval models are divided into two categories, namely, hand-crafted models and deep network models. Data labels are not used to guide hash codes' learning in Unsupervised models during model training. For instance, the subspace shared by different modal data is learned and then the correlation between similar different modal data is maximized in Canonical Correlation Analysis (CCA) [21]. Implicit factors of different modal data are learned and unified hash codes are generated based on matrix decomposition in Collective Matrix Factorization Hashing (CMFH) [22]. In latent semantic sparse hashing (LSSH), sparse coding and matrix decomposition are used to capture important structures in images and potential semantics in texts, respectively [23]. Semantic topics and semantic concepts for images and texts are learned and discrete characteristics of different modal data are maintained in Semantic topic multi-modal hashing (STMH) [25]. Cross-Modal Self-Taught Hashing (CMSTH) [24] applies semantic information to detect multimodal topics, and then uses robust matrix decomposition to convert these different modal data into hash codes that are suitable for quantization. Spectral Multimodal Hashing (SMH) [26] uses spectrum analysis of correlation matrices of multi-modal data, learning parameters from the distribution of multi-modal data to get hash codes. On the contrary, supervised models use available data labels to learn more accurate hash features, which is better than unsupervised models in performance. Semantic correlation maximization (SCM) [27] applies nonnegative matrix decomposition and the nearest neighbor preservation algorithm to preserve semantic consistency within modalities and between modalities. Semantic Preserving Hashing (SePH) [28] transforms the semantic matrix into a probability distribution, makes it as close as possible by minimizing the Kullback-Leibler (KL) divergence, and then applies logical regression to learn the hash function of each modal data [29]. Hash functions and binary codes can be learned simultaneously by the data's similarity matrix with discrete constraints in Enhanced Discrete Multi-modal Hashing (EDMH) [30].

However, the above unsupervised and supervised models all belong to hand-crafted models, which are unable to get the feature relevance between different modal data very well. With the continuous improvement of feature learning, deep neural networks are extensively applied in the cross-modal retrieval field. A deep neural network is introduced into feature learning in Deep Cross-Modal Hashing (DCMH) [31], so the unified model includes feature learning and the generation of hash codes. In Pairwise Relationship Deep Hashing (PRDH) [32], the similarity degree between different modal data is preserved in hash codes while taking into account the similarity between the same modal data. A high-level semantic similarity matrix of continuous values is constructed to guide the learning of hash codes in Deep Multi-level Semantic Hashing (DMSH) [33], which captures the degree of similarity between different modal data. To generate more representative image features, Mask Cross-Modal Hashing (MCMH) [34] effectively combines convolution features with mask features extracted by the Mask R-CNN. Self-supervised adversarial Hashing (SSAH) [35] introduces adversarial loss through the construction of a label network to shorten the distance between image and text distribution, which brings a better retrieval effect. Using cosine distance and Euclidean distance, the same measurement index can accurately reflect the similarity between different modal data in Deep Semantic Cross-Modal Hashing Based on Graph Similarity of Modal-Specific (DCMHGMS) [36]. The distance between similar data can be reduced by constructing ranking alignment loss to unearth the semantic structure between different modal data in Deep Rank Cross-modal Hashing (DRCH) [37, 38]. Semantic weight factors are constructed to guide the optimization of the loss function and obtain better retrieval performance in Multiple Deep neural networks with Multiple labels for Cross-modal Hashing (MDMCH) [39]. A label network is constructed to jointly guide the feature learning of different modal data and innovates discrete optimization strategies to learn hash codes in Deep Discrete Cross-modal Hashing (DDCH) [40]. To increase the correlation between hash codes, Deep Cross-Modal Hashing with Hashing Functions and Unified Hash Codes Jointly Learning (DCHUC) [41] has constructed a new unified joint hash code framework. To improve the accuracy of hash codes in comparative learning, Unsupervised Contrastive Cross-Modal Hashing (UCCH) [42] proposes a momentum optimizer to make the generated hash codes more accurate.

Transformer: The excellent performance of the Transformer is attributed to the exertion of the attention mechanism, and it is widely used in the field of Natural Language Processing (NLP) [43]. It can assign attention weight according to the input data, to determine which part of the data needs attention. On this basis, limited information processing resources are allocated to important parts and so the performance of the model is improved. Google Deep Mind [44] applied it to the computer vision field for the first time and achieved good performance by combining it with Recurrent Neural Network (RNN). Bahdanau et al. [45] prove the effectiveness of attention mechanisms in the NLP. In [46], Google has successfully constructed the Transformer network structure based on the attention mechanism. Due to the limited feature subspace, it is hard to enhance the performance of this ordinary attention mechanism. The multi-head attention mechanism is more likely to capture features from multiple dimensions by dividing attention operations. Inspired by this important achievement, many researchers tried to introduce Transformer structure into computer vision tasks and achieved good results. In 2020, the Vision Transformer (ViT) proposed by Dosovitskiy et al. [47] performed well in many image classification tasks, because it can capture contextual dependencies at different positions in an image. It is simple and effective, with strong scalability. The larger the amount of data, the better the performance of the ViT. When there is enough data for pre-training, the performance of the ViT is even better than that of the convolutional neural network model, which fully proves that ViT can extract excellent features from images.

3 Proposed method

The innovative networks of this paper will be introduced in this section, and the structural framework of MDFFH is shown in Figure 1. To facilitate comparison with other models, images and texts are selected in our model. Our model can be extended to other modalities easily.

3.1 Notations and problem definitions

Throughout this paper, vectors are denoted by lowercase bold letters (e.g., $\mathbf{z}$), matrices are represented by uppercase bold letters (e.g., $\mathbf{Z}$), and the transposition of the matrix $\mathbf{Z}$ is expressed as $\mathbf{Z}^{\mathrm{T}}$. For the matrix $\mathbf{Z}$, the *ith* row, the *jth* column, the element located in *ith* row and *jth* column and the Frobenius norm are denoted by $\mathbf{Z}_{i*}$, $\mathbf{Z}_{*j}$, Z_{ij} and $\|\mathbf{Z}\|_{\mathrm{F}}$, respetively. The sign function represented by $sign(x)$ is that the value is −1 when x is less than 0, otherwise, the value is 1.

Assume that $O = \{O_n\}_{n=1}^{N}$ denotes the image-text pair dataset, each sample $o_n = (x_n, y_n, l_n)$ includes three parts: one part $x_n \in R^{D_x}$ represents an image feature vector, another part $y \in R^{D_y}$ denotes a text feature vector, and the last part $l_n \in R^{C}$ denotes the corresponding category labels, where D_x, D_y and C are the dimensions of these two modal data's feature and the number of the category labels respectively. $\mathbf{S} \in \{0,1\}^{N\times N}$ is the matrix to measure the similarity degree between different modalities, called the similarity matrix. $S_{ij} = 0$ means that x_i and y_j are not similar to each other and $S_{ij} = 1$ denotes that these two data have at least one same category label. The input data is transformed into the corresponding hash codes and the similarity degree between different hash codes is obtained by calculating their Hamming distance in our model. The more similar the hash codes, the smaller the Hamming distance; the greater the difference between hash codes, the greater the Hamming distance. The formula for calculating Hamming distance is

$$d(c_i, c_j) = \frac{1}{2}\left(k - \langle c_i, c_j \rangle\right), \tag{1}$$

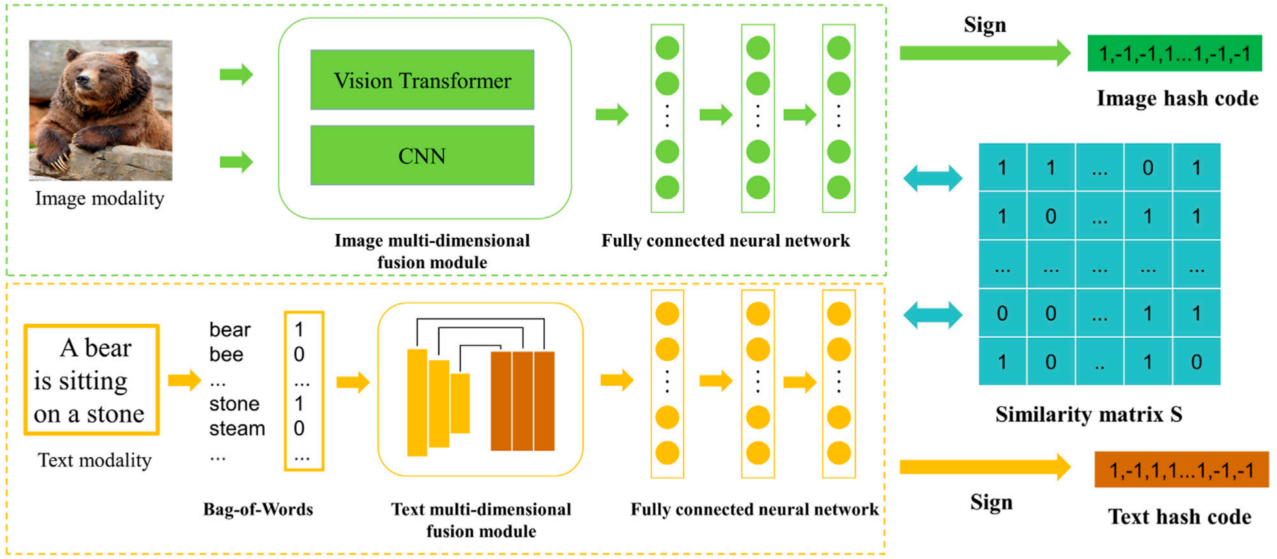
In Eq. 1, c_i and c_j are the hash codes for the vector x_i and y_j, $\langle c_i, c_j \rangle$ represents their inner product and k is the length of hash codes.

MDFFH aims to obtain two hash functions through training, one is $f(x_i; \theta_x)$ for images, and the other is $g(y_j; \theta_y)$ for texts while maintaining the similarity degree of the original data. Here, θ_x and θ_y denote parameters in the different networks. These hash functions can convert the data into hash codes with unified dimensions for comparison.

3.2 Network architecture

The specific details of the networks in our model are as follows.

Image network: Image network is mainly composed of an image multi-dimensional fusion module and a fully connected neural network. Specifically, the multi-dimensional image fusion module



1	1	...	0	1
1	0	...	1	1
...	...	...	...	...
0	0	...	1	1
1	0	..	1	0

FIGURE 1
The structural framework of MDFFH.

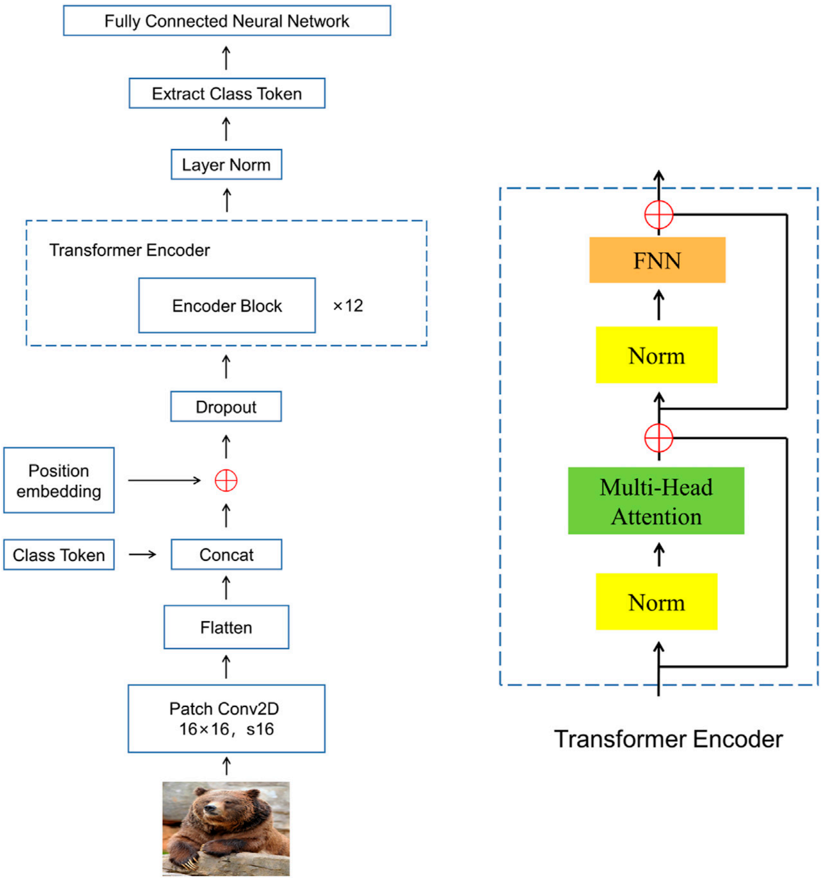


FIGURE 2
Detailed introduction of Vision Transformer.

includes a Vision Transformer network and a convolutional neural network. In the Vision Transformer network, the ViT-B/16 model is chosen as the basic framework and fine-tuned on this basis. We replace the last MLP Head used for the image classification in the ViT-B/16 model with a single-layer completely connected network with 4,096 neurons where the size of each image patch is 16 × 16. The transformer Encoder has 12 Encoder Blocks, which are shown in Figure 2. At the same time, the first six layers of CNN-F [48] are selected as the model of a convolution neural network. In addition, these two networks are pre-rained on ImageNet [49] to obtain initialization parameters. Finally, the output results of these two networks are fused into the multi-dimensional semantic features learned by the image fusion module by vector concatenation. The fully connected neural network has three layers, in which the number of neurons is 8,192, 4,096, and the hash code length in turn.

Text network: Bag-of-Words (BoW) is usually used to convert text into vectors, but the sparsity of vectors makes it impossible to fully capture the text's semantic information. Inspired by [28], we adopt a text multi-dimensional fusion module to solve this problem. The text multi-dimensional fusion module extracts the text semantic features in different dimensions through five average pool layers (the scales are 1a, 2a, 3a, 6a, and 10a, where "a" represents the parameter), and uses 1 × 1 convolution layer to integrate multiple features. At the end of this network, there is a three-layer completely connected network to extract the text's hash codes and the numbers of neurons in every layer are 4,096, 4,096, and the hash code length.

3.3 Hash code learning

The performance of the cross-modal hashing model depends on whether generated hash codes can effectively reflect the similarity degree between different modalities. Generally speaking, the Hamming distance of hash codes generated by similar original data should be small, and *vice versa*. To ensure that MDFFH can achieve excellent retrieval performance, we have established an objective function composed of two terms: semantic similarity loss and hashing code quantization loss. We apply $\boldsymbol{P}_{*i} = f(x_i; \theta_x)$ to denote the learned feature from the image network, where θ_x presents the network parameters. Let $\boldsymbol{Q}_{*i} = g(y_i; \theta_y)$ denote the learned feature from the text network, where θ_y refers to the network parameters.

To minimize the semantic gap, we transform different modal data to the same common semantic space to measure similarity. Here, the formula of the likelihood function can be written as follows:

$$p\left(S_{ij}|\boldsymbol{P}_{*i}, \boldsymbol{Q}_{*j}\right) = \begin{cases} \sigma\left(\Phi_{\mathrm{ij}}\right), & S_{ij} = 1 \\ 1 - \sigma\left(\Phi_{\mathrm{ij}}\right), & S_{ij} = 0 \end{cases} \tag{2}$$

In Eq. 2, $\Phi_{ij} = \frac{1}{2}\boldsymbol{P}_{*i}^{T}\boldsymbol{Q}_{*j}$ and $\sigma(\Phi_{ij}) = \frac{1}{1+e^{-\Phi_{ij}}}$. When $S_{ij} = 1$, the inner product of $\boldsymbol{P}_{*i}$ and $\boldsymbol{Q}_{*j}$ will be bigger, which is equivalent to that the two data are more similar. On the contrary, the more dissimilar the two data are when $S_{ij} = 0$.

The maximization of the likelihood function is equal to the maximization of the negative log-likelihood function. To facilitate the training of MDFFH, the above formula can be converted into the following formula:

$$J_{similarity} = -\sum_{i,j=1}^{N}\left(S_{ij}\Phi_{ij} - \log\left(1 + e^{\Phi_{ij}}\right)\right), \tag{3}$$

where $\Phi_{ij} = \frac{1}{2}\boldsymbol{P}_{*i}^{T}\boldsymbol{Q}_{*j}$.

Since the output of the continuous variables from the network is converted into hash binary codes through symbolic functions, there is a certain quantization loss. Therefore, we set the quantization loss term of hash binary codes to reduce this error:

$$J_{quantization} = \|\boldsymbol{H}^{x} - \boldsymbol{P}\|_F^2 + \|\boldsymbol{H}^{y} - \boldsymbol{Q}\|_F^2, \tag{4}$$

where $\boldsymbol{H}^{x} = sign(\boldsymbol{P})$ and $\boldsymbol{H}^{y} = sign(\boldsymbol{Q})$.

From Equations 3, 4, we can get the objective function for optimizing MDFFH as follows:

$$\begin{aligned} \min_{H,\theta_x,\theta_y} J &= J_{similarity} + \eta J_{quantization} \\ &= -\sum_{i,j=1}^{N}\left(S_{ij}\Phi_{ij} - \log\left(1 + e^{\Phi_{ij}}\right)\right) \\ &\quad +\eta\left(\|\boldsymbol{H}^{x} - \boldsymbol{P}\|_F^2 + \|\boldsymbol{H}^{y} - \boldsymbol{V}\|_F^2\right), \end{aligned} \tag{5}$$

In Eq. 5, η denotes the hyper-parameter of the hash code quantization loss. Inspired by Jiang et al. [31], we set $\boldsymbol{H} = \boldsymbol{H}^{x} = \boldsymbol{H}^{y}$ during model training.

3.4 Optimization

Given the discreteness of hash codes, we apply an alternating learning strategy to optimize MDFFH: at one time, only one parameter is optimized while the rest of the parameters are unchanged. In the optimization process, the model parameters are updated by the back-propagation with stochastic gradient descent (SGD). The optimization steps are shown in Algorithm 1. Generally, it includes three steps:

1. Optimize θ_x with θ_y and H fixed.

Select any image data x_i, and obtain the partial derivative of our objective function as following in Eq. 6:

$$\frac{\partial J}{\partial \boldsymbol{P}_{*i}} = \frac{1}{2}\sum_{j=1}^{N}\left(\sigma\left(\Phi_{ij}\right)\boldsymbol{Q}_{*j} - S_{ij}\boldsymbol{Q}_{*j}\right) + 2\eta\left(\boldsymbol{P}_{*i} - \boldsymbol{H}_{*i}\right). \tag{6}$$

Then through the chain derivation rule, we can get $\frac{\partial J}{\partial \theta_x}$ from $\frac{\partial J}{\partial \boldsymbol{P}_{*i}}$ and optimize θ_x according to BP.

2. Optimize θ_y with θ_x and H fixed.

Select any data y_i, and obtain the derivative of the objective function as following in Eq. 7:

$$\frac{\partial J}{\partial \boldsymbol{Q}_{*j}} = \frac{1}{2}\sum_{i=1}^{N}\left(\sigma\left(\Phi_{ij}\right)\boldsymbol{P}_{*i} - S_{ij}\boldsymbol{P}_{*i}\right) + 2\eta\left(\boldsymbol{Q}_{*j} - \boldsymbol{H}_{*j}\right). \tag{7}$$

Then through the chain derivation rule, we can get $\frac{\partial J}{\partial \theta_y}$ from $\frac{\partial J}{\partial \boldsymbol{Q}_{*j}}$ and optimize θ_y according to BP.

3. Optimize hash codes *H*.

The objective function can be converted into the formula as follows:

$$\begin{aligned} &\max_{H} tr\left(\boldsymbol{H}^{T}\left(\eta\left(\boldsymbol{P} + \boldsymbol{Q}\right)\right)\right) = tr\left(\boldsymbol{H}^{T}\boldsymbol{R}\right) = \sum_{i,j} H_{ij}R_{ij}, \\ &s.t.\boldsymbol{H} \in \{-1, +1\}^{k\times N} \end{aligned} \tag{8}$$

In Eq. 8 $\boldsymbol{R} = \eta(\boldsymbol{P} + \boldsymbol{Q})$. At last, the hash code matrix H is updated according to the feature matrixes of images and text as following in Eq. 9:

TABLE 1 MAP scores of different models.

Task	Model	MIRFLICKR-25K				NUS-WIDE			
		16 bits	32 bits	64 bits	Avg	16 bits	32 bits	64 bits	Avg
I → T	CCA	0.5442	0.5693	0.5787	0.5640	0.3743	0.3781	0.3805	0.3776
	CMFH	0.5526	0.5865	0.5907	0.5766	0.4427	0.4527	0.4623	0.4525
	SCM	0.6225	0.6379	0.6508	0.6370	0.4807	0.4845	0.4882	0.4844
	STMH	0.5984	0.6012	0.6074	0.6023	0.4501	0.4623	0.4779	0.4634
	SePH	0.6571	0.6652	0.6717	0.6646	0.5752	0.5838	0.5902	0.5830
	DCMH	0.7413	0.7462	0.7549	0.7474	0.5903	0.6031	0.6093	0.6009
	DDCH	0.7394	0.7450	0.7575	0.7473	0.5971	0.6083	0.6259	0.6104
	DCHUC	0.7118	0.7235	0.7377	0.7243	0.5879	0.5924	0.6068	0.5957
	UCCH	0.7392	0.7441	0.7548	0.7460	0.5942	0.6136	0.6366	0.6148
	OURS	**0.7552**	**0.7675**	**0.7879**	**0.7702**	**0.6077**	**0.6365**	**0.6583**	**0.6341**
T → I	CCA	0.5501	0.5713	0.5791	0.5668	0.378	0.3869	0.3874	0.3841
	CMFH	0.5638	0.5949	0.5972	0.5853	0.4515	0.4548	0.4614	0.4559
	SCM	0.6801	0.6889	0.6941	0.6877	0.4895	0.4917	0.5073	0.4961
	STMH	0.6103	0.6126	0.6215	0.6148	0.4476	0.4587	0.4592	0.4551
	SePH	0.7183	0.7247	0.7278	0.7236	0.5883	0.5943	0.6124	0.5983
	DCMH	0.7632	0.7643	0.7705	0.7660	0.6389	0.6511	0.6571	0.6490
	DDCH	0.7596	0.7662	0.7781	0.7679	0.6332	0.6407	0.6460	0.6399
	DCHUC	0.7107	0.7254	0.7318	0.7226	0.6185	0.6218	0.6253	0.6218
	UCCH	0.7253	0.7268	0.7435	0.7318	0.6442	0.6484	0.6509	0.6478
	OURS	**0.7657**	**0.7705**	**0.7860**	**0.7740**	**0.6478**	**0.6528**	**0.6650**	**0.6552**

The bold values highlight that our algorithm performs better compared to other algorithms and its variant.

$$H = sign(\eta(\boldsymbol{P} + \boldsymbol{Q})). \tag{9}$$

3.5 Out-of-sample extension

The hash codes of the data not used for training are generated by the hash functions learned by MDFFH. For example, given the query image x_q, we can get its hash codes by the hash function as following in Eq. 10:

$$h_q^x = sign\left(f\left(x_q; \theta_x\right)\right) \tag{10}$$

Similarly for text data y_q, we can get its hash codes by the hash function as following in Eq. 11:

$$h_q^y = sign\left(g\left(y_q; \theta_y\right)\right) \tag{11}$$

4 Experiments

Based on two commonly used data sets, namely, MIRFLICKR-25K [50] and NUS-WIDE [51], we conduct a large number of experiments comparing the results with some representative baselines to verify the validity of our model. It is noted that our model can be easily applied to other similar datasets.

4.1 Datasets

MIRFLICKR-25K [50]: There are 25,000 images from the Flickr website in this dataset, and every image has text descriptions and labels, thus forming data pairs. During the experiment, we only retain 20,015 data pairs, because there are too few text descriptions for some data pairs. For each text description, the Bag-of-Word model is applied to convert it into 1386-dimensional vector form, and the corresponding label is transformed into 24-dimensional vector form. 2000 data pairs are randomly selected for querying and the rest for retrieval. For model training, we select 10,000 data pairs from retrieval.

NUS-WIDE [51]: There are 269,648 data pairs in this dataset, and each includes images, text descriptions, and data labels. There is a total of 81 categories of original data labels in this dataset. We selected 21 of the most common data labels as the experimental dataset and finally retained 195,834 data pairs after processing. Text descriptions and data labels in each data pair are converted into 1,000 and 21-dimensional vector forms through the Bag-of-Word model. The partition of different sets for model training in this dataset is consistent with the MIRFLICKR-25 dataset.

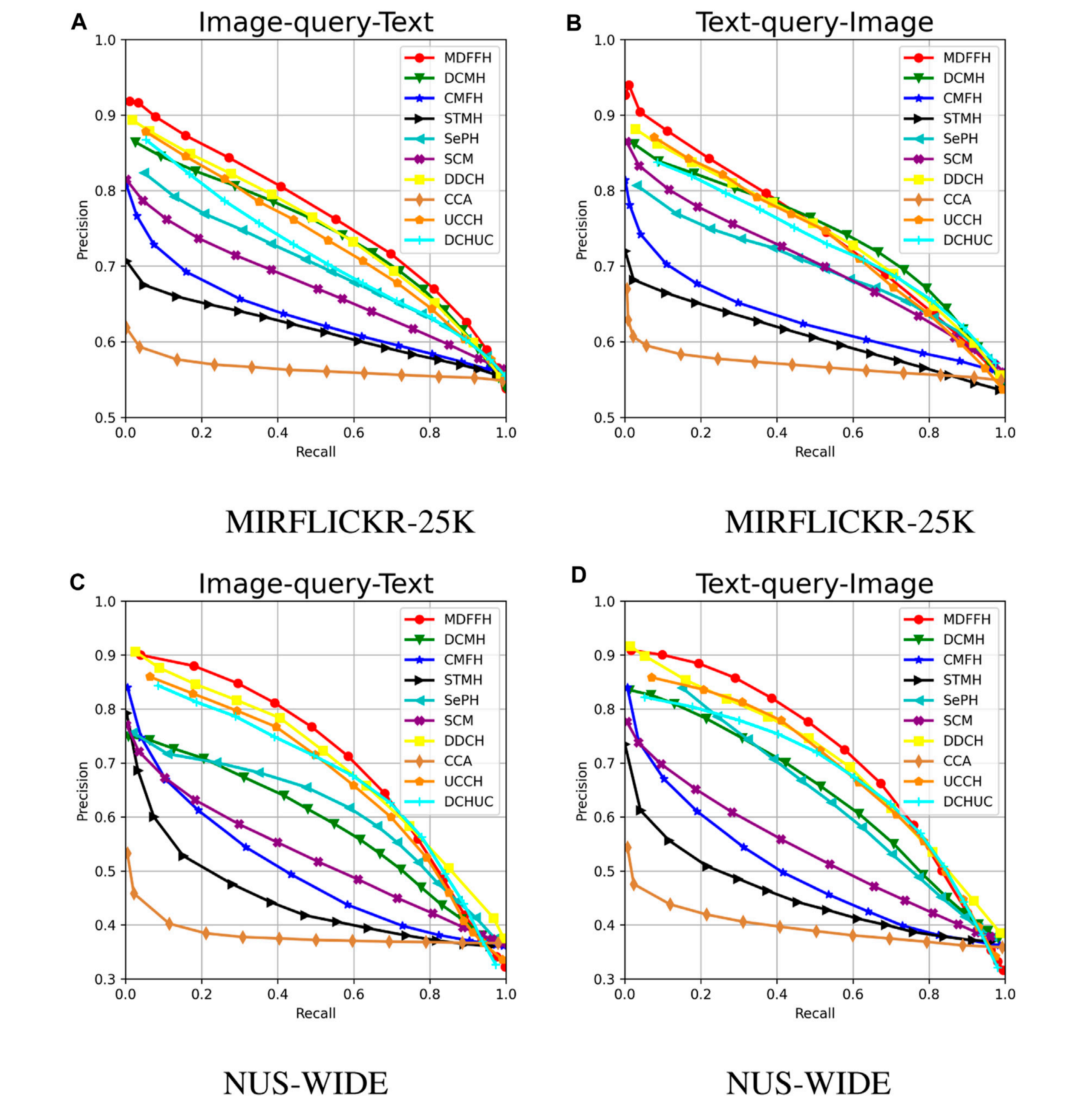


FIGURE 3
The PR curves with code length 16. **(A)** MIRFLICKR-25K. **(B)** MIRFLICKR-25K. **(C)** NUS-WIDE. **(D)** NUS-WIDE.

4.2 Evaluation and baselines

Evaluation: For cross-modal retrieval, researchers usually study two typical tasks: retrieving text with images and retrieving images with text.

To evaluate MDFFH's performance, we select the two most commonly used evaluation criteria, namely, the Precision-Recall (PR) Curve and Mean Average Precision (MAP) [52]. The average accuracy (AP) of any query data is calculated as follows:

$$AP = \frac{1}{K}\sum_{s=1}^{M} U(s)V(s), \tag{12}$$

where K and M are the numbers of retrieved relevant data and the retrieval set, $U(s)$ denotes the proportion of the first s retrieved data related to the query data, and $V(s)$ shows whether the retrieved sth data is related to the query data, which can be judged by the category label. If two data are related, $V(s) = 1$, otherwise, $V(s) = 0$. The MAP value can be calculated by averaging the APs of all query data and is positively correlated with model performance.

In addition, the PR curve is another indicator for evaluating the model performance. The performance can be directly judged by drawing a PR curve of this model: if the area under this curve is larger, the model performance is better. Moreover, the

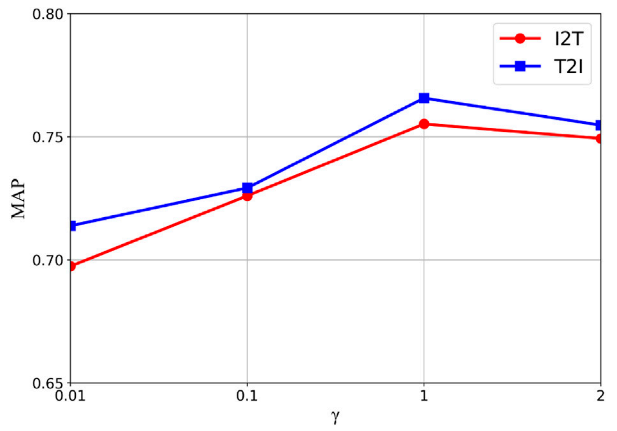


FIGURE 4
The sensitivity analysis of the hyper-parameter.

corresponding recall and precision can be obtained by altering the Hamming radius and drawing the PR curve.

Baselines: We compare our MDFFH with nine representative models, which are CCA, CMFH, SCM, STMH, SePH, DCMH, DDCH, DCHUC, and UCCH. The first four models belong to hand-crafted models and the rest are deep network models.

4.3 Implementation details

We use PyTorch, which is a deep-learning framework based on dynamic tensors, to implement our MDFFH on the NVIDIA RTX 3090 server and the iteration number is set to 300. In the iteration, the learning rate gradually decreases from 0.03 initialized to 10^{-6}. The hyper-parameter η is set to 1, and the detailed parameter analysis is in the section Parameter Analysis. For each model result, experiments have been run five times and the average value is obtained as a representative.

4.4 Performance

The MAP scores of MDFFH and nine baseline models based on two general datasets are shown in Table 1, where "I → T" represents from image to retrieve text and "T → I" represents from text to retrieve images. We can find that for hash codes with different lengths, our model is superior to baseline models. For example, when we select the MIRFLICKR-25K dataset, compared with DCMH which is the most representative deep cross-modal hashing model, MDFFH on "I → T" tasks increased by 3.05% on average, and its MAP score on text retrieval image tasks increased by 1.04% on average. On the NUS-WIDE dataset, compared with DCMH, MDFFH's MAP score on image retrieval text tasks increased by 5.52% on average, and its MAP score on text retrieval image tasks increased by 0.95% on average. In particular, compared with these five hand-crafted baseline models, MDFFH has been greatly improved. This proves that better performance can be achieved by integrating feature learning and the generation of hash codes into a unified end-to-end network. At the same time, MDFFH has a better performance compared with DCMH and DDCH. The reason is that DCMH and DDCH generate hash codes only using single-dimensional semantic features, ignoring the information complementation between multi-dimensional semantic features, which has certain limitations. On the contrary, MDFFH applies the image multi-dimensional fusion module and the text multi-dimensional fusion module to get the multi-dimensional semantic features of different modal data, which can mine richer semantic associations and establish more accurate modal relationships, thus helping to narrow the modal gap to greatly improve the retrieval accuracy.

TABLE 2 The MAP scores of MDFFH and its variant.

Task	Method	MIRFLICKR-25K		
		16bits	32bits	64bits
I → T	MDFFH	**0.7552**	**0.7675**	**0.7879**
	MDFFH-1	0.7521	0.7587	0.7649
T → I	MDFFH	**0.7657**	**0.7705**	**0.7860**
	MDFFH-1	0.7567	0.7606	0.7692

The bold values highlight that our algorithm performs better compared to other algorithms and its variant.

When the hash code length is set to 16 bits, the PR curves of MDFFH and baseline models under MIRFLICKR-25K and NUS-WIDE datasets are demonstrated in Figure 3. For PR curves of different models, which curve has a larger area represents better performance. From this figure, it is clear that the performance of MDFFH outperforms other baselines, which is consistent with the application of MAP as a performance evaluation index.

4.5 Parameter analysis

The influence of hyper-parameter values in the model based on the MIRFLICKR-25K dataset is studied in this section. The hash code length is uniformly 16 bits and the experimental results are shown in Figure 4. The MAP scores of two cross-modal retrieval tasks change with the hyper-parameter. During the manual adjustment of the hyper-parameter, the range of values is 0.01, 0.1, 1, and 2. The experimental results demonstrate the MDFFH performance can reach the best under the setting of $\gamma = 1$. The initial values of other network parameters are randomly generated and then determined through network learning.

4.6 Ablation study

We have designed one variant and carried out experiments to verify whether the innovative module in MDFFH improves the overall performance. MDFFH-1 is a variant of MOFFH without a Vision Transformer. The variant aims to check the important influence of the innovative image multi-dimensional fusion module on our model's retrieval performance. Table 2 shows

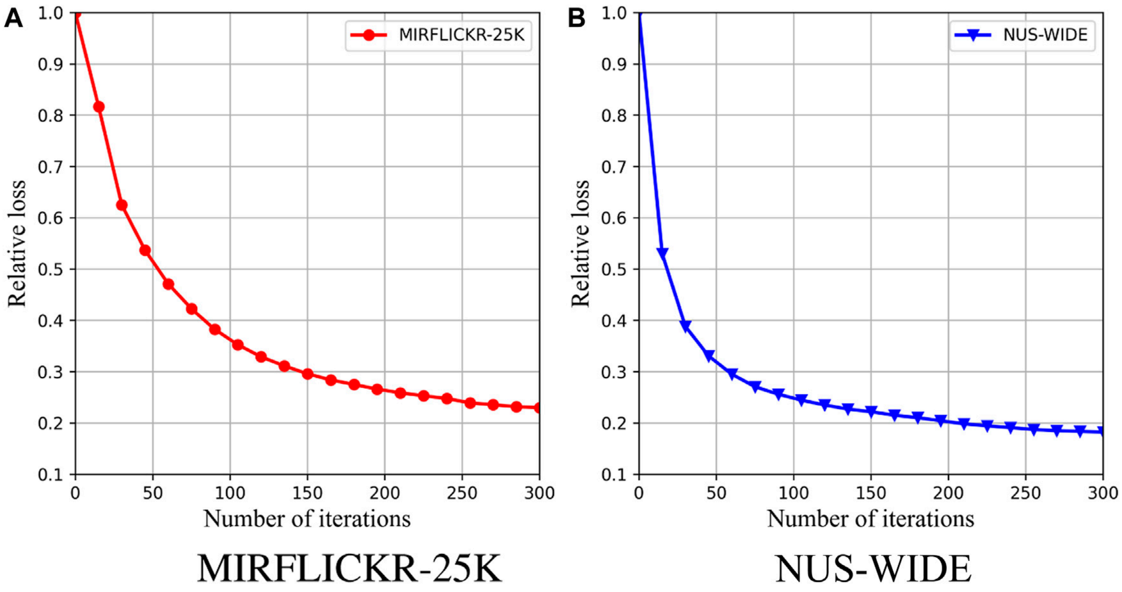


FIGURE 5
The convergence curve of MDFFH. **(A)** MIRFLICKR-25K. **(B)** NUS-WIDE.

the comparative results. From this table, it is clear that MDFFH's performance is better than MDFFH-1's performance on the MIRFLICKR-25K dataset because of the effective role of the image multi-dimensional fusion module. The image multi-dimensional fusion module effectively combines the global image information concerned by the Vision Transformer with the local image information concerned by the convolutional neural network to generate more representative multi-dimensional semantic features. This can more effectively get the semantic similarity between different data to learn more accurate hash mapping functions, and so improve our model performance.

4.7 Convergence analysis

For analyzing MDFFH's convergence, experiments are conducted on MIRFLICKR-25K and NUS-WIDE datasets. During the experiment, the hash code length is 16 bits and the relative loss is used as an evaluation criterion. The relative loss of the *ith* iteration is the ratio of the loss function value of the *ith* iteration divided by the loss function value of the first iteration and the experimental results are shown in Figure 5. With the number of iterations increasing, the relative loss value decreases rapidly and becomes stable, which means our optimization algorithm is effective.

5 Conclusion

A new cross-modal hashing model named MDFFH is proposed from the perspective of multi-dimensional semantic features. The image multi-dimensional fusion module constructed effectively combines the convolutional neural network and Vision Transformer and can generate multi-dimensional semantic features of images with richer semantic information. Similarly, we apply the text multi-dimensional fusion module to generate more representative text multi-dimensional semantic features, which provides a basis for mining richer semantic associations and building more accurate modal relationships, thus making the generated hash code more semantic. Experimental analysis of two general datasets can verify that our MDFFH model improves the performance of cross-modal retrieval. In future work, we will attempt to investigate its applications in the field of multimodal generation, multimodal question answering, and health and medical big data retrieval.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

DR: Conceptualization, Data curation, Formal Analysis, Investigation, Resources, Supervision, Writing–review and editing. WX: Writing–original draft and Writing–review and editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

Author WX was employed by the company Industrial and Commercial Bank of China.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Li Q, Li L, Li Y. Developing ChatGPT for biology and medicine: a complete review of biomedical question answering. *Biophys Rep* (2024) 9:1–20. doi:10.52601/bpr.2024.240004

2. Mandal RC, Kler R, Tiwari A, Keshta I, Abonazel MR, Tageldin EM, et al. Enhancing stock price prediction with deep cross-modal information fusion network. *Fluctuation Noise Lett* (2024) 23(02). doi:10.1142/s0219477524400170

3. Ma Y, Yu C, Yan M, Sangaiah AK, Wu Y. Dark-side avoidance of mobile applications with data biases elimination in socio-cyber world. *IEEE Trans Comput Soc Syst* (2023) 1–10. doi:10.1109/TCSS.2023.3264696

4. Gionis A, Indyk P, Motwani R. *Similarity search in high dimensions via hashing* (2000).

5. Luo K, Zhang C, Li H, Jia X, Chen C. Adaptive marginalized semantic hashing for unpaired cross-modal retrieval. *IEEE Trans Multimedia* (2022) 25:9082–95. doi:10.1109/tmm.2023.3245400

6. Kebaili A, Lapuyade-Lahorgue J, Su R. Deep learning approaches for data augmentation in medical imaging: a review. *J Imaging* (2023) 9(4):81. doi:10.3390/jimaging9040081

7. Wang J, Shen HT, Song J, Ji J. *Hashing for similarity search: a survey* (2019). *arXiv preprint* 2019, arXiv:1408.2927.

8. Su MY, Gu GH, Ren XL, Fu H, Zhao Y. Semi-supervised knowledge distillation for cross-modal hashing. *IEEE Trans Multimedia* (2023) 25:662–75. doi:10.1109/TMM.2021.3129623

9. Long J, Sun L, Hua L, Yang Z. Discrete semantics-guided asymmetric hashing for large-scale multimedia retrieval. *Appl Sci* (2021) 11:8769. doi:10.3390/app11188769

10. Yao D, Li ZX, Li B, Zhang CL, Ma HF. Similarity graph-correlation reconstruction network for unsupervised cross-modal hashing. *Expert Syst Appl* (2023) 237:121516. doi:10.1016/j.eswa.2023.121516

11. Lu X, Zhang HX, Sun JD, Wang ZH, Guo PL, Wan WB. Discriminative correlation hashing for supervised cross-modal retrieval. In: *Signal processing: image communication* (2018).

12. Shen F, Shen C, Liu W, Shen HT. Supervised discrete hashing. In: *Proceedings of the 2015 IEEE conference on computer vision and pattern recognition*. Boston, MA, USA: CVPR (2015).

13. Song J, Yang Y, Huang Z, Yang Y, Shen HT. Inter-media hashing for large-scale retrieval from heterogeneous data sources. In: *Proceedings of the ACM SIGMOD* (2013). p. 785–96.

14. Hong L, Ji R, Wu Y, Huang F, Zhang B. Cross-modality binary code learning via fusion similarity hashing. In: *Proceedings of the 2017 IEEE conference on computer vision and pattern recognition*. Honolulu, HI, USA: CVPR (2017).

15. Ren D, Xu W, Wang Z, Sun Q. Deep label feature fusion hashing for cross-modal retrieval. *IEEE Access* (2022) 10:100276–85. doi:10.1109/access.2022.3208147

16. Zou X, Wang X, Bakker EM, Wu S. Multi-label semantics preserving based deep cross-modal hashing. *Signal Processing: Image Communicationbr* (2021) 93:116131. doi:10.1016/j.image.2020.116131

17. Qiang H, Wan Y, Liu Z, Xiang L, Meng X. Discriminative deep asymmetric supervised hashing for cross-modal retrieval. *Knowledge-Based Syst* (2020) 204:106188. doi:10.1016/j.knosys.2020.106188

18. Jin M, Zhang HX, Zhu L, Sun JD, Liu L. Coarse-to-fine dual-level attention for video-text cross-modal retrieval. *Knowledge-Based Syst* (2022) 242:108354. doi:10.1016/j.knosys.2022.108354

19. Wang Z, Wang M, He P, Xu J, Lu G. Unsupervised cross-modal retrieval based on deep convolutional neural networks. In: *2022 4th international conference on advances in computer technology, information science and communications (CTISC)* (2022).

20. Wang T, Zhu L, Cheng ZY, Li JJ, Gao Z. Unsupervised deep cross-modal hashing with virtual label regression. *Neurocomputing* (2020) 386:84–96. doi:10.1016/j.neucom.2019.12.058

21. Hotelling H. Relations between two sets of variates. *Breakthroughs Stat* (1992) 162–90.

22. Ding G, Guo Y, Zhou J. Collective matrix factorization hashing for multimodal data. In: *Proceedings of the 2014 IEEE conference on computer vision and pattern recognition*. Columbus, OH, USA: CVPR (2014).

23. Zhou J, Ding G, Guo Y. Latent semantic sparse hashing for cross-modal similarity search. In: *Proceedings of the ACM SIGIR* (2014). p. 415–24.

24. Xie L, Zhu L, Yan P. Cross-Modal Self-Taught Hashing for large-scale image retrieval. In: *Signal processing*. 124. The Official Publication of the European Association for Signal Processing (2016).

25. Wang D, Gao X, Wang X, He L. Semantic topic multimodal hashing for cross-media retrieval. In: *Proceedings of the 2015 international joint conference on artificial intelligence*. Buenos Aires Argentina: IJCAI (2015).

26. Zhen Y, Gao Y, Yeung D, Zha H, Li X. Spectral multimodal hashing and its application to multimedia retrieval. *IEEE Trans Cybernetics* (2016) 46:27–38. doi:10.1109/tcyb.2015.2392052

27. Zhang D, Li W. Large-scale supervised multimodal hashing with semantic correlation maximization. In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. Québec City, Québec Canada: AAAI (2014).

28. Lin Z, Ding G, Hu M, Wang J. Semantics-preserving hashing for cross-view retrieval. In: *Proceedings of the 2015 IEEE conference on computer vision and pattern recognition*. Boston, MA, USA: CVPR (2015).

29. Qi XJ, Zeng XH, Wang SM, Xie YC, Xu LM. Cross-modal variable-length hashing based on hierarchy. *Intell Data Anal* (2021) 25(3):669–85. doi:10.3233/IDA-205162

30. Chen Y, Zhang H, Tian Z, Wang D, Li X. Enhanced discrete multi-modal hashing: more constraints yet less time to learn. *IEEE Trans Knowledge Data Eng* (2022) 34:1177–90. doi:10.1109/tkde.2020.2995195

31. Jiang Q, Li W. Deep cross-modal hashing. In: *Proceedings of the 2017 IEEE conference on computer vision and pattern recognition*. Honolulu, HI, USA: CVPR (2017).

32. Yang E, Deng C, Liu W, Liu X, Tao D, Gao X. Pairwise relationship guided deep hashing for cross-modal retrieval. In: *Proceedings of the 2017 association for the advancement of artificial intelligence*. San Francisco, California, USA: AAAI (2017).

33. Ji Z, Yao W, Wei W, Song H, Pi H. Deep multi-level semantic hashing for cross-modal retrieval. *IEEE Access* (2019) 7:23667–74. doi:10.1109/access.2019.2899536

34. Lin Q, Cao W, He Z, He Z. Mask cross-modal hashing networks. *IEEE Trans Multimedia* (2020) 14:550–8. doi:10.1109/tmm.2020.2984081

35. Li C, Deng C, Li N, Liu W, Gao X, Tao D. Self-supervised adversarial hashing networks for cross-modal retrieval. In: *Proceedings of the 2018 IEEE conference on computer vision and pattern recognition*. Salt Lake City, UT, USA: CVPR (2018).

36. Li J. Deep semantic cross-modal hashing based on graph similarity of modal-specific. *IEEE Access* (2021) 9:96064–75. doi:10.1109/access.2021.3093357

37. Liu X, Zeng H, Shi Y, Zhu J, Ma K. Deep Rank cross-modal hashing with semantic consistent for image-text retrieval. In: *IEEE international conference on acoustics, speech and signal processing* (2022). p. 4828–32.

38. Zhu X, Cai L, Zou Z, Zhu L. Deep multi-semantic fusion-based cross-modal hashing. *Mathematics* (2022) 10:430–20. doi:10.3390/math10030430

39. Xie Y, Zeng X, Wang T, Xu L, Wang D. Multiple deep neural networks with multiple labels for cross-modal hashing retrieval. *Eng Appl Artif Intelligence* (2022) 114:105090. doi:10.1016/j.engappai.2022.105090

40. Yu E, Ma J, Sun J, Chang X, Zhang H, Hauptmann AG. Deep discrete cross-modal hashing with multiple supervision. *Neurocomputing* (2022) 486:215–24. doi:10.1016/j.neucom.2021.11.035

41. Tu R, Mao X, Ma B, Hu Y, Yan T, Huang H, et al. Deep cross-modal hashing with hashing functions and unified hash codes jointly learning. *IEEE Trans Knowledge Data Eng* (2022) 34:560–72. doi:10.1109/tkde.2020.2987312

42. Hu P, Zhu H, Lin J, Peng D, Zhao Y, Peng X. Unsupervised contrastive cross-modal hashing, *IEEE Trans Pattern Anal Mach Intell* (2023) 45:3877–89. doi:10.1109/TPAMI.2022.3177356

43. Tay Y, Dehghani M, Bahri D, Metzler D. *Efficient transformers: a survey* (2009). arXiv: 2009.06732.

44. Mnih V, Heess N, Graves A, Kavukcuoglu K. Recurrent models of visual attention. *Adv Neural Inf Process Syst* (2014) 27:2204–12. doi:10.48550/arXiv.1406.6247

45. Bahdanau D, Cho K, Bengio Y. *Neural machine translation by jointly learning to align and translate* (2014). arXiv preprint 2014, arXiv: 1409.0473.

46. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A, et al. *Attention is all you need* (2014). arXiv preprint 2017, arXiv: 1706.03762.

47. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, et al. *An image is worth 16x16 words: transformers for image recognition at scale* (2020). arXiv preprint 2020, arXiv: 2010.11929v2.

48. Chatfield K, Simonyan K, Vedaldi A, Zisserman A. *Return of the devil in the details: delving deep into convolutional nets* (2014). arXiv preprint 2014, arXiv: 1405.3531.

49. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, et al. ImageNet large scale visual recognition challenge. *Int J Comput Vis* (2015) 115:211–52. doi:10.1007/s11263-015-0816-y

50. Huiskes MJ, Lew MS. The MIR Flickr retrieval evaluation. In: *Proceedings of the 1st ACM international conference on Multimedia information retrieval* (2008). p. 39–43.

51. Chua T, Tang J, Hong R, Li H, Luo Z, Zheng Y. NUS-WIDE: a real-world web image database from the National University of Singapore. In: *Proceedings of the ACM international conference on image and video retrieval* (2009). p. 1–9.

52. Liu W, Mu C, Kumar S, Chang S. Discrete graph hashing. *Adv Neural Inf Process Syst* (2014) 4:3419–27.

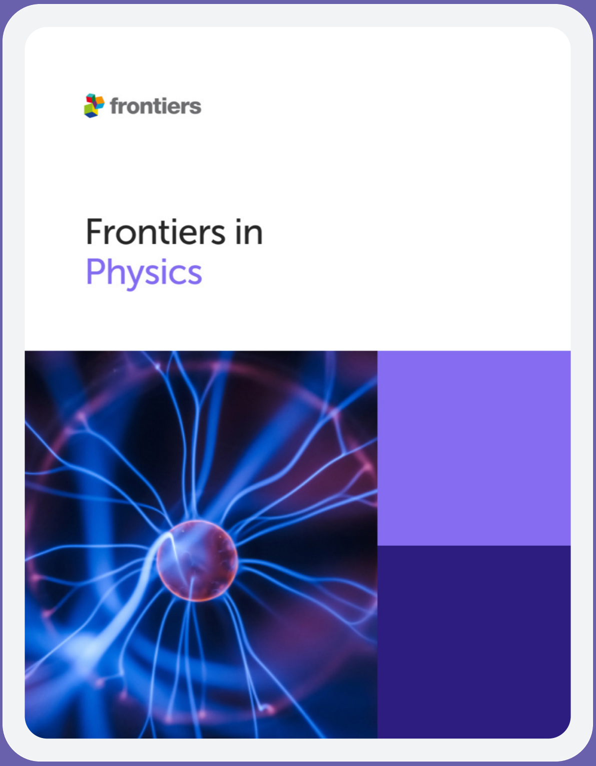
Frontiers in
Physics

Investigates complex questions in physics to understand the nature of the physical world

Addresses the biggest questions in physics, from macro to micro, and from theoretical to experimental and applied physics.

Discover the latest Research Topics

See more →



frontiers

Frontiers in
Physics

Frontiers
Avenue du Tribunal-Fédéral 34
1005 Lausanne, Switzerland
frontiersin.org

Contact us
+41 (0)21 510 17 00
frontiersin.org/about/contact