frontiers | Research Topics

Machine learning for biological sequence analysis

Edited by
Quan Zou and Zhibin Lv

Published in
Frontiers in Genetics

ISSN 1664-8714
ISBN 978-2-83251-601-0
DOI 10.3389/978-2-83251-601-0

Machine learning for biological sequence analysis

Topic editors
Quan Zou — University of Electronic Science and Technology of China, China
Zhibin Lv — Sichuan University, China

Topic Coordinators
Yansu Wang — University of Electronic Science and Technology of China, China

Citation
Zou, Q., Lv, Z., eds. (2023). *Machine learning for biological sequence analysis*. Lausanne: Frontiers Media SA. doi: 10.3389/978-2-83251-601-0

Table of contents

TYPE Editorial
PUBLISHED 03 February 2023
DOI 10.3389/fgene.2023.1150688

*CORRESPONDENCE
Quan Zou,
zouquan@nclab.net

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 24 January 2023
ACCEPTED 30 January 2023
PUBLISHED 03 February 2023

CITATION
Lv Z, Li M, Wang Y and Zou Q (2023), Editorial: Machine learning for biological sequence analysis.
Front. Genet. 14:1150688.
doi: 10.3389/fgene.2023.1150688

Editorial: Machine learning for biological sequence analysis

Zhibin Lv[1], Mingxin Li[1], Yansu Wang[2] and Quan Zou[2,3]*

[1]College of Biomedical Engineering, Sichuan University, Chengdu, China, [2]Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, China, [3]Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou, China

KEYWORDS

machine learning (ML), biosequence analysis, biomarker, BERT classification algorithm, single-cell, imblanced data

Editorial on the Research Topic
Machine learning for biological sequence analysis

1 Introduction

Biomacromolecules, primarily proteins, DNA, and RNA, are crucial for vital physiological processes. Biomacromolecules can generally be represented by sequences, comprising series of strings, which are referred to as bio-sequences and represent the primary structures of proteins, DNA, and RNA. The development of sequencing technologies, particularly next-generation sequencing and tandem mass spectrometry, has led to the production of vast amounts of bio-sequence data. It is established that structure generally determines function; in particular, determination of tertiary structure is critical for functional analysis of biomacromolecules. Nevertheless, determination of the tertiary structure of bio-sequences is relatively difficult; therefore, directly obtaining functional information from primary structures (i.e., bio-sequences) is extremely challenging and an important problem that requires urgent resolution.

Machine learning technologies, based on statistical theory and data mining, provide new tools for bio-sequence analysis that are effective for biological function analysis of genes and proteins, as well as determining relationships between primary structure and function. There are two basic problems in the use of machine learning in this context that have yet to be satisfactorily resolved. One is how to extract sufficient and effective features from bio-sequences. Usually, bio-sequences comprise series of strings that must be converted into numerical vectors before input into machine learning models, a process referred to as feature extraction. Only by effectively extracting the hidden numerical features in primary structure sequences can they be successfully mined by the machine learning model and achieve optimal function recognition. The other problem is that of data imbalance, which refers to the fact that the ratio of positive to negative sample sequences are not 1 to 1; in actual application, there are generally fewer positive than negative samples. To obtain the best results, machine learning models often need to be trained with balanced data, and unbalanced data will greatly affect training of machine learning models and their application in real-world scenarios. At present, some methods have been proposed to solve the problem of data imbalance, but they still cannot satisfactorily solve this fundamental issue. In this Research Topic, we focus on the two challenges described above, as well as collating the results of recent research on related Research Topic. The total of 12 articles can be divided into three categories, as follows:

9 papers on the identification of functions and interactions based on bio-sequences, 1 paper on a bioinformatics tool recommendation platform, and 2 papers on biomarker mining and analysis.

2 Identification of functions and interactions of bio-sequences

Identification of the biological functions of macromolecular sequences directly from their primary structures has been a hotspot in the application of machine learning. We collected 9 papers related to this Research Topic, which explore a variety of feature extraction techniques and machine learning methods for different bio-sequences, and achieved the most advanced accuracy in corresponding function identification.

Sucrose transporter (SUT) is a transmembrane protein that occurs widely in plant species and has important roles in sucrose transport and sucrose-specific signal transduction. Chen et al. built a model named ISTRF, based on a random forest algorithm, to identify SUT proteins by constructing an in-house dataset comprising SUT and non-SUT sequences, then using feature extraction tools including: protein amino acid composition, transition, and distribution; position-specific scoring matrix (PSSM) composition; and k-separated-bigrams-PSSM. They also applied the Borderline-SMOTE algorithm to solve the problem of data imbalance. ISTRF achieved an independent test accuracy of 96.1%.

Moonlighting proteins are present in many animals, plants, and microorganisms and play important roles in signal transduction, cell growth and motility, tumor suppression, DNA synthesis and repair, and metabolism of biological macromolecules. Chen et al. used linear discriminant analysis (LDA) and a support vector machine ensemble with bagging (bagging-SVM) to build a bioinformatics tool that can effectively identify moonlighting proteins. The tool uses three embedded features to encode proteins, a linear discriminant method for feature selection, and a SVM as the classifier. The authors found that the LDA method can effectively screen out sequence features identifying moonlight proteins, and that the bagging-SVM is superior to a classic SVM algorithm, achieving accuracy of 93.25%.

As a good substitute for antibiotics, antimicrobial peptides (AMPs) can effectively kill bacteria in organisms, resulting numerous therapeutic effects, such as antibacterial, wound healing, antioxidant, and immune regulation activities. Dong et al. proposed a deep learning model that fuses multiple sequence feature representations (four types) as input to identify AMPs. They adopted a convolutional layer structure and fully connected layers to construct a deep learning network; model accuracy was 97.8%.

Vesicle transporters are membrane proteins that function by regulating the interaction of specific molecules with vesicle membranes. Fan et al.established a hypergraph regularized K-local hyperplane distance nearest neighbor machine learning model to distinguish vesicle transporters from non-vesicle transporters [4]. The sequence encoding feature used by this model is PsePSSM. The research showed that the classifier outperformed traditional classifiers and achieved an accuracy of approximately 84%.

Protein-protein interactions (PPIs) are fundamental to deep understanding of proteome functional mechanisms and are highly valuable in medical applications of novel diagnostic and therapeutic targets. Yang et al. developed a new tool for PPI data and functional analysis [5]. A key feature of the tool is that each protein involved in a PPI is encoded using Gene Ontology and Kyoto Encyclopedia of Genes and Genomes (KEGG) pathway annotations. The tool uses minimum absolute shrinkage and selection operators, gradient boosting machines, maximum correlation, and minimum redundancy to rank features for importance analysis. Then, the most significant features were selected as critical functional items identified by PPI.

Pseudouridine is an abundant RNA modification that can affect RNA stability and immunoreducibility, among other characteristics, and its mutation is associated with numerous malignancies, including lung and gastric cancers. Zhang et al.proposed a new machine learning model, PseU-ST, to identify RNA pseudo-uridine modification sites in *Homo sapiens*, *Saccharomyces cerevisiae*, and *Mus musculus*. They used six feature extraction methods to encode RNA, and chi-square analysis for feature selection. In addition, stacking ensemble learning was applied. The accuracy values of PseU-ST for data from *H. sapiens*, *S. cerevisiae*, and *M. musculus* were approximately 94%, 88%, and 89%, respectively.

Inspired by the hypothesis that pathogen-derived immunological epitopes can mediate CD8$^+$ T cell-associated host adaptive immune responses, Hu et al. used available positive and negative CD8$^+$ T cell epitope (TCE) data to propose a novel predictor, CD8TCEI-EukPath, to detect CD8$^+$ TCEs in eukaryotic pathogens. The authors aimed to develop a method to enable rapid screening of epitope-based vaccine candidates. CD8TCEI-EukPath integrated three hybrid features, adopted an MRMD tool for feature selection, and used a LightGBM classifier to distinguish CD8$^+$ TCEs from non-CD8$^+$ TCEs, thereby achieving accuracy values of approximately 79% and 78% in cross-validation and independent testing, respectively.

The success of a transformer model with a unique self-attention mechanism in natural language processing inspired Mai et al. to use it to predict and analyze promoters in *Synechococcus* sp. and *Synechocystis* sp. They named the tool, TSSNote-CyaPromBERT, and it facilitates large dataset extraction, model training, and promoter prediction from public dRNA-seq datasets. The model of TSSNote-CyaPromBERT achieved an area under the receiver operating curve value of 0.92 for distinguishing promoter and non-promoter nucleotides, as well as relatively good performance in cross-species verification testing. Monte Carlo sampling and attention score visualizations can be used to explain the model behavior.

Histone modifications affect various chromatin-dependent processes, including DNA replication, repair, and transcription. Chen et al. proposed a new deep learning model, TransferChrome, with the aim of solving the problem of inaccurate gene expression prediction across cell lines, based on use of a self-attention mechanism to predict the effects of histone modifications on gene expression, and used a transfer learning model to achieve cross-cell gene expression prediction. The authors trained and tested TransferChrome on 56 different cell lines from the REMC database, and achieved a mean area under the curve score of 84.79%.

3 Mining of disease-related markers

COVID-19 triggers a complex immune response, where CD8$^+$ T cells play a particularly important role in controlling disease severity. The mechanisms underlying the regulatory effects of CD8$^+$ T cells on COVID-19 remain poorly studied. Lu et al. applied single-cell omics data to target three CD8$^+$ T cell subtypes and three COVID-19 disease states, using CD8$^+$ T single cell data expression profiles, combined with multiple feature selection methods, to screen out biomarkers, including ZFP36, DUSP1, TCR, and IL7R, among other molecules. They proposed that these genes can be confirmed to play an immunomodulatory role in the processes of infection with and recovery from COVID-19 disease. Simultaneously, the authors used the characteristics of CD8$^+$ T cell subtypes to establish a machine learning model that can distinguish COVID-19 disease severity.

Discovering tumor markers related to cancer has long been a focus of considerable research attention. Zhao et al. analyzed the expression level of KRAS, a signal transduction protein that binds to GTP in the MAPK pathway. To assess the tumor microenvironment, they used 22 immune-infiltrating cell expression datasets to calculate immune and stromal scores. They also used 33 tumor expression datasets to construct a PPI network by establishing KRAS, immune checkpoint, and interacting genes. By performing gene set enrichment analysis, they generated results suggesting that KRAS may be a reliable prognostic biomarker for diagnosing patients with cancer that can be incorporated into tumor-targeted drugs.

4 An online platform for recommendation of bioinformatics tools

How to choose a suitable tool for structural variation analysis of bio-sequence data is a particularly interesting problem. Numerous bioinformatics tools have been developed, but their applicability to real data and universality are serious concerns, and it is unrealistic to test each tool individually. Wang et al. noticed this problem and developed a meta-learning framework to establish the relationship between data features and bioinformatics tool performance. Using random forest analysis, the authors identified the relationships between 8 selected data features and the optimal caller, and used these relationship to recommend callers. Testing the algorithm of the automatic recommendation tool constructed showed that the applicable samples varied among different callers. Hence, different tools are often suitable for various types of bio-sequencing data analyses. The accuracy of recommended tools was maintained above a mean of 80%, which is far superior to random selection or fixed selection strategies. The authors also built an online website and provided the source code.

In conclusion, the papers discussed in this Research Topic demonstrate significant roles for machine learning techniques in various bio-sequence analysis applications, and we sincerely hope that this Research Topic will be widely read and benefit readers. In particular, this Research Topic collates insightful explanations and applications that can contribute to developments and advances in biology. Finally, we wish to convey our appreciation for all the efforts of the authors, reviewers, and staff of the *Frontiers in Genetics* editorial office.

Author contributions

ZL and ML wrote the manuscript. YW and QZ edited the manuscript. QZ supervised the program.

Funding

The work was supported by the National Natural Science Foundation of China (No. 62250028, No. 62131004, and No. 62001090), the Sichuan Provincial Science Fund for Distinguished Young Scholars (2021JDJQ0025), and the Municipal Government of Quzhou (No. 2022D040).

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

ORIGINAL RESEARCH
published: 13 July 2022
doi: 10.3389/fgene.2022.960388

Identification of Vesicle Transport Proteins *via* Hypergraph Regularized K-Local Hyperplane Distance Nearest Neighbour Model

Rui Fan[1,2†], *Bing Suo*[3†] *and Yijie Ding*[2*]

[1]*Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, China,* [2]*Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou, China,* [3]*Beidahuang Industry Group General Hospital, Harbin, China*

The prediction of protein function is a common topic in the field of bioinformatics. In recent years, advances in machine learning have inspired a growing number of algorithms for predicting protein function. A large number of parameters and fairly complex neural networks are often used to improve the prediction performance, an approach that is time-consuming and costly. In this study, we leveraged traditional features and machine learning classifiers to boost the performance of vesicle transport protein identification and make the prediction process faster. We adopt the pseudo position-specific scoring matrix (PsePSSM) feature and our proposed new classifier hypergraph regularized k-local hyperplane distance nearest neighbour (HG-HKNN) to classify vesicular transport proteins. We address dataset imbalances with random undersampling. The results show that our strategy has an area under the receiver operating characteristic curve (AUC) of 0.870 and a Matthews correlation coefficient (MCC) of 0.53 on the benchmark dataset, outperforming all state-of-the-art methods on the same dataset, and other metrics of our model are also comparable to existing methods.

Keywords: transport proteins, protein function prediction, hypergraph learning, local hyperplane, membrane proteins

OPEN ACCESS

Edited by:
Zhibin Lv,
Sichuan university, China

Reviewed by:
Changli Feng,
Taishan University, China
Liangzhen Jiang,
Chengdu University, China

Correspondence:
Yijie Ding
wuxi_dyj@csj.uestc.edu.cn

†These authors have contributed equally to this work

Specialty section:
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

Received: *02 June 2022*
Accepted: *22 June 2022*
Published: *13 July 2022*

Citation:
Fan R, Suo B and Ding Y (2022) Identification of Vesicle Transport Proteins via Hypergraph Regularized K-Local Hyperplane Distance Nearest Neighbour Model. Front. Genet. 13:960388. doi: 10.3389/fgene.2022.960388

1 INTRODUCTION

Proteins are the basis of most life activities and perform important functions in different biochemical reactions. Proteins with different amino acid sequences and folding patterns have different functions. Understanding the factors that influence protein function has practical biological implications. Therefore, protein function prediction has been an important topic since the birth of bioinformatics. In recent years, machine learning-based protein function prediction methods have been widely used in many studies (Shen et al., 2019; Zhang J. et al., 2021; Zulfiqar et al., 2021; Ding et al., 2022b; Zhang et al., 2022), such as drug discovery (Ding et al., 2020c; Chen et al., 2021; Song et al., 2021; Xiong et al., 2021), protein gene ontology (Hong et al., 2020b; Zhang W. et al., 2021), DNA-binding proteins (Zou et al., 2021), enzyme proteins (Feehan et al., 2021; Jin et al., 2021), and protein subcellular localization (Ding et al., 2020b; Su et al., 2021; Wang et al., 2021; Zeng et al., 2022). In this study, we propose a novel method to identify vesicular transporters with machine learning.

Vesicular transport proteins are membrane proteins. The cell membrane separates the cell's internal environment from the outside and controls the transport of substances into and out of the

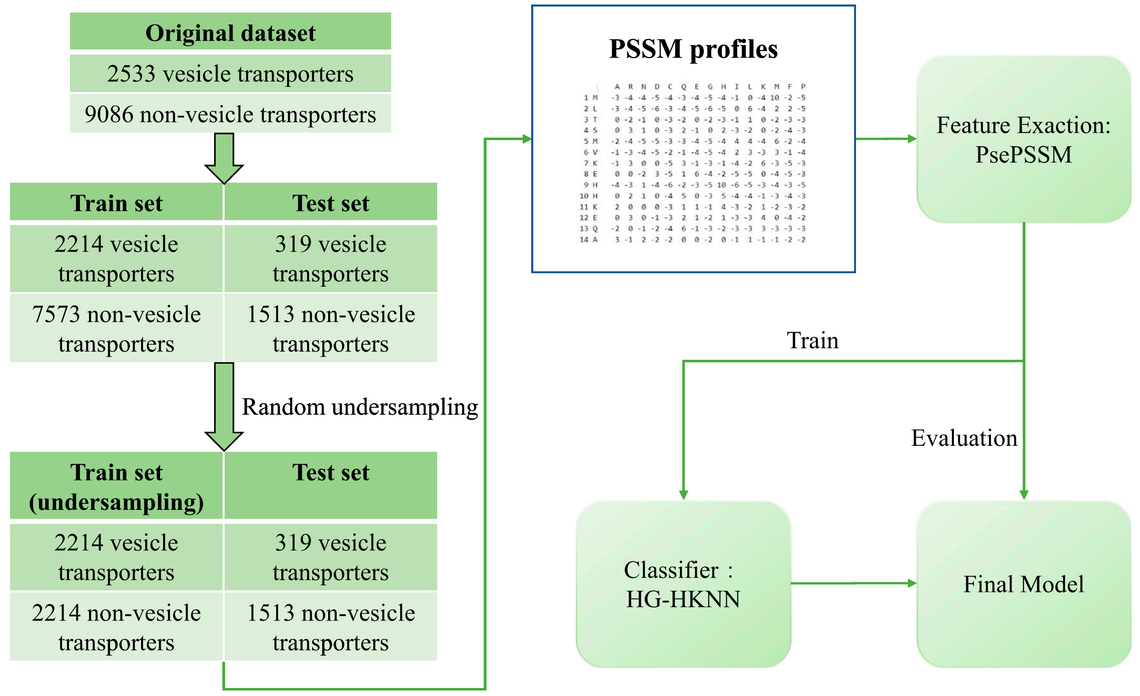

Original dataset
2533 vesicle transporters
9086 non-vesicle transporters
Train set
Test set
2214 vesicle transporters
319 vesicle transporters
7573 non-vesicle transporters
1513 non-vesicle transporters
Random undersampling
Train set (undersampling)
Test set
2214 vesicle transporters
319 vesicle transporters
2214 non-vesicle transporters
1513 non-vesicle transporters
PSSM profiles
Feature Exaction: PsePSSM
Train
Evaluation
Classifier : HG-HKNN
Final Model

FIGURE 1 | Flowchart of our model.

cell. Different substances enter and leave cells in different ways, and the transport of macromolecular substances is called vesicular transport. In vesicular transport, cells first surround substances and form vesicles. Vesicles move within cells and release their contents through vesicle rupture or membrane fusion. The process of vesicle transport exists widely in life activities. Vesicular transport proteins play an important role in vesicle transport by regulating the interactions of specific molecules with the vesicle membrane. In biology, there have been many studies on vesicular transport proteins, such as (Cheret et al., 2021; Li et al., 2021; Fu T. et al., 2022). Many human diseases are associated with abnormal vesicle transport proteins, such as those described in (Buck et al., 2021; Mazere et al., 2021; Zhou et al., 2022).

With the development of protein sequencing technology, an increasing number of vesicle transport protein sequences have been discovered. The need to rapidly identify vesicle transporter protein sequences conflicts with traditional experimental techniques, which are costly and time-consuming. Therefore, it is imperative to develop a fast and efficient computational method. To date, there have been few studies on the computational identification of vesicle transport proteins.

Computational identification of protein, RNA and DNA sequences has similar steps, and their processes can be described as two steps of feature extraction and classification. In 2019, Le et al. proposed a method (Vesicular-GRU) to identify vesicle transporters using position-specific scoring matrix (PSSM) features and a neural network classifier based on a convolutional neural network (CNN) and gated recurrent unit (GRU) and released the dataset used in their study (Le et al., 2019). In 2020, Tao et al. (Tao et al., 2020) attempted to classify vesicular transport proteins with fewer feature dimensions. Their model used the composition part of the method of composition, transition, and distribution (CTDC) features and a support vector machine (SVM) classifier. After dimensionality reduction with the Maximum Relevance Maximum Distance (MRMD) method, they obtained a comparatively satisfactory accuracy with fewer feature dimensions on the Le et al. dataset.

In our study, we propose a new model to identify vesicular transporters using pseudo position-specific scoring matrix (PsePSSM) features and a classifier called hypergraph regularized k-local hyperplane distance nearest neighbour (HG-HKNN). The main contributions of our work are as follows: 1) a better identification model of vesicle transport protein, with fewer feature dimensions and better results than the state-of-the-art model; and 2) a classifier called HG-HKNN that combines hypergraph learning (Zhou et al., 2006; Ding et al., 2020a) with k-local hyperplane distance nearest neighbours (HKNN) (Vincent and Bengio, 2001; Liu et al., 2021). The flowchart of our study is illustrated in **Figure 1**.

2 MATERIALS AND METHODS

2.1 Dataset

The dataset we use to build and evaluate the model is the benchmark dataset released by Le et al. (Le et al., 2019). In the construction of the benchmark dataset, experimentally validated vesicular transport proteins were screened from the universal protein (UniProt) database (Consortium, 2019) and the gene ontology (GO) database (Consortium, 2004).

TABLE 1 | Details of the dataset used in our study.

	Original	Train Set	Train Set (RUS)	Test Set
Vesicular transport	2533	2214	2214	319
Non-vesicular transport	9086	7573	2214	1513

For the positive dataset, the authors collected protein sequences by searching the UniProt database for the keyword "vesicular transport" or the gene ontology term "vesicular transport". Likewise, for the negative dataset, the authors collected a set of universal protein (membrane protein) sequences and excluded vesicular transporters from them. Next, protein sequences annotated by biological experiments were selected in the original dataset, and all protein sequences that were not validated experimentally were filtered out. The authors then eliminated homologous sequences on the positive and negative datasets, respectively, with a 30% cut-off level by the basic local alignment search tool (BLAST) clustering (Johnson et al., 2008). The BLAST clustering ensures that any two sequences in the dataset have less than 30% pairwise sequence similarity. Finally, protein sequences with noncanonical amino acids (X, U, B, Z) were removed from the dataset.

The benchmark dataset contains 2533 vesicular transport proteins and 9086 non-vesicular transport proteins, and the dataset is divided into a training set and a test set. The training set consists of 2144 vesicular transporters and 7573 non-vesicular transporters, and the test set consists of 319 vesicular transporters and 1513 non-vesicular transporters. We perform random undersampling (RUS) on the training set to balance the proportions of positive and negative samples. In random undersampling, we randomly select a sample from the class with more samples in the training set to represent its class, and repeat until there are the same number of vesicular transport proteins and non-vesicular transport proteins in the training set. The randomly undersampled training set has 2214 positive samples and 2214 negative samples. The details of the dataset are listed in **Table 1**.

2.2 Feature Extraction

The feature type we use is PsePSSM (Chou and Shen, 2007), and the PSSM profile used to build PsePSSM is directly downloaded from the open-source data of Le et al. (Le et al., 2019). The authors of (Le et al., 2019) constructed these PSSM profiles by searching all sequences one by one in the non-redundant (NR) database with BLAST software. The PSSM matrix is an $L*20$ matrix similar to the following formula (Zhu et al., 2019). Each PSSM matrix corresponds to a protein sequence.

$$\boldsymbol{P}_{\mathrm{PSSM}} = \begin{bmatrix} \mathbb{E}_{1\to1} & \mathbb{E}_{1\to2} & \cdots & \mathbb{E}_{1\to20} \\ \mathbb{E}_{2\to1} & \mathbb{E}_{2\to2} & \cdots & \mathbb{E}_{2\to20} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbb{E}_{i\to1} & \mathbb{E}_{i\to2} & \cdots & \mathbb{E}_{i\to20} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbb{E}_{L\to1} & \mathbb{E}_{L\to2} & \cdots & \mathbb{E}_{L\to20} \end{bmatrix}. \tag{1}$$

In this formula, L is the length of the protein sequence. $\mathrm{E}_{i\to j}$ represents the relationship between the amino acid at position i of the protein sequence and the amino acid of type j in the homologous sequence. j is the amino acid type number ranging from 1 to 20. The PSSM matrix contains the position-specific frequency information of amino acids in the protein homologous sequences, which is used to decode the evolutionary information of proteins. Compared with other protein information (such as amino acid frequency and physicochemical properties), the PSSM matrix of proteins not only contains the information of the proteins in the dataset but also contains the motif information of the protein homologous sequences in the NR database. However, the dimension of the PSSM matrix is too large, so further PsePSSM feature extraction is required.

The PsePSSM feature we use is a $(\xi + 1)*20$ dimension feature, which can be calculated with this formula:

$$\boldsymbol{P}^{\xi}_{\mathrm{PsePSSM}} = \left[\bar{\mathbb{E}}_1 \cdots \bar{\mathbb{E}}_{20} G^1_1 \cdots G^1_{20} \cdots G^{\xi}_1 \cdots G^{\xi}_{20}\right]^{\boldsymbol{T}}. \tag{2}$$

where $\bar{\mathrm{E}}_j$ is the average value of each column of the PSSM matrix, and the calculation of G^{ξ}_j can be expressed by the following formula:

$$G^{\xi}_j = \frac{1}{L-\xi} \sum_{i=1}^{L-\xi} \left[\mathbb{E}_{i\to j} - \mathbb{E}_{(i+\xi)\to j}\right]^2 \quad (j = 1, 2, \cdots, 20; \xi < L). \tag{3}$$

G^{ξ}_j is the correlation factor obtained by coupling the ξ th-most contiguous PSSM scores along the protein chain with amino acid type j. Clearly, $\bar{\mathrm{E}}_j$ and G^0_j are the same. Note that the maximum value of ξ must be less than the length of the shortest protein sequence in the benchmark dataset. The value of ξ we choose is 6, so $\boldsymbol{P}^{\xi}_{PsePSSM}$ is a feature vector with 140 dimensions. When ξ increases, the evaluation metric first increases and then decreases and reaches the maximum value when ξ is 6.

2.3 Method for Classification

The hypergraph regularized k-local hyperplane distance nearest neighbour model (HG-HKNN) is a new classifier that combines the k-local hyperplane distance nearest neighbour algorithm (HKNN) and hypergraph learning.

2.3.1 HKNN

In the HKNN (Vincent and Bengio, 2001) workflow, multiple hyperplanes are constructed first, each hyperplane corresponds to a class in the training set, and the hyperplane is constructed by the k samples of the same class that is closest to the test sample. Then, the HKNN predicts the class of the test sample by comparing the distance between the test sample and the hyperplanes and assigns the test sample to the class corresponding to the nearest hyperplane (Ding et al., 2022c). **Figure 2** shows a sketch of an HKNN, where sample x obtains its class by comparing the distances to hyperplane 1 and hyperplane 2.

In class c, when x represents the test sample, the hyperplane can be expressed as the following formula:

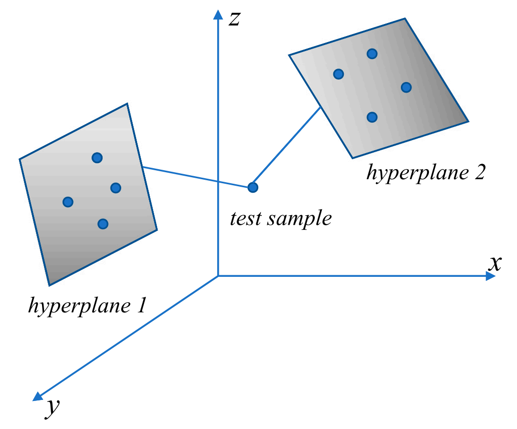


FIGURE 2 | Sketch of an HKNN.

$$LH_k^c(x) = \left\{ p^c \mid p^c = \bar{N}^c + \sum_{i=1}^{k} \alpha_i^c V_i^c, \alpha_{1\ldots k}^c \in R^k \right\}. \tag{4}$$

where k means that k nearest neighbour samples are taken to construct the hyperplane, and the i -th sample in class c can be expressed as N_i^c (i from 1 to k). Let $\bar{N}^c$ represent the centre of N_i^c, and let $V_i^c = N_i^c - \bar{N}^c$, where α_i^c is an undetermined parameter; then, p^c is a point on this hyperplane.

The mean squared distance of the test sample x to each hyperplane can be expressed as follows:

$$(LH_k^c(x))^2 = \left\| x - \bar{N}^c - \sum_{i=1}^{k} \alpha_i^c V_i^c \right\|^2 + \lambda \sum_{i=1}^{k} (\alpha_i^c)^2. \tag{5}$$

where λ is the regularization parameter of α_i^c, which is used to reduce the complexity of the model. α^c is obtained by minimizing the distance. Finally, the classification result of the HKNN can be judged by the following formula:

$$c = argmin_c \left\| x - \bar{N}^c - \sum_{i=1}^{k} \alpha_i^c V_i^c \right\|^2. \tag{6}$$

HKNN has relatively good performance on unbalanced datasets because the same number of samples are selected in each class. However, since the distribution of samples cannot be fully expressed by a hyperplane, the performance of the HKNN is disturbed by the distribution of samples.

2.3.2 Hypergraph Learning

In machine learning, we can express the similarity between two samples by calculating the inner product of the features of the two samples to form a pairwise similarity matrix (Yang et al., 2020). However, the relationship between samples cannot simply be determined by pairwise similarity. Therefore, hypergraphs (Zhou et al., 2006) are proposed to express the relationship between three or more samples.

In a hypergraph, each hyperedge consists of multiple vertices. **Figure 3** is a hypergraph and its association matrix H. In our study, each hyperedge weights 1. When hyperedge e_j contains vertex v_i, then H_{ij} is 1; otherwise, it is 0.

Formally, the association matrix H, the degree of each hyperedge, and the degree of each vertex can be expressed as:

$$H(v,e) = \begin{cases} 1, & \text{if } v \in e \\ 0, & \text{if } v \notin e \end{cases}, \tag{7a}$$

$$\delta(e) = \sum_{v \in V} H(v,e), \tag{7b}$$

$$d(v) = \sum_{e \in E} H(v,e). \tag{7c}$$

The Laplacian matrix of a hypergraph association matrix H can be calculated as:

$$L_H = I - D_v^{-\frac{1}{2}} H A D_e^{-1} H^T D_v^{-\frac{1}{2}}. \tag{8}$$

where D_v and D_e are the diagonal matrices formed by $d(v)$ and $\delta(e)$, respectively, and A is the same as the identity matrix I in our study. We construct the association matrix H with the k -nearest neighbour algorithm proposed by Zhou et al. (Zhou et al., 2006). Given a set of samples, we choose the k nearest neighbours of each sample and construct a hyperedge containing these k vertices. Finally, we construct N hyperedges for a dataset of N samples.

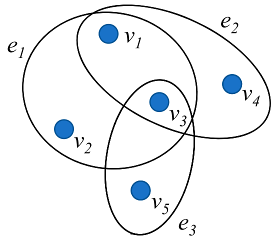
2.3.3 HG-HKNN

The HG-HKNN rewrites the mean squared distance from the test sample x to each hyperplane in the HKNN into the following form:

$$(LH_k^c(x))^2 = \left\| \phi(\bar{x}) - \sum_{i=1}^{k} \alpha_i^c \phi(V_i^c) \right\|^2 + \lambda \sum_{i=1}^{k} (\alpha_i^c)^2$$

$$+ \mu \sum_{p=1}^{k} \sum_{q=1}^{k} w_{p,q}^c (\alpha_p^c - \alpha_q^c)^2. \tag{9}$$

The kernel trick (Hofmann, 2006; Ding et al., 2019) is used to solve this problem, and the map ϕ maps the feature space to higher dimensions. $\bar{x} = x - \bar{N}^c$ is a simple rewrite. The third term in this formula is the Laplacian regularization term, which improves classification performance by smoothing the feature space (Ding et al., 2021). μ is the Laplacian regularization parameter, and $w_{p,q}^c$ is the similarity between the p -th nearest and the q -th nearest samples in the k samples in class c, which is



	v_1	v_2	v_3	v_4	v_5
e_1	1	1	1	0	0
e_2	1	0	1	1	0
e_3	0	0	1	0	1

FIGURE 3 | A hypergraph and its association matrix H.

calculated by the kernel function (Ding et al., 2022a). $K(x, y) = \phi(x), \phi(y)$ represents the kernel function, which is the radial basis function (RBF) in our study.

By minimizing the distance and making the partial derivative of $(LH_k^c(x))^2$ with respect to α^c zero, then the solution of α^c is obtained as follows:

$$\begin{gathered}\frac{\partial\left(\left(LH_k^c(x)\right)^2\right)}{\partial\alpha^c} = 0,\\ \left(\phi(V^c)^T\phi(V^c) + \lambda I + \mu L\right)\alpha^c = \phi(V^c)^T\phi(\bar{x}),\\ \alpha^c = \left(\phi(V^c)^T\phi(V^c) + \lambda I + \mu L\right)^{-1}\phi(V^c)^T\phi(\bar{x}),\\ \alpha^c = \left(K(V^c, V^c) + \lambda I + \mu L\right)^{-1}K\left(V^c, \bar{x}\right).\end{gathered} \tag{10}$$

We construct the hypergraph and use the Laplacian matrix of the hypergraph to replace the Laplacian matrix in the above formula:

$$\alpha^c = \left(K(V^c, V^c) + \lambda I + \mu L_H\right)^{-1}K\left(V^c, \bar{x}\right). \tag{11}$$

Note that the original Laplacian matrix contains pairwise similarities between samples, while our hypergraph Laplacian matrix contains more complex relationships between samples.

Now the distance from sample x to the c -th hyperplane can be expressed as follows:

$$\begin{aligned}distance_c &= \left\|\phi\left(\bar{x}\right) - \sum_{i=1}^{k}\alpha_i^c\phi\left(V_i^c\right)^2\right\|,\\ &= \left(\phi(\bar{x}) - \phi(V^c)\alpha^c\right)^T\left(\phi(\bar{x}) - \phi(V^c)\alpha^c\right),\\ &= \left(K\left(\bar{x}, \bar{x}\right) - 2(\alpha^c)^T K\left(V^c, \bar{x}\right) + (\alpha^c)^T K(V^c, V^c)\alpha^c\right).\end{aligned} \tag{12}$$

Finally, we assign the test sample x to class c:

$$c = argmin_c(distance_c). \tag{13}$$

We define the prediction score as follows:

$$score_c = \frac{\sqrt{distance_c}}{\sum_{i=1}^{C}\sqrt{distance_i}}, i = 1,\ 2,\ \ldots,\ C. \tag{14}$$

The process of HG-HKNN is listed in Algorithm 1

Algorithm 1. Algorithm of HG-HKNN

```
Input: A test sample $x$, a training set $N$ with $C$ types of classes, five parameters $k$, $\lambda$, $\mu$, $\gamma$
and $k_H$;

Output: The prediction label and score of test sample $x$;

1.  for $1 \le c \le C$ do
2.    Getting $k$ nearest neighbourhoods $N_i^c$ ($i$ from 1 to $k$) of sample $x$ in $c$-th class;
3.    Calculating $\bar{N}^c = \frac{1}{k}\sum_{i=1}^{k} N_i^c$, $V_i^c = N_i^c - \bar{N}^c$ and $\bar{x} = x - \bar{N}^c$;
4.    Obtaining $K(\bar{x},\bar{x})$, $K(V^c,\bar{x})$ and $K(V^c,V^c)$ with the radial basis function and its
      parameter $\gamma$;
5.    Getting $k_H$ nearest neighbours of each sample with kernel matrix $K(V^c,V^c)$,;
6.    Constructing the hypergraph association matrix $H$, and obtaining the diagonal matrices
      $D_v$, $D_e$ and $A$;
7.    Calculating the Laplacian matrix of the hypergraph with $L_H = I - D_v^{-\frac{1}{2}}HAD_e^{-1}H^T D_v^{-\frac{1}{2}}$
8.    Obtaining $\alpha^c$ with $\alpha^c = (K(V^c,V^c) + \lambda I + \mu L_H)^{-1}K(V^c,\bar{x})$;
9.    Calculating $distance_c = (K(\bar{x},\bar{x}) - 2(\alpha^c)^T K(V^c,\bar{x}) + (\alpha^c)^T K(V^c,V^c)\alpha^c)$;
10. end for
11. Assign sample $x$ to class $c$ with $c = argmin_c(distance_c)$;
12. Calculating prediction score for sample $x$ with $score_c = \frac{\sqrt{distance_c}}{\sum_{i=1}^{C}\sqrt{distance_i}}$;
13. Return the label and score of test sample $x$.
```

3 RESULTS AND DISCUSSION

3.1 Evaluation

In this section, we will introduce the evaluation methods and metrics we use. We use positive to describe vesicular transport proteins and negative to describe non-vesicular transport proteins. We optimize the parameters with cross-validation (CV) on the training set and then evaluate our model on the test set.

Cross-validation sets aside a small portion of the dataset for validating the model, while the rest of the dataset is used for training the model (Zhang D. et al., 2021; Lv et al., 2021; Yang et al., 2021; Zheng et al., 2021; Li F. et al., 2022; Li X. et al., 2022). The leave-one-out cross-validation (LOOCV) is a classic cross-validation method (Qiu et al., 2021). LOOCV takes only one sample in the dataset at a time for validation and uses other samples in the dataset to train the model. Until all samples are left out once for validation, the leave-one-out method obtains statistical values for multiple results. However, the leave-one-out method is too time-consuming, so we adopted another cross-validation method: k -fold cross-validation (K-CV). K-CV divides the dataset into k subsets. Each time, one of the subsets is taken for validation, and the remaining $k - 1$ subsets are used for training the model. In this way, k prediction results are obtained, and we take the average of these k results as the result of k -fold cross-validation.

The evaluation indicators we take include sensitivity, precision, specificity, accuracy (ACC), Matthews correlation coefficient (MCC), and area under the receiver operating characteristic curve (AUC), which have been widely used in previous studies (Hong et al., 2020a; Tang et al., 2020; Pan et al., 2022; Song et al., 2022).

$$sensitivity = \frac{TP}{TP + FN}, \tag{15a}$$

$$precision = \frac{TP}{TP + FP}, \tag{15b}$$

$$specificity = \frac{TN}{TN + FP}, \tag{15c}$$

$$ACC = \frac{TP + TN}{TP + FN + FP + FN}, \tag{15d}$$

$$MCC = \frac{1 - \left(\left(\frac{FN}{TP} + FN\right) + \left(\frac{FP}{TN} + FP\right)\right)}{\sqrt{\left(1 + \left(FP - \frac{FN}{TP} + FN\right)\right)\left(1 + \left(FN - \frac{FP}{TN} + FP\right)\right)}}. \tag{15e}$$

where TP, FP, TN, and FN represent true positives, false positives, true negatives, and false negatives, respectively. In addition, the AUC is obtained by integrating the receiver operating characteristic curve (ROC) (Fu J. et al., 2022). The ROC curve plots sensitivity and specificity at different classification thresholds (Tzeng et al., 2022). The more meaningful ones are AUC and precision since our test set is a class-imbalanced dataset. In our model, we perform 10-fold cross-validation on a training set of 4428 samples (2214 positive and 2214 negative). The binary

TABLE 2 | Details in parameter tuning of ***k***.

k	AUC	ACC	Precision	Specificity
200	0.8127	0.7256	0.7677	0.8035
350	0.8241	0.7319	0.7897	0.8311
500	0.8284	0.7362	0.7940	0.8338
650	0.8292	0.7398	0.7954	0.8333
800	0.8287	0.7425	0.7927	0.8265
950	0.8279	0.7437	0.7840	0.8134

TABLE 3 | Comparison of classification metrics among different kernels.

Kernel Type	AUC	MCC	ACC	Precision	Specificity
Linear	0.7618	0.3739	0.6719	0.7833	0.8686
Polynomial	0.8021	0.4664	0.7322	0.7519	0.7687
Laplacian	0.8243	0.5153	0.7575	0.7592	0.7597
RBF	0.8309	0.5099	0.7538	0.7760	0.7922

classification threshold is set to the default 0.5. Finally, the trained model is evaluated on the test set, which has 319 positive samples and 1513 negative samples.

3.2 Parameter Tuning

In this section, we describe the parameter tuning process for our model. Classification metrics are largely influenced by parameter tuning. The HG-HKNN has five parameters: k, λ, μ, γ, and k_H. k represents the number of neighbour samples selected when constructing the hyperplane. λ is the regularization parameter in L_2 regularization and μ is the Laplacian regularization parameter. γ is a parameter in the radial basis function. k_H is the number of neighbours used to construct the hypergraph.

We first adjust the k parameters among them. We set k, μ and γ to be 0.2, 0.2 and 0.2, respectively, and k_H to be 2. We perform 10-fold cross-validation for different values of k, and the best parameter k is determined to be 650; the details are shown in **Table 2**.

For λ, μ, γ and k_H, we adopt the grid search method for parameter tuning. The grid search method enumerates the possible values of each parameter, combines the possible values of all parameters into groups, and then trains the model with each group of parameters to obtain the best set of parameters. In our grid search, the possible values of λ, μ and γ are all 0.1, 0.2, 0.4, and 0.8, and the k_H values in the hypergraph range from 2 to 10. The best parameters for choosing λ, μ and γ are 0.4, 0.4 and 0.4, respectively. The best parameter k_H is 2, and the best AUC is 0.8309.

In our dataset, the dimension of features is much smaller than the number of samples, which is regarded as a sign that the dataset is linearly inseparable. On linearly inseparable datasets, the RBF kernel generally performs better than the linear or polynomial kernel. Formally, the Laplacian kernel is similar to the RBF kernel, and they usually have similar performance, but the Laplacian kernel function requires additional computational cost. We regard the type of kernel function used by HG-HKNN as an additional hyperparameter and conduct comparative experiments. The details of the experimental results are shown in **Table 3**. The results show that the RBF kernel has the best performance.

TABLE 4 | Comparison of classification metrics among different models.

Techniques	AUC	MCC	ACC	Precision	Specificity
KNN	0.7824	0.4189	0.7078	0.6886	0.6519
RF	0.8019	0.4576	0.7285	0.7267	0.7231
SVM	0.8091	0.4820	0.7405	0.7466	0.7502
HKNN	0.8203	0.4976	0.7484	0.7442	0.7371
OG-HKNN	0.8289	0.4944	0.7446	0.7843	0.8130
HG-HKNN	0.8309	0.5099	0.7538	0.7760	0.7922

3.3 Comparison With Traditional Machine Learning Methods

In the previous section, we have chosen the best parameters for our model. Our model is trained with traditional PsePSSM features, with nothing special in feature extraction. In this section, to highlight the effect of our proposed classifier HG-HKNN, we train some models with different traditional machine learning classifiers, the same training set, and the same PsePSSM feature extraction method. We perform 10-fold cross-validation on these models and compare the evaluation metrics of these models with ours. Note that the only difference between these models is the classifier.

We implement and train these models with the programming language's built-in library of functions. With the help of the parameter optimization function, we can automatically train the SVM model with the best evaluation metrics. After parameter tuning, the parameters in the other models are as follows: $K = 20$ in the k-nearest neighbour model(KNN), $ntrees = 60$ in the random forest model (RF), and $k = 30$ and $\lambda = 10$ in HKNN. **Table 4** shows the comparison of our model with other traditional machine learning models in 10-fold cross-validation.

Among them, the prediction effect of HKNN is better than that of the KNN algorithm. Intuitively explained in principle, although the classical K-nearest neighbour algorithm can fit the training samples well, it does not work well for the unseen samples located near the decision boundary. This is the overfitting problem of the KNN algorithm, and overfitting is more obvious in small data sets. HKNN constructs a hyperplane for k-nearest neighbour samples and then compares the distances between the test sample and the hyperplanes. The construction of the hyperplane can be analogous to adding more sample points to the k-nearest neighbours, which will reduce the interference of extreme samples on the decision boundary. Therefore, compared with KNN, the HKNN model has a smoother decision boundary, avoiding the disadvantage of overfitting in KNN.

Our proposed HG-HKNN model outperforms the other models on almost all metrics at the same level of comparison. By introducing Laplacian regularization in manifold learning, the HG-HKNN model incorporates local similarity information in the feature space into the construction process of the hyperplane.

TABLE 5 | Comparison of our model with other existing technologies.

Techniques	AUC	MCC	ACC	Sensitivity	Precision	Specificity
GRU	0.848	0.44	79.2	70.8	44.0	81.0
BLSTM	0.846	0.46	84.6	54.2	55.8	90.9
BLAST	0.82	0.43	83.6	54.1	52.8	89.8
Vesicular-GRU	0.861	0.52	82.3	79.2	48.7	82.9
HG-HKNN	0.870	0.53	84.1	72.1	53.2	86.7

Compared with the HKNN model, the HG-HKNN model not only reduces the disturbance of extreme samples to the decision boundary, but also preserves the local similarity information in the feature space. In the HG-HKNN model, we replace the ordinary graph with a hypergraph for Laplacian regularization. Hypergraph learning allows us to represent feature space local structures with more complex relationships than just pairwise similarity relationships. This further improves the performance of our HG-HKNN model. To highlight the effect of hypergraph learning, we add an ordinary graph regularized HKNN model (OG-HKNN) to our comparison, and the details are also listed in **Table 4**. The parameter tuning process of the OG-HKNN model is the same as that of the HG-HKNN. The best parameters for choosing λ, μ, γ and k are 0.2, 0.8, 0.4 and 350, respectively. The experimental results show that the AUC, MCC and ACC of the HG-HKNN model are better than the OG-HKNN model.

One disadvantage of our model is that HG-HKNN increases computation time and memory usage compared to HKNN. In terms of memory usage, the storage of hypergraphs, Laplacian matrices, and kernel matrices in HG-HKNN increases memory usage. In terms of operating efficiency, we conduct experiments on the test set with the same parameter $k = 20$, HKNN completes the computation in 362 milliseconds, while HG-HKNN completes the computation in 640 milliseconds. Such computational time cost is acceptable, especially considering the performance of HG-HKNN and time-consuming deep learning models in vesicle transporter identification.

3.4 Comparison With Previous Techniques

In this section, we aim to compare our model with previous techniques to highlight the performance of our proposed model on benchmark datasets. After optimizing the parameters with cross-validation, we obtain the optimal values of each parameter in HG-HKNN, where λ is 0.4, k is 650, γ is 0.4, μ is 0.4, and the value of k_H in the hypergraph part is 2. With these parameters, we no longer perform cross-validation on the training set but instead feed the entire training set into our model and then evaluate our final model on the test set. Among the metrics, the AUC is 87.0%, and the MCC is 0.53. Compared with the existing state-of-the-art Vesicular-GRU method with an AUC of 86.1% and MCC of 0.52, our model has higher AUC and MCC values, fewer feature dimensions (140 dimensions) and fewer parameters.

We compare our model with several other existing methods, among which the GRU model is a prediction method using traditional PSSM features and GRU and BLAST is a general-purpose protein prediction tool (Johnson et al., 2008). BLSTM is a commonly used prediction method in protein research (Li et al., 2020). The state-of-the-art method Vesicular-GRU (Le et al., 2019), a prediction method based on 1D CNN and GRU, is also listed in the comparison. The details of the comparison are shown in **Table 5**.

The meaning of the indicators has been described in the previous section. Experimental results show that our model achieves the best AUC and MCC metrics on this imbalanced benchmark dataset. Deep learning is involved in most of the methods in the comparison. The black box is an unavoidable problem for deep learning-based methods, and it is difficult to intuitively understand which factors lead to the predicted results. In deep learning models, researchers need to optimize a large number of parameters to improve the performance of the network, and these parameters are directly tuned through back-propagation of the prediction results, resulting in overfitting and the curse of dimensionality. The neural network in the Vesicular-GRU model has hundreds of thousands of parameters, which makes the Vesicular-GRU model a potential risk of overfitting on the training set. Our HG-HKNN has only five parameters, and the performance of our model is mainly attributable to hypergraph regularization and hyperplane rather than fitting to the parameters. Local hyperplane models have better performance on imbalanced datasets because the same number of samples are selected in each class. Like many biological sequence datasets, the vesicle transporter dataset is a typically imbalanced dataset, which is where the local hyperplane model excels. Furthermore, HG-HKNN applies kernel tricks to handle high-dimensional features, avoiding the curse of dimensionality. Although there is an increase in time and memory usage compared to HKNN, our model is faster relative to deep learning models trained with huge parameters via backpropagation. With only five parameters, our model avoids the black box, overfitting and curse of dimensionality problems in deep learning and makes predictions faster, and the performance of our model is equal to or higher than all the mentioned techniques, especially in terms of MCC and AUC.

4 CONCLUSION

In this study, we propose a novel approach for predicting vesicular transport proteins. The existing methods are typically performed with complex neural networks or by extracting a large number of features. Our method classifies vesicular transport proteins with PsePSSM features and our proposed HG-HKNN model. We completed the prediction of vesicle transporters with only 140-dimensional features and 5 parameters with satisfactory results. Experimental results show that our method has the best AUC of 0.870 and MCC of 0.53 on the benchmark dataset and outperforms the state-of-the-art method (Vesicular-GRU) in ACC, MCC and AUC. Other metrics of our model are also comparable to other methods. A traditional machine learning computational model is used in our approach, avoiding some of the drawbacks of deep learning. Compared with another study (Tao et al., 2020) using traditional machine learning on the same dataset, their study achieved 72.2% accuracy and 0.34 MCC with 21-dimensional CTDC features after MRMD (He et al., 2020) dimensionality reduction, while our model achieves 84.1% accuracy and 0.53 MCC with 140-dimensional PsePSSM features. Furthermore, like CTDC features, the classical features we used imply that amino acids have a certain regularity in the arrangement of the protein sequence. Since PSSM matrix information is a commonly used motif representation, our study may help scholars to judge whether an unknown protein is a vesicle transporter.

The proposed method also has the following limitations: 1) In the case of large parameter k, the prediction takes a long time; 2) Our model uses the PsePSSM feature without incorporating sequence information for prediction; and 3) Feature selection and dimensionality reduction are not performed in our model. For the first limitation, parallel optimization can be used to solve the problem of computation time. For the second question, adding sequence features such as amino acid frequency or composition of k-spaced amino acid pairs (CKSAAP) to our model may further improve the prediction accuracy. For the third question, the dataset can be processed with feature selection and dimensionality reduction tools that remove redundant features. The results of this study can provide a basis for further studies in computational biology to identify vesicle transport proteins with classical features and traditional machine learning classifiers.

DATA AVAILABILITY STATEMENT

Our experimental code can be obtained from https://github.com/ferryvan/HG-HKNN, and the datasets used in this study can be found in (Le et al., 2019).

AUTHOR CONTRIBUTIONS

RF performed the experiment and wrote the manuscript; BS helped perform the experiment with constructive discussions; YD contributed to the conception of the study.

FUNDING

This work was supported by the Municipal Government of Quzhou under Grant Number 2020D003 and 2021D004.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2022.960388/full#supplementary-material

REFERENCES

Buck, S. A., Steinkellner, T., Aslanoglou, D., Villeneuve, M., Bhatte, S. H., Childers, V. C., et al. (2021). Vesicular Glutamate Transporter Modulates Sex Differences in Dopamine Neuron Vulnerability to Age-Related Neurodegeneration. *Aging cell* 20 (5), e13365. doi:10.1111/acel.13365

Chen, Y., Ma, T., Yang, X., Wang, J., Song, B., and Zeng, X. (2021). MUFFIN: Multi-Scale Feature Fusion for Drug-Drug Interaction Prediction. *Bioinformatics* 37 (17), 2651–2658. doi:10.1093/bioinformatics/btab169

Cheret, C., Ganzella, M., Preobraschenski, J., Jahn, R., and Ahnert-Hilger, G. (2021). Vesicular Glutamate Transporters (SLCA17 A6, 7, 8) Control Synaptic Phosphate Levels. *Cell Rep.* 34 (2), 108623. doi:10.1016/j.celrep.2020.108623

Chou, K.-C., and Shen, H.-B. (2007). MemType-2L: a Web Server for Predicting Membrane Proteins and Their Types by Incorporating Evolution Information through Pse-PSSM. *Biochem. biophysical Res. Commun.* 360 (2), 339–345. doi:10.1016/j.bbrc.2007.06.027

Consortium, G. O. (2004). The Gene Ontology (GO) Database and Informatics Resource. *Nucleic acids Res.* 32 (Suppl. l_1), D258–D261. doi:10.1093/nar/gkh036

Consortium, U. (2019). UniProt: a Worldwide Hub of Protein Knowledge. *Nucleic Acids Res.* 47 (D1), D506–D515. doi:10.1093/nar/gky1049

Ding, Y., Tang, J., and Guo, F. (2019). Protein Crystallization Identification via Fuzzy Model on Linear Neighborhood Representation. *IEEE/ACM Trans. Comput. Biol. Bioinform* 18 (5), 1986–1995. doi:10.1109/TCBB.2019.2954826

Ding, Y., He, W., Tang, J., Zou, Q., and Guo, F. (2021). Laplacian Regularized Sparse Representation Based Classifier for Identifying DNA N4-Methylcytosine Sites via L2, 1/2-matrix Norm. *IEEE/ACM Trans. Comput. Biol. Bioinforma.* 99, 1. doi:10.1109/tcbb.2021.3133309

Ding, Y., Jiang, L., Tang, J., and Guo, F. (2020a). Identification of Human microRNA-Disease Association via Hypergraph Embedded Bipartite Local Model. *Comput. Biol. Chem.* 89, 107369. doi:10.1016/j.compbiolchem.2020.107369

Ding, Y., Tang, J., and Guo, F. (2020b). Human Protein Subcellular Localization Identification via Fuzzy Model on Kernelized Neighborhood Representation. *Appl. Soft Comput.* 96, 106596. doi:10.1016/j.asoc.2020.106596

Ding, Y., Tang, J., and Guo, F. (2020c). Identification of Drug-Target Interactions via Dual Laplacian Regularized Least Squares with Multiple Kernel Fusion. *Knowledge-Based Syst.* 204, 106254. doi:10.1016/j.knosys.2020.106254

Ding, Y., Tang, J., Guo, F., and Zou, Q. (2022a). Identification of Drug–Target Interactions via Multiple Kernel-Based Triple Collaborative Matrix Factorization. *Briefings Bioinforma.* 23 (2), bbab582. doi:10.1093/bib/bbab582

Ding, Y., Tiwari, P., Zou, Q., Guo, F., and Pandey, H. M. (2022b). C-Loss Based Higher-Order Fuzzy Inference Systems for Identifying DNA N4-Methylcytosine Sites. *IEEE Trans. Fuzzy Syst.* 2022, 12. doi:10.1109/tfuzz.2022.3159103

Ding, Y., Yang, C., Tang, J., and Guo, F. (2022c). Identification of Protein-Nucleotide Binding Residues via Graph Regularized K-Local Hyperplane Distance Nearest Neighbor Model. *Appl. Intell.* 52 (6), 6598–6612. doi:10.1007/s10489-021-02737-0

Feehan, R., Franklin, M. W., and Slusky, J. S. G. (2021). Machine Learning Differentiates Enzymatic and Non-enzymatic Metals in Proteins. *Nat. Commun.* 12 (1), 1–11. doi:10.1038/s41467-021-24070-3

Fu, J., Zhang, Y., Wang, Y., Zhang, H., Liu, J., Tang, J., et al. (2022a). Optimization of Metabolomic Data Processing Using NOREVA. *Nat. Protoc.* 17 (1), 129–151. doi:10.1038/s41596-021-00636-9

Fu, T., Li, F., Zhang, Y., Yin, J., Qiu, W., Li, X., et al. (2022b). VARIDT 2.0: Structural Variability of Drug Transporter. *Nucleic Acids Res.* 50 (D1), D1417–D1431. doi:10.1093/nar/gkab1013

He, S., Guo, F., and Zou, Q. (2020). MRMD2. 0: a python Tool for Machine Learning with Feature Ranking and Reduction. *Curr. Bioinforma.* 15 (10), 1213–1221. doi:10.2174/1574893615999200503030350

Hofmann, M. (2006). Support Vector Machines-Kernels and the Kernel Trick. *Notes* 26 (3), 1–16.

Hong, J., Luo, Y., Mou, M., Fu, J., Zhang, Y., Xue, W., et al. (2020a). Convolutional Neural Network-Based Annotation of Bacterial Type IV Secretion System Effectors with Enhanced Accuracy and Reduced False Discovery. *Brief. Bioinform* 21 (5), 1825–1836. doi:10.1093/bib/bbz120

Hong, J., Luo, Y., Zhang, Y., Ying, J., Xue, W., Xie, T., et al. (2020b). Protein Functional Annotation of Simultaneously Improved Stability, Accuracy and False Discovery Rate Achieved by a Sequence-Based Deep Learning. *Brief. Bioinform* 21 (4), 1437–1447. doi:10.1093/bib/bbz081

Jin, S., Zeng, X., Xia, F., Huang, W., and Liu, X. (2021). Application of Deep Learning Methods in Biological Networks. *Briefings Bioinforma.* 22 (2), 1902–1917. doi:10.1093/bib/bbaa043

Johnson, M., Zaretskaya, I., Raytselis, Y., Merezhuk, Y., McGinnis, S., and Madden, T. L. (2008). NCBI BLAST: a Better Web Interface. *Nucleic Acids Res.* 36 (Suppl. l_2), W5–W9. doi:10.1093/nar/gkn201

Le, N. Q. K., Yapp, E. K. Y., Nagasundaram, N., Chua, M. C. H., and Yeh, H.-Y. (2019). Computational Identification of Vesicular Transport Proteins from Sequences Using Deep Gated Recurrent Units Architecture. *Comput. Struct. Biotechnol. J.* 17, 1245–1254. doi:10.1016/j.csbj.2019.09.005

Li, F., Eriksen, J., Finer-Moore, J., Edwards, R. H., and Stroud, R. M. (2021). Structure of a Vesicular Glutamate Transporter Determined by Cryo-Em. *Biophysical J.* 120 (3), 104a. doi:10.1016/j.bpj.2020.11.844

Li, F., Zhou, Y., Zhang, Y., Yin, J., Qiu, Y., Gao, J., et al. (2022a). POSREG: Proteomic Signature Discovered by Simultaneously Optimizing its Reproducibility and Generalizability. *Brief. Bioinform* 23 (2), bbac040. doi:10.1093/bib/bbac040

Li, J., Pu, Y., Tang, J., Zou, Q., and Guo, F. (2020). DeepAVP: a Dual-Channel Deep Neural Network for Identifying Variable-Length Antiviral Peptides. *IEEE J. Biomed. Health Inf.* 24 (10), 3012–3019. doi:10.1109/jbhi.2020.2977091

Li, X., Ma, S., Liu, J., Tang, J., and Guo, F. (2022b). Inferring Gene Regulatory Network via Fusing Gene Expression Image and RNA-Seq Data. *Bioinformatics* 38 (6), 1716–1723. doi:10.1093/bioinformatics/btac008

Liu, X., Zhang, X., Zhang, Y., Ding, Y., Shan, W., Huang, Y., et al. (2021). Kernelized K-Local Hyperplane Distance Nearest-Neighbor Model for Predicting Cerebrovascular Disease in Patients with End-Stage Renal Disease. *Front. Neurosci.* 15, 773208. doi:10.3389/fnins.2021.773208

Lv, H., Dao, F.-Y., Guan, Z.-X., Yang, H., Li, Y.-W., and Lin, H. (2021). Deep-Kcr: Accurate Detection of Lysine Crotonylation Sites Using Deep Learning Method. *Brief. Bioinform* 22 (4), bbaa255. doi:10.1093/bib/bbaa255

Mazere, J., Dilharreguy, B., Catheline, G., Vidailhet, M., Deffains, M., Vimont, D., et al. (2021). Striatal and Cerebellar Vesicular Acetylcholine Transporter Expression Is Disrupted in Human DYT1 Dystonia. *Brain* 144 (3), 909–923. doi:10.1093/brain/awaa465

Pan, X., Lin, X., Cao, D., Zeng, X., Yu, P. S., He, L., et al. (2022). Deep Learning for Drug Repurposing: Methods, Databases, and Applications. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* 2022, e1597. doi:10.1002/wcms.1597

Qiu, Y., Ching, W. K., and Zou, Q. (2021). Matrix Factorization-Based Data Fusion for the Prediction of RNA-Binding Proteins and Alternative Splicing Event Associations during Epithelial-Mesenchymal Transition. *Brief. Bioinform* 22 (6), bbab332. doi:10.1093/bib/bbab332

Shen, Y., Tang, J., and Guo, F. (2019). Identification of Protein Subcellular Localization via Integrating Evolutionary and Physicochemical Information into Chou's General PseAAC. *J. Theor. Biol.* 462, 230–239. doi:10.1016/j.jtbi.2018.11.012

Song, B., Li, F., Liu, Y., and Zeng, X. (2021). Deep Learning Methods for Biomedical Named Entity Recognition: a Survey and Qualitative Comparison. *Brief. Bioinform* 22 (6), bbab282. doi:10.1093/bib/bbab282

Song, B., Luo, X., Luo, X., Liu, Y., Niu, Z., and Zeng, X. (2022). Learning Spatial Structures of Proteins Improves Protein-Protein Interaction Prediction. *Briefings Bioinforma.* 23, bbab558. doi:10.1093/bib/bbab558

Su, R., He, L., Liu, T., Liu, X., and Wei, L. (2021). Protein Subcellular Localization Based on Deep Image Features and Criterion Learning Strategy. *Brief. Bioinform* 22 (4), bbaa313. doi:10.1093/bib/bbaa313

Tang, J., Fu, J., Wang, Y., Li, B., Li, Y., Yang, Q., et al. (2020). ANPELA: Analysis and Performance Assessment of the Label-free Quantification Workflow for Metaproteomic Studies. *Brief. Bioinform* 21 (2), 621–636. doi:10.1093/bib/bby127

Tao, Z., Li, Y., Teng, Z., and Zhao, Y. (2020). A Method for Identifying Vesicle Transport Proteins Based on LibSVM and MRMD. *Comput. Math. Methods Med.* 2020, 8926750. doi:10.1155/2020/8926750

Tzeng, S., Chen, C.-S., Li, Y.-F., and Chen, J.-H. (2022). On Summary ROC Curve for Dichotomous Diagnostic Studies: an Application to Meta-Analysis of COVID-19. *J. Appl. Statistics*, 1–17. doi:10.1080/02664763.2022.2041565

Vincent, P., and Bengio, Y. (2001). K-Local Hyperplane and Convex Distance Nearest Neighbor Algorithms. *Adv. neural Inf. Process. Syst.* 14, 985–992.

Wang, H., Ding, Y., Tang, J., Zou, Q., and Guo, F. (2021). Identify RNA-Associated Subcellular Localizations Based on Multi-Label Learning Using Chou's 5-steps Rule. *Bmc Genomics* 22 (1), 56. doi:10.1186/s12864-020-07347-7

Xiong, G., Wu, Z., Yi, J., Fu, L., Yang, Z., Hsieh, C., et al. (2021). ADMETlab 2.0: an Integrated Online Platform for Accurate and Comprehensive Predictions of ADMET Properties. *Nucleic Acids Res.* 49 (W1), W5–W14. doi:10.1093/nar/gkab255

Yang, H., Ding, Y., Tang, J., and Guo, F. (2021). Drug-disease Associations Prediction via Multiple Kernel-Based Dual Graph Regularized Least Squares. *Appl. Soft Comput.* 112, 107811. doi:10.1016/j.asoc.2021.107811

Yang, Q., Li, B., Tang, J., Cui, X., Wang, Y., Li, X., et al. (2020). Consistent Gene Signature of Schizophrenia Identified by a Novel Feature Selection Strategy from Comprehensive Sets of Transcriptomic Data. *Brief. Bioinform* 21 (3), 1058–1068. doi:10.1093/bib/bbz049

Zeng, X., Tu, X., Liu, Y., Fu, X., and Su, Y. (2022). Toward Better Drug Discovery with Knowledge Graph. *Curr. Opin. Struct. Biol.* 72, 114–126. doi:10.1016/j.sbi.2021.09.003

Zhang, D., Chen, H.-D., Zulfiqar, H., Yuan, S.-S., Huang, Q.-L., Zhang, Z.-Y., et al. (2021a). iBLP: An XGBoost-Based Predictor for Identifying Bioluminescent Proteins. *Comput. Math. Methods Med.* 2021, 6664362. doi:10.1155/2021/6664362

Zhang, J., Zhang, Z., Pu, L., Tang, J., and Guo, F. (2021b). AIEpred: An Ensemble Predictive Model of Classifier Chain to Identify Anti-inflammatory Peptides. *IEEE/ACM Trans. Comput. Biol. Bioinf.* 18 (5), 1831–1840. doi:10.1109/tcbb.2020.2968419

Zhang, W., Xue, X., Xie, C., Li, Y., Liu, J., Chen, H., et al. (2021c). CEGSO: Boosting Essential Proteins Prediction by Integrating Protein Complex, Gene Expression, Gene Ontology, Subcellular Localization and Orthology Information. *Interdiscip. Sci. Comput. Life Sci.* 13 (3), 349–361. doi:10.1007/s12539-021-00426-7

Zhang, Z.-Y., Sun, Z.-J., Yang, Y.-H., and Lin, H. (2022). Towards a Better Prediction of Subcellular Location of Long Non-coding RNA. *Front. Comput. Sci.* 16 (5), 1–7. doi:10.1007/s11704-021-1015-3

Zheng, Y., Wang, H., Ding, Y., and Guo, F. (2021). CEPZ: A Novel Predictor for Identification of DNase I Hypersensitive Sites. *IEEE/ACM Trans. Comput. Biol. Bioinf.* 18 (6), 2768–2774. doi:10.1109/tcbb.2021.3053661

Zhou, D., Huang, J., and Schölkopf, B. (2006). Learning with Hypergraphs: Clustering, Classification, and Embedding. *Adv. neural Inf. Process. Syst.* 19, 1601–1608.

Zhou, Y., Zhang, Y., Lian, X., Li, F., Wang, C., Zhu, F., et al. (2022). Therapeutic Target Database Update 2022: Facilitating Drug Discovery with Enriched Comparative Data of Targeted Agents. *Nucleic Acids Res.* 50 (D1), D1398–D1407. doi:10.1093/nar/gkab953

Zhu, X.-J., Feng, C.-Q., Lai, H.-Y., Chen, W., and Hao, L. (2019). Predicting Protein Structural Classes for Low-Similarity Sequences by Evaluating Different Features. *Knowledge-Based Syst.* 163, 787–793. doi:10.1016/j.knosys.2018.10.007

Zou, Y., Wu, H., Guo, X., Peng, L., Ding, Y., Tang, J., et al. (2021). MK-FSVM-SVDD: a Multiple Kernel-Based Fuzzy SVM Model for Predicting DNA-Binding Proteins via Support Vector Data Description. *Cbio* 16 (2), 274–283. doi:10.2174/1574893615999200607173829

Zulfiqar, H., Yuan, S.-S., Huang, Q.-L., Sun, Z.-J., Dao, F.-Y., Yu, X.-L., et al. (2021). Identification of Cyclin Protein Using Gradient Boost Decision Tree Algorithm. *Comput. Struct. Biotechnol. J.* 19, 4123–4131. doi:10.1016/j.csbj.2021.07.013

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.

ORIGINAL RESEARCH
published: 22 July 2022
doi: 10.3389/fgene.2022.935989

CD8TCEI-EukPath: A Novel Predictor to Rapidly Identify CD8$^+$ T-Cell Epitopes of Eukaryotic Pathogens Using a Hybrid Feature Selection Approach

Rui-Si Hu[1], *Jin Wu*[2], *Lichao Zhang*[3], *Xun Zhou*[4]* and *Ying Zhang*[5]*

[1]Yangtze Delta Region Institute, University of Electronic Science and Technology of China, Quzhou, China, [2]School of Management, Shenzhen Polytechnic, Shenzhen, China, [3]School of Intelligent Manufacturing and Equipment, Shenzhen Institute of Information Technology, Shenzhen, China, [4]Beidahuang Industry Group General Hospital, Harbin, China, [5]Department of Anesthesiology, Hospital (T.C.M) Affiliated of Southwest Medical University, Luzhou, China

OPEN ACCESS

Edited by:
Quan Zou,
University of Electronic Science and Technology of China, China

Reviewed by:
Chen Cao,
University of Calgary, Canada
Ru Xiaoqing,
University of Tsukuba, Japan

***Correspondence:**
Xun Zhou
zhouxun20220428@163.com
Ying Zhang
zhangying021210@163.com

Specialty section:
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

Received: 04 May 2022
Accepted: 24 May 2022
Published: 22 July 2022

Citation:
Hu R-S, Wu J, Zhang L, Zhou X and Zhang Y (2022) CD8TCEI-EukPath: A Novel Predictor to Rapidly Identify CD8$^+$ T-Cell Epitopes of Eukaryotic Pathogens Using a Hybrid Feature Selection Approach. Front. Genet. 13:935989. doi: 10.3389/fgene.2022.935989

Computational prediction to screen potential vaccine candidates has been proven to be a reliable way to provide guarantees for vaccine discovery in infectious diseases. As an important class of organisms causing infectious diseases, pathogenic eukaryotes (such as parasitic protozoans) have evolved the ability to colonize a wide range of hosts, including humans and animals; meanwhile, protective vaccines are urgently needed. Inspired by the immunological idea that pathogen-derived epitopes are able to mediate the CD8$^+$ T-cell-related host adaptive immune response and with the available positive and negative CD8$^+$ T-cell epitopes (TCEs), we proposed a novel predictor called CD8TCEI-EukPath to detect CD8$^+$ TCEs of eukaryotic pathogens. Our method integrated multiple amino acid sequence-based hybrid features, employed a well-established feature selection technique, and eventually built an efficient machine learning classifier to differentiate CD8$^+$ TCEs from non-CD8$^+$ TCEs. Based on the feature selection results, 520 optimal hybrid features were used for modeling by utilizing the LightGBM algorithm. CD8TCEI-EukPath achieved impressive performance, with an accuracy of 79.255% in ten-fold cross-validation and an accuracy of 78.169% in the independent test. Collectively, CD8TCEI-EukPath will contribute to rapidly screening epitope-based vaccine candidates, particularly from large peptide-coding datasets. To conduct the prediction of CD8$^+$ TCEs conveniently, an online web server is freely accessible (http://lab.malab.cn/~hrs/CD8TCEI-EukPath/).

Keywords: eukaryotic pathogens, T-cell epitopes, machine learning, hybrid features, LightGBM

INTRODUCTION

Pathogen-derived antigen epitopes displayed on the surface of host antigen-presenting cells can be presented by major histocompatibility complex (MHC) molecules (also called human leukocyte antigen in humans) to the different subsets of T cells. Typically, MHC-I molecules present relatively fixed peptide lengths (usually 8–11 residues) to CD8$^+$ T cells, thereby activating cytotoxic T lymphocytes to destroy invading pathogens (Trolle et al., 2016), whereas MHC-II molecules with an open peptide-binding groove have the ability to recognize peptides of highly variable lengths (usually 9–22 residues) that activate $CD4^+$ helper or regulatory T cells (Holland et al., 2013). Obviously, antigen epitopes that trigger $CD8^+$ T cells or $CD4^+$ T cells bear essential differences during the process of host adaptive immune responses. Therefore, identifying what pathogen peptides will be presented to specific T cells is critical information for understanding infectious etiologies, developing diagnostic assays, and designing epitope-based vaccines against infectious agents.

Conventional approaches for T-cell epitope identification have depended entirely upon experimental technologies and experiences and are obviously time-consuming and costly. As a result, alternative computational approaches to implement antigen epitope identification have become powerful methods in immunology and vaccinology research and have significantly decreased the experimental load associated with epitope identification (Brusic et al., 2004; Zhang et al., 2012). To date, most T-cell epitope prediction tools have been developed using machine learning algorithms to train various experimental data, which are generally available in specialized epitope databases, such as the Immune Epitope Database (IEDB) (Vita et al., 2019). Since the first computational approach for epitope prediction was introduced more than 30 years (Sette et al., 1989), the performance of prediction methods in recent years has obtained significant advancement with the accumulation of positive epitope data, the development of machine learning algorithms, and the reduction of computational cost. These advancements are seen in the development of machine learning models to identify T-cell epitopes in various infectious agents, including pathogenic prokaryotes (such as bacteria) (Pamer et al., 1991; Nagpal et al., 2018; Zadeh Hosseingholi et al., 2020), viruses (Bukhari et al., 2021; Sharma et al., 2021; Xu et al., 2021), and pathogenic eukaryotes (such as parasitic protozoans) (Goodswen et al., 2014; Goodswen et al., 2021).

Among infectious agents, eukaryotic pathogens have evolved into several distinct phylogenetic lineages and bear resourceful abilities to affect a wide range of hosts, including humans and animals, resulting in significant effects on the aspects of global public health and considerable economic loss to the agricultural community (Haldar et al., 2006). Since a high level of MHC polymorphism in infected hosts and a large number of unknown functional proteins exist in eukaryotic pathogens (Hu et al., 2022), this undoubtedly produces challenges for T-cell epitope identification. Although presently some available software systems for *in silico* T-cell epitope prediction have been developed, including the NetCTL server (Larsen et al., 2005), the NetMHCpan server (Jurtz et al., 2017), and the MHCflurry server (O'Donnell et al., 2018), there is no guarantee that all these tools produce good quality predictions (Resende et al., 2012; Bordbar et al., 2020; Zawawi et al., 2020). Moreover, a general analysis of MHC-peptide binding prediction, overlooking specific patterns of MHC-presented peptides recognized by different types of T-cell receptors, may lead to lower predictive accuracy.

Given the wealth of state-of-the-art machine learning algorithms available and public experimental data, it is necessary to keep comparing the performance of different methods reciprocally and develop effective tools for the identification of T-cell epitopes in pathogen biology research. In the present study, based on MHC-I T-cell peptides collected from the IEDB database and experimentally validated neoantigen epitopes available from previous Review articles, we developed a novel machine learning-based method to identify $CD8^+$ T-cell immunogenic epitopes in eukaryotic pathogens. Our method adopted the best hybrid feature descriptor and classifier to establish a prediction model and finally achieved an accuracy of 79.255% in ten-fold cross-validation and an accuracy of 78.169% in the independent test. Finally, a user-friendly web server named CD8TCEI-EukPath was developed, which will be helpful for scientists to rapidly screen epitope-based vaccine candidates from a plethora of mass spectrometry peptidome data.

METHODS AND MATERIALS

Dataset Preparation

Eukaryotic pathogens (Eukpaths), such as protozoans and fungi, are important causative agents that cause serious infectious diseases in humans and animals; however, there is a lack of systematic collection of Eukpath-derived antigenic epitopes associated with the host immune response. Additionally, many previous works have pointed out that stringent datasets are considered important for the performance of a predictive model. In particular, peptide sequences for most T-cell epitopes (TCEs) usually have a short length, which easily leads to biased estimates if peptide sequences in a dataset have high similarity.

The present study collected datasets concerning positive and negative $CD8^+$ TCEs available from the IEDB database (http://www.iedb.org/), following the search strategy: Eukaryote T-cell and class I MHC restriction (accessed on 15 October 2021). After data processing, 809 TCEs and 1,715 non-TCEs for Eukpaths were retained as positive and negative datasets, respectively. A detailed description of Eukpaths is included in **Supplementary Table S1**. In addition, we also obtained 371 experimentally determined peptide sequences for host $CD8^+$ T cells that are described in the latest Review articles, in which have gave a detailed list regarding peptide sequences in three important parasites [i.e., *Plasmodium falciparum* (Heide et al., 2019), *Toxoplasma gondii* (Javadi Mamaghani et al., 2019), and *Trypanosoma cruzi* (Ferragut et al., 2021)] and, eventually, a total of 1,180 TCEs were reorganized as the positive dataset.

Before dividing the positive and negative datasets into training and testing sets, we performed data preprocessing, such as removing repeat sequences and sequences with high sequence identity. Repeat peptide sequences in the positive sample were removed using SeqKit software (Shen et al., 2016), and peptide sequences in positive and negative samples with more than 90% sequence identity were removed using the CD-HIT Suite server (Huang et al., 2010). Finally, a total of 706 TCEs from the positive dataset were retained, and an equal number of non-TCEs were randomly selected from negative datasets.

Regarding machine learning, training datasets are used to train a predictive model, and based on evaluation through testing sets, an optimal classifier was selected. We randomly selected 80% of datasets from both positive and negative

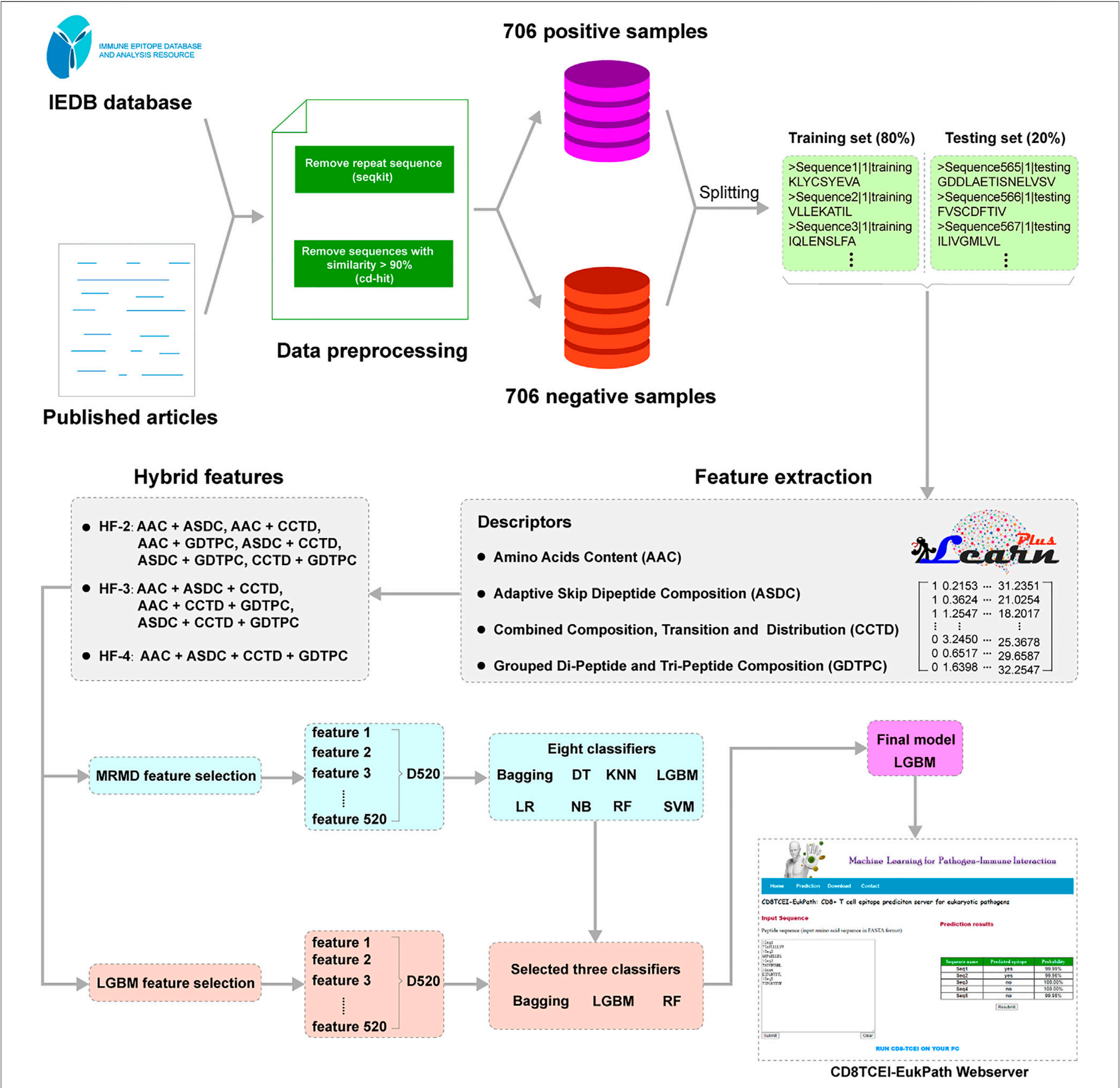


FIGURE 1 | A technology roadmap of the machine learning model proposed in this study.

samples as training sets and the remaining 20% as testing sets. Note that the training and testing sets can be downloaded from http://lab.malab.cn/~hrs/CD8TCEI-EukPath/download.html.

An Overview of the Established Predictor

A modeling overview of the proposed approach is illustrated in **Figure 1**. CD8TCEI-EukPath allows users to utilize a large volume of peptide sequences, such as peptide-coding datasets available from mass spectrometry peptidomics, to serve as input sequences. First, each sequence is subjected to the feature representation based on the proposed hybrid feature scheme. Regarding machine learning modeling, hybrid feature identification is a useful approach for improving prediction performance and has been extensively applied to the identification of specific peptide sequences, such as anticancer T-cell antigen epitopes (Wei et al., 2018; Beltran Lissabet et al., 2019; Charoenkwan et al., 2020; Jiao et al., 2021). The detailed hybrid feature representation method can be seen in the subsequent **Section 2.3**. Then, hybrid features for each sequence are transmitted to the well-trained prediction model. In the final evaluation of the models, we choose the LightGBM (LGBM) classifier as the optimal training model. Eventually, the LGBM-based model will give an estimated score in the prediction results to differentiate TCEs from non-TCEs. If the prediction possibility of more than 50% is considered to indicate the true TCE and lower values indicate non-TCEs, the prediction possibility is calculated with a range from 0 to 100%.

Feature Extraction

Protein or peptide sequences are composed of amino acids. In the standard amino acid alphabet, 20 different amino acids can be represented as {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, and Y}. To establish a machine learning model, an essential step is to extract amino acid features from protein or peptide sequences, typically regarding structural and physicochemical properties of amino acids through a transformation from sequence to numerical vector (Chen et al., 2018; Chen et al., 2020; Chen et al., 2021). In this study, the iLearnPlus platform (Chen et al., 2021) was used to conduct feature extraction of the peptide sequence, as described below.

1) Amino Acid Composition (AAC). AAC is a commonly used descriptor that has been successfully applied to protein classification and anticancer peptide prediction (Bhasin and Raghava, 2004; Wei et al., 2018; Jiao et al., 2021). AAC is encoded based on calculating the occurrence frequency of each amino acid in a peptide sequence. The frequencies of AAC can be calculated as follows:

where $N(i)$ represents the number of amino acid type i, L represents the length of a peptide sequence, and $f(i)$ is the calculated composition frequency for a specific amino acid type i.

2) Adaptive Skip Dinucleotide Composition (ASDC). The ASDC descriptor is a modified dipeptide composition proposed by Wei et al., 2017a and Wei et al., 2017b. This descriptor has the advantage that it not only fully considers the relevant information between adjacent residues but also considers the intermediate residue. For a given protein or peptide sequence, this descriptor can generate 400-dimensional feature vectors (*fv*) that are presented by:

$$ASDC = (fv1,\ fv2,\ fv3, \ldots, fv400)$$

$$fvi = \frac{\sum_{k=1}^{M-1} N_i^k}{\sum_{i=1}^{400} \sum_{k=1}^{M-1} N_i^k}$$

$f(i) = \frac{N(i)}{L}$, $i \in \{\text{A, C, D...Y}\}$

In the formula, *fvi* indicates the occurrence frequency of all possible dipeptide pairs with $\leq M-1$ intervening amino acids.

3) Combined Composition, Transition, and Distribution (CCTD). The CCTD features represent a global description of amino acids' structural or physicochemical attributes, such as hydrophobicity, normalized van der Waals volume, polarity, polarizability, charge, secondary structures, and solvent accessibility of a peptide sequence (Dubchak et al., 1995; Tomii and Kanehisa, 1996; Dubchak et al., 1999). The CCTD contains three descriptors, namely, composition (C), transition (T), and distribution (D).

- Composition: The composition descriptor computes the percentage frequency of polar (RKEDQN), neutral (GASTPHY), and hydrophobic (CLVIMFW) residues in a given peptide sequence. It can be calculated as follows:

$$C = \frac{N(i)}{L},\ i \in \{polar,\ neutral,\ or\ hydrophobic\ residues\}$$

N (i) represents the number of amino acid type *i* in the encoded sequence, and *L* represents the length of the peptide sequence.

- Transition: The transition descriptor indicates the percentage frequency of amino acids that transition between the three groups, i.e., polar, neutral, and hydrophobic groups. The formula can be defined as follows:

$$T(i,j) = \frac{N(i,j) + N(j,i)}{L-1}$$
$$i, j \in \{(polar,\ neutral),\ (neutral,\ hydrophobic),\ (hydrophobic, polar)\}$$

where $N(i,j)$ and $N(j,i)$ represent the number of dipeptides that appeared in '*i, j*' and '*j, i*', respectively, and *L* represents the length of a peptide sequence.

- Distribution: The distribution descriptor describes the distribution of amino acids for each of the three groups (polar, neutral, and hydrophobic) in the sequence. There are five descriptor values for each group, and they are the corresponding position fractions in the entire sequence concerning first residues, 25% residues, 50% residues, 70% residues, and 100% residues.

4) Grouped Di-Peptide/Tri-Peptide Composition (GDTPC). The 20 different amino acids can be categorized into five classes, including aliphatic group–*g1* (GAVLMI), aromatic group–*g2* (FYW), positively charged group–*g3* (KRH), negatively charged group–*g4* (DE), and uncharged group–*g5* (STCPNQ), according to their physicochemical properties, such as hydrophobicity, charge and molecular size of amino acids in a peptide sequence (Lee et al., 2011). In this study, the grouped di-peptide composition (GDPC) and the grouped tri-peptide composition (GTPC) are combined to present the feature vector in the peptide sequence. The GDPC is a variation of the di-peptide composition descriptor and can generate 25 descriptors (Chen et al., 2018; Chen et al., 2021). It is defined as follows:

$$f(i,j) = \frac{N_{ij}}{L-1},\ i, j \in \{g1, g2, g3, g4, g5\}$$

where N_{ij} is the number of dipeptides coded by amino acid type groups i and j, and L represents the length of a peptide sequence.

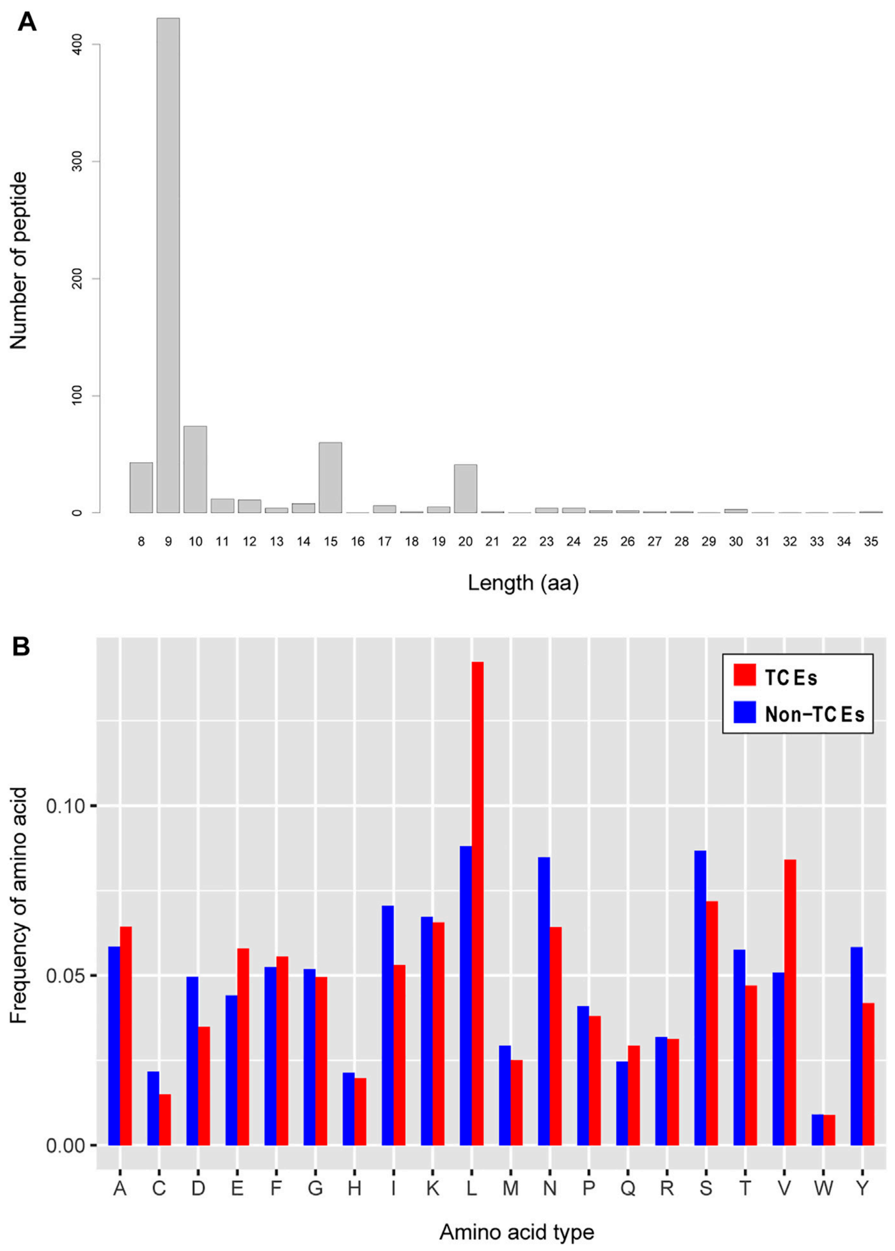


FIGURE 2 | Analysis of amino acid sequence features. **(A)** Length distribution of the positive CD8$^+$ T-cell epitopes. The horizontal axis represents the length of amino acids, and the vertical axis represents the number of epitopes in positive samples. **(B)** Distribution features of amino acid types with respect to the positive and negative CD8$^+$ T-cell epitopes. The horizontal axis represents the twenty amino acids, and the vertical axis represents the occurrence frequency of an amino acid in all sequences.

The GTPC is also a variation of the tri-peptide composition descriptor, and a total of 125 descriptors can be generated for a given peptide sequence (Chen et al., 2018; Chen et al., 2021). It is defined as follows:

$$f(i, j, s) = \frac{N_{ijs}}{L-1}, \ i, j, s \in \{g1, g2, g3, g4, g5\}$$

where N_{ijs} is the number of tripeptides coded by amino acid type groups i, j, and s, and L represents the length of a peptide sequence.

Feature Selection

Feature selection is an important process that can effectively reduce the number of redundant variables and the computational cost as well as solve overfitting problems in machine learning modeling. A variety of feature selection tools have been developed and applied to the identification of peptide sequences (Wang et al., 2013; Zou et al., 2016; Jung et al., 2019; He et al., 2020; Meng et al., 2020; Mostafa et al., 2020). For the first method applied to the optimal feature selection, we decided to utilize the MRMD

TABLE 1 | The accuracy (Acc) results of a single feature descriptor classified by different machine learning algorithms.

Feature descriptors		Classifiers and Acc values (%)							
		Bagging	DT	KNN	LGBM	LR	NB	RF	SVM
Ten-fold cross-validation	AAC	71.454	66.667	68.174	**73.670**	65.160	65.603	**73.670**	67.730
	ASDC	72.252	65.160	67.642	**75.443**	66.223	66.755	74.468	74.291
	CTDC	67.199	62.145	66.933	**68.528**	64.628	61.259	68.351	68.351
	CTDT	66.667	60.638	64.805	**67.996**	63.032	60.372	67.908	66.401
	CTDD	71.986	64.982	69.326	**75.089**	66.312	66.401	72.606	68.883
	GDPC	64.894	59.309	60.638	65.514	59.929	57.624	**66.223**	63.209
	GTPC	67.287	62.057	63.564	68.174	60.284	59.663	**68.528**	65.071
Independent test	AAC	**76.408**	67.606	73.592	76.056	66.901	65.493	74.648	67.606
	ASDC	75.352	65.845	69.366	**76.761**	67.254	69.014	75.704	74.296
	CTDC	72.535	63.028	66.197	70.070	67.254	63.380	**73.944**	71.479
	CTDT	65.493	60.915	65.141	65.141	66.197	60.211	**68.662**	64.085
	CTDD	69.014	62.324	66.197	72.183	68.662	66.901	**72.887**	70.775
	GDPC	63.028	50.704	59.859	64.789	61.268	61.972	**67.254**	65.141
	GTPC	66.197	59.859	65.141	**72.887**	63.380	63.028	68.662	66.197

The best Acc values to reflect the performance of different classifiers were highlighted in bold font.

tool (http://lab.malab.cn/soft/MRMD3.0/index.html) (Zou et al., 2016; He et al., 2020) following the PageRank strategy. MRMD is a feature ranking method based on function distance calculated by Pearson's correlation coefficient to measure the independence of every feature and generates a sub-feature set with a low redundancy but strong relevance with the target class. The second method was the LGBM algorithm (Ke et al., 2017), which was used to select the best feature subsets based on the ranking of feature significance calculated by the LGBM classifier. Finally, the features selected by the MRMD and LGBM methods were used for modeling.

Classifier Selection

In this study, eight popular machine learning algorithms were used, including Bagging, decision tree (DT), neighbors (KNN), light gradient boosting machine (LGBM), logistic regression (LR), GaussianNB (NB), random forest (RF) and support vector machines (SVM), to select a suitable algorithm for machine learning modeling. These algorithms are built into the scikit-learn toolkit package (Pedregosa et al., 2011), which can run in the *Python* program. Based on the feature selection matrix generated from the MRMD program, with regard to the eight algorithms, default hyperparameters were used for the initial evaluation during the process of classification performance. Additionally, we optimized the hyperparameters and selected the three most suitable classifiers, namely, RF, LGBM, and Bagging, for further comparative analysis. The best parameters were determined by grid search techniques, and the detailed settings are compiled in **Supplementary Table S2**.

Performance Evaluation and Methods

For each predictive model, the quality was evaluated by measurement metrics for ten-fold cross-validation and an independent test method. In terms of measurement metrics, we used four standard evaluation metrics, namely, sensitivity (Se), specificity (Sp), accuracy (Acc), and Matthew correlation coefficient (MCC), to evaluate a model's performance. These metrics were formulated as follows:

$$Se = \frac{TP}{TP + FN} \times 100\%$$

$$Sp = \frac{TN}{TN + FP} \times 100\%$$

$$Acc = \frac{TP + TN}{TP + TN + FN + FP} \times 100\%$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FN) \times (TP + FP) \times (TN + FP) \times (TN + FN)}}$$

where TP, TN, FP, and FN indicate the sample numbers of true positives, true negatives, false-positives, and false negatives, respectively. The Se of a test is also called the true positive rate and refers to the proportion of samples that are correctly classified as positive samples in the dataset among all real positive samples. The Sp of a test is also called the true negative rate and is the proportion of samples that are correctly classified as negative samples in the dataset among all real negative samples. Another two metrics, Acc and MCC, can comprehensively evaluate the performance of a predictor on balanced data. The Acc metric represents the ratio of a sample number of correct predictions to all numbers of input samples, but the MCC metric takes the ratio of positive and negative elements into account. Therefore, for unbalanced data, MCC would display a better predictive quality than Acc (Chicco and Jurman, 2020).

Additionally, the area under the receiver operating characteristic (auROC, or AUC) curve was introduced to evaluate the performance of a predictor. The auROC curve is plotted with a true positive rate on the *Y*-axis and the false-positive rate on the *X*-axis, with values ranging from 0 to 1. Having the auROC curve near the upper left or an auROC curve value = 1 reflects perfect prediction, while having an auROC curve value of 0.5 suggests random prediction of a model.

TABLE 2 | The classification results of different hybrid feature combinations detected by the LGBM classifier.

Hybrid features	Ten-fold cross-validation				Independent test			
	Acc (%)	MCC	Se (%)	Sp (%)	Acc (%)	MCC	Se (%)	Sp (%)
AAC + ASDC + CCTD + GDTPC	**79.255**	**0.585**	**77.837**	**80.674**	**78.169**	**0.563**	**78.873**	77.465
ASDC + CCTD + GDTPC	78.103	0.562	76.596	79.610	76.056	0.521	77.465	74.648
AAC + ASDC + CCTD	77.660	0.553	76.064	79.255	77.113	0.542	76.056	78.169
AAC + CCTD + GDTPC	77.305	0.546	76.596	78.014	75.352	0.507	74.648	76.056
ASDC + CCTD	77.482	0.550	76.950	78.014	76.761	0.535	78.169	75.352
AAC + CCTD	76.684	0.534	75.887	77.482	75.352	0.507	76.056	74.648
ASDC + GDTPC	76.152	0.523	74.468	77.837	75.704	0.515	72.535	**78.873**
CCTD + GDTPC	76.064	0.522	74.468	77.660	77.465	0.550	**78.873**	76.056
AAC + ASDC	75.621	0.512	75.000	76.241	72.535	0.451	70.423	74.648
AAC + GDTPC	74.911	0.499	73.227	76.596	77.817	0.556	76.761	**78.873**
CCTD	75.621	0.513	74.645	76.596	74.648	0.495	**78.873**	70.423
GDTPC	69.592	0.392	68.262	70.922	72.535	0.451	71.831	73.239

The best metric values were highlighted in bold font.

TABLE 3 | A comparison of classification results by a pairwise combination of two feature selection techniques (MRMD and LGBM) and three optimal classifiers (LGBM, RF, and Bagging).

Method	Ten-fold cross-validation				Independent test			
	Acc (%)	MCC	Se (%)	Sp (%)	Acc (%)	MCC	Se (%)	Sp (%)
MRMD + LGBM	**79.255**	**0.585**	77.837	**80.674**	**78.169**	**0.563**	**78.873**	**77.465**
LGBM + LGBM	78.457	0.569	**78.014**	78.901	77.113	0.542	77.465	76.761
MRMD + RF	75.887	0.518	73.404	78.369	74.648	0.493	72.535	76.761
LGBM + RF	75.355	0.507	74.645	76.064	74.648	0.493	73.944	75.352
MRMD + Bagging	73.316	0.466	72.163	74.468	75.352	0.507	73.239	77.465
LGBM + Bagging	74.202	0.484	72.695	75.709	75.704	0.514	76.056	75.352

The best metric values were highlighted in bold font.

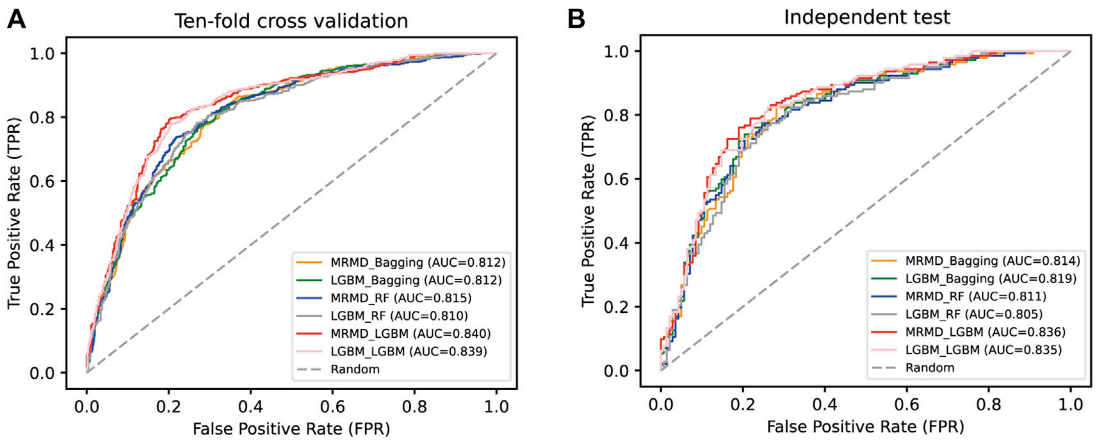


FIGURE 3 | A comparison of the AUC curve in ten-fold cross-validation **(A)** and independent test **(B)**. Results were by a pairwise combination of two feature selection techniques (MRMD and LGBM) and three optimal classifiers (LGBM, RF, and Bagging).

RESULTS AND DISCUSSION

Analysis of Peptide Sequence Features

In terms of peptide length, antigen epitopes that are presented to CD8$^+$ T cells by MHC-I molecules are typical peptides between 8 and 11 amino acids in length, and occasionally a few noncanonical lengths overstep this range (Trolle et al., 2016). Additionally, the sequence characteristics of T-cell epitopes should largely reflect the specific binding ability to the MHC allele in the process of eliciting immune responses induced by pathogen infection. Motivated by these observations, we first investigated the length distribution of positive T-cell epitope sequences. The results are illustrated in **Figure 2A**, which shows the main distribution of sequence length is 9-mer

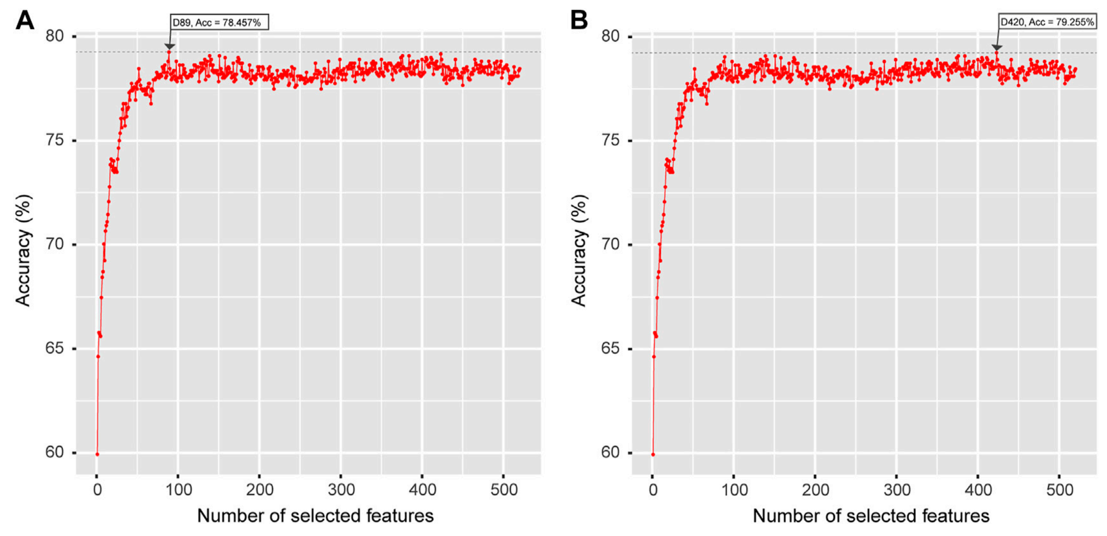


FIGURE 4 | The optimal feature sets selected by LGBM feature importance ranking **(A)** and a well-established MRMD strategy **(B)**. The horizontal axis represents the number of selected features, and the vertical axis represents the accuracy value calculated by the LGBM classifier.

peptides and that much longer peptides reach a length of up to 35 aa. As shown in **Figure 2B**, we also observed significant preferences in terms of amino acid appearance frequency between TCEs and non-TCEs, especially for leucine (L). Previous evidence has demonstrated that L is an important amino acid to mediate the adaptive immune response; specifically, L can play a role in T-cell activation and proliferation of immune cells (Ananieva et al., 2016). This implies that the preference of L in positive TCEs is essential feature information, in which the role of L not only serves as a biological activator of T-cell immunity but also may contribute to discriminating TCEs from non-TCEs.

Initial Evaluation of a Single Feature Descriptor on Different Classifiers

In our scheme on feature learning, we evaluated the performance of individual feature descriptors by the utilization of eight extensively used machine learning classifiers, i.e., Bagging, DT, KNN, LGBM, LR, NB, RF, and SVM. These models were evaluated thoroughly by ten-fold cross-validation, and their performances were compared reciprocally. A detailed summary of these evaluation results is compiled in **Supplementary Table S3**, and the Acc values of all classifiers are shown in **Table 1**. Given that each descriptor has a fair comparison with the eight classifiers, as shown in **Table 1**, we noticed that in each feature descriptor, there were three classifiers, namely, Bagging, LGBM, and RF, that had better performances than other classifiers (the best Acc values are highlighted in bold font). In the results of ten-fold cross-validation, the LGBM classifier had the best performance on five feature descriptors (AAC, ASDC, CTDC, CTDT, and CTDD), followed by the RF classifier on three feature descriptors (AAC, GDPC, and GTPC); however, in the individual test result, there were four feature descriptors (CTDC, CTDT, CTDD, and GDPC) on the RF classifier that had the best performance, followed by two feature descriptors (ASDC and GTPC) on the LGBM classifier and one feature descriptor (AAC) on the Bagging classifier. Remarkably, the feature descriptor ASDC worked on the LGBM classifier and was able to obtain the best prediction results in both ten-fold cross-validation and the independent test, with Acc values of 75.443% and 76.761%, respectively. Therefore, the LGBM classifier can be chosen as the most suitable classifier for model deployment, if only a single feature descriptor is being considered.

Comparison of Hybrid Multisource Features on Different Classifiers

Compared to machine learning techniques, in general, the sequence feature is a more critical element to achieve high accuracy in biological sequence classification, especially for the extensive applications of combining hybrid multisource features in machine learning modeling (Zhang et al., 2016; Mohan et al., 2019; Charoenkwan et al., 2020; Ao et al., 2021; Jiao et al., 2021). Based on the feature descriptors mentioned in **Section 3.2**, we combined similar feature types as a hybrid group, including CTDC + CTDT + CTDD (CCTD) and GDPC + GTPC (GDTPC), and the performances of four groups (AAC, ASDC, CCTD, and GDTPC) were compared thoroughly on the eight classifiers using ten-fold cross-validation. The detailed prediction results are summarized in **Supplementary Table S2**, and we reconfirmed that LGBM is a satisfactory classifier to differentiate TCEs from non-TCEs. To compare the performances of various hybrid features, as shown in **Table 2**, the prediction results of the LGBM classifier were generated based on the ten-fold cross-validation and independent test. Strikingly, the majority of prediction results of LGBM using hybrid features had an Acc value of more than 75%, which indicated that the prediction ability was greatly improved when compared to the single features. We also observed from **Table 2** that the ten-fold cross-validation results of the AAC + ASDC + CCTD + GDTPC combination in particular, with metric values of 79.255% Acc, 0.585 MCC, 77.837% Se, and 80.674% Sp outperformed all the single or hybrid features, and therefore, this combination feature was selected for the subsequent analyses.

MRMD Serves as a Powerful Feature Selection Technology That Determines the Optimal Feature Space

Various feature selection technologies can be used for representation learning features. In the present study, we compared two feature selection technologies (MRMD and LGBM) by calculating the feature importance values, including the PageRank-based value for MRMD and Gini-based feature importance value for LGBM. Among the feature list results obtained by the two methods, we selected the top 520 features and employed the incremental feature selection (IFS) strategy to determine the optimal feature vector spaces, which are subsequently predicted on LGBM, RF, and Bagging classifiers.

As shown in **Table 3**, the ten-fold cross-validation results suggested that the MRMD + LGBM combination yielded the best prediction capability, with 79.255% Acc, 0.585 MCC, and 80.674% Sp, compared to the other five models (LGBM + LGBM, MRMD + RF, LGBM + RF, MRMD + Bagging, and LGBM + Bagging), except that the Se of MRMD + LGBM of 77.837% was lower than that of the LGBM + LGBM model; however, an independent test indicated that the MRMD + LGBM model was better than the other five combinations in all metrics. Furthermore, a separate AUC curve analysis is shown in **Figure 3** and further illustrated that the MRMD + LGBM model with an AUC value of 0.840 in ten-fold cross-validation and an AUC value of 0.836 in the independent test outperformed the other five models.

The optimal feature vector spaces detected by the IFS strategy suggested that the maximum accuracy of the LGBM + LGBM model was 78.457% with 89 features, which was less than the maximum accuracy of the MRMD + LGBM model of 79.255% with 420 features **Figure 4**. In the case of evaluating the computational cost of both models and considering the stability and robustness of the models, the MRMD + LGMB combination was finally selected as the best strategy for modeling and webserver development.

User Guide of the Established Webserver

A user-friendly web server called CD8TCEI-EukPath was developed, and the users can freely enter the homepage *via* the link http://lab.malab.cn/~hrs/CD8TCEI-EukPath/. The prediction interface can be accessed by clicking the "Prediction" or "CD8TCEI-EukPath" hyperlink, where the users can utilize amino acid sequences (FASTA format) to identify whether the input sequences are CD8TCEs or non-CD8TCEs. Briefly, the users should use short peptide sequences (generally 8–11 aa in length), paste the FASTA sequences in the left box, and click the "Submit" button for calculation. Immediately, the prediction results will be shown in the right box, which includes the protein name, predicted epitope (yes or no), and probability of belonging to $CD8^+$ TCEs. If starting a new task, the users can click the "Resubmit" button and/or click the "Clear" button and paste new sequences to conduct computational predictions. Note that the computing resources of the webserver are limited for high-volume predictions, and the maximum number of sequences should be 1,000 at a time. In addition, using the established model, we also provided the prediction results of five important pathogen species (*Plasmodium*, *Toxoplasma*, *Trypanosoma*, *Leishmania*, and *Giardia*) based on the available mass spectrometry peptidome data obtained from the ProteomeXchange database (http://www.proteomexchange.org/). These prediction results can be downloaded freely from our web server and need to be further evaluated by MHC-peptide binding predictions and biological experiments.

Conclusion

By comparing the performances of various single feature descriptors and hybrid feature descriptors using eight different classifiers, we selected a set of best features (AAC + ASDC + CCTD + GDTPC) and a satisfactory classifier (LGBM) for machine learning modeling. Following the state-of-the-art feature selection strategy of MRMD 3.0, we developed an effective sequence-based predictor named CD8TCEI-EukPath, capable of rapidly identifying eukaryotic pathogen-derived antigen epitopes for host $CD8^+$ T cells from large peptide-coding datasets. As a first sequence-based predictor to identify T-cell epitopes in eukaryotic pathogens, CD8TCEI-EukPath achieved 79.255% Acc in ten-fold cross-validation and 78.169% Acc, 0.563 MCC, 78.873% Se and 77.465% Sp in the independent test. Meanwhile, a user-friendly web server was developed in the present work. We believe that this tool is helpful for scientists to evaluate the immunogenicity of a given peptide sequence before performing biological experiments. The current tool only applies to the identification of $CD8^+$ T-cell epitopes in eukaryotic pathogens. In future works, we will apply deep representation learning features and state-of-the-art classification algorithms for $CD4^+$ T-cell epitope and B-cell epitope prediction. By leveraging machine learning models to develop auxiliary tools, their combinations will assist in the development of peptide-based vaccines.

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/s.

AUTHOR CONTRIBUTIONS

XZ and YZ contributed to the conception of the study; R-SH built the model and wrote the original manuscript; JW and LZ participated in the data analysis. The authors read and approved the final manuscript.

FUNDING

This work was supported by the grant from the National Natural Science Foundation of China (62101353 and 62101354), and the Special Science Foundation of Quzhou (2021D004).

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2022.935989/full#supplementary-material

REFERENCES

Ananieva, E. A., Powell, J. D., and Hutson, S. M. (2016). Leucine Metabolism in T Cell Activation: mTOR Signaling and beyond. *Adv. Nutr.* 7, 798S–805S. doi:10.3945/an.115.011221

Ao, C., Yu, L., and Zou, Q. (2021). RFhy-m2G: Identification of RNA N2-Methylguanosine Modification Sites Based on Random Forest and Hybrid Features. *Methods* S1046-2023 (21), 00142–0. doi:10.1016/j.ymeth.2021.05.016

Beltrán Lissabet, J. F., Herrera Belén, L., and Farias, J. G. (2019). TTAgP 1.0: A Computational Tool for the Specific Prediction of Tumor T Cell Antigens. *Comput. Biol. Chemistry* 83, 107103. doi:10.1016/j.compbiolchem.2019.107103

Bhasin, M., and Raghava, G. P. S. (2004). Classification of Nuclear Receptors Based on Amino Acid Composition and Dipeptide Composition. *J. Biol. Chemistry* 279, 23262–23266. doi:10.1074/jbc.M401932200

Bordbar, A., Bagheri, K. P., Ebrahimi, S., and Parvizi, P. (2020). Bioinformatics Analyses of Immunogenic T-Cell Epitopes of LeIF and PpSP15 Proteins from *Leishmania Major* and Sand Fly Saliva Used as Model Antigens for the Design of A Multi-Epitope Vaccine to Control Leishmaniasis. *Infect. Genet. Evol.* 80, 104189. doi:10.1016/j.meegid.2020.104189

Brusic, V., Bajic, V. B., and Petrovsky, N. (2004). Computational Methods for Prediction of T-Cell Epitopes-A Framework for Modelling, Testing, and Applications. *Methods* 34, 436–443. doi:10.1016/j.ymeth.2004.06.006

Bukhari, S. N. H., Jain, A., Haq, E., Khder, M. A., Neware, R., Bhola, J., et al. (2021). Machine Learning-Based Ensemble Model for Zika Virus T-Cell Epitope Prediction. *J. Healthc. Eng.* 2021, 1–10. doi:10.1155/2021/9591670

Charoenkwan, P., Nantasenamat, C., Hasan, M. M., and Shoombuatong, W. (2020). iTTCA-Hybrid: Improved and Robust Identification of Tumor T Cell Antigens by Utilizing Hybrid Feature Representation. *Anal. Biochemistry* 599, 113747. doi:10.1016/j.ab.2020.113747

Chen, Z., Zhao, P., Li, C., Li, F., Xiang, D., Chen, Y.-Z., et al. (2021). iLearnPlus: A Comprehensive and Automated Machine-Learning Platform for Nucleic Acid and Protein Sequence Analysis, Prediction and Visualization. *Nucleic Acids Res.* 49, e60. doi:10.1093/nar/gkab122

Chen, Z., Zhao, P., Li, F., Leier, A., Marquez-Lago, T. T., Wang, Y., et al. (2018). iFeature: A Python Package and Web Server for Features Extraction and Selection from Protein and Peptide Sequences. *Bioinformatics* 34, 2499–2502. doi:10.1093/bioinformatics/bty140

Chen, Z., Zhao, P., Li, F., Marquez-Lago, T. T., Leier, A., Revote, J., et al. (2020). iLearn: An Integrated Platform and Meta-Learner for Feature Engineering, Machine-Learning Analysis and Modeling of DNA, RNA and Protein Sequence Data. *Brief. Bioinform* 21, 1047–1057. doi:10.1093/bib/bbz041

Chicco, D., and Jurman, G. (2020). The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation. *BMC Genomics* 21, 6. doi:10.1186/s12864-019-6413-7

Dubchak, I., Muchnik, I., Holbrook, S. R., and Kim, S. H. (1995). Prediction of Protein Folding Class Using Global Description of Amino Acid Sequence. *Proc. Natl. Acad. Sci. U.S.A.* 92, 8700–8704. doi:10.1073/pnas.92.19.8700

Dubchak, I., Muchnik, I., Mayor, C., Dralyuk, I., and Kim, S.-H. (1999). Recognition of a Protein Fold in the Context of the SCOP Classification. *Proteins* 35, 401–407. doi:10.1002/(sici)1097-0134(19990601)35:4<401:aid-prot3>3.0.co;2-k

Ferragut, F., Acevedo, G. R., and Gómez, K. A. (2021). T Cell Specificity: A Great Challenge in Chagas Disease. *Front. Immunol.* 12, 674078. doi:10.3389/fimmu.2021.674078

Goodswen, S. J., Kennedy, P. J., and Ellis, J. T. (2021). Applying Machine Learning to Predict the Exportome of Bovine and Canine Babesia Species that Cause Babesiosis. *Pathogens* 10, 660. doi:10.3390/pathogens10060660

Goodswen, S. J., Kennedy, P. J., and Ellis, J. T. (2014). Vacceed: A High-Throughput *In Silico* Vaccine Candidate Discovery Pipeline for Eukaryotic Pathogens Based on Reverse Vaccinology. *Bioinformatics* 30, 2381–2383. doi:10.1093/bioinformatics/btu300

Haldar, K., Kamoun, S., Hiller, N. L., Bhattacharje, S., and Van Ooij, C. (2006). Common Infection Strategies of Pathogenic Eukaryotes. *Nat. Rev. Microbiol.* 4, 922–931. doi:10.1038/nrmicro1549

He, S., Guo, F., Zou, Q., and HuiDing, fnm. (2021). MRMD2.0: A Python Tool for Machine Learning with Feature Ranking and Reduction. *Cbio* 15, 1213–1221. doi:10.2174/1574893615999200503030350

Heide, J., Vaughan, K. C., Sette, A., Jacobs, T., and Schulze Zur Wiesch, J. (2019). Comprehensive Review of Human *Plasmodium Falciparum*-Specific CD8+ T Cell Epitopes. *Front. Immunol.* 10, 397. doi:10.3389/fimmu.2019.00397

Holland, C. J., Cole, D. K., and Godkin, A. (2013). Re-Directing CD4+ T Cell Responses with the Flanking Residues of MHC Class II-Bound Peptides: The Core Is Not Enough. *Front. Immunol.* 4, 172. doi:10.3389/fimmu.2013.00172

Hu, R.-S., Hesham, A. E.-L., and Zou, Q. (2022). Machine Learning and its Applications for Protozoal Pathogens and Protozoal Infectious Diseases. *Front. Cell. Infect. Microbiol.* 12 (12), 882995. doi:10.3389/fcimb.2022.882995

Huang, Y., Niu, B., Gao, Y., Fu, L., and Li, W. (2010). CD-HIT Suite: A Web Server for Clustering and Comparing Biological Sequences. *Bioinformatics* 26, 680–682. doi:10.1093/bioinformatics/btq003

Javadi Mamaghani, A., Fathollahi, A., Spotin, A., Ranjbar, M. M., Barati, M., Aghamolaie, S., et al. (2019). Candidate Antigenic Epitopes for Vaccination and Diagnosis Strategies of *Toxoplasma Gondii* Infection: A Review. *Microb. Pathog.* 137, 103788. doi:10.1016/j.micpath.2019.103788

Jiao, S., Zou, Q., Guo, H., and Shi, L. (2021). iTTCA-RF: A Random Forest Predictor for Tumor T Cell Antigens. *J. Transl. Med.* 19, 449. doi:10.1186/s12967-021-03084-x

Jung, Y., Zhang, H., and Hu, J. (2019). Transformed Low-Rank ANOVA Models for High-Dimensional Variable Selection. *Stat. Methods Med. Res.* 28, 1230–1246. doi:10.1177/0962280217753726

Jurtz, V., Paul, S., Andreatta, M., Marcatili, P., Peters, B., and Nielsen, M. (2017). NetMHCpan-4.0: Improved Peptide-MHC Class I Interaction Predictions Integrating Eluted Ligand and Peptide Binding Affinity Data. *J. I.* 199, 3360–3368. doi:10.4049/jimmunol.1700893

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., et al. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Adv. neural Inf. Process. Syst.* 30, 3146–3154.

Larsen, M. V., Lundegaard, C., Lamberth, K., Buus, S., Brunak, S., Lund, O., et al. (2005). An Integrative Approach to CTL Epitope Prediction: A Combined Algorithm Integrating MHC Class I Binding, TAP Transport Efficiency, and Proteasomal Cleavage Predictions. *Eur. J. Immunol.* 35, 2295–2303. doi:10.1002/eji.200425811

Lee, T.-Y., Lin, Z.-Q., Hsieh, S.-J., Bretaña, N. A., and Lu, C.-T. (2011). Exploiting Maximal Dependence Decomposition to Identify Conserved Motifs from A Group of Aligned Signal Sequences. *Bioinformatics* 27, 1780–1787. doi:10.1093/bioinformatics/btr291

Meng, C., Wu, J., Guo, F., Dong, B., and Xu, L. (2020). CWLy-pred: A Novel Cell Wall Lytic Enzyme Identifier Based on an Improved MRMD Feature Selection Method. *Genomics* 112, 4715–4721. doi:10.1016/j.ygeno.2020.08.015

Mohan, S., Thirumalai, C., and Srivastava, G. (2019). Effective Heart Disease Prediction Using Hybrid Machine Learning Techniques. *IEEE access* 7, 81542–81554. doi:10.1109/ACCESS.2019.2923707

Mostafa, S. S., Morgado-Dias, F., and Ravelo-García, A. G. (2020). Comparison of SFS and mRMR for Oximetry Feature Selection in Obstructive Sleep Apnea Detection. *Neural Comput. Applic* 32, 15711–15731. doi:10.1007/s00521-018-3455-8

Nagpal, G., Usmani, S. S., and Raghava, G. P. S. (2018). A Web Resource for Designing Subunit Vaccine against Major Pathogenic Species of Bacteria. *Front. Immunol.* 9, 2280. doi:10.3389/fimmu.2018.02280

O'donnell, T. J., Rubinsteyn, A., Bonsack, M., Riemer, A. B., Laserson, U., and Hammerbacher, J. (2018). MHCflurry: Open-Source Class I MHC Binding Affinity Prediction. *Cell. Syst.* 7, 129–132 e124. doi:10.1016/j.cels.2018.05.014

Pamer, E. G., Harty, J. T., and Bevan, M. J. (1991). Precise Prediction of A Dominant Class I MHC-Restricted Epitope of *Listeria Monocytogenes*. *Nature* 353, 852–855. doi:10.1038/353852a0

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12, 2825–2830.

Resende, D. M., Rezende, A. M., Oliveira, N. J., Batista, I. C., Corrêa-Oliveira, R., Reis, A. B., et al. (2012). An Assessment on Epitope Prediction Methods for Protozoa Genomes. *BMC Bioinforma.* 13, 309. doi:10.1186/1471-2105-13-309

Sette, A., Buus, S., Appella, E., Smith, J. A., Chesnut, R., Miles, C., et al. (1989). Prediction of Major Histocompatibility Complex Binding Regions of Protein Antigens by Sequence Pattern Analysis. *Proc. Natl. Acad. Sci. U.S.A.* 86, 3296–3300. doi:10.1073/pnas.86.9.3296

Sharma, G., Rana, P. S., and Bawa, S. (2021). Hybrid Machine Learning Models for Predicting Types of Human T-Cell Lymphotropic Virus. *IEEE/ACM Trans. Comput. Biol. Bioinf.* 18, 1524–1534. doi:10.1109/TCBB.2019.2944610

Shen, W., Le, S., Li, Y., and Hu, F. (2016). SeqKit: A Cross-Platform and Ultrafast Toolkit for FASTA/Q File Manipulation. *PLoS One* 11, e0163962. doi:10.1371/journal.pone.0163962

Tomii, K., and Kanehisa, M. (1996). Analysis of Amino Acid Indices and Mutation Matrices for Sequence Comparison and Structure Prediction of Proteins. *Protein Eng. Des. Sel.* 9, 27–36. doi:10.1093/protein/9.1.27

Trolle, T., Mcmurtrey, C. P., Sidney, J., Bardet, W., Osborn, S. C., Kaever, T., et al. (2016). The Length Distribution of Class I-Restricted T Cell Epitopes Is Determined by Both Peptide Supply and MHC Allele-specific Binding Preference. *J. I.* 196, 1480–1487. doi:10.4049/jimmunol.1501721

Vita, R., Mahajan, S., Overton, J. A., Dhanda, S. K., Martini, S., Cantrell, J. R., et al. (2019). The Immune Epitope Database (IEDB): 2018 Update. *Nucleic Acids Res.* 47, D339–D343. doi:10.1093/nar/gky1006

Wang, J., Zhang, D., and Li, J. (2013). PREAL: Prediction of Allergenic Protein by Maximum Relevance Minimum Redundancy (mRMR) Feature Selection. *BMC Syst. Biol.* 7 (Suppl. 5), S9. doi:10.1186/1752-0509-7-S5-S9

Wei, L., Tang, J., and Zou, Q. (2017a). SkipCPP-Pred: An Improved and Promising Sequence-Based Predictor for Predicting Cell-Penetrating Peptides. *BMC Genomics* 18, 742. doi:10.1186/s12864-017-4128-1

Wei, L., Xing, P., Su, R., Shi, G., Ma, Z. S., and Zou, Q. (2017b). CPPred-RF: A Sequence-Based Predictor for Identifying Cell-Penetrating Peptides and Their Uptake Efficiency. *J. Proteome Res.* 16, 2044–2053. doi:10.1021/acs.jproteome.7b00019

Wei, L., Zhou, C., Chen, H., Song, J., and Su, R. (2018). ACPred-FL: A Sequence-Based Predictor Using Effective Feature Representation to Improve the Prediction of Anti-cancer Peptides. *Bioinformatics* 34, 4007–4016. doi:10.1093/bioinformatics/bty451

Xu, Z., Luo, M., Lin, W., Xue, G., Wang, P., Jin, X., et al. (2021). DLpTCR: An Ensemble Deep Learning Framework for Predicting Immunogenic Peptide Recognized by T Cell Receptor. *Brief. Bioinform* 22. doi:10.1093/bib/bbab335

Zadeh Hosseingholi, E., Neghabi, N., Molavi, G., Gheibi Hayat, S. M., and Shahriarpour, H. (2020). *In Silico* Identification and Characterization of Antineoplastic Asparaginase Enzyme from Endophytic Bacteria. *IUBMB Life* 72, 991–1000. doi:10.1002/iub.2237

Zawawi, A., Forman, R., Smith, H., Mair, I., Jibril, M., Albaqshi, M. H., et al. (2020). *In Silico* Design of A T-Cell Epitope Vaccine Candidate for Parasitic Helminth Infection. *PLoS Pathog.* 16, e1008243. doi:10.1371/journal.ppat.1008243

Zhang, J., Ju, Y., Lu, H., Xuan, P., and Zou, Q. (2016). Accurate Identification of Cancerlectins through Hybrid Machine Learning Technology. *Int. J. Genomics* 2016, 1–11. doi:10.1155/2016/7604641

Zhang, L., Udaka, K., Mamitsuka, H., and Zhu, S. (2012). Toward More Accurate Pan-specific MHC-Peptide Binding Prediction: A Review of Current Methods and Tools. *Briefings Bioinforma.* 13, 350–364. doi:10.1093/bib/bbr060

Zou, Q., Zeng, J., Cao, L., and Ji, R. (2016). A Novel Features Ranking Metric with Application to Scalable Visual and Bioinformatics Data Classification. *Neurocomputing* 173, 346–354. doi:10.1016/j.neucom.2014.12.123

Conflict of Interest: XZ was employed by the company Beidahuang Industry Group General Hospital.

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.

The handling editor QZ declared a past co-authorship with the author RSH.

TYPE Methods
PUBLISHED 15 August 2022
DOI 10.3389/fgene.2022.963349

A method for identifying moonlighting proteins based on linear discriminant analysis and bagging-SVM

Yu Chen, Sai Li and Jifeng Guo*

College of Information and Computer Engineering, Northeast Forestry University, Harbin, China

EDITED BY
Quan Zou,
University of Electronic Science and Technology of China, China

REVIEWED BY
Yangfan Wang,
Ocean University of China, China
Kumar Shashvat,
Alliance University, India
Sammed Mandape,
University of North Texas Health Science Center, United States

*CORRESPONDENCE
Jifeng Guo,
guojifeng@nefu.edu.cn

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 07 June 2022
ACCEPTED 18 July 2022
PUBLISHED 15 August 2022

CITATION
Chen Y, Li S and Guo J (2022), A method for identifying moonlighting proteins based on linear discriminant analysis and bagging-SVM.
Front. Genet. 13:963349.
doi: 10.3389/fgene.2022.963349

Moonlighting proteins have at least two independent functions and are widely found in animals, plants and microorganisms. Moonlighting proteins play important roles in signal transduction, cell growth and movement, tumor inhibition, DNA synthesis and repair, and metabolism of biological macromolecules. Moonlighting proteins are difficult to find through biological experiments, so many researchers identify moonlighting proteins through bioinformatics methods, but their accuracies are relatively low. Therefore, we propose a new method. In this study, we select SVMProt-188D as the feature input, and apply a model combining linear discriminant analysis and basic classifiers in machine learning to study moonlighting proteins, and perform bagging ensemble on the best-performing support vector machine. They are identified accurately and efficiently. The model achieves an accuracy of 93.26% and an F-sorce of 0.946 on the MPFit dataset, which is better than the existing MEL-MP model. Meanwhile, it also achieves good results on the other two moonlighting protein datasets.

KEYWORDS
moonlighting proteins, protein recognition, machine learning, linear discriminant analysis, bagging-SVM

1 Introduction

With the continuous expansion of proteomic data and the continuous study of protein functions by researchers, multifunctional proteins have gradually attracted people's attention. Among multifunctional proteins, people have found a new type of protein that can perform multiple functions autonomously without partitioning these into separate domains, and they are called moonlighting proteins (MPs) (Huberts et al., 2010). Under the influence of certain specific factors, such as cell localization, cell type, substrate or different cofactor, moonlighting proteins can switch their executive functions (Jeffery, 1999). At present, moonlighting proteins have been found in a variety of animals, plants and microorganisms, and a large number of studies have shown that moonlighting proteins play an important role in organisms. They can be used as enzymes for catalytic reactions, as well as secreted cytokines, transcription factors and DNA stabilizers. Through the study of moonlighting proteins, it is found that they can play an important role in the development of new therapies for some diseases (Jeffery, 2018). For example, moonlighting proteins can be used as targets for active medicines in the treatment of hepatitis B virus, cancer, and bacterial infections (Adamo et al., 2021; Zakrzewicz and Geyer, 2022). Due to the excellent performance of moonlighting proteins in disease treatment, the discovery of new moonlighting proteins is of great significance for solving many medical problems. Therefore, the prediction of moonlighting proteins has become a hot research direction.

At present, there are several online available moonlighting protein databases that can obtain protein sequences. Jeffery's laboratory manually collected some strict moonlighting proteins from peer journals, and built a searchable and Internet-based MoonProt database, which has been updated to MoonProt 3.0 (Chen C. et al., 2021). Luis et al. constructed a multi-functional protein database MultitaskProtDB, designed to provide a free online database for researchers using bioinformatics methods to predict multifunctional proteins, and has been updated to MultitaskProtDB-II (Franco-Serrano et al., 2018). Bo et al. established PlantMP, the first plant moonlighting protein database, enabling researchers to conveniently collect and process plant-specific raw data (Su et al., 2019).

Based on these public moonlighting protein databases, researchers have constructed several models to predict moonlighting proteins. In 2016, Khan and Kihara et al. developed a moonlighting protein prediction model called MPFit, which achieved 98% accuracy when protein gene ontology (GO) annotations were available, and 75% accuracy using omics features when no GO annotations were available (Khan and Kihara, 2016). In 2017, Khan et al. proposed a new solution: they built DextMP based on three types of textual information of proteins (title, abstracts from literature and function description in UniProt) and machine learning classifier, achieving 91% accuracy (Khan et al., 2017). In 2021, Li et al. proposed a multimodal deep ensemble learning architecture called MEL-MP. Firstly, they extracted four sequence-based features: primary protein sequence information, evolutionary information, physical and chemical properties, and secondary protein structure information; secondly, they selected a specific classifier for each feature; finally, they applied stacked ensemble to integrate the output of each classifier. The method showed excellent predictive performance, which achieved an F-score of 0.891 (Li et al., 2021). In the same year, Shirafkan et al. constructed a new moonlighting protein dataset to identify MPs and non-MPs through the SVM method of SAAC feature, and established a well-judged scheme to detect outlier proteins (Shirafkan et al., 2021). Liu et al. believed that an appropriate method was needed to identify plant moonlighting proteins, so they used the combination of Tri-Peptide composition (TPC) and XGBoost to construct IdentPMP, which was a plant moonlighting protein prediction tool (Liu et al., 2021).

For MPFit and DextMP, although high accuracy can be obtained, GO annotations and text information of protein samples need to be provided, which is very restrictive. Other experiments have shortcomings such as relatively low model accuracy and low efficiency due to the complexity of the model (Li et al., 2021; Shirafkan et al., 2021). In order to solve the above problems, we propose a new scheme. Firstly, we extract the SVMProt-188D feature, which contains information of protein composition and eight physicochemical properties that are effective in showing the properties of moonlighting proteins (Zou et al., 2016). Secondly, linear discriminant analysis (LDA) is used to reduce the dimensionality of the feature set to achieve separation of positive and negative samples. Finally, bagging ens is performed on SVM to classify moonlighting proteins. The main contributions of this paper are as follows: 1) We propose a method combining LDA and Bagging-SVM to classify moonlighting proteins. 2) We conduct extensive experiments on MPFit dataset, Shirafkan's dataset, and plant moonlighting protein dataset, and the model achieves excellent performance on these datasets.

2 Materials and methods

Our research is mainly divided into four parts: benchmark dataset acquisition; feature extraction; model construction; model evaluation. The experimental process is shown in Figure 1. Firstly, we use MPFit as the benchmark dataset (a). Secondly, we extract SVMProt-188D as a feature and compare the classification results of this feature with Pse-AAC and Pse-PSSM (b). Thirdly, we combine LDA with Bagging-SVM for protein classification, and compare the classification results with other base classifiers to verify the superiority of the classifier (c). Finally, we use multiple datasets to validate our method and compare the classification results with state-of-the-art models to demonstrate the effectiveness of our method (d).

2.1 Benchmark dataset

In this study, we use the benchmark dataset constructed by Khan and Kihara et al. (MPFit dataset) [9]. The dataset contains 268 MPs and 162 non-MPs. The positive examples in the dataset are derived from 268 proteins with Uniprot ID extracted from MoonProt database, and their biological origins are shown in Table 1(Mani et al., 2015). Screening of suitable proteins from four genomes of human, E. coli, yeast and mouse as negative example of moonlighting proteins (single-function proteins). The screening criterias are as follows: 1) target protein with at least eight GO term annotations; 2) when clustering GO terms in the biological process (BD) category using a semantic similarity score threshold between 0.1 and 0.5, no more than one cluster is obtained at each threshold; 3) there is no more than one cluster of

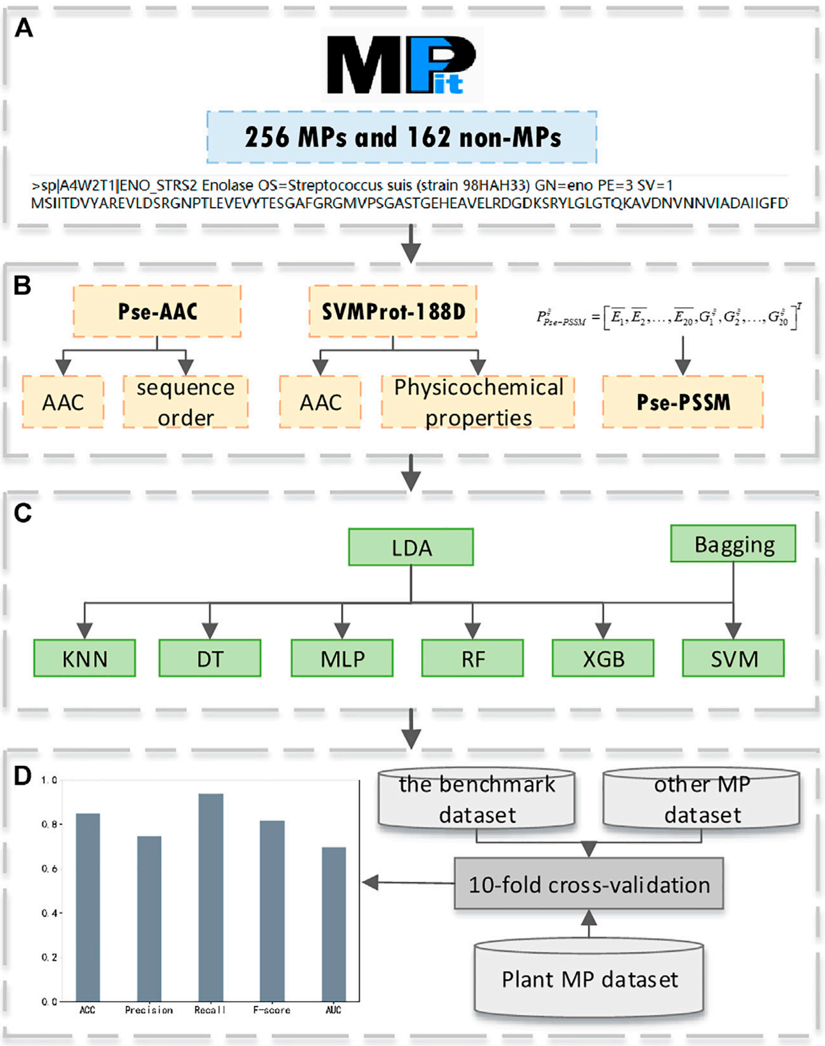


FIGURE 1
The pipeline of our experiment, **(A)** benchmark dataset acquisition; **(B)** feature extraction; **(C)** model construction; **(D)** model evaluation.

TABLE 1 Composition of the benchmark dataset.

Organism	MPs		Non-MPs	
	Number	Percentage (%)	Number	Percentage (%)
Human	45	16.8	60	37.0
Escherichia coli	30	11.19	16	9.88
Yeast	27	10.1	34	20.9
Mouse	11	4.1	52	32.1
Other	155	57.81	0	0.0
Total	268	100	162	100

GO terms for molecular function (MF) with semantic similarity scores between 0.1 and 0.5. After removing non-MPs with more than 25% similarity to MPs, 162 negative samples were obtained (Table 1) (Khan and Kihara, 2016). This dataset has been used in several experiments on moonlighting protein prediction and is very authoritative in the field (Khan and Kihara, 2016; Khan et al., 2017; Li et al., 2021; Shirafkan et al., 2021). Therefore, it is suitable as the benchmark dataset for this study. Also, we have conducted experiments on the state-of-the-art dataset of Shirafkan et al. (2021).

2.2 Feature extraction

Feature extraction is a crucial step in the process of identifying proteins. This process is the conversion of the amino acid sequence of a protein into discrete data of a certain length, and the representation of a sample of the protein by features composed of discrete data. At present, a variety of features have been used in the study of protein classification, such as amino acid composition, positional information, physicochemical properties, evolutionary information and secondary structure. Pse-AAC, SVMProt-188D and Pse-PSSM reflect positional information, physicochemical properties and evolutionary information respectively, which is important for protein recognition. Therefore, we choose these three features as the feature vectors of this study. The details are as follows.

2.2.1 Pse-AAC

Since the amino acid composition does not take into account the influence of sequence order information, the researchers propose the feature of pseudo-amino acids (Pse-AAC). The feature combines regular amino acid composition (frequency of occurrence of 20 amino acids) with a set of discrete sequence correlation factors, which are primarily used to address the problem that sequence information cannot be directly incorporated into the prediction algorithm due to different lengths of amino acid sequences (Chou, 2001; Ding et al., 2009; Tang et al., 2016; Awais et al., 2021). The specific descriptions are as follows.

$$X = [x_1 \cdots x_{20}, x_{20+1} \cdots x_{20+\lambda}]^T$$

Where X represents Pse-AAC, x_1 to x_{20} represent the regular amino acid composition, and x_{20+1} to $x_{20+\lambda}$ represent the information of sequence order. x_i in X is expressed as follows.

$$x_i = \begin{cases} \dfrac{f_i}{\sum_{j=1}^{20} f_j + \omega \sum_{k=1}^{\lambda} \theta_k} \, (1 \le i \le 20) \\ \dfrac{\omega \theta_{i-20}}{\sum_{j=1}^{20} f_j + \omega \sum_{k=1}^{\lambda} \theta_k} \, (20 + 1 \le i \le 20 + \lambda) \end{cases}$$

Where f_i is the frequency of occurrence of the 20 amino acids, θ_k is the k-layer sequence correlation factor, and ω is the weighting factor for sequence order effects, $\omega = 0.05$ in our study. The λ components can be defined by the user at will (Yan et al., 2020). In this experiment, hydrophilic, hydrophobic, mass, pK1, pK2, pI, rigidity, flexibility, and irreplaceability are added, resulting in a 65-dimensional feature vector.

2.2.2 SVMProt-188D

The SVMProt-188D includes the frequency of 20 amino acids (i.e., “ACDEFGHIKLMNPQRSTVWY”) and eight physical and chemical properties (hydrophobicity, normalized van der Waals volume, polarity, polarizability, charge, secondary structure, solvent accessibility, and surface tension) (Cai et al., 2003). The details are shown in Table 2, and will be introduced separately below.

The frequency of 20 amino acids can be calculated by the following formula:

$$F_i = \frac{N_i}{L}, \quad (i = A, C, D, \ldots, Y)$$

Where N_i is the number of amino acid type i, and L is the length of a protein sequence.

Eight physicochemical properties are studied on the composition, transition, and distribution of amino acids, and each property is divided into three groups (Dubchak et al., 1995; Wang et al., 2017; Xiong et al., 2018; Zou et al., 2019).

2.2.2.1 Composition

Taking the hydrophobicity attribute as an example, “RKEDQN” is polar, “GASTPHY” is neutral, and “CVLIMFW” is hydrophobic. The frequency of each group can be expressed as:

$$C_i = \frac{N_i}{L}, \ i \in \{polar, neutral, hydrophobic\}$$

2.2.2.2 Transition

The transition from polar group to neutral group is the frequency of polar residue following neutral residue or neutral residue following polar residue. The transition between neutral group and hydrophobic group, and the transition between hydrophobic group and polar group have similar definitions. It can be expressed by the following formula:

$$T(i_1, i_2) = \frac{N(i_1, i_2) + N(i_2, i_1)}{L - 1},$$
$$(i_1, i_2) \in \{(polar, neutral), (neutral, hydrophobic), (hydrophobic, polar)\}$$

2.2.2.3 Distribution

The distribution represents the position of the first, 25%, 50%, 75%, and last of each group category in the amino acid sequence.

TABLE 2 Eight physical and chemical properties of the 188-dimensions.

Attribute	Division		
hydrophobicity	Polar:RKEDQN	Neutral:GASTPHY	Hydrophobicity:CVLIMFW
Normalized van der waals volume	Small:GASCTPD	Medium:NVEQIL	Large:MHKFRYW
polarity	Low:LIFWCMVY	Medium:PATGS	High:HQRKNED
polarizability	Low:GASDT	Medium:GPNVEQIL	High:KMHFRYW
charge	Positive:KR	Neutral:ANCQGHILMFPSTWYV	Negative:DE
Secondary structure	Helix:EALMQKRH	Strand:VIYCWFT	Coil:GNPSD
Solvent accessibility	Buried:ALFCGIVW	Exposed:RKQEND	Intermediate:MPSTHY
Surface tension	Large:GQDNAHR	Medium:KTSEC	Small:ILMFPWYV

2.2.3 Pse-PSSM

Inspired by Pse-AAC signatures, and combining with evolutionary information, Chou et al. proposed a new signature, Pse-PSSM (Chou and Shen, 2007; Wang et al., 2020). The original PSSM profile P_{PSSM} was generated by running the position-specific iterative basic local alignment search tool (PSI-BLAST) against Uniref50 database, and setting the E-value to 0.001 for 3 iterations (Ding et al., 2014).

$$P_{PSSM} = \begin{bmatrix} E_{1\to 1} & \cdots & E_{1\to 20} \\ \vdots & \ddots & \vdots \\ E_{L\to 1} & \cdots & E_{L\to 20} \end{bmatrix}$$

Where $E_{i\to j}$ represents the score of the amino acid residue at the i-th position of the protein sequence being changed to amino acid residue type j during the evolutionary process, L is the length of the protein sequence, k from 1 to 20 indicate the 20 natural amino acid types. Implement the following standardised procedures:

$$E_{i\to j} = \frac{E^0_{i\to j} - \frac{1}{20}\sum_{k=1}^{20} E^0_{i\to k}}{\sqrt{\frac{1}{20}\sum_{u=1}^{20}\left(E^0_{i\to j} - \frac{1}{20}\sum_{k=1}^{20} E^0_{i\to k}\right)^2}}$$

In order to make the dimension size of the PSSM descriptors consistent, the following operations are performed:

$$\overline{P_{PSSM}} = \left[\overline{E_1}, \overline{E_2}, \cdots, \overline{E_{20}}\right]^T$$

$$\overline{E_j} = \frac{1}{L}\sum_{i=1}^{L} E_{i\to j}$$

Where $\overline{E_j}$ is the average score of the i-th amino acid in the protein sequence P over the course of biological evolution. In order to preserve sequence order information, the concept of pseudo-amino acid composition is used to obtain the final 40-dimensional Pse-PSSM by considering the correlation between two amino acids $E_{i\to j}$.

$$P^{\xi}_{Pse-PSSM} = \left[\overline{E_1}, \overline{E_2}, \cdots, \overline{E_{20}}, G^{\xi}_1, G^{\xi}_2, \cdots, G^{\xi}_{20}\right]^T$$

$$G^{\xi}_j = \frac{1}{L-\xi}\sum_{i=1}^{L-\xi}\left[E_{i\to j} - E_{(i+\xi)\to j}\right]^2$$

2.3 Feature selection

Linear discriminant analysis (LDA) is a feature selection technique (Arjmandi and Pooyan, 2012; Xie et al., 2018; Yang et al., 2020; Chen Y. et al., 2021). It can effectively reduce the feature dimension and reduce the error caused by redundant data. The idea of LDA is to project samples from high-dimensional space onto low-dimensional space where the distance between samples of the same category is minimized and the distance between samples of different categories is maximized, thus making the samples more easily distinguishable and obtaining better classification results. Therefore, this study uses LDA for dimensionality reduction. The diagram of LDA applied to a binary classification algorithm is shown in Figure 2.

2.3.1 The linear discriminant analysis process is as follows

Suppose we have N protein samples which can be denoted as $\{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$, where x_i is the features of the protein sample and y_i is the label of the protein sample, $y_i \in (0, 1)$. Our aim is to find a projection line W such that the projection $Y = W^T x_i$ of sample x_i on the line minimizes the intra-class distance and maximizes the inter-class distance. Firstly, calculate the mean vector for each class:

$$\mu_j = \frac{1}{N_j}\sum_{x\in X_j} x\,(j = 0, 1)$$

Where N_j is the number of samples of class j and X_j is the set of samples of class j, μ_j is the mean vector of the j-th class of samples.

Then, calculate the within-class scatter matrix S_W:

$$S_W = \sum_0 + \sum_1 = \sum_{x\in X_0}(x-\mu_0)(x-\mu_0)^T + \sum_{x\in X_1}(x-\mu_1)(x-\mu_1)^T$$

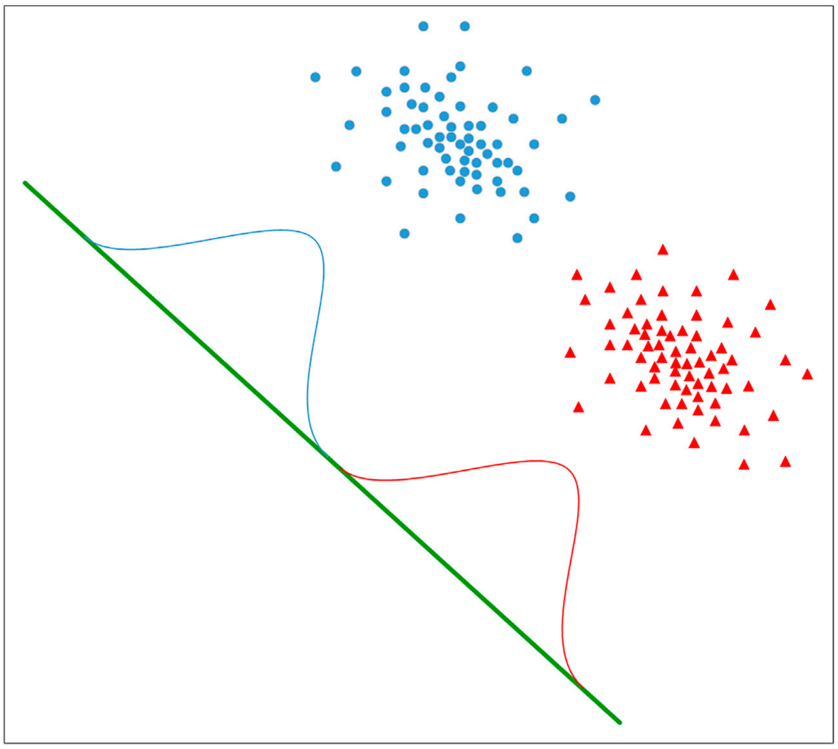


FIGURE 2
The diagram of LDA applied to a binary classification algorithm.

Where $\sum_j$ is the covariance matrix of samples of class j (strict lack of covariance matrix of the numerator), expressed by the following formula:

$$\sum_j = \sum_{x \in X_j} \left(x - \mu_j\right)\left(x - \mu_j\right)^T (j = 0, 1)$$

Calculating the between-class scatter matrix S_B:

$$S_B = (\mu_0 - \mu_1)(\mu_0 - \mu_1)^T$$

Finally, the optimization objective is:

$$arg\,max\,J(W) = \frac{W^T S_B W}{W^T S_W W}$$

Simplify the above formula to get the target projection line W^*:

$$W^* = arg\,max \left\{ \frac{W^T S_B W}{W^T S_W W} \right\} = S_W^{-1}(\mu_0 - \mu_1)$$

The original set of samples is projected onto the one-dimensional space W to obtain the 1-dimensional feature vector after dimensionality reduction (Chen Y. et al., 2021).

2.4 Classifier

In the experiments, we use six popular base classifiers, including K-nearest-neighbor (KNN) (Deng et al., 2016), Decision Tree (DT) (Safavian and Landgrebe, 1991), Multilayer Perceptrons (MLP) (Lee et al., 2020), Random Forests (RF) (Breiman, 2001), XGBoost (Chen et al., 2016; Chen et al., 2020) and Support Vector Machine (SVM). Experimental parameters for all classifiers can be found in Supplementary Table S1. After evaluation on the benchmark dataset, the support vector machine works best, and can avoid overfitting when the number of samples is small (Gong et al., 2021). Through bagging ensemble of SVM, the model performance is further improved.

SVM is a type of supervised learning proposed by Vladimir Vapnik and is widely used in machine learning, computer vision and data mining, such as image recognition, text classification and protein sequence classification (Zhao et al., 2015; Ding et al., 2017; Manavalan et al., 2018; Zhang et al., 2019). In binary classification problems, the main idea of SVM is to find a segmentation hyperplane that maximizes the distance of the segmentation hyperplane from the nearest point. Given a training sample $x_i \in R^P$, i = 1, . . . , n, and a vector $y \in \{0, 1\}^n$, our goal is to find $w \in R^P$ and $b \in R$ for a given prediction $sign(w^T\phi(x) + b)$ that predicts correctly for most samples. In this experiment, we use the SVC algorithm for classification and set the kernel function to linear function and the penalty parameter C to 1.0.

Bagging is a common ensemble learning method that integrates the prediction results of multiple base classifiers into the final strong classifier prediction result. Its integration strategy is to obtain training subsets by sampling from the original sample set, and each training subset trains a model. Finally, the classification results of samples are obtained by voting strategy (Breiman, 1996; Zaman and Hirose, 2008).

2.5 Performance assessment

We used these indicators to evaluate the performance of the experiment: accuracy (ACC), Precision, Recall, F-score and AUC (area of ROC curve) (Wei et al., 2017; Shan et al., 2019; Basith et al., 2020; Zhang et al., 2020; Wang et al., 2021). These evaluation indicators are the results of the confusion matrix calculation obtained from the experiment, and the calculation formulas are as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2{*}Precision{*}Recall}{Precision + Recall}$$

Where TP represents the number of correctly predicted MPs, TN represents the number of correctly predicted non-MPs, FP represents the number of incorrectly predicted MPs as non-MPs, and FN represents the number of incorrectly predicted non-MPs as MPs.

3 Results and discussion

3.1 Performance evaluation of different feature extraction

To ensure the accuracy of the experimental results, the 10-fold cross-validation (i.e., The training samples are divided into ten folds, nine of which are adopted for training, one of which is adopted for testing. The process repeats 10 times and the average value is taken as the final result.) is applied on the benchmark dataset. To select suitable input data, Pse-AAC, SVMProt-188D and Pse-PSSM are experimented with multiple classifiers respectively (Table 3). It is clear from the table that the SVMProt-188D performs best on all indicators, with the most accuracy rates exceeding 90% (Figure 3). In contrast, Pse-AAC and Pse-PSSM don't perform as well as SVMProt-188D. From this, we hypothesize that: On the one hand, MPs can change their functions under certain conditions, such as substrate concentration or cofactor change, and there are great differences in physicochemical properties between them and non-MPs; on the other hand, SVMProt-188D is a linear feature of the protein, which can be easily identified by the classifier after LDA.

3.2 Performance evaluation of different classifiers

Six classifiers from scikit-learn are used in this study for comparison experiments, namely KNN, DT, MLP, RF, XGBoost, and SVM. From the data, SVM obtains an accuracy rate of 92.7907%, which is the highest accuracy rate. Despite the unbalanced benchmark dataset used in this experiment, with 268 positive and 162 negative samples, the classifier achieves high scores of 0.943, 0.942 and 0.925 on the three metrics of percision, F-score and AUC (Figure 4). The MLP is second only to the SVM and also achieves high scores in various metrics. Of these, surprisingly, DT obtains the highest recall value, 0.946. Because we use accuracy as the main metric, SVM is the most suitable classifier for this experiment. Furthermore, we compare this model with the model without LDA (Figure 5). From the figure, we can observe that the LDA dimensionality reduction method has greatly improved the experimental results, proving that it is very effective in the identification of MPs.

3.3 Comparison of Bagging-SVM and single SVM

The above experiments prove that the combination of SVMProt-188D and support vector machine has the best effect. Based on the excellent performance of bagging ensemble algorithm in the field of machine learning, we use bagging to integrate SVM and verify the classification performance of the integrated model (Chen and Association for Computing Machinery, 2018; Kaur et al., 2019; Raihan-Al-Masud and Mondal, 2020). The results are shown in Table 4 (The experimental results of bagging integration with all classifiers can be obtained from the Supplementary Figures S1, S2). As can be seen from the table, ACC, Precision, Recall, F-score and AUC all improved, which indicates that Bagging-SVM is effective for the classification of moonlighting proteins. Bagging-SVM can reduce the error caused by a single support vector machine to the experimental results, improving the stability of the model, and have stronger convincing.

TABLE 3 The results of 10-fold cross-validation using a variety of classifiers and hybrid features.

Feature	Method	ACC (%)	Precision	Recall	F-score	AUC
Pse-AAC	KNN	87.4419	0.885	0.923	0.901	0.863
	DT	87.2093	0.892	0.909	0.898	0.865
	MLP	88.8372	0.898	0.931	0.912	0.878
	RF	85.3488	0.883	0.887	0.883	0.847
	XGB	86.0465	0.891	0.891	0.888	0.856
	SVM	87.907	0.9	0.913	0.904	0.872
SVMProt-188D	KNN	91.3953	0.919	0.944	0.931	0.906
	DT	91.1628	0.918	0.946	0.929	0.906
	MLP	92.5581	0.939	0.941	0.939	0.922
	RF	89.3023	0.917	0.911	0.912	0.891
	XGB	89.5349	0.92	0.911	0.914	0.893
	SVM	92.7907	0.943	0.942	0.942	0.925
Pse-PSSM	KNN	85.8514	0.886	0.886	0.884	0.848
	DT	84.4189	0.884	0.868	0.872	0.839
	MLP	86.5116	0.917	0.868	0.888	0.869
	RF	82.5581	0.858	0.862	0.858	0.815
	XGB	84.186	0.869	0.883	0.873	0.833
	SVM	87.6744	0.921	0.883	0.898	0.879

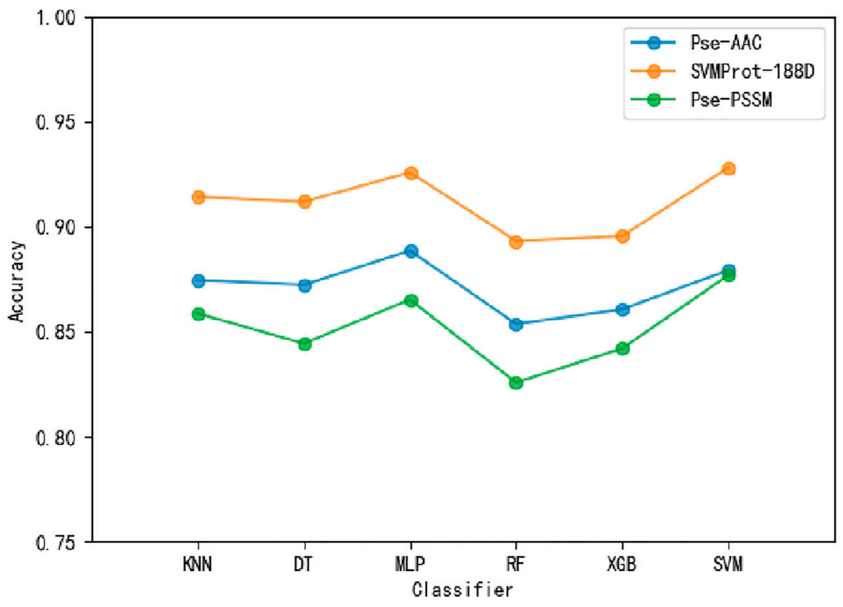


FIGURE 3
The accuracy of different features in each classifier.

3.4 Comparison with other methods

We compare with the more current MP classification models, including Khan's MPFit (Khan and Kihara, 2016), Li's MEL-MP(Li et al., 2021) and Shirafkan's method (Shirafkan et al., 2021). The results of the comparison are shown in Table 5 (Where '*' is for data not given in the comparison papers). The experimental results for all three models above are obtained with the MPFit dataset, mostly using 10-fold cross-validation. Therefore, they are very suitable for comparison with our model. As can be observed from the table, our

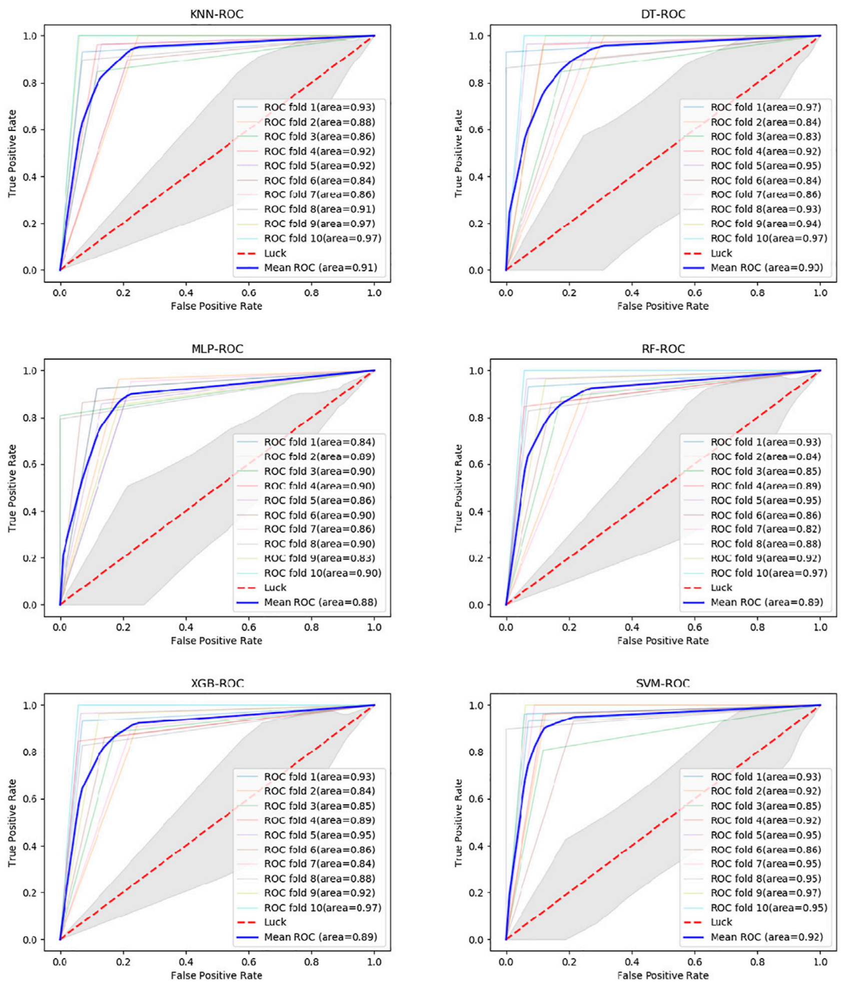


FIGURE 4
ROC curves of different classifiers on SVMProt-188D.

model outperforms the other prediction methods on all the remaining evaluation indicators except for the AUC. In particular, the F-score of 0.946 is 5.4% higher than the second highest, MEL-MP (F-score = 0.892).

3.5 Performance on other MPs datasets

To verify that our model can effectively classify moonlighting proteins, we obtain a state-of-the-art moonlighting protein dataset from Shirafkan's paper, which includes 215 positive samples and 136 negative samples (Shirafkan et al., 2021). Similarly, feature extraction is performed on this dataset to obtain SVMProt-188D features, and then, using 10-fold cross-validation, classification is performed on our model. In order to verify the generalization ability of our model, MPFit dataset is used as the training set and Shirafkan's dataset is used as the independent testing set to conduct the experiment again. The experimental results are shown in Table 6. Method 1 is the result of 10-fold cross-validation, and method 2 is the result of independent testing. On this dataset, we still obtain an accuracy rate higher than 91%, and the other four indicators also achieve high scores, proving that our model has a strong generalization ability.

To verify that the model can effectively classify plant moonlighting proteins, we obtain the Uniprot ID of the plant moonlighting protein dataset from Liu et al. and obtain protein sequences from the corresponding databases according to the UniprotID (Liu et al., 2021). In order to compare with IdentPMP, 10-fold cross-validation is used on the same dataset, and the experimental results are shown in Figure 6. On the dataset of plant MP, the accuracy of 94.9692% is obtained by 10-fold cross-validation, far exceeding IdentPMP in F-score and AUC.

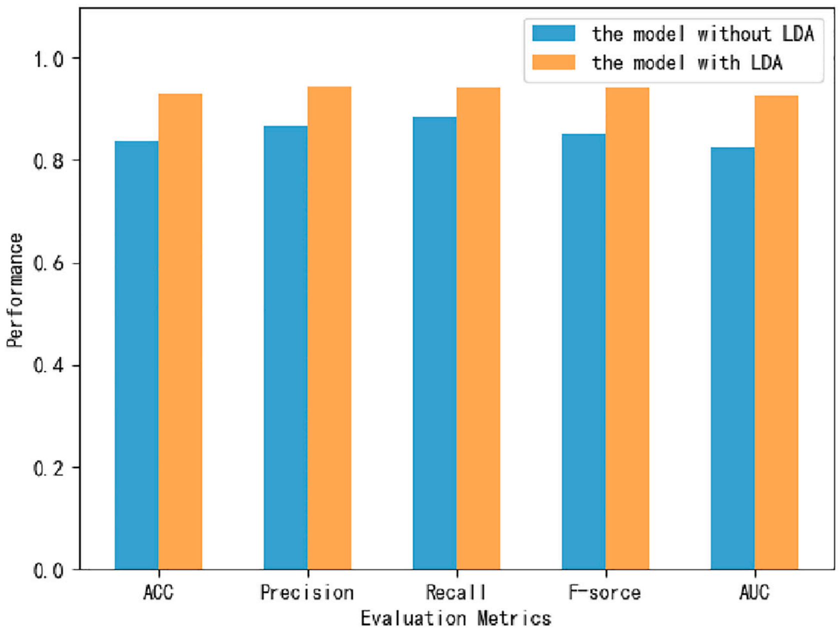


FIGURE 5
The performance of the model after and before the implementation of LDA.

TABLE 4 The results of Bagging-SVM and Single SVM.

Method	ACC (%)	Precision	Recall	F-score	AUC
SVM	92.7907	0.943	0.942	0.942	0.925
Bagging_SVM	93.2558	0.944	0.949	0.946	0.928

TABLE 5 Comparison with other methods.

Method	ACC (%)	Precision	Recall	F-score	AUC
MPFit	75	*	*	0.784	*
MEL-MP	*	0.895	0.893	0.892	0.947
Shirafkan's	81.7	0.813	*	0.802	0.806
Our	92.7907	0.943	0.942	0.942	0.925

TABLE 6 The results of other dataset on our model.

Method	ACC (%)	Precision	Recall	F-score	AUC
Method1	91.1746	0.91	0.949	0.929	0.901
Method2	91.4530	0.907	0.958	0.932	0.902

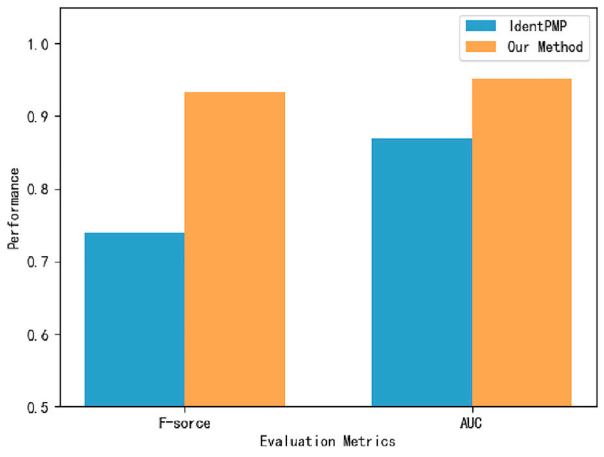


FIGURE 6
The performance of the plant MPs dataset on our model.

4 Conclusion

In this paper, we propose a method for identifying moonlighting proteins based on bagging-SVM ensemble learning classifier. Firstly, feature extraction is carried out on the collected benchmark dataset, and after comparison, SVMprot-188D is selected. Then, we use the feature selection method of LDA to reduce the dimension of the feature. Finally, the Bagging-SVM ensemble learning algorithm is used to construct the prediction model. The experimental results show that our model achieves good results in various indicators and is superior to the current advanced models. In order to prove that our model has strong generalization ability, we also use the dataset in Shirafkan's paper to conduct experiments, and the accuracy rate has exceeded 91%. In addition, plant MPs are found to be equally applicable to our method, which is a great improvement compared with the previous experimental method. However, the depth of machine learning model is relatively shallow. In the future, we will try to use deep learning model to identify MPs, and hope to make new breakthroughs in this field.

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found in the article/Supplementary Material.

Author contributions

SL collected the datasets, performed the experiments, analyzed the experiments' result, and drafted the manuscript. YC designed the experiments and revised the manuscript. SL, YC, and JG provided suggestions for the study design and the writing of the manuscript. All authors contributed to the article and approved the submitted version.

Funding

This work was supported by the Fundamental Research Funds for the Central Universities (2572021BH01) and the National Natural Science Foundation of China (62172087).

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

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2022.963349/full#supplementary-material

References

Adamo, A., Frusteri, C., Pallotta, M. T., Pirali, T., Sartoris, S., and Ugel, S. (2021). Moonlighting proteins are important players in cancer immunology. *Front. Immunol.* 11, 613069. doi:10.3389/fimmu.2020.613069

Arjmandi, M. K., and Pooyan, M. (2012). An optimum algorithm in pathological voice quality assessment using wavelet-packet-based features, linear discriminant analysis and support vector machine. *Biomed. Signal Process. Control* 7 (1), 3–19. doi:10.1016/j.bspc.2011.03.010

Awais, M., Hussain, W., Khan, Y. D., Rasool, N., Khan, S. A., and Chou, K. C. (2021). iPhosH-PseAAC: identify phosphohistidine sites in proteins by blending statistical moments and position relative features according to the chou's 5-step rule and general pseudo amino acid composition. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 18 (2), 596–610. doi:10.1109/tcbb.2019.2919025

Basith, S., Manavalan, B., Shin, T. H., and Lee, G. (2020). Machine intelligence in peptide therapeutics: a next-generation tool for rapid disease screening. *Med. Res. Rev.* 40 (4), 1276–1314. doi:10.1002/med.21658

Breiman, L. (1996). Bagging predictors. *Mach. Learn.* 24 (2), 123–140. doi:10.1007/bf00058655

Breiman, L. (2001). Random forests. *Mach. Learn.* 45 (1), 5–32. doi:10.1023/a:1010933404324

Cai, C. Z., Han, L. Y., Ji, Z. L., Chen, X., and Chen, Y. Z. (2003). SVM-prot: web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic Acids Res.* 31 (13), 3692–3697. doi:10.1093/nar/gkg600

Chen, C., Liu, H. P., Zabad, S., Rivera, N., Rowin, E., Hassan, M., et al. (2021a). MoonProt 3.0: an update of the moonlighting proteins database. *Nucleic Acids Res.* 49 (D1), D368–D372. doi:10.1093/nar/gkaa1101

Chen, T. H., Wang, X. G., Chu, Y. Y., Wang, Y. J., Jiang, M. M., Wei, D. Q., et al. (2020). T4SE-XGB: interpretable sequence-based prediction of type IV secreted effectors using eXtreme gradient boosting algorithm. *Front. Microbiol.* 11, 580382. doi:10.3389/fmicb.2020.580382

Chen, T. Q., and Guestrin, C.,and Association for Computing Machinery. (2016). "XGBoost: A scalable tree boosting system", in: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)), 785–794.

Chen, Y., Chang, R., and Guo, J. F. (2021b). Emotion recognition of EEG signals based on the ensemble learning method: AdaBoost. *Math. Problems Eng.* 2021, 1–12. doi:10.1155/2021/8896062

Chen, Y. F.,and Association for Computing Machinery, (2018). "A selective under-sampling based bagging SVM for imbalanced data learning in biomedical event trigger recognition", in: 2nd International Conference on Biomedical Engineering and Bioinformatics ICBEB, 112–119.

Chou, K. C. (2001). Prediction of protein cellular attributes using pseudo-amino acid composition. *Proteins* 43 (3), 246–255. doi:10.1002/prot.1035

Chou, K. C., and Shen, H. B. (2007). MemType-2L: A web server for predicting membrane proteins and their types by incorporating evolution information through pse-PSSM. *Biochem. Biophys. Res. Commun.* 360 (2), 339–345. doi:10.1016/j.bbrc.2007.06.027

Deng, Z. Y., Zhu, X. S., Cheng, D. B., Zong, M., and Zhang, S. C. (2016). Efficient kNN classification algorithm for big data. *Neurocomputing* 195, 143–148. doi:10.1016/j.neucom.2015.08.112

Ding, H., Luo, L. F., and Lin, H. (2009). Prediction of cell wall lytic enzymes using chou's amphiphilic pseudo amino acid composition. *Protein Pept. Lett.* 16 (4), 351–355. doi:10.2174/092986609787848045

Ding, S. Y., Yan, S. J., Qi, S. H., Li, Y., and Yao, Y. H. (2014). A protein structural classes prediction method based on PSI-BLAST profile. *J. Theor. Biol.* 353, 19–23. doi:10.1016/j.jtbi.2014.02.034

Ding, Y. J., Tang, J. J., and Guo, F. (2017). Identification of drug-target interactions via multiple information integration. *Inf. Sci.* 418, 546–560. doi:10.1016/j.ins.2017.08.045

Dubchak, I., Muchnik, I., Holbrook, S. R., and Kim, S. H. (1995). Prediction of protein-folding class using global description of amino-acid-sequence. *Proc. Natl. Acad. Sci. U. S. A.* 92 (19), 8700–8704. doi:10.1073/pnas.92.19.8700

Franco-Serrano, L., Hernandez, S., Calvo, A., Severi, M. A., Ferragut, G., Perez-Pons, J., et al. (2018). MultitaskProtDB-II: an update of a database of multitasking/moonlighting proteins. *Nucleic Acids Res.* 46 (D1), D645–D648. doi:10.1093/nar/gkx1066

Gong, Y. X., Liao, B., Wang, P., and Zou, Q. (2021). DrugHybrid_BS: Using hybrid feature combined with bagging-SVM to predict potentially druggable proteins. *Front. Pharmacol.* 12, 771808. doi:10.3389/fphar.2021.771808

Huberts, D., Venselaar, H., Vriend, G., Veenhuis, M., and van der Klei, I. J. (2010). The moonlighting function of pyruvate carboxylase resides in the non-catalytic end of the TIM barrel. *Biochim. Biophys. Acta* 1803 (9), 1038–1042. doi:10.1016/j.bbamcr.2010.03.018

Jeffery, C. J. (1999). Moonlighting proteins. *Trends biochem. Sci.* 24 (1), 8–11. doi:10.1016/s0968-0004(98)01335-8

Jeffery, C. J. (2018). Protein moonlighting: what is it, and why is it important? *Philos. Trans. R. Soc. Lond. B Biol. Sci.* 372, 20160523. doi:10.1098/rstb.2016.0523

Kaur, A., Verma, K., Bhondekar, A. P., and Shashvat, K. (2019). Implementation of bagged SVM ensemble model for classification of epileptic states using EEG. *Curr. Pharm. Biotechnol.* 20 (9), 755–765. doi:10.2174/1389201020666190618112715

Khan, I. K., Bhuiyan, M., and Kihara, D. (2017). DextMP: Deep dive into text for predicting moonlighting proteins. *Bioinformatics* 33 (14), I83–I91. doi:10.1093/bioinformatics/btx231

Khan, I. K., and Kihara, D. (2016). Genome-scale prediction of moonlighting proteins using diverse protein association information. *Bioinformatics* 32 (15), 2281–2288. doi:10.1093/bioinformatics/btw166

Lee, J., Hyeon, D. Y., and Hwang, D. (2020). Single-cell multiomics: technologies and data analysis methods. *Exp. Mol. Med.* 52 (9), 1428–1442. doi:10.1038/s12276-020-0420-2

Li, Y., Zhao, J. N., Liu, Z. Q., Wang, C. K., Wei, L. Z., Han, S. Y., et al. (2021). De novo prediction of moonlighting proteins using multimodal deep ensemble learning. *Front. Genet.* 12, 630379. doi:10.3389/fgene.2021.630379

Liu, X. Y., Shen, Y. Y., Zhang, Y. H., Liu, F., Ma, Z. Y., Yue, Z. Y., et al. (2021). IdentPMP: Identification of moonlighting proteins in plants using sequence-based learning models. *Peerj* 9, e11900. doi:10.7717/peerj.11900

Manavalan, B., Shin, T. H., and Lee, G. (2018). PVP-SVM: Sequence-Based prediction of phage virion proteins using a support vector machine. *Front. Microbiol.* 9, 476. doi:10.3389/fmicb.2018.00476

Mani, M., Chen, C., Amblee, V., Liu, H. P., Mathur, T., Zwicke, G., et al. (2015). MoonProt: a database for proteins that are known to moonlight. *Nucleic Acids Res.* 43 (D1), D277–D282. doi:10.1093/nar/gku954

Raihan-Al-Masud, M., and Mondal, M. R. H. (2020). Data-driven diagnosis of spinal abnormalities using feature selection and machine learning algorithms. *Plos One* 15 (2), e0228422. doi:10.1371/journal.pone.0228422

Safavian, S. R., and Landgrebe, D. (1991). A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man. Cybern.* 21 (3), 660–674. doi:10.1109/21.97458

Shan, X. Q., Wang, X. G., Li, C. D., Chu, Y. Y., Zhang, Y. F., Xiong, Y., et al. (2019). Prediction of CYP450 enzyme-substrate selectivity based on the network-based label space division method. *J. Chem. Inf. Model.* 59 (11), 4577–4586. doi:10.1021/acs.jcim.9b00749

Shirafkan, F., Gharaghani, S., Rahimian, K., Sajedi, R. H., and Zahiri, J. (2021). Moonlighting protein prediction using physico-chemical and evolutional properties via machine learning methods. *Bmc Bioinforma.* 22 (1), 261. doi:10.1186/s12859-021-04194-5

Su, B., Qian, Z., Li, T. S., Zhou, Y. W., and Wong, A. (2019). PlantMP: A database for moonlighting plant proteins. *Database.*, 2019. baz050. doi:10.1093/database/baz050

Tang, H., Chen, W., and Lin, H. (2016). Identification of immunoglobulins using Chou's pseudo amino acid composition with feature selection technique. *Mol. Biosyst.* 12 (4), 1269–1275. doi:10.1039/c5mb00883b

Wang, C. Y., Li, J. L., Liu, X. Y., and Guo, M. Z. (2020). Predicting sub-golgi apparatus resident protein with primary sequence hybrid features. *Ieee Access* 8, 4442–4450. doi:10.1109/access.2019.2962821

Wang, H., Tang, J. J., Ding, Y. J., and Guo, F. (2021). Exploring associations of non-coding RNAs in human diseases via three-matrix factorization with hypergraph-regular terms on center kernel alignment. *Brief. Bioinform.* 22 (5), bbaa409. doi:10.1093/bib/bbaa409

Wang, Y. B., Ding, Y. J., Guo, F., Wei, L. Y., and Tang, J. J. (2017). Improved detection of DNA-binding proteins via compression technology on PSSM information. *Plos One* 12 (9), e0185587. doi:10.1371/journal.pone.0185587

Wei, L. Y., Xing, P. W., Zeng, J. C., Chen, J. X., Su, R., and Guo, F. (2017). Improved prediction of protein-protein interactions using novel negative samples, features, and an ensemble classifier. *Artif. Intell. Med.* 83, 67–74. doi:10.1016/j.artmed.2017.03.001

Xie, Q., Liu, Z. T., and Ding, X. W. (2018). "Electroencephalogram emotion recognition based on a stacking classification model", in: 37th Chinese Control Conference (CCC)), 5544–5548.

Xiong, Y., Wang, Q. K., Yang, J. C., Zhu, X. L., and Weil, D. Q. (2018). PredT4SE-Stack: Prediction of bacterial type IV secreted effectors from protein sequences using a stacked ensemble method. *Front. Microbiol.* 9, 2571. doi:10.3389/fmicb.2018.02571

Yan, Z. H., Chen, D., Teng, Z. X., Wang, D. H., and Li, Y. J. (2020). SMOPredT4SE: An effective prediction of bacterial type IV secreted effectors using SVM training with SMO. *Ieee Access* 8, 25570–25578. doi:10.1109/access.2020.2971091

Yang, L. W., Gao, H., Wu, K. Y., Zhang, H. T., Li, C. Y., and Tang, L. X. (2020). Identification of cancerlectins by using cascade linear discriminant analysis and optimal g-gap tripeptide composition. *Curr. Bioinform.* 15 (6), 528–537. doi:10.2174/1574893614666190730103156

Zakrzewicz, D., and Geyer, J. (2022). Multitasking Na+/Taurocholate cotransporting polypeptide (NTCP) as a drug target for HBV infection: from protein engineering to drug discovery. *Biomedicines* 10 (1), 196. doi:10.3390/biomedicines10010196

Zaman, F., and Hirose, H. (2008). "A robust bagging method using median as a combination rule", in: 8th IEEE International Conference on Computer and Information Technology), 55–60.

Zhang, L., Xiao, X., and Xu, Z. C. (2020). iPromoter-5mC: a novel fusion decision predictor for the identification of 5-methylcytosine sites in genome-wide DNA promoters. *Front. Cell Dev. Biol.* 8, 614. doi:10.3389/fcell.2020.00614

Zhang, M., Li, F. Y., Marquez-Lago, T. T., Leier, A., Fan, C., Kwoh, C. K., et al. (2019). MULTiPly: a novel multi-layer predictor for discovering general and specific types of promoters. *Bioinformatics* 35 (17), 2957–2965. doi:10.1093/bioinformatics/btz016

Zhao, Y. M., Wang, F., and Juan, L. R. (2015). MicroRNA promoter identification in arabidopsis using multiple histone markers. *Biomed. Res. Int.*, 2015. 861402. doi:10.1155/2015/861402

Zou, Q., Wan, S. X., Ju, Y., Tang, J. J., and Zeng, X. X. (2016). Pretata: predicting TATA binding proteins with novel features and dimensionality reduction strategy. *BMC Syst. Biol.* 10, 114. doi:10.1186/s12918-016-0353-5

Zou, Q., Xing, P. W., Wei, L. Y., and Liu, B. (2019). Gene2vec: gene subsequence embedding for prediction of mammalian N-6-methyladenosine sites from mRNA. *Rna* 25 (2), 205–218. doi:10.1261/rna.069112.118

frontiers | Frontiers in Genetics

TYPE Original Research
PUBLISHED 12 September 2022
DOI 10.3389/fgene.2022.1011659

Check for updates

OPEN ACCESS

EDITED BY
Quan Zou,
University of Electronic Science and Technology of China, China

REVIEWED BY
Yue Zhang,
Harbin Engineering University, China
Xu Liu,
Guangxi University, China

*CORRESPONDENCE
Tao Huang,
tohuangtao@126.com
Yu-Dong Cai,
cai_yud@126.com

†These authors have contributed equally to this work

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 04 August 2022
ACCEPTED 22 August 2022
PUBLISHED 12 September 2022

CITATION
Yang L, Zhang Y-H, Huang F, Li Z, Huang T and Cai Y-D (2022), Identification of protein–protein interaction associated functions based on gene ontology and KEGG pathway. *Front. Genet.* 13:1011659.
doi: 10.3389/fgene.2022.1011659

Identification of protein–protein interaction associated functions based on gene ontology and KEGG pathway

Lili Yang[1†], Yu-Hang Zhang[2†], FeiMing Huang[3†], ZhanDong Li[1], Tao Huang[4,5]* and Yu-Dong Cai[3]*

[1]Measurement Biotechnique Research Center, School of Biological and Food Engineering, Jilin Engineering Normal University, Changchun, China, [2]Channing Division of Network Medicine, Brigham and Women's Hospital, Harvard Medical School, Boston, MA, United States, [3]School of Life Sciences, Shanghai University, Shanghai, China, [4]Bio-Med Big Data Center, CAS Key Laboratory of Computational Biology, Shanghai Institute of Nutrition and Health, University of Chinese Academy of Sciences, Chinese Academy of Sciences, Shanghai, China, [5]CAS Key Laboratory of Tissue Microenvironment and Tumor, Shanghai Institute of Nutrition and Health, University of Chinese Academy of Sciences, Chinese Academy of Sciences, Shanghai, China

Protein–protein interactions (PPIs) are extremely important for gaining mechanistic insights into the functional organization of the proteome. The resolution of PPI functions can help in the identification of novel diagnostic and therapeutic targets with medical utility, thus facilitating the development of new medications. However, the traditional methods for resolving PPI functions are mainly experimental methods, such as co-immunoprecipitation, pull-down assays, cross-linking, label transfer, and far-Western blot analysis, that are not only expensive but also time-consuming. In this study, we constructed an integrated feature selection scheme for the large-scale selection of the relevant functions of PPIs by using the Gene Ontology and Kyoto Encyclopedia of Genes and Genomes (KEGG) pathway annotations of PPI participants. First, we encoded the proteins in each PPI with their gene ontologies and KEGG pathways. Then, the encoded protein features were refined as features of both positive and negative PPIs. Subsequently, Boruta was used for the initial filtering of features to obtain 5684 features. Three feature ranking algorithms, namely, least absolute shrinkage and selection operator, light gradient boosting machine, and max-relevance and min-redundancy, were applied to evaluate feature importance. Finally, the top-ranked features derived from multiple datasets were comprehensively evaluated, and the intersection of results mined by three feature ranking algorithms was taken to identify the features with high correlation with PPIs. Some functional terms were identified in our study, including cytokine–cytokine receptor interaction (hsa04060), intrinsic component of membrane (GO:0031224), and protein-binding biological process (GO:0005515). Our newly proposed integrated computational approach offers a novel perspective of the large-scale mining of biological functions linked to PPI.

KEYWORDS
protein-protein interaction, gene ontology, KEGG pathway, enrichment, feature analysis

1 Introduction

In living creatures, protein–protein interactions (PPIs) are one of the basic formats of molecular interactions that regulate various important biological functions, including cell proliferation, differentiation, and apoptosis. Traditionally, PPIs can be identified by using experimental methods, such as co-immunoprecipitation, pull-down assays, cross-linking, label transfer, and far-Western blot analysis (Hall, 2015; Evans and Paliashvili, 2022; Lyu et al., 2022). Various significant PPIs have been identified by using complex but accurate experiment-based methods. The identified PPIs can be divided into two groups: 1) PPIs that transport cell signals for downstream biological functions. For example, 14-3-3 protein complexes have been reported to interact as cell-signaling transporters with multiple protein molecules via PPIs to regulate inflammatory effects (Munier et al., 2021). 2) PPIs that establish stable complexes. The stable complex of ferrtin is formed by two subunits: the ferrtin heavy chain and the ferrtin light chain (Blankenhaus et al., 2019). Interactions between these two subunits form the stable ferrtin complex and further play a specific role in iron metabolism (Neves et al., 2019).

Although experiment-based approaches have been widely used to recognize various functional PPIs, they are not only expensive but also time-consuming. With the establishment of the PPI databases, advanced computational algorithms, especially machine learning methods, have been introduced to explore new PPIs and identify connections between biological functions and PPIs (Balogh et al., 2022; Gao et al., 2022; Ieremie et al., 2022). Three major aspects of PPIs have been widely reported with the application of machine learning methods: 1) Microbe–host protein interactions. Early in 2019, researchers summarized the optimized methods for selecting features to describe viral protein–host protein interactions; this effort indicated that microbe–host interactions can be predicted by using computational methods (Zheng et al., 2019). 2) Protein interactions in human malignant diseases, such as cancer. In 2020, predicted PPIs were applied to recognize glioma stages; this approach indicated that predicted PPIs can also predict disease progression and thus extended the application of PPIs based on machine learning models (Niu et al., 2020). 3) Predicted protein interactions in drug development. Through the integration of PPIs predicted by a machine learning method and drug physical scoring (Guedes et al., 2021), newly identified PPIs were shown to be robust for drug discovery and pharmalogical mechanism exploration.

Therefore, machine learning methods become more and more popular for new PPI recognition and PPI function exploration. They have been deemed to be one of the major novel tools for PPI studies. As introduced above, PPIs are one of the basic approaches for molecular interactions regulating essential biological functions in all living creatures. Machine learning methods can help recognize key functional potentials that can be attributed to PPIs. In this study, multiple machine learning methods were employed to conduct the investigation. First, each PPI was represented by lots of features derived from gene ontology (GO) terms or Kyoto Encyclopedia of Genes and Genomes (KEGG) pathways of two proteins in the PPI. Then, several machine learning methods, including Boruta (Kursa and Rudnicki, 2010), least absolute shrinkage and selection operator (LASSO) regression (Tibshirani, 1996), light gradient boosting machine (LightGBM) (Ke et al., 2017), and max-relevance and min-redundancy (mRMR) (Peng et al., 2005), were adopted to deeply analyze these features. Key features yielded by different methods were integrated by a comprehensive evaluation method to obtain most essential features. Their corresponding GO terms and KEGG pathways, such as cytokine–cytokine receptor interaction (hsa04060), intrinsic component of membrane (GO:0031224), and protein-binding biological process (GO: 0005515), were analyzed to uncover their relationships to PPIs. This study reflected the important and irreplaceable roles of GO terms and KEGG pathways for PPIs.

2 Materials and methods

2.1 Data acquisition

All human PPIs used in this research were retrieved from STRING (https://string-db.org/, version 9.1) (Franceschini et al., 2013). These interactions were obtained through the following sources: high-throughput experiments, genomic context, (conserved) co-expression, and previous knowledge. PPIs with "Experimental" scores greater than zero were selected, which indicated that these PPIs had been experimentally confirmed. 309,287 human PPIs involving 16,571 proteins were accessed. However, if all of this PPI information was adopted, the subsequent calculations would introduce significant noise due to redundant protein sequences and unmanifested protein functions. The following screening processes were performed to create a well-defined PPI dataset: 1) By applying CD-HIT (Fu et al., 2012), similar proteins were excluded. The similarity of any two remaining proteins was less than 0.25. 2) Proteins without GO terms or KEGG pathways were also discarded. After the above filtering process, 6623 proteins and 70,392 pairs of PPIs were retained. These PPI comprised the positive sample set.

Pairs of proteins without PPIs are also necessary to study the specific function of PPIs. We randomly selected two proteins from the 6623 proteins obtained through the above screening to constitute pairs of PPIs. If the pair did not exist in the positive sample set, it was treated as a negative sample. Through random combination, 21,928,753 pairs can be obtained, including 21,858,361 negative samples and 70,392 positive samples. However, the considerably higher number of negative samples than that of positive samples indicated that the constructed dataset was extremely imbalanced. Direct analysis of such imbalanced dataset would produce bias. As the negative samples were 310 times as many as positive samples, the negative samples were divided into 310 subsets randomly and equally. Each subset was combined with the positive sample set to form a balanced learning dataset. As a result, 310 datasets for subsequent analysis were created.

2.2 Representation of protein–protein function associations

GO terms and KEGG pathways are well-known functional information for deciphering and describing the molecular functions, cellular components, and biological processes of proteins or genes (Kanehisa et al., 2012; Gene Ontology Consortium, 2015). As in our prior study, we used such functional terms (GO terms and KEGG pathways) of proteins to generate the representations of PPIs (Yuan et al., 2019; Zhang et al., 2021). Based on the GO information of a protein p, it can be encoded as

$$v_{GO}(p) = [g_1^p, g_2^p, \ldots, g_n^p]^T, \tag{1}$$

where n is the total number of GO terms (n = 17916 in this study). g_i^p equals 1 if the protein p is annotated by the i -th GO term. Otherwise, g_i^p equals 0. Likewise, p can be encoded as the following vector using its KEGG pathway information

$$v_{kegg}(p) = [k_1^p, k_2^p, \ldots, k_m^p]^T, \tag{2}$$

where g_i^p and k_i^p are also similar in value, and m stands for the number of pathways ($m = 279$ in this study). For a PPI, we cannot simply combine the features of two proteins when generating the features of PPI because the order information of the PPI should be excluded. We utilized the following scheme, which has been employed in some studies (Chen et al., 2013; Ran et al., 2022), to construct the feature vectors of PPIs. The feature vectors for GO terms and KEGG pathways of a PPI consisting of p_1 and p_2 were constructed by using the following scheme:

$$\begin{aligned} V_{GO}(PPI) &= v_{GO}(p_1) \otimes v_{GO}(p_2) \\ &= \left[g_1^{p_1} + g_1^{p_2}, \left|g_1^{p_1} - g_1^{p_2}\right|, \ldots, g_n^{p_1} + g_n^{p_2}, \left|g_n^{p_1} - g_n^{p_2}\right|\right]^T, \end{aligned} \tag{3}$$

$$\begin{aligned} V_{KEGG}(PPI) &= v_{KEGG}(p_1) \otimes v_{KEGG}(p_2) \\ &= \left[k_1^{p_1} + k_1^{p_2}, \left|k_1^{p_1} - k_1^{p_2}\right|, \ldots, k_m^{p_1} + k_m^{p_2}, \left|k_m^{p_1} - k_m^{p_2}\right|\right]^T \end{aligned} \tag{4}$$

By integrating above two feature vectors, we can finally represent the feature vector of the PPI as follows:

$$V(PPI) = V_{GO}(PPI) \otimes V_{KEGG}(PPI) = \begin{bmatrix} V_{GO}(PPI) \\ V_{KEGG}(PPI) \end{bmatrix} \tag{5}$$

2.3 Feature filtering with boruta

A large number of features were used to describe PPIs by using GO terms and KEGG pathways. Evidently, lots of features were unrelated to distinguish positive and negative samples, which must be filtered to reduce the noise in subsequent calculations. Here, Boruta was adopted to exclude irrelated features and retain relevant ones.

Boruta, a wrapper-based feature selection method, uses random forest as the classifier to filter out a set of features that are relevant to the target variable (Kursa and Rudnicki, 2010; Zhang et al., 2020; Chen et al., 2021; Ding et al., 2021; Zhou et al., 2022). It is implemented through the following steps: 1) The features are randomly shuffled and then stitch together with the actual feature matrix to form a new feature matrix. 2) The importance of the shuffled and actual features is obtained by inputting the new feature matrix into the random forest. 3) The actual features with importance greater than the maximum importance of the shuffled features are retained. By iterating the above steps several times, the important features are identified by Boruta.

For this study, the Boruta program retrieved from https://github.com/scikit-learn-contrib/boruta_py was used, which was executed with its default parameters on each of 310 datasets.

2.4 Feature ranking algorithms

Through Boruta, some relevant features can be screened out. However, their contributions for distinguishing positive and negative samples were not same. They should be further analyzed. Here, we ranked these features in accordance with their importance by using three efficient feature ranking algorithms: LASSO (Tibshirani, 1996), LightGBM (Ke et al., 2017), and mRMR (Peng et al., 2005). These feature ranking algorithms are briefly described as below.

In 1996, Tibshirani et al. proposed the LASSO algorithm, which is primarily used to select variables (Tibshirani, 1996). The LASSO method constructs a regression model by employing a penalty function with coefficients, each of which corresponds to one feature. The coefficients of features can be an indicator to measure the importance of features. Accordingly, features can be ranked based on their corresponding coefficients. In this study, the LASSO package collected in Scikit-learn (Pedregosa et al., 2011) was adopted and applied to all 310 datasets for generating feature lists. Such obtained lists were called LASSO feature lists in this study.

LightGBM is a gradient boosting decision tree algorithm that was proposed by Ke et al., in 2017 (Ke et al., 2017; Ding et al., 2022). This method consists of multiple decision trees, and the weights of each tree are considered in the classification. The importance of a feature is determined by the number of times it is used in the constructed decision trees. Accordingly, features can be sorted in a list with the decreasing order of such times. The present study used the LightGBM program downloaded from https://lightgbm.readthedocs.io/en/latest/, which was performed on 310 datasets. For convenience, the lists yielded by LightGBM were called LightGBM feature lists.

The mRMR algorithm is a heuristic feature selection method in which the original features are ranked in accordance with a well-defined scheme (Peng et al., 2005; Wang et al., 2018; Zhao et al., 2018; Chen et al., 2022). This scheme considers that the importance of features is determined by two aspects: relevance to target variable and redundancies to other features. The feature with high relevance to target variable and low redundancies to other features should be assigned a high rank in the final feature list. A loop procedure determines the rank of all features. In each round, the feature with greatest difference between its relevance to target variable and redundancies to already-selected features is selected and appended to the list. This study adopted the mRMR program obtained from http://home.penglab.com/proj/mRMR/. It was executed on each of 310 datasets. The generated lists were termed as mRMR feature lists.

2.5 Comprehensive evaluation of feature lists

Given that the negative samples were randomly chosen and divided into 310 datasets, the features that were selected by Boruta from 310 datasets were distinctive. Given a certain feature ranking algorithm described in Section 2.4, 310 feature lists can be generated, denoted by $F_1, F_2, \cdots F_{310}$. Features occurring in these lists were collected. For one feature f, its rank in F_i was denoted by $R_i(f)$. In particular, if the list did not contain this feature. Its rank was denoted by 0. Furthermore, count the number of lists containing feature f, denoted by $N(f)$. The importance of feature f was measured by the following importance score

$$Importance\ score\ (f) = \frac{M(f)}{W(f)}, \tag{6}$$

where $M(f)$ was the mean ranks of f, calculated by $M(f) = \sum_{i=1}^{310} R_i(f) \Big/ N(f)$, and $W(f)$ represented the weight of f, defined as $N(f)/310$. The numerator in Eq. 6 considered the evaluation results yielded by the feature ranking algorithm on different datasets, whereas the denominator further considered the evaluation results of Boruta on different datasets. Generally, a high weight, i.e., the feature was selected by Boruta on many datasets, suggested the feature was important. In this case, the penalty, the reciprocal of weight, on the mean rank was small. Thus, the smaller the importance score, the more important the feature. All features were ranked in terms of the increasing order of their importance scores. Under such operation, 310 feature lists were integrated into one feature list.

As three feature ranking algorithms were used, three integrated feature lists can be obtained. Top 100 features in each integrated list were picked up. The features that ranked high in all three feature lists were most relevant to PPIs, which were valuable for giving detailed analysis.

3 Results

This study utilized advanced machine learning methods to investigate relevant functional terms of PPIs. The whole analysis process is illustrated Figure 1. The results generated in each step are then described in detail.

3.1 Results of boruta

Our data included 21,928,753 pairs of 6,623 proteins, where 70,392 were positive samples and rest 21,858,361 were negative samples. Negative samples were divided into 310 parts, thereby constructing 310 datasets. PPIs in each dataset were represented by 17,916 features for GO terms and 279 features for KEGG pathways. For each dataset, all features were analyzed by Boruta. Relevant features were selected. Figure 2 shows the number of selected features from each dataset. The number of selected features ranged from 3200 to 3600 with the median of 3423. The majority of datasets selected 3350–3500 features, suggesting that these numbers did not differ considerably. The detailed features selected from each dataset can be found in Supplementary Table S1. Furthermore, we obtained 5684 different features by combining the selected features derived from 310 datasets, which are provided in Supplementary Table S2. Among these 5684 features, 226 features were about KEGG pathways, whereas 5458 features were about GO terms. These features were used in the subsequent comprehensive assessment.

3.2 Results of feature ranking and comprehensive evaluation

Several features were selected by Boruta on each dataset. These features were further analyzed by each feature ranking algorithm, resulting in one feature list. Accordingly, each feature ranking algorithm generated 310 feature lists, which were further integrated into one feature list by comprehensive evaluation method described in Section 2.5. Each of 5684 features was assigned an importance score, which is listed in Supplementary Table S3. The integrated feature list was generated according to above score, which is also provided in Supplementary Table S3.

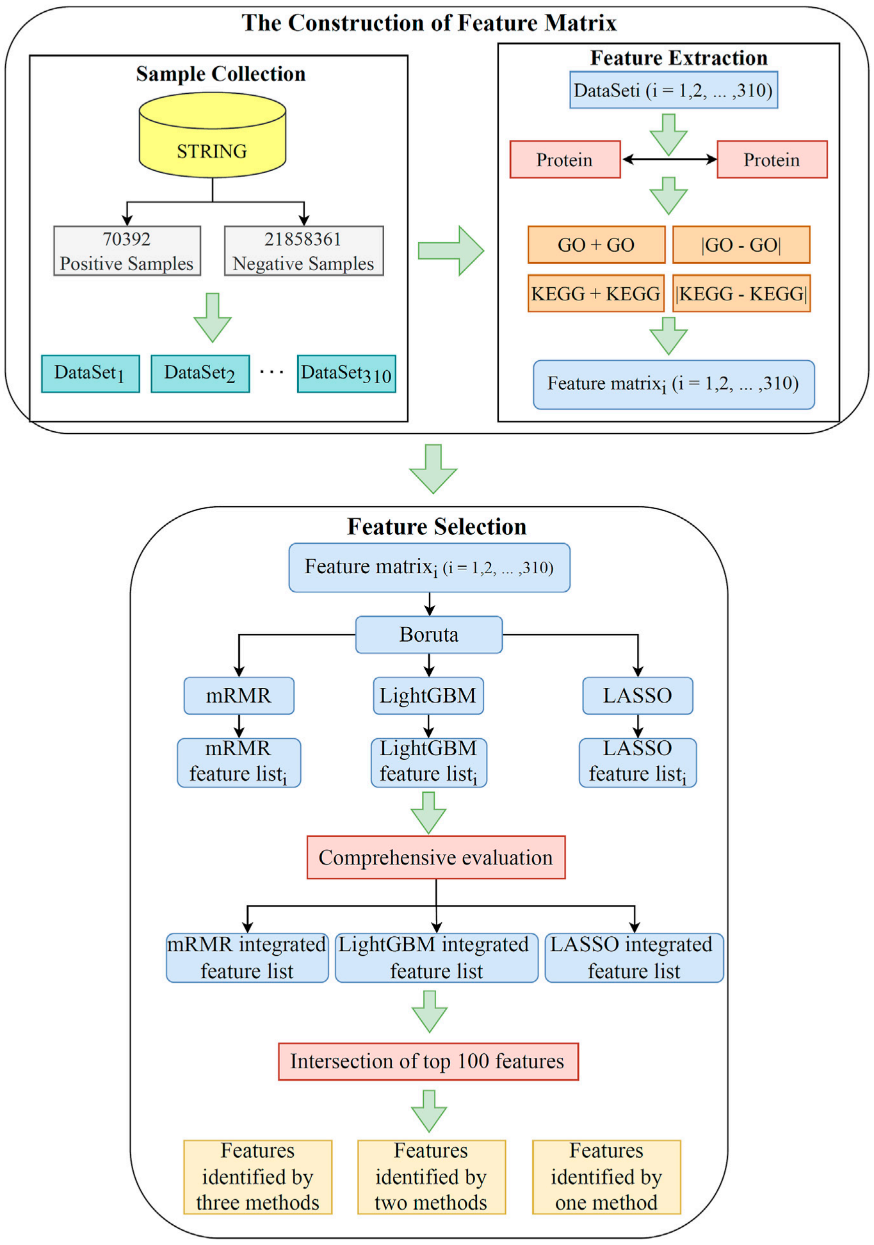

The Construction of Feature Matrix
Sample Collection
STRING
70392 Positive Samples
21858361 Negative Samples
DataSet1
DataSet2
DataSet310
Feature Extraction
DataSeti (i = 1,2, ... ,310)
Protein
Protein
GO + GO
|GO - GO|
KEGG + KEGG
|KEGG - KEGG|
Feature matrixi (i = 1,2, ... ,310)
Feature Selection
Feature matrixi (i = 1,2, ... ,310)
Boruta
mRMR
LightGBM
LASSO
mRMR feature listi
LightGBM feature listi
LASSO feature listi
Comprehensive evaluation
mRMR integrated feature list
LightGBM integrated feature list
LASSO integrated feature list
Intersection of top 100 features
Features identified by three methods
Features identified by two methods
Features identified by one method

FIGURE 1
Flow chart of the whole analytical process. A total of 21,928,753 pairs of PPIs acquired from the STRING database are divided into 70,392 positive samples and 21,858,361 negative samples. The negative samples are randomly and equally divided into 310 subsets, yielding 310 datasets. Each dataset is characterized by using GO terms and KEGG pathways. Subsequently, the features in each dataset are filtered and ranked by using Boruta, LASSO, LightGBM, and mRMR. Finally, the features in 310 datasets are comprehensively evaluated. The intersection of the last three ranked feature lists are taken to obtain essential functional terms that may be highly relevant to the PPI.

From each integrated feature list, top 100 features were picked up for further analysis. The distribution of 100 features selected from each integrated list on GO terms and KEGG pathways is provided in Figure 3. It can be observed that features for GO terms were more than those for KEGG pathways regardless of the feature ranking algorithms. However, the quantities were not same. LASSO identified much less features for GO terms than other two methods. By using multiple algorithms, some common functional terms can be discovered and exclusive terms can be mined by a special algorithm. Comprehensive analysis of functional terms identified by three algorithms

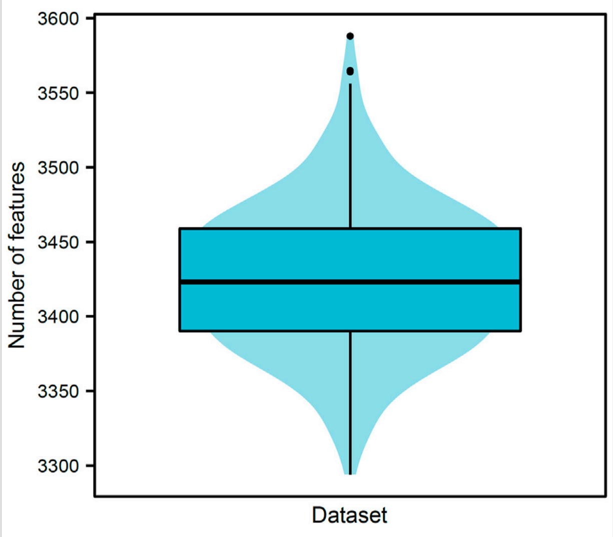


FIGURE 2
Violin plot of the number of features selected by Boruta on 310 datasets. The numbers of selected features vary from 3200 to 3600, and 3350–3500 features are selected in majority datasets (~88.06%). This result indicates that the numbers of selected features are not considerably difference despite the different negative samples in different datasets.

can make the result more complete. In view of this, the intersection operation was performed on the above three feature subsets selected from the integrated feature lists. A Venn diagram was plotted to show the intersections, as illustrated in Figure 4. The detailed features contained in three, two or one subsets are provided in Supplementary Table S4. Eight features occurred in three subsets, which are listed in Table 1. These features were identified and ranked high by all three feature ranking algorithms, indicating they may provide essential contributions for distinguishing positive and negative samples. At the same time, their corresponding GO terms and KEGG pathways can be used to depict PPIs. Furthermore, 50 features were highly ranked by two algorithms, i.e., they contained in two feature subsets. They may also important for uncovering the essential differences between PPIs and general protein pairs. As for the features contained in one subset, i.e., they were identified by one feature ranking algorithm, they can supplement some exclusive differences between PPIs and general protein pairs, which cannot be uncovered by other algorithms. In Section 4, GO terms and KEGG pathways corresponding to some above features would be discussed.

4 Discussion

By using the three feature ranking algorithms of LASSO, LightGBM, and mRMR, we identified some essential biological functional terms that were deemed to be associated with PPIs. We discussed some PPI-associated functional terms identified by using three, two or one algorithms, which are listed in Table 2.

4.1 Key features found by all three feature ranking algorithms

Eight biological functional terms were shown to be associated with the PPIs, which were identified by all three algorithms. The first GO term was intrinsic component of

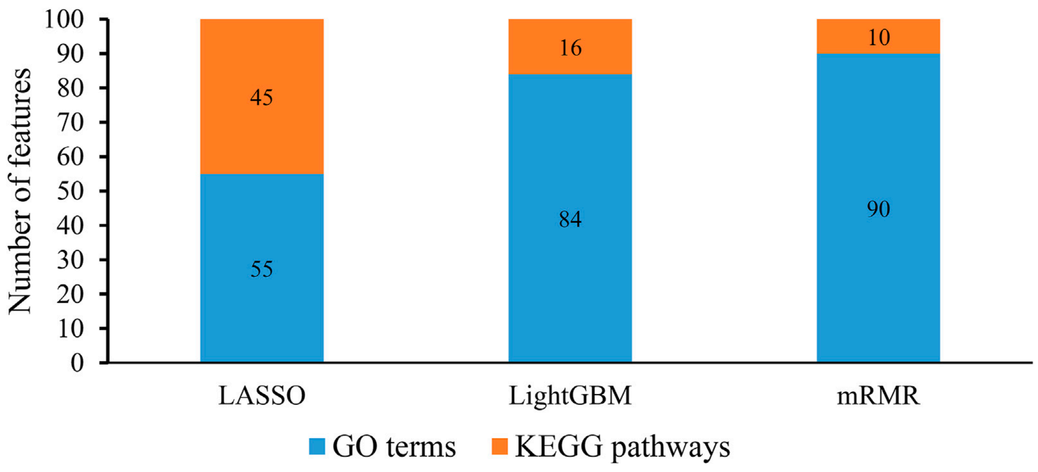


FIGURE 3
Distribution of top 100 features identified by each feature ranking algorithm on gene ontology (GO) terms and KEGG pathways. The identified features for GO terms are more than those for KEGG pathways.

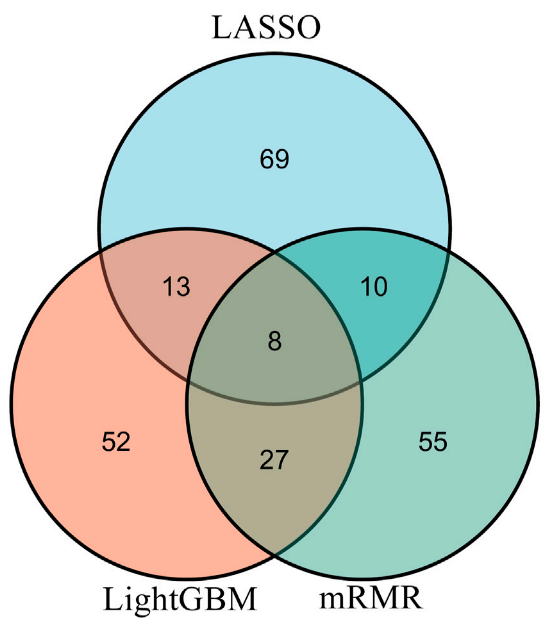


FIGURE 4
Venn diagram of top 100 features in three integrated feature lists obtained by mRMR, LightGBM, and LASSO methods. The overlapping circles indicate the features that are identified by different ranking algorithms. Eight features are identified and ranked highly by all three feature ranking algorithms.

membrane (**GO:0031224**). This term contained multiple functional protein complexes, including anchored component of membrane with PPIs between gp130 and IL-6/IL-6R complex (Narazaki et al., 1993). The linkage of multiple functional PPIs, such as predicted cellular component, to the intrinsic component of membrane validated the efficacy and accuracy of our analysis. Another identified PPI-associated functional term was cytokine–cytokine receptor interaction (**hsa04060**) (Dey et al., 2009), which describes the interaction between membrane-based receptors and soluble cytokines. Considering that cytokines, such as the IL-2, IL-1 and IL-17 family, are small effective soluble proteins, the interactions between cytokines and their respective matched receptors are functional PPIs.

4.2 Key features found by any two feature ranking algorithms

Fifty features were identified by exact two algorithms, which involved 48 biological functional terms. The first predicted GO term was a general description of the protein-binding biological process (**GO:0005515**). The next predicted biological function was the cell cycle (**hsa04110**). Recent publications have shown that cell cycle biological processes involve multiple PPIs. The establishment of PPI networks for the cell cycle in *Saccharomyces cerevisiae* early in 2012 confirmed that the cell cycle involves multiple PPIs (Alberghina et al., 2012; Lu et al., 2020). Further studies on human beings and other eukaryotic creatures also validated the role of such identified PPIs in human beings. These PPIs included interactions between TP53 and MDM2 (Lu et al., 2020) and interactions among PDK1, AKT, and the mTOR complex (Pennington et al., 2018). Therefore, the cell cycle is an effective biological process that involves multiple functional PPIs across different eukaryotic species.

4.3 Key features found by one of the feature ranking algorithms

Although the remaining 176 features were identified by only one algorithm, some of them may also be important. These features were about 149 functional terms. **GO:0043232** describes intracellular nonmembrane-bound organelle. Few PPIs have been observed to be associated with intracellular nonmembrane-bound organelles. Fewer PPIs may be related to nonmembrane bound organelles than to intracellular membrane-based subcellular structures because biological

TABLE 1 Eight features with high ranks yielded by all three feature ranking algorithms.

Feature	Description	Group
abs (GO:0031224_1-GO:0031224_2)	Intrinsic component of membrane	Cellular Component
abs (GO:0044425_1-GO:0044425_2)	Obsolete membrane part	Cellular Component
abs (GO:0005615_1-GO:0005615_2)	Extracellular space	Cellular Component
abs (hsa04060_1-hsa04060_2)	Cytokine-cytokine receptor interaction	KEGG pathway
abs (GO:0071944_1-GO:0071944_2)	Cell periphery	Cellular Component
abs (GO:0007186_1-GO:0007186_2)	G protein-coupled receptor Signaling pathway	Biological Process
abs (hsa04514_1-hsa04514_2)	Cell adhesion molecules	KEGG pathway
hsa04060_1 + hsa04060_2	Cytokine-cytokine receptor interaction	KEGG pathway

TABLE 2 Discussed gene ontology (GO) terms and KEGG pathways.

IDs of GO terms or KEGG pathways	Description	Number of algorithms identified the functional term
GO:0031224	Intrinsic component of membrane	3
hsa04060	Cytokine-cytokine receptor interaction	3
GO:0005515	protein binding	2
hsa04110	Cell cycle	2
GO:0043232	intracellular nonmembrane-bound organelle	1

processes generally involve PPIs, such as cell signaling, immune recognition, and exosome intake, that all depend on biomembrane systems. Therefore, although some pieces of experimental evidence imply that intracellular nonmembrane-bound organelles also involve some PPIs, such as interactions between peptide synthetase and related synthesized proteins (Jaremko et al., 2020).

All in all, as we have discussed above, the biological functional terms predicted by multiple machine learning algorithms have all been confirmed by recent publications with solid experimental support. Therefore, our analyses validated that machine learning algorithms are effective tools for exploring the potential biological functions of PPIs. The application of multiple machine learning algorithms simultaneously may help recognize additional potential PPI-associated functions, thus providing a novel workflow for identifying the biological significance of PPIs.

5 Conclusion

In this research, an integrated feature selection method on GO terms and KEGG pathways was established to distinguish significant PPIs. First, Boruta was applied to obtain a set of features that were highly correlated with PPI functions. Three efficient feature ranking algorithms, namely, LASSO, LightGBM, and mRMR, were adopted to rank the filtered features. The intersection of the top-ranked features in three different feature ranking lists was performed to extract most essential GO terms and KEGG pathways. Some essential PPI-associated functional terms, including cytokine–cytokine receptor interaction, intrinsic component of membrane, and protein-binding biological process, were identified. Furthermore, the functional terms mined in our study were analyzed by reviewing the literature.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found here: https://string-db.org/.

Author contributions

TH and Y-DC designed the study. LY and FH performed the experiments. Y-HZ and ZL analyzed the results. LY, Y-HZ and FH wrote the manuscript. All authors contributed to the research and reviewed the manuscript.

Funding

This work was supported by the Strategic Priority Research Program of Chinese Academy of Sciences [XDB38050200, XDA26040304], National Key R&D Program of China [2018YFC0910403], the Fund of the Key Laboratory of Tissue Microenvironment and Tumor of Chinese Academy of Sciences [202002].

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

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2022.1011659/full#supplementary-material

References

Alberghina, L., Mavelli, G., Drovandi, G., Palumbo, P., Pessina, S., Tripodi, F., et al. (2012). Cell growth and cell cycle in *Saccharomyces cerevisiae*: basic regulatory design and protein–protein interaction network. *Biotechnol. Adv.* 30, 52–72. doi:10.1016/j.biotechadv.2011.07.010

Balogh, O. M., Benczik, B., Horváth, A., Pétervári, M., Csermely, P., Ferdinandy, P., et al. (2022). Efficient link prediction in the protein-protein interaction network using topological information in a generative adversarial network machine learning model. *BMC Bioinforma.* 23, 78. doi:10.1186/s12859-022-04598-x

Blankenhaus, B., Braza, F., Martins, R., Bastos-Amador, P., González-García, I., Carlos, A. R., et al. (2019). Ferritin regulates organismal energy balance and thermogenesis. *Mol. Metab.* 24, 64–79. doi:10.1016/j.molmet.2019.03.008

Chen, L., Li, B. Q., Zheng, M. Y., Zhang, J., Feng, K. Y., and Cai, Y. D. (2013). Prediction of effective drug combinations by chemical interaction, protein interaction and target enrichment of KEGG pathways. *Biomed. Res. Int.* 2013, 723780. doi:10.1155/2013/723780

Chen, L., Li, Z., Zeng, T., Zhang, Y. H., Li, H., Huang, T., et al. (2021). Predicting gene phenotype by multi-label multi-class model based on essential functional features. *Mol. Genet. Genomics.* 296, 905–918. doi:10.1007/s00438-021-01789-8

Chen, L., Li, Z., Zhang, S., Zhang, Y.-H., Huang, T., and Cai, Y.-D. (2022). Predicting RNA 5-methylcytosine sites by using essential sequence features and distributions. *Biomed. Res. Int.* 2022, 4035462. doi:10.1155/2022/4035462

Dey, R., Ji, K., Liu, Z., and Chen, L. (2009). A cytokine-cytokine interaction in the assembly of higher-order structure and activation of the interleukine-3:receptor complex. *PLoS One* 4, e5188. doi:10.1371/journal.pone.0005188

Ding, S., Li, H., Zhang, Y. H., Zhou, X., Feng, K., Li, Z., et al. (2021). Identification of pan-cancer biomarkers based on the gene expression profiles of cancer cell lines. *Front. Cell Dev. Biol.* 9, 781285. doi:10.3389/fcell.2021.781285

Ding, S., Wang, D., Zhou, X., Chen, L., Feng, K., Xu, X., et al. (2022). Predicting heart cell types by using transcriptome profiles and a machine learning method. *Life* 12, 228. doi:10.3390/life12020228

Evans, I. M., and Paliashvili, K. (2022). Co-immunoprecipitation assays. *Methods Mol. Biol.* 2475, 125–132. doi:10.1007/978-1-0716-2217-9_8

Franceschini, A., Szklarczyk, D., Frankild, S., Kuhn, M., Simonovic, M., Roth, A., et al. (2013). STRING v9. 1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Res.* 41, D808–D815. doi:10.1093/nar/gks1094

Fu, L., Niu, B., Zhu, Z., Wu, S., and Li, W. (2012). CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics* 28, 3150–3152. doi:10.1093/bioinformatics/bts565

Gao, M., Nakajima An, D., Parks, J. M., and Skolnick, J. (2022). AF2Complex predicts direct physical interactions in multimeric proteins with deep learning. *Nat. Commun.* 13, 1744. doi:10.1038/s41467-022-29394-2

Gene Ontology Consortium (2015). Gene ontology Consortium: going forward. *Nucleic Acids Res.* 43, D1049–D1056. doi:10.1093/nar/gku1179

Guedes, I. A., Barreto, A., Marinho, D., Krempser, E., Kuenemann, M. A., Sperandio, O., et al. (2021). New machine learning and physics-based scoring functions for drug discovery. *Sci. Rep.* 11, 3198. doi:10.1038/s41598-021-82410-1

Hall, R. A. (2015). Studying protein-protein interactions via blot overlay/far Western blot. *Methods Mol. Biol.* 1278, 371–379. doi:10.1007/978-1-4939-2425-7_24

Ieremie, I., Ewing, R. M., and Niranjan, M. (2022). TransformerGO: Predicting protein-protein interactions by modelling the attention between sets of gene ontology terms. *Bioinformatics* 38, 2269–2277. doi:10.1093/bioinformatics/btac104

Jaremko, M. J., Davis, T. D., Corpuz, J. C., and Burkart, M. D. (2020). Type II non-ribosomal peptide synthetase proteins: structure, mechanism, and protein–protein interactions. *Nat. Prod. Rep.* 37, 355–379. doi:10.1039/c9np00047j

Kanehisa, M., Goto, S., Sato, Y., Furumichi, M., and Tanabe, M. (2012). KEGG for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Res.* 40, D109–D114. doi:10.1093/nar/gkr988

Ke, G., Meng, Q., Finely, T., Wang, T., Chen, W., Ma, W., et al. (2017). "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in neural information processing systems* (Redmond, WA, United States: NIP), 30.

Kursa, M. B., and Rudnicki, W. R. (2010). Feature selection with the Boruta package. *J. Stat. Softw.* 36, 1–13. doi:10.18637/jss.v036.i11

Lu, H., Zhou, Q., He, J., Jiang, Z., Peng, C., Tong, R., et al. (2020). Recent advances in the development of protein–protein interactions modulators: mechanisms and clinical trials. *Signal Transduct. Target. Ther.* 5, 213. doi:10.1038/s41392-020-00315-3

Lyu, S., Zhang, C., Hou, X., and Wang, A. (2022). Tag-based pull-down assay. *Methods Mol. Biol.* 2400, 105–114. doi:10.1007/978-1-0716-1835-6_11

Munier, C. C., Ottmann, C., and Perry, M. W. D. (2021). 14-3-3 modulation of the inflammatory response. *Pharmacol. Res.* 163, 105236. doi:10.1016/j.phrs.2020.105236

Narazaki, M., Yasukawa, K., Saito, T., Ohsugi, Y., Fukui, H., Koishihara, Y., et al. (1993). Soluble forms of the interleukin-6 signal-transducing receptor component gp130 in human serum possessing a potential to inhibit signals through membrane-anchored gp130. *Blood* 82, 1120–1126. doi:10.1182/blood.v82.4.1120.1120

Neves, J., Haider, T., Gassmann, M., and Muckenthaler, M. U. (2019). Iron homeostasis in the lungs—a balance between health and disease. *Pharmaceuticals* 12, 5. doi:10.3390/ph12010005

Niu, B., Liang, C., Lu, Y., Zhao, M., Chen, Q., Zhang, Y., et al. (2020). Glioma stages prediction based on machine learning algorithm combined with protein-protein interaction networks. *Genomics* 112, 837–847. doi:10.1016/j.ygeno.2019.05.024

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* 12, 2825–2830.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 1226–1238. doi:10.1109/TPAMI.2005.159

Pennington, K., Chan, T., Torres, M., and Andersen, J. (2018). The dynamic and stress-adaptive signaling hub of 14-3-3: emerging mechanisms of regulation and context-dependent protein–protein interactions. *Oncogene* 37, 5587–5604. doi:10.1038/s41388-018-0348-3

Ran, B., Chen, L., Li, M., Han, Y., and Dai, Q. (2022). Drug-Drug interactions prediction using fingerprint only. *Comput. Math. Methods Med.* 2022, 7818480. doi:10.1155/2022/7818480

Tibshirani, R. J. (1996). Regression shrinkage and selection via the lasso: a retrospective. *J. R. Stat. Soc. Ser. B* 73, 273–282. doi:10.1111/j.1467-9868.2011.00771.x

Wang, T., Chen, L., and Zhao, X. (2018). Prediction of drug combinations with a network embedding method. *Comb. Chem. High. Throughput Screen.* 21, 789–797. doi:10.2174/1386207322666181226170140

Yuan, F., Pan, X., Chen, L., Zhang, Y. H., Huang, T., and Cai, Y. D. (2019). Analysis of protein-protein functional associations by using gene ontology and KEGG pathway. *Biomed. Res. Int.* 2019, 4963289. doi:10.1155/2019/4963289

Zhang, Y. H., Li, H., Zeng, T., Chen, L., Li, Z., Huang, T., et al. (2020). Discriminating origin tissues of tumor cell lines by methylation signatures and dys-methylated rules. *Front. Bioeng. Biotechnol.* 8, 507. doi:10.3389/fbioe.2020.00507

Zhang, Y. H., Zeng, T., Chen, L., Huang, T., and Cai, Y. D. (2021). Determining protein-protein functional associations by functional rules based on gene ontology and KEGG pathway. *Biochim. Biophys. Acta. Proteins Proteom.* 1869, 140621. doi:10.1016/j.bbapap.2021.140621

Zhao, X., Chen, L., and Lu, J. (2018). A similarity-based method for prediction of drug side effects with heterogeneous information. *Math. Biosci.* 306, 136–144. doi:10.1016/j.mbs.2018.09.010

Zheng, N., Wang, K., Zhan, W., and Deng, L. (2019). Targeting virus-host protein interactions: Feature extraction and machine learning approaches. *Curr. Drug Metab.* 20, 177–184. doi:10.2174/1389200219666180829121038

Zhou, X., Ding, S., Wang, D., Chen, L., Feng, K., Huang, T., et al. (2022). Identification of cell markers and their expression patterns in skin based on single-cell RNA-sequencing profiles. *Life* 12, 550. doi:10.3390/life12040550

Frontiers in Genetics

TYPE Original Research
PUBLISHED 12 September 2022
DOI 10.3389/fgene.2022.1012828

Check for updates

OPEN ACCESS

EDITED BY
Quan Zou,
University of Electronic Science and Technology of China, China

REVIEWED BY
Chunyu Wang,
Harbin Institute of Technology, China
Ping Xuan,
Heilongjiang University, China

*CORRESPONDENCE
Yu Chen,
nefu_chenyu@163.com

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 05 August 2022
ACCEPTED 22 August 2022
PUBLISHED 12 September 2022

CITATION
Chen D, Li S and Chen Y (2022), ISTRF: Identification of sucrose transporter using random forest.
Front. Genet. 13:1012828.
doi: 10.3389/fgene.2022.1012828

ISTRF: Identification of sucrose transporter using random forest

Dong Chen[1], Sai Li[2] and Yu Chen[2]*

[1]College of Electrical and Information Engineering, Qu Zhou University, Quzhou, China, [2]College of Information and Computer Engineering, Northeast Forestry University, Harbin, China

Sucrose transporter (SUT) is a type of transmembrane protein that exists widely in plants and plays a significant role in the transportation of sucrose and the specific signal sensing process of sucrose. Therefore, identifying sucrose transporter is significant to the study of seed development and plant flowering and growth. In this study, a random forest-based model named ISTRF was proposed to identify sucrose transporter. First, a database containing 382 SUT proteins and 911 non-SUT proteins was constructed based on the UniProt and PFAM databases. Second, k-separated-bigrams-PSSM was exploited to represent protein sequence. Third, to overcome the influence of imbalance of samples on identification performance, the Borderline-SMOTE algorithm was used to overcome the shortcoming of imbalance training data. Finally, the random forest algorithm was used to train the identification model. It was proved by 10-fold cross-validation results that k-separated-bigrams-PSSM was the most distinguishable feature for identifying sucrose transporters. The Borderline-SMOTE algorithm can improve the performance of the identification model. Furthermore, random forest was superior to other classifiers on almost all indicators. Compared with other identification models, ISTRF has the best general performance and makes great improvements in identifying sucrose transporter proteins.

KEYWORDS

machine learning, biological sequence analysis, protein identification, sucrose transporter, random forest

1 Introduction

Sucrose is a kind of disaccharide, which is formed by the condensation of fructose and glucose molecules through dehydration and is widely found in various tissues of plants. In the process of plant photosynthesis, carbon transport is mainly in the form of sucrose (Kühn et al., 1999). Therefore, the distribution of sucrose directly affects the growth and yield of plants (Aluko et al., 2021; Mangukia et al., 2021). In terms of physical properties, sucrose is a non-reducing sugar, which can carry a large amount of carbon. In terms of chemical properties, its properties are very stable, and it is not easy to combine with other compounds during transportation, so it has a certain protective effect on carbon. In terms of biological properties, due to the carbon in sucrose having a higher osmotic potential, the transport speed of sucrose is faster in a sieve tube. Sucrose transporters affect the transport of sucrose, which is mainly distributed in parenchyma cells, companion cells, and vacuolar membranes. They are the mediators of sucrose transport from source leaves to the phloem. In addition, sucrose transporters also exist in sink organs, such as stems, seeds, and fruits. Sucrose transporters can promote sucrose transport under Pi starvation, salinity, and drought stress (Al-Sheikh Ahmed et al., 2018). At present, many experts have carried out a lot of research studies on sucrose transporters and found sucrose transporters in a variety of plant species, such as rice (Aoki et al., 2003), maize (Tran et al., 2017), grapevine, and tobacco (Wang et al., 2019). Endler et al. (2006) discovered a new sucrose transporter on the vacuolar membrane. They used liquid chromatography–tandem mass spectrometry to analyze tonoplast proteins and identified 101 proteins, including sucrose transporters. By studying the sucrose transporter gene RUSUT2 in blackberry, Yan et al. (2021) found that the sucrose content of mature leaves of the transgenic tobacco is enhanced by the overexpression of RUSUT2. At the same time, they found that Rusut2 has transport activity and may participate in sucrose transport during the growth and development of blackberry plants.

With the development of bioinformatics, more and more scholars used machine learning methods to identify sugar transporters. Mishra et al. (2014) developed a new model that incorporated the PSSM profile, amino acid composition, and biochemical composition of transporter proteins. The SVM algorithm was used as a classifier to classify transporters. Based on Mishra's experiments, Alballa et al. (2020) used a series of features including position information, evolutionary information, and amino acid composition to improve the accuracy and MCC of transporter classification. Ho et al. (2019) used word embedding technology to extract effective features from protein sequences and then adopted traditional machine learning methods to classify a variety of transporters (including sugar transporters). It has been proved that machine learning can effectively solve some problems of protein classification. All of the above studies focused on sugar transporters, while Shah et al. proposed to use natural language processing technology BERT to carry out feature extraction of glucose transporters in sugar transporters and classify three glucose transporters through an SVM classifier (Shah et al., 2021). Using machine learning methods to identify special proteins has become a trend, and a machine learning frame has been employed to identify sugar transporters. All these previous works guide us to build a frame for identifying sucrose transporters. In this study, we constructed an identification model named ISTRF to identify sucrose transporters. First, a dataset is built. Second, protein sequences are encoded with k-separated-bigrams-PSSM. Third, the Borderline-SMOTE algorithm is used to augment the positive samples. Finally, the identification model is trained by the random forest algorithm.

2 Materials and methods

2.1 Frame chart of ISTRF

In the study, we proposed a novel identification model called ISTRF, the frame chart of which is shown in Figure 1. First of all, the sucrose and non-sucrose transporter datasets are obtained using sequence homology analysis technology based on the Uniprot and Pfam databases, and then the CD-HIT program was used to remove redundancy and delete the protein sequences with more than 60% similarity. The sucrose transporter samples are construed for the training identification model. Second, we extracted the k-separated-bigrams-PSSM feature to represent samples. Third, we augment the positive samples to balance the training samples by using the Borderline-SMOTE technology. Finally, we built a random forest-based classifier that takes the balancing feature vectors as input. In the following sections, the dataset, feature extraction, sample balancing, and classifiers will be, respectively, introduced in detail.

2.2 Dataset

In this study, a self-built dataset is constructed and used. To obtain a reliable experimental result, it is necessary to use a high-quality benchmark data set, and then the initial data must be processed strictly and standardly. UniProt (Consortium, 2019) database is an authoritative protein database, in which we searched by the keyword "sucrose transporter" to obtain the initial positive sample data set. From the protein family database PFAM (Mistry et al., 2021), families containing positive samples were deleted, and the protein sequence with the longest length was extracted from every remaining family as a negative sample to construct the initial negative sample data set. Next, we processed the initial data set. The first step was to delete the protein sequences containing illegal characters; the second step was to delete the protein sequences with a length less than 50; in the third step, the CD-HIT (Fu et al., 2012) program was used to remove redundancy and delete the protein sequences with more than 60% similarity. We eventually obtained 382 SUTs and 9,109 non-SUTs. This data set is extremely unbalanced, so we divided the negative sample data set into ten equally and took one as the experimental data, which is 911 non-SUTs. We divided the data set into an 80% training dataset and a 20% testing dataset and constructed the dataset as shown in Table 1.

2.3 Feature extraction

In the process of protein identification, feature extraction is a crucial step (Yang and Jiao, 2021). To improve the identification performance of the model, we tried to extract features with high identification and good specificity. In this study, we considered this problem from two perspectives, namely, physicochemical properties and evolutionary information. We tried three features and their various combinations. Finally, the k-separated-bigrams-PSSM which has the best performance according to the experimental result is selected as the feature representation method in our model.

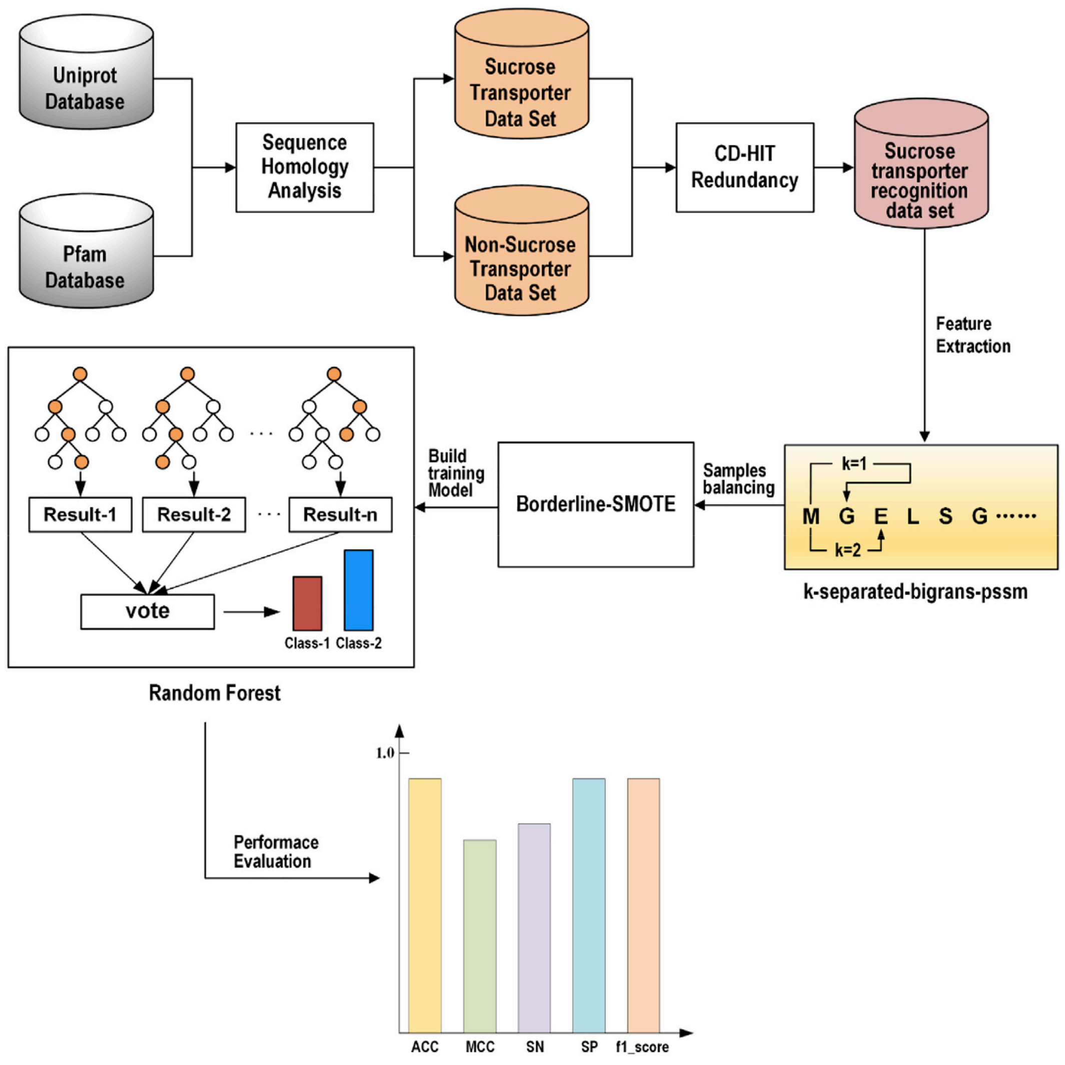


FIGURE 1
Frame chart of ISTRF.

TABLE 1 Self-built dataset.

Dataset	SUT	Non-SUT
Training dataset	306	729
Testing dataset	76	182

2.3.1 188D

188D includes the frequency of 20 amino acids and eight physical and chemical properties (Cai et al., 2003).

The formula for calculating the frequency of 20 amino acids is as follows:

$$F_i = \frac{N_i}{L}, (i = A, C, D, \ldots, Y),$$

where N_i is the number of amino acid type i, and L is the length of a protein sequence.

The composition, transition, and distribution are used to describe eight physicochemical properties of proteins (Xiong et al., 2018; Zou et al., 2019; Masoudi-Sobhanzadeh et al., 2021). Taking the hydrophobicity attribute as an example, “RKEDQN” is polar, “GASTPHY” is neutral, and “CVLIMFW” is hydrophobic. The frequency of each group can be expressed as follows:

$$C_i = \frac{N_i}{L}, i \in \{polar, neutral, hydrophobic\}.$$

The transition from polar group to neutral group is the frequency of polar residue following neutral residue or neutral residue following polar residue. The transition between the neutral group and hydrophobic group and the transition between the hydrophobic group and polar group have similar definitions. It can be expressed by the following formula:

$$T(i_1, i_2) = \frac{N(i_1, i_2) + N(i_2, i_1)}{L-1}, (i_1, i_2) \in \{(polar, neutral), (neutral, hydrophobic), (hydrophobic, polar)\}.$$

The distribution consists of five values, which are the first, 25, 50, 75, and 100% positions of each group of amino acid in the sequence.

2.3.2 PSSM composition

PSSM composition is a feature that describes the evolutionary information of protein sequences, and it is also used to identify a variety of proteins (Wang et al., 2018; Ali et al., 2020; Qian et al., 2021). First, we run the PSI-BLAST tool (Ding et al., 2014) against the Uniref50 database with the e-value set to 0.001. We can obtain the original PSSM profile. Then, we summed the same amino acid rows together and divided the results by the number of amino acids in the protein sequence. Finally, a 400-dimensional PSSM composition was obtained.

2.3.3 The k-separated-bigrams-PSSM

The k-separated-bigrams-PSSM is generated from the original PSSM profile by column transformation. It represents the transition probabilities from one amino acid to another amino acid in a protein sequence (Wang et al., 2020), and the interval of the two amino acids is K. N represents the PSSM matrix, and L is the number of amino acids in the protein sequence and also the number of rows in the PSSM matrix. The transition from the m-th amino acid to the n-th amino acid can be expressed by the following formulas:

$$T_{m,n}(k) = \sum_{i=1}^{L-k} N_{i,m} N_{i+k,n},$$

where $1 \le m \le 20$, $1 \le n \le 20$, and $1 \le k \le K$.

$$T(k) = [T_{1,1}(k), T_{1,2}(k), \ldots, T_{1,20}(k), T_{2,1}(k), \ldots, T_{2,20}(k), \ldots, T_{20,1}(k), \ldots, T_{20,20}(k)].$$

For each k, T(k) is a 400-dimensional feature that represents 400 amino acid transitions. The k ranges from 1 to 11. When k is set to 1, it represents the transition probabilities between neighboring amino acids; when k is set to 2, it represents the transition probabilities between amino acids with one amino acid between them. We can obtain k-separated-bigrams-PSSM (k = 1) and a PSSM-related transformation matrix through POSSUM (Wang et al., 2017). The website is open, and users can easily obtain the required features.

2.4 Sample balancing

The training dataset constructed in Section 2.2 is an imbalance dataset, on which the classifier trained is biased to identify the unseen sample as the majority class (Shabbir et al., 2021). Therefore, we use the Borderline-SMOTE algorithm to balance the feature set. The SMOTE (Chawla et al., 2002) algorithm is an oversampling technique for synthesizing minority classes. It uses the KNN algorithm to calculate the k nearest neighbors of each minority class sample, randomly selects N samples, and performs random linear interpolation on the k nearest neighbors to construct new minority class samples. However, it does not consider the position of the adjacent majority class samples, which usually leads to the phenomenon of sample overlap and affects the classification effect (Chen et al., 2021). Borderline-SMOTE (Han et al., 2005) is an improved oversampling algorithm based on SMOTE. Because the boundary samples are more likely to be misclassified than those far from the boundary, the algorithm only oversamples the boundary samples of the minority class. In the Borderline-SMOTE algorithm, we used the KNN algorithm with k = 5 to balance the feature set of sucrose transporters, so that the 306 SUTs and 729 non-SUTs in the training set were expanded to 729 SUTs and 729 non-SUTs.

2.5 Classifier

In this study, we tried a lot of classification algorithms such as SVM, naive Bayes, SGD, and random forest (Ao et al., 2022). Eventually, we selected the random forest as our classifier based on the experimental results shown in Section 3.3. These machine learning algorithms can be implemented by the WEKA (Holmes et al., 1994; Garner, 1995) software. WEKA is an open data mining platform that can perform data processing such as classification, regression, and clustering. It contains a variety of machine learning algorithms and is simple to operate.

SVM is a supervised learning algorithm and is implemented by the SMO (sequential minimal optimization) algorithm in WEKA (Vapnik, 2006). The classical SVM algorithm has been applied to many problems of bioinformatics, especially in binary classification (Manavalan et al., 2018; Zhang et al., 2019; Ao et al., 2021; Zeng et al., 2021). The main idea is to find an optimal segmentation hyperplane and measure the maximum geometric distance between the nearest sample and the hyperplane so as to divide the data set correctly. The SMO algorithm is an improved support vector machine algorithm that aims to improve the efficiency of the support vector machine. It breaks the large quadratic programming (QP) problem into many smaller QP problems and avoids the problem that the time-consuming numerical QP optimization is used in the inner loop (Platt, 1998).

Naive Bayes is a very classical and simple classification algorithm (Cao et al., 2003). The idea of the algorithm is also very simple. For a given sample to be classified, the probability that it belongs to the positive sample and the negative sample is solved firstly. And then the sample will be classified into the category with the higher probability. It assumes that each input variable is independent. Although real life cannot meet this assumption, it is still valid for most complex problems.

Stochastic gradient descent (SGD) is often used to learn linear classifiers under convex loss functions such as logistic regression and support vector machines (Bottou, 2010). The SGD algorithm is proposed to solve the problem that batch gradient descent needs to use all the samples for each parameter update, and the speed is slow when the number of samples is large. The characteristic of the SGD algorithm is that in each iteration, a group of samples is randomly chosen for training. After N iterations, it finds out the coefficient which leads to the minimum error of these models.

Random forest is based on the idea of ensemble learning, and it integrates multiple decision trees to obtain classification results (Breiman, 2001). First of all, select k samples repeatedly and randomly from the original training sample set N to generate a new training sample set. Then, n decision trees are generated using the training sample set as input. These decision trees form a random forest. Each decision tree is a classifier. As many decision trees as there are, there are as many classification results. Finally, the random forest integrates the classification results of n decision trees and identifies the class with most votes as the classification result of the sample. Because of this randomness, the random forest has a good anti-noise capability and is very suitable for processing high-dimensional data and avoiding overfitting. In many studies, random forest has shown a good classification effect (Lv et al., 2019; Ru et al., 2019; Ao et al., 2020; Lv et al., 2020; Petry et al., 2020; Zhang et al., 2021a).

2.6 Measurement

We used five indicators to evaluate the performance of our identification model: sensitivity (SN), specificity (SP), accuracy (ACC), Marshall correlation coefficient (MCC), and F-measure (Basith et al., 2020; Zhang et al., 2021b; Lee et al., 2021). These evaluation indicators were the results of the confusion matrix calculation obtained from the experiment, and the calculation formula is as follows:

$$SN = \frac{TP}{TP + FN}$$

$$SP = \frac{TN}{TN + FP}$$

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}$$

$$FR = \frac{TP}{TP + FP}$$

$$F - Measure = \frac{2 \times SN \times PR}{SN + PR}$$

where TP represents the number of correctly predicted sucrose transporters, TN represents the number of correctly predicted non-sucrose transporters, FP represents the number of incorrectly predicted sucrose transporters as non-sucrose transporters, and FN represents the number of incorrectly predicted non-sucrose transporters as sucrose transporters.

3 Results and discussion

3.1 Performance of different features

As shown in the frame chart of ISTRF in Section 2.1, our model extracted the k-separated-bigrams-PSSM feature to encode samples. To prove the effectiveness of our feature extraction method, we conducted experiments to compare the performance of different feature extraction algorithms. Specifically, we selected 188D, PSSM composition, k-separated-bigrams-PSSM, and their combinations. 188D feature reflected the frequency of 20 amino acids and eight physical and chemical properties, while PSSM composition and k-separated-bigrams-PSSM reflected the evolutionary information of protein sequences. We used the random forest as a classifier and did not apply Borderline-SMOTE to the extracted feature, and the experimental results of different features on 10-fold cross-validation are shown in Table 2. Bold values in the table indicate the best results. According to the number of indicators with the highest value, the number of k-separated-bigrams-PSSM is 4, the number of combinational features of 188D and k-separated-bigrams-PSSM is 4, and the number of other features and combinational features is lower or equal to 1. The k-separated-bigrams-PSSM has fewer feature numbers than the combination of 188D and k-separated-bigrams-PSSM; therefore, the former has the best performance according to the number of indicators with the highest value. According to the indicator of ACC and MCC, k-separated-bigrams-PSSM still has the highest value, and it verified that k-separated-bigrams-PSSM has the best general performance. Considering the indicator of SN, our used k-separated-bigrams-PSSM also has the maximum value. It verifies that our feature extraction method has better performance than other methods in predicting sucrose transporter protein from positive examples. Considering the indicator of SP, our feature extraction method is slightly lower than the combinational feature of PSSM composition and k-separated-bigrams- PSSM and is equal to or higher than other methods. However, the indicators of SN, MCC, and ACC of our feature extraction method are obviously larger than the combinational feature of PSSM composition and k-separated-bigrams-PSSM, which verify that

TABLE 2 Result of various feature extraction methods using random forest without Borderline-SMOTE on 10-fold cross-validation.

Feature	SN	SP	ACC	MCC	F-measure
188D	0.895	0.970	0.948	0.874	0.910
PSSM composition	0.876	0.967	0.940	0.855	0.896
k-separated-bigrams-PSSM	**0.925**	0.973	**0.958**	**0.900**	**0.929**
188D + PSSM composition	0.895	0.973	0.950	0.878	0.913
188D + k-separated-bigrams-PSSM	**0.925**	0.973	**0.958**	**0.900**	**0.929**
PSSM composition + k-separated-bigrams-PSSM	0.908	**0.978**	0.957	0.897	0.927
188D + PSSM composition + k-separated-bigrams-PSSM	0.918	0.973	0.957	0.895	0.926

Bold values in the table indicate the best results.

TABLE 3 Result of various feature extraction methods using SGD without Borderline-SMOTE on 10-fold cross-validation.

Feature	SN	SP	ACC	MCC	F-measure
188D	0.866	0.951	0.926	0.821	0.873
PSSM composition	0.873	0.956	0.931	0.834	0.883
k-separated-bigrams-PSSM	**0.964**	0.952	**0.956**	**0.897**	**0.928**
188D + PSSM composition	0.902	0.959	0.942	0.861	0.902
188D + k-separated-bigrams-PSSM	0.912	0.952	0.940	0.857	0.900
PSSM composition + k-separated-bigrams-PSSM	0.905	**0.967**	0.949	0.877	0.913
188D + PSSM composition + k-separated-bigrams-PSSM	0.912	0.960	0.946	0.870	0.909

Bold values in the table indicate the best results.

TABLE 4 Result of various features using random forest with Borderline-SMOTE on 10-fold cross-validation.

Feature	SN	SP	ACC	MCC	F-measure
188D	0.989 + 9.4	0.937–3.3	0.963 + 1.5	0.927 + 5.3	0.964 + 5.4
PSSM composition	0.982 + 10.6	0.952–1.5	0.967 + 2.7	0.935 + 8	0.968 + 7.2
k-separated-bigrams-PSSM	0.986 + 6.1	0.970–0.3	0.978 + 2	0.956 + 5.6	0.978 + 4.9
188D + PSSM composition	0.982 + 8.7	0.952–2.1	0.967 + 1.7	0.935 + 5.7	0.968 + 5.5
188D + k-separated-bigrams-PSSM	0.984 + 5.9	0.945–2.8	0.964 + 0.6	0.929 + 2.9	0.965 + 3.6
PSSM composition + k-separated-bigrams-PSSM	0.984 + 7.6	0.957–2.1	0.971 + 1.4	0.941 + 4.4	0.971 + 4.4
188D + PSSM composition + k-separated-bigrams-PSSM	0.985 + 6.7	0.949–2.4	0.967 + 1	0.935 + 4	0.968 + 4.2

the combinational feature of PSSM composition and k-separated-bigrams-PSSM trends to identify a protein as a non-sucrose transporter protein. Based on the fact that training data are an unbalanced data set in which negative samples are larger than positive ones, our feature extraction method is less affected by unbalanced data. After balancing the training data, the SN of our feature method is larger than the combinational feature of PSSM composition and k-separated-bigrams-PSSM, and the detailed experimental results are shown in Section 3.2. Therefore, from the overall perspective, our method obviously performs better than all other methods.

To further illustrate that our feature extraction method also has better performance using other classifiers, we also conducted experiments on different features using an SGD classifier. Table 3 shows the experimental results. As we can see from Table 3, our feature extraction method has better performance than other methods according to the number of indicators with the highest value or ACC indicator or MCC indicator. All in all, our feature extraction method performs better than other feature extraction methods.

3.2 Experiments on sample balancing

As shown in the frame chart of ISTRF in Section 2.1, the sucrose transporter database built in this study has more negative samples than positive ones, and it is an imbalanced dataset that influences the classification performance of the machine learning

TABLE 5 Result of various features using SGD with Borderline-SMOTE on 10-fold cross-validation.

Feature	SN	SP	ACC	MCC	F-measure
188D	0.966 + 10	0.909–4.2	0.938 + 1.2	0.877 + 5.6	0.939 + 6.6
PSSM composition	0.975 + 10.2	0.938–1.8	0.957 + 2.6	0.914 + 8	0.958 + 7.5
k-separated-bigrams-PSSM	0.997 + 3.3	0.942–1	0.970 + 1.4	0.941 + 4.4	0.971 + 4.3
188D + PSSM composition	0.985 + 8.3	0.931–2.8	0.958 + 1.6	0.918 + 5.7	0.959 + 5.7
188D + k-separated-bigrams-PSSM	0.985 + 7.3	0.931–2.1	0.958 + 1.8	0.918 + 6.1	0.959 + 5.9
PSSM composition + k-separated-bigrams-PSSM	0.984 + 7.9	0.951–1.6	0.967 + 1.8	0.945 + 6.8	0.968 + 5.5
188D + PSSM composition + k-separated-bigrams-PSSM	0.988 + 7.6	0.940–2	0.964 + 1.8	0.928 + 5.8	0.965 + 5.6

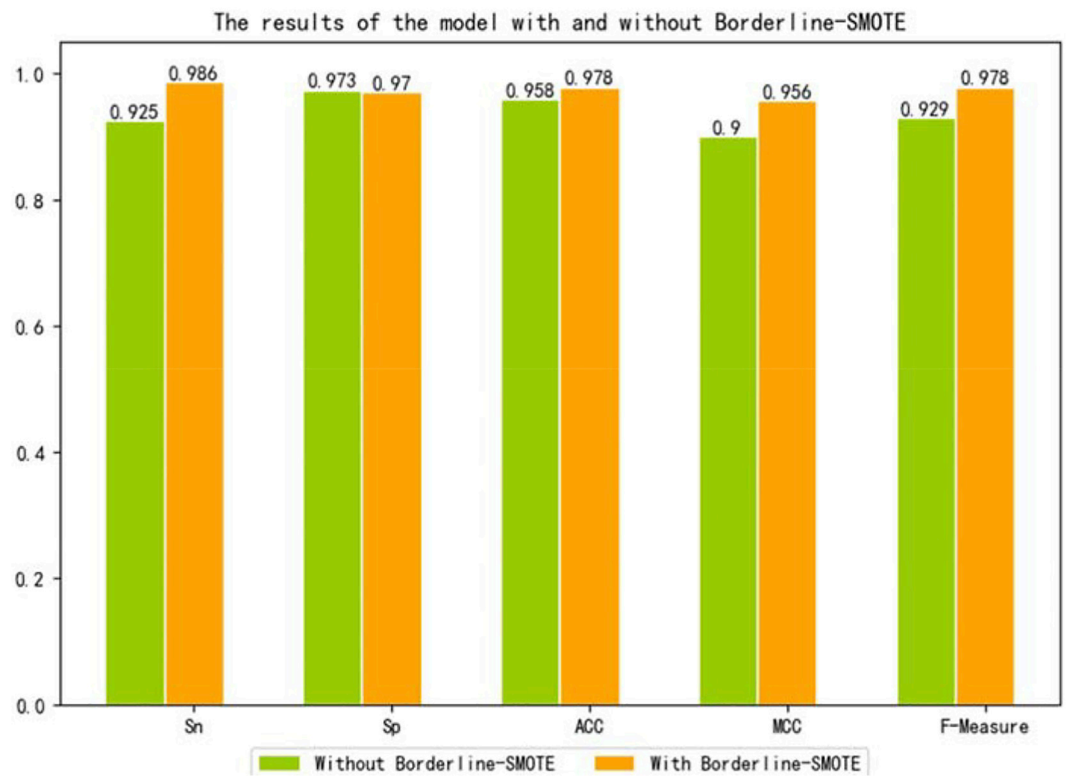


FIGURE 2
Results of the model with or without Borderline-SMOTE on 10-fold cross-validation.

algorithm. We adopted Borderline-SMOTE to augment the positive samples, and finally the number of positive samples is equal to negative samples. To verify that Borderline-SMOTE is effective for our model, we, respectively, conducted experiments using random forest and SGD on the basis of different features with Borderline-SMOTE. Experimental results are shown in Tables 4 and 5. The first number is the experimental result using Borderline-SMOTE, the second number is the percentage of increase or decrease relative to one without Borderline-SMOTE, and the plus sign denotes an increase, while the minus sign denotes a decrease. By comparing Table 4 with Table 2, we can see that the performance of features using Borderline-SMOTE is better than features not using Borderline-SMOTE in all indicators except indicator SP. The same conclusion is also obtained by comparing Table 5 with Table 3. In general, the features of Borderline-SMOTE can improve classification performance.

To further verify that our model can use Borderline-SMOTE to improve the classification performance, that is, Borderline-SMOTE is effective in our model. We compared the performance of our model with Borderline-SMOTE and

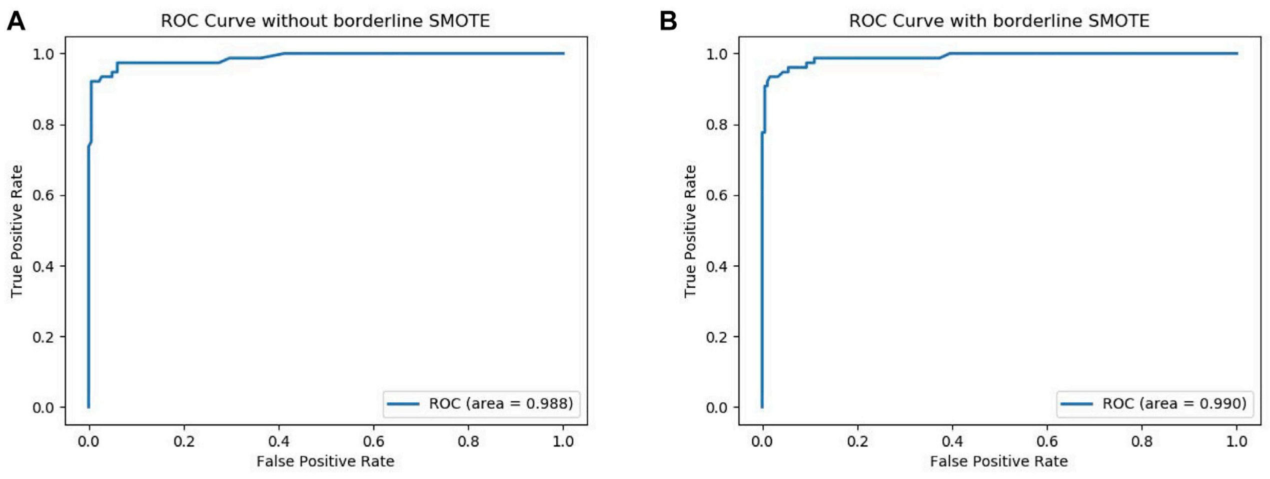


FIGURE 3
ROC curve with or without Borderline-SMOTE. **(A)** ROC curve without Borderline-SMOTE. **(B)** ROC curve with Borderline-SMOTE.

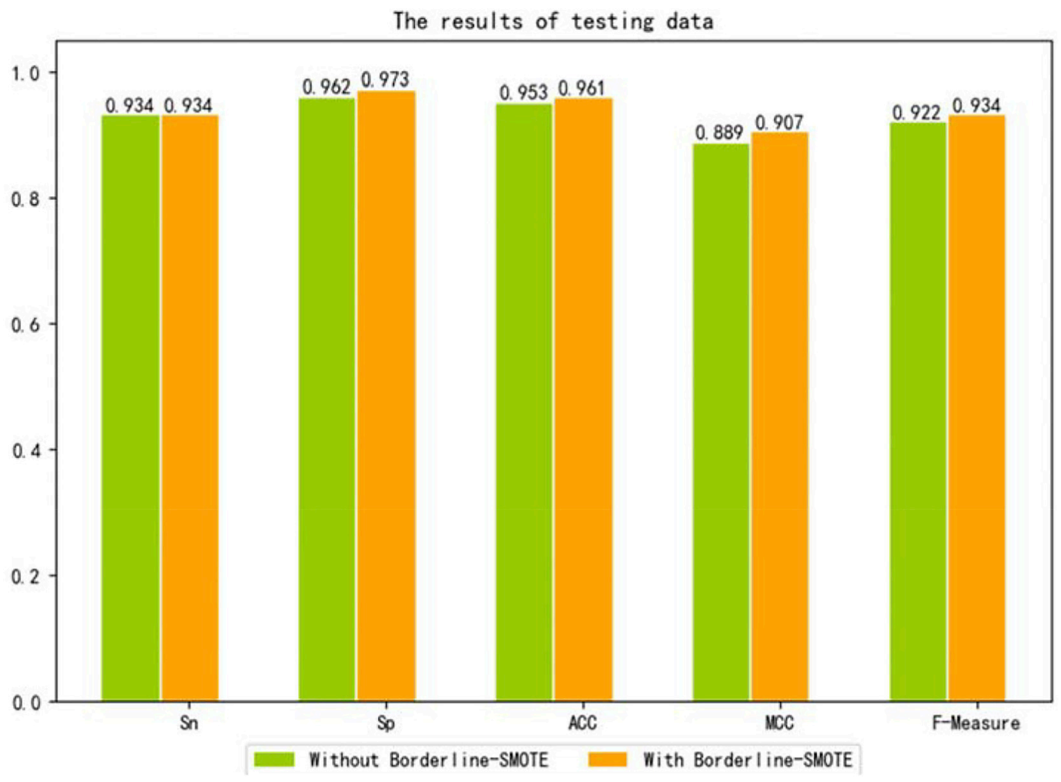


FIGURE 4
Results of the model with or without Borderline-SMOTE on the test dataset.

without Borderline-SMOTE on 10-fold cross-validation, and the experimental result is shown in Figure 2. Except for a slight decrease in SP, all other indicators improved by 2.0–6.1% in Figure 2, especially the indicator SN, which improved to its maximum. The decrease of SP verified that Borderline-SMOTE avoids our model being biased to classifying samples into

TABLE 6 Result of various classifiers using k-separated-bigrams-PSSM feature without Borderline-SMOTE on 10-fold cross-validation.

Classifier	SN	SP	ACC	MCC	F-measure
SVM	0.948	0.944	0.945	0.872	0.911
NB	**0.984**	0.782	0.842	0.703	0.786
SGD	0.964	0.952	0.956	0.897	0.928
RF	0.925	**0.973**	**0.958**	**0.900**	**0.929**

Bold values in the table indicate the best results.

TABLE 7 Result of various classifiers using k-separated-bigrams-PSSM feature with Borderline-SMOTE on 10-fold cross-validation.

Classifier	SN	SP	ACC	MCC	F-measure
SVM	**0.997**	0.877	0.937	0.880	0.940
NB	0.989	0.774	0.881	0.781	0.893
SGD	0.997	0.942	0.970	0.941	0.971
RF	0.986	**0.970**	**0.978**	**0.956**	**0.978**

Bold values in the table indicate the best results.

negative samples. The increase of SN verified that Borderline-SMOTE improves our model's identification ability of positive samples. The improvement of indicators of ACC, MCC, and F-measure verified that Borderline-SMOTE improved our model performance from a general perspective. Furthermore, the ROC curves of our model are plotted in Figure 3, and it can be seen that ISTRF with Borderline-SMOTE is superior to ISTRF without Borderline-SMOTE in the prediction of sucrose transporter protein.

To further evaluate the performance of Borderline-SMOTE in an unseen data set, we conducted experiments on the unseen data set. We used the testing set containing 76 sucrose transporters and 182 non-sucrose transporters to verify the model, and the experimental result is shown in Figure 4. By comparing the two models without and with Borderline-SMOTE, it was found that the latter performs better, which proves once again that Borderline-SMOTE improves the performance of our model.

3.3 Performance of various classifiers

As shown in the frame chart of ISTRF, we adopt random forest as a classifier to train the identification model. To verify that random forest has a better performance than other classifiers, we compared random forest with SVM, NB, and SGD. Table 6 showed the experimental result of 10-fold cross-validation using the k-separated-bigrams-PSSM feature without the Borderline-SMOTE as input. Table 7 showed the experimental result of 10-fold cross-validation using the k-separated-bigrams-PSSM feature with the Borderline-SMOTE as input.

TABLE 8 Experimental result of using different methods.

Model	ACC	MCC	SN	SP
ISTRF	**0.961**	**0.907**	**0.934**	0.973
BioSeq-SVM	0.9457	0.8694	0.9079	0.9615
BioSeq-RF	0.938	0.8505	0.8026	**0.9945**

Bold values in the table indicate the best results.

In Table 6, although the random forest classifier is slightly lower than BN on the SN indicator, it is obviously superior to the other four indicators. According to the number of indicators with the highest value, random forest obtained the four highest values and performs better than the compared classifiers. It is seen in Table 7 that random forest also performs better than the compared classifiers. All in all, random forest is effective in identifying sucrose transporter proteins.

3.4 Comparison with existing methods

To further evaluate the performance of ISTRF, our model is compared with the existing prediction method BioSeq-Analysis (Liu et al., 2019). The online address for this method is http://bioinformatics.hitsz.edu.cn/BioSeq-Analysis/PROTEIN/Kmer/. The SVM and random forest algorithms are used in the BioSeq-Analysis prediction method. We compared them separately. The prediction results are shown in Table 8. It can be seen from Table 8 that our identification model outperforms the compared models on the indicators of ACC, MCC, and SN. It verified that our identification model performs better in general.

4 Conclusion

A large number of experiments have proved that sucrose transporters play an important role in plant growth and crop yield. Therefore, the identification of sucrose transporters has become particularly important. With the rapid development of high-throughput sequencing technology, protein sequences can be easily obtained. In contrast, traditional biochemical technology needs a lot of human, material, and financial resources, and the identification of proteins through bioinformatics methods has become a popular trend. In this study, we introduced k-separated-bigrams-PSSM as the input feature, random forest as the classifier, and the Borderline-SMOTE algorithm to balance the training set. We achieved 0.978 accuracy, 0.986 SN, 0.970 SP, 0.956 MCC, and 0.978 F-measure on the training set. In order to verify the effectiveness of the model, the testing set was used for experiments, and the accuracy was 0.961. In the future, we will continue to find breakthroughs, optimize the experimental model, and strive to obtain better results.

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

Conceptualization, DC and YC; data curation, SL; formal analysis, YC and DC; project administration, DC; writing—original draft, SL; and writing—review and editing, DC and YC.

Funding

This work is supported by the Research Start-up Funding Project of Qu Zhou University (BSYJ202112, BSYJ202109), the National Natural Science Foundation of China (61901103, 61671189), and the Natural Science Foundation of Heilongjiang Province (LH2019F002).

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

References

Al-Sheikh Ahmed, S., Zhang, J., Ma, W., and Dell, B. (2018). Contributions of TaSUTs to grain weight in wheat under drought. *Plant Mol. Biol.* 98 (4), 333–347. doi:10.1007/s11103-018-0782-1

Alballa, M., Aplop, F., and Butler, G. (2020). TranCEP: Predicting the substrate class of transmembrane transport proteins using compositional, evolutionary, and positional information. *PloS one* 15 (1), e0227683. doi:10.1371/journal.pone.0227683

Ali, F., Arif, M., Khan, Z. U., Kabir, M., Ahmed, S., and Yu, D. J. (2020). SDBP-Pred: Prediction of single-stranded and double-stranded DNA-binding proteins by extending consensus sequence and K-segmentation strategies into PSSM. *Anal. Biochem.* 589, 113494. doi:10.1016/j.ab.2019.113494

Aluko, O. O., Li, C., Wang, Q., and Liu, H. (2021). Sucrose utilization for improved crop yields: A review article. *Int. J. Mol. Sci.* 22 (9), 4704. doi:10.3390/ijms22094704

Ao, C., Yu, L., and Zou, Q. (2021). Prediction of bio-sequence modifications and the associations with diseases. *Brief. Funct. Genomics* 20 (1), 1–18. doi:10.1093/bfgp/elaa023

Ao, C., Zhou, W., Gao, L., Dong, B., and Yu, L. (2020). Prediction of antioxidant proteins using hybrid feature representation method and random forest. *Genomics* 112 (6), 4666–4674. doi:10.1016/j.ygeno.2020.08.016

Ao, C., Zou, Q., and Yu, L. (2022). NmRF: Identification of multispecies RNA 2'-O-methylation modification sites from RNA sequences. *Brief. Bioinform.* 23 (1), bbab480. doi:10.1093/bib/bbab480

Aoki, N., Hirose, T., Scofield, G. N., Whitfeld, P. R., and Furbank, R. T. (2003). The sucrose transporter gene family in rice. *Plant Cell. Physiol.* 44 (3), 223–232. doi:10.1093/pcp/pcg030

Basith, S., Manavalan, B., Hwan Shin, T., and Lee, G. (2020). Machine intelligence in peptide therapeutics: A next-generation tool for rapid disease screening. *Med. Res. Rev.* 40 (4), 1276–1314. doi:10.1002/med.21658

Bottou, L. (2010). "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, Hamburg (Physica-Verlag HD), 177–186.

Breiman, L. (2001). Random forests. *Mach. Learn.* 45 (1), 5–32. doi:10.1023/a:1010933404324

Cai, C. Z., Han, L. Y., Ji, Z. L., Chen, X., and Chen, Y. Z. (2003). SVM-prot: Web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic Acids Res.* 31 (13), 3692–3697. doi:10.1093/nar/gkg600

Cao, J., Panetta, R., Yue, S., Steyaert, A., Young-Bellido, M., and Ahmad, S. (2003). A naive Bayes model to predict coupling between seven transmembrane domain receptors and G-proteins. *Bioinformatics* 19 (2), 234–240. doi:10.1093/bioinformatics/19.2.234

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* 16, 321–357. doi:10.1613/jair.953

Chen, Y., Chang, R., and Guo, J. (2021). Effects of data augmentation method borderline-SMOTE on emotion recognition of EEG signals based on convolutional neural network. *IEEE Access* 9, 47491–47502. doi:10.1109/access.2021.3068316

Consortium, UniProt (2019). UniProt: A worldwide hub of protein knowledge. *Nucleic Acids Res.* 47 (D1), D506–D515. doi:10.1093/nar/gky1049

Ding, S., Yan, S., Qi, S., Li, Y., and Yao, Y. (2014). A protein structural classes prediction method based on PSI-BLAST profile. *J. Theor. Biol.* 353, 19–23. doi:10.1016/j.jtbi.2014.02.034

Endler, A., Meyer, S., Schelbert, S., Schneider, T., Weschke, W., Peters, S. W., et al. (2006). Identification of a vacuolar sucrose transporter in barley and Arabidopsis mesophyll cells by a tonoplast proteomic approach. *Plant Physiol.* 141 (1), 196–207. doi:10.1104/pp.106.079533

Fu, L., Niu, B., Zhu, Z., Wu, S., and Li, W. (2012). CD-HIT: Accelerated for clustering the next-generation sequencing data. *Bioinformatics* 28 (23), 3150–3152. doi:10.1093/bioinformatics/bts565

Garner, S. R. (1995). Weka: The waikato environment for knowledge analysis. *Proc. N. Z. Comput. Sci. Res. students Conf.* 1995, 57–64.

Han, H., Wang, W. Y., and Mao, B. H. (2005). "Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning," in *International conference on intelligent computing* (Berlin, Heidelberg: Springer), 878–887.

Ho, Q. T., Phan, D. V., and Ou, Y. Y. (2019). Using word embedding technique to efficiently represent protein sequences for identifying substrate specificities of transporters. *Anal. Biochem.* 577, 73–81. doi:10.1016/j.ab.2019.04.011

Holmes, G., Donkin, A., and Witten, I. H. (1994). "Weka: A machine learning workbench," in Proceedings of ANZIIS'94-Australian New Zealnd Intelligent Information Systems Conference, Brisbane, QLD, Australia, 29 November 1994 - 02 December 1994, 357–361.

Kühn, C., Barker, L., Bürkle, L., and Frommer, W. B. (1999). Update on sucrose transport in higher plants. *J. Exp. Bot.* 50, 935–953. doi:10.1093/jexbot/50.suppl_1.935

Lee, Y. W., Choi, J. W., and Shin, E. H. (2021). Machine learning model for predicting malaria using clinical information. *Comput. Biol. Med.* 129, 104151. doi:10.1016/j.compbiomed.2020.104151

Liu, B., Gao, X., and Zhang, H. (2019). BioSeq-Analysis2. 0: An updated platform for analyzing DNA, RNA and protein sequences at sequence level and residue level based on machine learning approaches. *Nucleic Acids Res.* 47 (20), e127. doi:10.1093/nar/gkz740

Lv, H., Dao, F. Y., Zhang, D., Guan, Z. X., Yang, H., Su, W., et al. (2020). iDNA-MS: an integrated computational tool for detecting DNA modification sites in multiple genomes. *Iscience* 23 (4), 100991. doi:10.1016/j.isci.2020.100991

Lv, Z., Jin, S., Ding, H., and Zou, Q. (2019). A random forest sub-Golgi protein classifier optimized via dipeptide and amino acid composition features. *Front. Bioeng. Biotechnol.* 7, 215. doi:10.3389/fbioe.2019.00215

Manavalan, B., Shin, T. H., and Lee, G. (2018). DHSpred: Support-vector-machine-based human DNase I hypersensitive sites prediction using the optimal features selected by random forest. *Oncotarget* 9, 1944–1956. doi:10.18632/oncotarget.23099

Mangukia, N., Rao, P., Patel, K., Pandya, H., and Rawal, R. M. (2021). Identifying potential human and medicinal plant microRNAs against SARS-CoV-2 3' utr region: A computational genomics assessment. *Comput. Biol. Med.* 136, 104662. doi:10.1016/j.compbiomed.2021.104662

Masoudi-Sobhanzadeh, Y., Jafari, B., Parvizpour, S., Pourseif, M. M., and Omidi, Y. (2021). A novel multi-objective metaheuristic algorithm for protein-peptide docking and benchmarking on the LEADS-PEP dataset. *Comput. Biol. Med.* 138, 104896. doi:10.1016/j.compbiomed.2021.104896

Mishra, N. K., Chang, J., and Zhao, P. X. (2014). Prediction of membrane transport proteins and their substrate specificities using primary sequence information. *PloS one* 9 (6), e100278. doi:10.1371/journal.pone.0100278

Mistry, J., Chuguransky, S., Williams, L., Qureshi, M., Salazar, G. A., Sonnhammer, E. L., et al. (2021). Pfam: The protein families database in 2021. *Nucleic Acids Res.* 49 (D1), D412–D419. doi:10.1093/nar/gkaa913

Petry, D., de Godoy Marques, C. M., and Marques, J. L. B. (2020). Baroreflex sensitivity with different lags and random forests for staging cardiovascular autonomic neuropathy in subjects with diabetes. *Comput. Biol. Med.* 127, 104098. doi:10.1016/j.compbiomed.2020.104098

Platt, J. (1998). *Sequential minimal optimization: A fast algorithm for training support vector machines*. Redmond, Washington, United States: Microsoft Research.

Qian, L., Jiang, Y., Xuan, Y. Y., Yuan, C., and SiQiao, T. (2021). PsePSSM-based prediction for the protein-ATP binding sites. *Curr. Bioinform.* 16 (4), 576–582. doi:10.2174/1574893615999200918183543

Ru, X., Li, L., and Zou, Q. (2019). Incorporating distance-based top-n-gram and random forest to identify electron transport proteins. *J. Proteome Res.* 18 (7), 2931–2939. doi:10.1021/acs.jproteome.9b00250

Shabbir, S., Asif, M. S., Alam, T. M., and Ramzan, Z. (2021). Early prediction of malignant mesothelioma: An approach towards non-invasive method. *Curr. Bioinform.* 16 (10), 1257–1277. doi:10.2174/1574893616666210616121023

Shah, S. M. A., Taju, S. W., Ho, Q. T., and Ou, Y. Y. (2021). GT-Finder: Classify the family of glucose transporters with pre-trained BERT language models. *Comput. Biol. Med.* 131, 104259. doi:10.1016/j.compbiomed.2021.104259

Tran, T. M., Hampton, C. S., Brossard, T. W., Harmata, M., Robertson, J. D., Jurisson, S. S., et al. (2017). *In vivo* transport of three radioactive [18F]-fluorinated deoxysucrose analogs by the maize sucrose transporter ZmSUT1. *Plant Physiol. biochem.* 115, 1–11. doi:10.1016/j.plaphy.2017.03.006

Vapnik, V. (2006). *Estimation of dependences based on empirical data*. Berlin, Heidelberg: Springer Science Business Media.

Wang, C., Li, J., Zhang, Y., and Guo, M. (2020). Identification of Type VI effector proteins using a novel ensemble classifier. *IEEE Access* 8, 75085–75093. doi:10.1109/access.2020.2985111

Wang, J., Yang, B., Revote, J., Leier, A., Marquez-Lago, T. T., Webb, G., et al. (2017). Possum: A bioinformatics toolkit for generating numerical sequence feature descriptors based on PSSM profiles. *Bioinformatics* 33 (17), 2756–2758. doi:10.1093/bioinformatics/btx302

Wang, S., Yang, J., Xie, X., Li, F., Wu, M., Lin, F., et al. (2019). Genome-wide identification, phylogeny, and expression profile of the sucrose transporter multigene family in tobacco. *Can. J. Plant Sci.* 99 (3), 312–323. doi:10.1139/cjps-2018-0187

Wang, Y. B., You, Z. H., Li, L. P., Huang, D. S., Zhou, F. F., and Yang, S. (2018). Improving prediction of self-interacting proteins using stacked sparse auto-encoder with PSSM profiles. *Int. J. Biol. Sci.* 14 (8), 983–991. doi:10.7150/ijbs.23817

Xiong, Y., Wang, Q., Yang, J., Zhu, X., and Wei, D. Q. (2018). PredT4SE-stack: Prediction of bacterial type IV secreted effectors from protein sequences using a stacked ensemble method. *Front. Microbiol.* 9, 2571. doi:10.3389/fmicb.2018.02571

Yan, Z. X., Yang, H. Y., Zhang, C. H., Wu, W. L., and Li, W. L. (2021). Functional analysis of the blackberry sucrose transporter gene RuSUT2. *Russ. J. Plant Physiol.* 68 (2), 246–253. doi:10.1134/s1021443721020217

Yang, L., and Jiao, X. (2021). Distinguishing enzymes and non-enzymes based on structural information with an alignment free approach. *Curr. Bioinform.* 16 (1), 44–52. doi:10.2174/1574893615666200324134037

Zeng, R., Lu, Y., Long, S., Wang, C., and Bai, J. (2021). Cardiotocography signal abnormality classification using time-frequency features and Ensemble Cost-sensitive SVM classifier. *Comput. Biol. Med.* 130, 104218. doi:10.1016/j.compbiomed.2021.104218

Zhang, H., Zhang, Q., Jiang, W., and Lun, X. (2021). Clinical significance of the long non-coding RNA NEAT1/miR-129-5p axis in the diagnosis and prognosis for patients with chronic heart failure. *Exp. Ther. Med.* 16 (4), 512–523. doi:10.3892/etm.2021.9943

Zhang, L., Huang, Z., and Kong, L. (2021). CSBPI_Site: Multi-information sources of features to RNA binding sites prediction. *Curr. Bioinform.* 16 (5), 691–699. doi:10.2174/1574893615666210108093950

Zhang, M., Li, F., Marquez-Lago, T. T., Leier, A., Fan, C., Kwoh, C. K., et al. (2019). MULTiPly: A novel multi-layer predictor for discovering general and specific types of promoters. *Bioinformatics* 35 (17), 2957–2965. doi:10.1093/bioinformatics/btz016

Zou, Q., Xing, P., Wei, L., and Liu, B. (2019). Gene2vec: Gene subsequence embedding for prediction of mammalian N6-methyladenosine sites from mRNA. *Rna* 25 (2), 205–218. doi:10.1261/rna.069112.118

TYPE Original Research
PUBLISHED 18 October 2022
DOI 10.3389/fgene.2022.1024920

Check for updates

OPEN ACCESS

EDITED BY
Zhibin Lv,
Sichuan university, China

REVIEWED BY
Chen P. L,
Huazhong Agricultural University, China
Junxiao Chen,
Hubei Academy of Agricultural Sciences, China
Wei Han,
Hangzhou Medical College, China

*CORRESPONDENCE
Lei Xu,
csleixu@szpt.edu.cn

†These authors have contributed equally to this work

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 22 August 2022
ACCEPTED 20 September 2022
PUBLISHED 18 October 2022

CITATION
Zhao D, Wang L, Chen Z, Zhang L and Xu L (2022), KRAS is a prognostic biomarker associated with diagnosis and treatment in multiple cancers. *Front. Genet.* 13:1024920.
doi: 10.3389/fgene.2022.1024920

KRAS is a prognostic biomarker associated with diagnosis and treatment in multiple cancers

Da Zhao[1,2†], Lizhuang Wang[3†], Zheng Chen[1,2], Lijun Zhang[2] and Lei Xu[4*]

[1]Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, China, [2]School of food and drug, Shenzhen Polytechnic, Shenzhen, China, [3]Beidahuang Industry Group General Hospital, Harbin, China, [4]School of Electronic and Communication Engineering, Shenzhen Polytechnic, Shenzhen, China

KRAS encodes K-Ras proteins, which take part in the MAPK pathway. The expression level of KRAS is high in tumor patients. Our study compared *KRAS* expression levels between 33 kinds of tumor tissues. Additionally, we studied the association of *KRAS* expression levels with diagnostic and prognostic values, clinicopathological features, and tumor immunity. We established 22 immune-infiltrating cell expression datasets to calculate immune and stromal scores to evaluate the tumor microenvironment. *KRAS* genes, immune check-point genes and interacting genes were selected to construct the PPI network. We selected 79 immune checkpoint genes and interacting related genes to calculate the correlation. Based on the 33 tumor expression datasets, we conducted GSEA (genome set enrichment analysis) to show the *KRAS* and other co-expressed genes associated with cancers. *KRAS* may be a reliable prognostic biomarker in the diagnosis of cancer patients and has the potential to be included in cancer-targeted drugs.

KEYWORDS
KRAS, tumor stage, cell infiltration, functional enrichment analysis, tumor microenvironment, prognostic biomarker

1 Introduction

Cancer is a severe life-threatening disease that affects a large number of patients worldwide (Cao et al., 2021). Breast cancer has become the most common cancer in new possibilities, followed by prostate cancer, lung cancer, *etc.* (Wang et al., 2021). *KRAS* was first recognized in the Kirsten rat sarcoma virus, which encodes the p21 protein to induce virus transformation (Scolnick et al., 1979). The proteins of *KRAS* are located in the cell membranes, and on its C-terminus, there is an isoprene group (Welman et al., 2000). *KRAS* protein encodes the GTPase enzyme, which is a part of the MAPK pathway and acts as a switch on/off of transformation between GTP (guanosine triphosphate) and GDP molecules (Tsuchida et al., 1982). In mammalian cells, *KRAS* has two kinds of protein products, K-Ras4A and K-Ras4B, which are encoded by alternative exon4 (Welman et al., 2000). Many studies have observed that *KRAS* effectors affect the interactions of cells and their extracellular environment, which could also regulate cell growth, cell motility and cell metabolism (Tang et al., 2018; Gu et al., 2021; Hu et al., 1990; Hu et al., 2020; Yu et al., 2018).

KRAS is a signal transducer protein that binds to GTP in the MAPK pathway; the mutation of *KRAS* has been discovered in a quarter of human cancers (Pantsar, 2020). Based on the data in COSMIC, missense mutations of *KRAS* frequently occur in pancreatic, colorectal and lung cancers (Forbes et al., 2011; Ao et al., 2021; Li et al., 2021; Luo et al., 2021; Yu et al., 2021). Homozygous deletions of *KRAS* are the most frequent genetic alteration in pancreatic epithelial adenocarcinoma (PDAC), which results in cell metastasis and the transformation of cancer tumors (Chang et al., 2014). The proto-oncogenes are closely related to multiple cancers, such as cardio-facio-cutaneous syndrome (Niihori et al., 2006), ductal carcinoma of the pancreas (Hartman et al., 2012), leukemias (Singh et al., 2021), mucinous adenoma (Hartman et al., 2012), and noonan syndrome (Ando et al., 2021). The sequences of mutations in *KRAS* affect the function of genes, oncogenes, tumor-suppressor genes and stability genes and are the critical element for tumorigenesis (Vogelstein and Kinzler, 2004). *KRAS* mutations have been well characterized in 30%–50% of colorectal cancers (Andreyev et al., 2001). *KRAS* proteins are activated when transmembrane receptors are present, which include serine/threonine kinase, GTP enzyme activating protein (GAP), phosphatidylinositol 3-kinase (PI3K) and GEF (Shields et al., 2000). The abnormal activation of GTP binding of mutated *KRAS* protein leads to the unregulated growth of downstream cells (Arrington et al., 2012). Mutations at codon 12 and position 2 (GGT-GAT) of *KRAS* appear to be most common in colorectal cancer (Capella et al., 1991). The wild-type allele of KRAS is a suppressor in mouse lung cancer (Westcott et al., 2015).

Mutation sites of the *KRAS* gene have to be considered to be an effective way to develop new cancer treatment schemes (Hu et al., 2022a). Efforts to utilize KRAS and related genes as targets to explore drugs for cancer have been undertaken for years, and inhibitor drugs to block KARS^{G12C} have been developed (Xu et al., 2022). RAF1 could be an essential target to block KRAS mutant cancers (Drosten and Barbacid, 2020). EFGR-inhibiting drugs suppress the *KRAS* expression level in A549 lung cancer cells to inhibit cell proliferation (Zarredar et al., 2019).

Possibility, *KRAS* may be considered a genetic diagnosis potential biomarker of multiple malignant neoplastic diseases. First, we compared the survival data of patients with 33 kinds of tumors. Second, we analyzed the *KRAS* expression level with the characteristics of tumor patients, tumor stage, tumor microenvironment, and immune cell infiltration of 33 kinds of tumors; finally, we investigated the molecular mechanisms by GO and KEGG analysis. This study explores the molecular relationships between *KRAS* genes and cancer, which is crucial for developing new biomarkers and effective prevention and treatment of tumors.

2 Materials and methods

2.1 Data sets collection and process

We obtained 33 kinds of tumors sequencing datasets(10,327 tumor samples and 730 normal samples) somatic mutation and survival data from the UCSC Xena database (http://xena.ucsc.edu/). Subsequently, we used R language to convert the gene ID, extract the transcription data and analyze the differential expression of genes, analyze and draw, and the packages involved were R (version 4.2.1), BiomaRt (version 2.52.0, Functional Annotation Retrieval), dplyr (version 2.52.0, Data manipulation) and ggpubr (version 0.4.0, Data visualization).

2.2 Correlation analysis between *KRAS* expression and prognosis of tumor patients

In this experiment, we used existing data to investigate the survival status information from the UCSC Xena database, which contained 10,327 tumor samples and 730 normal samples. We gathered the survival status, which included disease-specific survival (DSS), disease-free interval (DFI) and progression-free interval (PFI) status data and time information for prognostic analysis. We divided the differentially expressed *KRAS* data into high-value and low-value groups. Furthermore, the prognostic value in 33 tumors was calculated by the Kaplan–Meier survival estimate method. The R packages utilized for this analysis were limma (version 3.9, Analyzing microarray and RNA-seq data), survival (version 3.3-1, Survival analysis), survminer (version 0.4.9, Drawing survival curves) and forestplot (version 2.0.1, Advanced Forest plot using 'grid' graphics).

2.3 Correlation analysis of *KRAS* gene expression and tumor stage, tumor mutation burden and microsatellite instability

Stage information containing the diagnostic and prognostic value of cancers was downloaded from the UCSC Xena database(http://xena.ucsc.edu/); approximately 8,099 tumor samples were divided into 3-4 stages. The limma and ggpubr packages of R were introduced to calculate the *KRAS* expression quantity and show the relevance between *KRAS* genes and tumor stage. Approximately 11,057 samples combined with *KRAS* expression data were used for TMB analysis by the Spearman correlation test. The fmsb(version 0.7.3, Medical and Health Data Analysis) package was used to create a correlation radar plot. We calculated the microsatellite instability (MSI) scores combined with the *KRAS* expression data by the Spearman correlation test, and a rader plot between tumors and *KRAS* genes was created. The fmsb package was used to create a correlation radar plot.

2.4 Correlation analysis of *KRAS* gene expression and the tumor microenvironment

The tumor microenvironment (TME) is considered to be the cells, tissues and matrix around a tumor; immune cells and stromal cells are considered diagnostic indicators of cancer development. We utilized ESTIMATE for predicting tumor purity by calculating stromal and immune cell infiltration (Yoshihara et al., 2013). Approximately 11,057 samples from 33 tumors were used to construct expression matrix data of *KRAS* genes. We calculated correlation between TME, MSI and expression data, tested by Spearman test methods.

2.5 Correlation analysis of *KRAS* gene expression and immune cell infiltration

We conducted CIBERSORT (https://cibersortstanfordedu/) to estimate the infiltration percentage of 22 immune cell types in tumors (Newman et al., 2015). According to the tumor expression data matrix file, we estimated the immune scores of all the tumor samples. Filtering through data from tumors, we approximate the correlation between cell infiltration levels and gene expression levels by Spearman's correlation test.

2.6 Protein–protein interaction networks, correlation of *KRAS* with marker genes and analysis of gene enrichment

The *KRAS* genes interacted with other tumor-associated genes may help us understand tumorigenesis molecular mechanisms. We selected 33 *KRAS* coexpressed genes to construct protein–protein interaction (PPI) networks on the STRING database (https://www.string-db.org/) (Szklarczyk et al., 2019; Yu et al., 2020). We selected 79 immune checkpoint genes and interacting related genes to calculate the correlation. Based on the 33 tumor expression datasets, we conducted GSEA (genome set enrichment analysis) to show the *KRAS* and other coexpressed genes associated with cancers. This research utilized the KEGG database (https://www.kegg.jp/) and the GO database to enrich the genes (FDR <0.5).

3 Results

3.1 Transcriptional data of *KRAS* in 33 pancancer

We obtained the transcription datasets, somatic mutations of 33 cancer tumors from TCGA, and the survival data from the UCSC Xena database. The transcription data contained 11,057 samples which included tumor samples and normal samples. When we compared the differences between normal and tumor tissues, in most tumor tissues the expression levels of *KRAS* were higher than normal tissues. A total of 12 kinds of tumor tissues has significant difference in *KRAS* expression levels, of which 9 tumor tissues BRCA (Breast invasive carcinoma), CHOL (Cholangiocarcinoma), COAD (Colon adenocarcinoma), KIRC (Kidney renal clear cell carcinoma), LUAD (Lung adenocarcinoma), LUSC (Lung squamous cell carcinoma), READ (Rectum adenocarcinoma), STAD (Stomach adenocarcinoma), and UCEC (Uterine Corpus Endometrial Carcinoma) had an extremely significant difference; meanwhile, 3 tumor tissues (GBM (Glioblastoma multiforme), LIHC(Liver hepatocellular carcinoma), and THCA(Thyroid carcinoma) had a significant difference (Figure 1). These results indicated that cancer could lead to the abnormal expression of *KRAS*.

3.2 The prognostic value of the *KRAS* gene across cancers

We analyzed the prognostic value of each tumor sample according to the *KRAS* expression levels with different tumors. High *KRAS* expression levels were associated with overall survival (OS) in ACC, LUAD, PAAD and UCEC (Figure 2A); poor disease-specific survival (DSS) in ACC, LUAD and PAAD (Figure 2B); poor disease-free interval (DFI) in ACC, LUAD, PAAD and STAD (Figure 2C); and poor progression-free interval (PFI) in adrenocortical carcinoma (ACC), LUAD, PAAD, STAD and uveal melanoma (UVM) (Figure 2D).

To evaluate how effective *KRAS* is in single cancer, we calculated *p* values to analyze the association of clinical data with single cancers. High *KRAS* expression levels correlated with OS in ACC, ESCA, KIRC, PAAD and THYM of 0.01, 0.01, 0.001, 0.022, 0.035 and 0.005, respectively (Figures 3E–J); correlated with poor DSS in ACC, ESCA, KIRC, KIPP, MESO and PAAD by 0.002, 0.036, 0.004, 0.010, 0.038 and 0.039, respectively (Figures 2K–P); correlated with poor DFI in ACC, LGG, LUAD, OV and PAAD by 0.026, 0.005, 0.004 m 0.046 and 0.001, respectively (Figures 2Q–U); and correlated with poor PFI in ACC, CESC, KIRC, KIPP, PAAD and SARC by 0.005, 0.028, 0.001, 0.007, 0.020 and 0.042, respectively (Figure 2 V-AA).

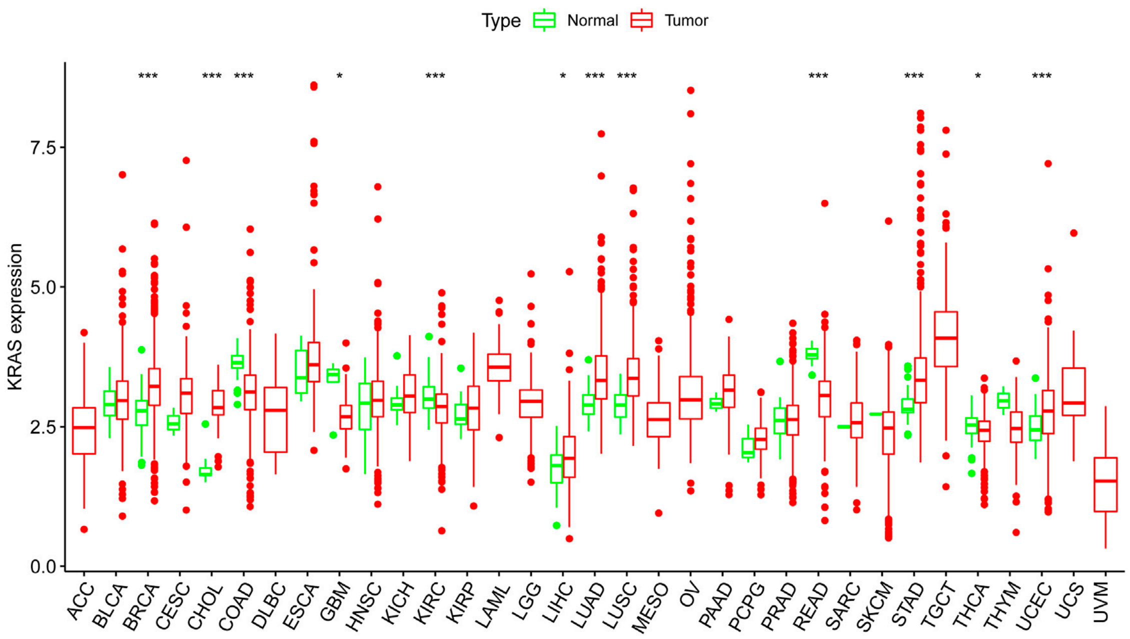


FIGURE 1
The *KRAS* expression level in cancer patients. Thirty-three kinds of tumor cancers. Expression of *KRAS* in tumors were performed by ggpubr (version 4.0.4) package of R (***$p < 0.001$, **$p < 0.01$, *$p < 0.05$).

3.3 The relationships between *KRAS* expression levels with clinicopathological features

We explored the connection between the clinicopathological characteristics and various tumor stages according to *KRAS* expression levels in different pathology grades. These data suggest that high *KRAS* expression levels had effects at severe stages; correlation analysis showed in ACC, COAD, ESCA, KIRC, KIRP, LIHC, MESO, SKCM and STAD (Figures 3A–I). In addition, high *KRAS* expression levels showed a correlation with TMB in 14 tumors: BLCA, COAD, HNSC, LIHC, LUAD, LUSC, LUSC, PAAD, OV, PRAD, SARC, STAD, THYM, UCEC and UVM (Figure 3J); high *KRAS* expression levels showed a correlation with MSI in 8 tumors: BRCA, COAD, DLBC, READ, SKCM, STAD, TGCT and UCEC (Figure 3K).

3.4 The relationships between *KRAS* expression levels and tumor microenvironment

A total of 22 immune-infiltrating cells were introduced to analyze the tumor microenvironment by 33 tumor expression data. Looking at Figure 4A, it is apparent that the immune scores of ACC, CESC, GBM, HNSC, KIRC, KIRP, LGG, LUAD, LUSC, TGCT and UCEC were negatively associated with *KRAS* expression; from Figure 4B above we can see that the stromal scores in THCA showed a causal negative relationship with *KRAS* expression levels, and GBM, LGG, LUSC, TGCT and UECE showed a causal negative relationship with *KRAS* expression levels.

3.5 The associations of *KRAS* expression levels with tumor immune cell infiltration levels

We established 22 cell expression datasets based on the immune-infiltrating levels to explore the potential effect of *KRAS* expression levels on the tumor immune cell infiltration. The results of the correlational analysis are shown in Figure 5; we could demonstrate that higher expression levels of *KRAS* were positively correlated with the infiltration levels of follicular helper T cells, activated memory CD4 T cells, activated dendritic cells, activated NK cells, naïve B cells, resting plasma cells, resting mast cells, resting CD4 T cells and follicular helper T cells, which were negatively correlated with the infiltration levels of activated NK cells, plasma cells, CD8 T cells, regulatory T cells (Tregs), M2 macrophages, M0 macrophages, memory B cells, activated memory T cells, memory CD4 T cells, activated NK cells, CD8 T cells, neutrophils, and monocytes in tumor cancers. High *KRAS* expression levels were positively correlated with the infiltration level of follicular helper T cells in BLCA. High *KRAS*

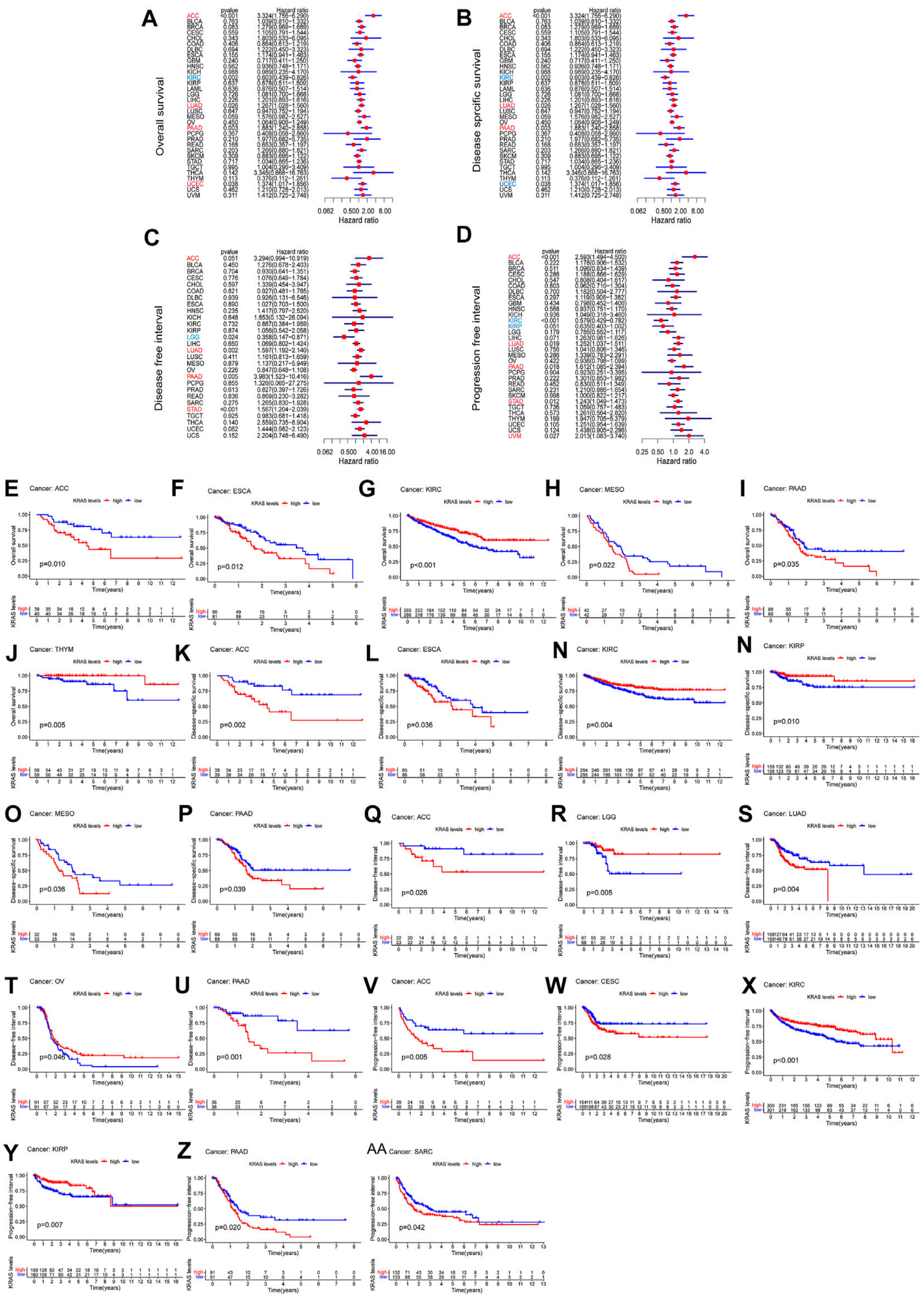


FIGURE 2
The relationship between the expression of *KRAS* and the prognosis of patients with 33 kinds of cancers in the TCGA database. **(A,E–J)** OS (overall survival) **(B,K–P)** DSS (disease-specific survival) **(C,Q–U)** DFI (disease-free interval) **(D,V–AA)** PFI (progression-free interval).

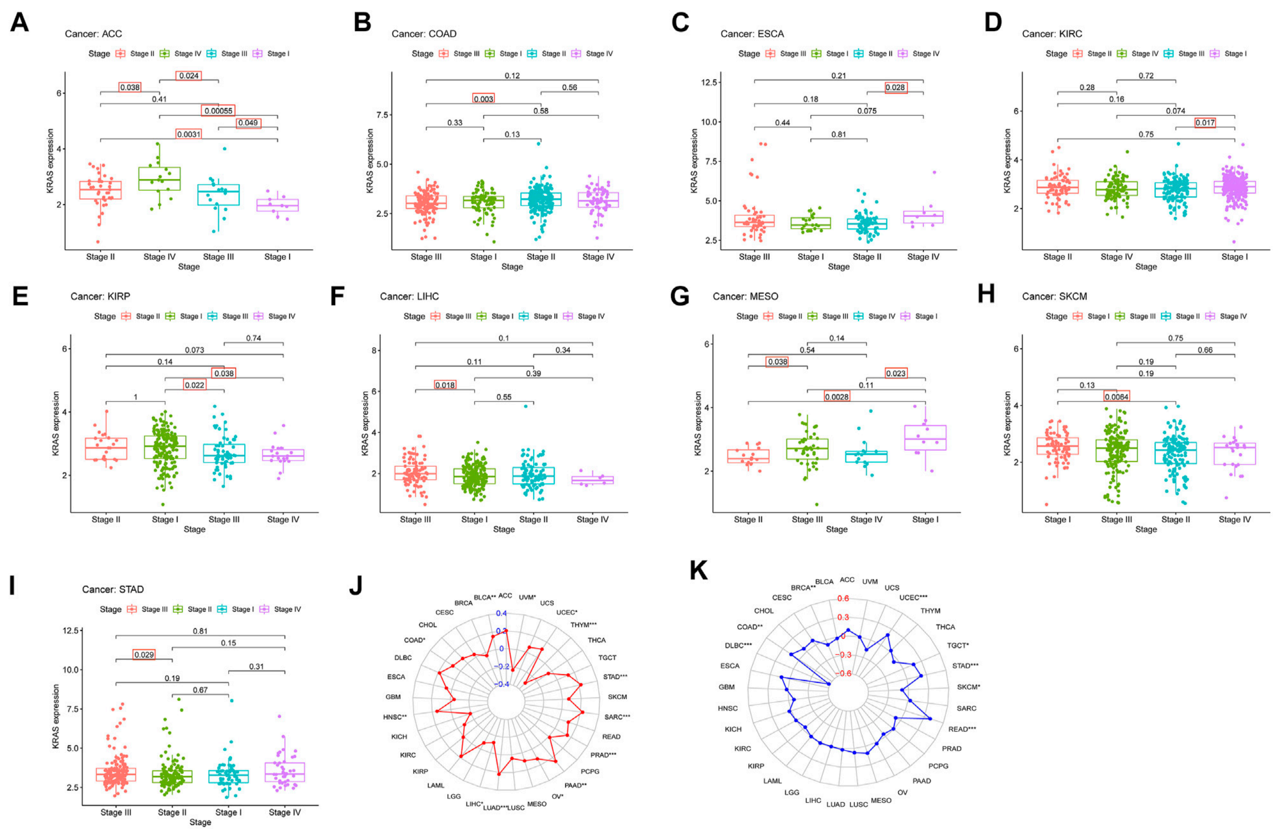


FIGURE 3
The relationship between *KRAS* expression level and pathological characteristics of tumor patients. (tumor mutation burden, microsatellite instabilitiy). **(A–I)** Relationship of *KRAS* expression level and stage grade. **(J)** Tumor mutation burden (TMB). **(K)** Microsatellite instability (MSI). $*p < 0.05$. $**p < 0.01$, $***p < 0.001$.

expression levels were positively correlated with the infiltration level of resting CD4 T cells in BRCA. High *KRAS* expression levels were negatively correlated with the infiltration levels of activated NK cells, plasma cells, CD8 T cells and regulatory T cells (Tregs) in BRCA. High *KRAS* expression levels were positively correlated with the infiltration level of resting CD4 T cells in CESC. High *KRAS* expression levels were negatively correlated with the infiltration levels of M2 macrophages in CESC. High *KRAS* expression levels were positively correlated with the infiltration level of activated dendritic cells and resting CD4 T cells in COAD. High *KRAS* expression levels were negatively correlated with the infiltration levels of M0 macrophages in COAD. High *KRAS* expression levels were positively correlated with the infiltration level of activated memory CD4 T cells in DLBC. High *KRAS* expression levels were negatively correlated with the infiltration levels of memory B cells in ESCA. High *KRAS* expression levels were negatively correlated with the infiltration levels of activated memory CD4 T cells in GBM. High *KRAS* expression levels were positively correlated with the infiltration level of naïve B cells, plasma cells and memory CD4 T cells resting in HNSC. High *KRAS* expression levels were negatively correlated with the infiltration levels of activated NK cells and CD8 T cells in HNSC. High *KRAS* expression levels were positively correlated with the infiltration level of M2 macrophages, resting memory CD4 T cells, resting MAST cells and neutrophils in KIRC. High KRAS expression levels were negatively correlated with the infiltration levels of regulatory T cells (Tregs), CD8 T cells and plasma cells in KIRC. High KRAS expression levels were positively correlated with the infiltration level of resting mast cells and resting memory CD4 T cells in KIRP. High *KRAS* expression levels were negatively correlated with the infiltration levels of regulatory T cells (Tregs) in KIRP. High *KRAS* expression levels were positively correlated with the infiltration level of memory resting memory CD4 T cells in LAML. High KRAS expression levels were positively correlated with the infiltration levels of memory-activated CD4 T cells in LUAD. High KRAS expression levels were positively clinically relevant to the infiltration levels of memory B cells in LUSC. High KRAS expression levels were negatively clinically relevant to the infiltration levels of neutrophils in LUSC. High KRAS expression levels were negatively correlated with the infiltration levels of activated NK cells in OV. High *KRAS* expression levels were positively correlated with the infiltration

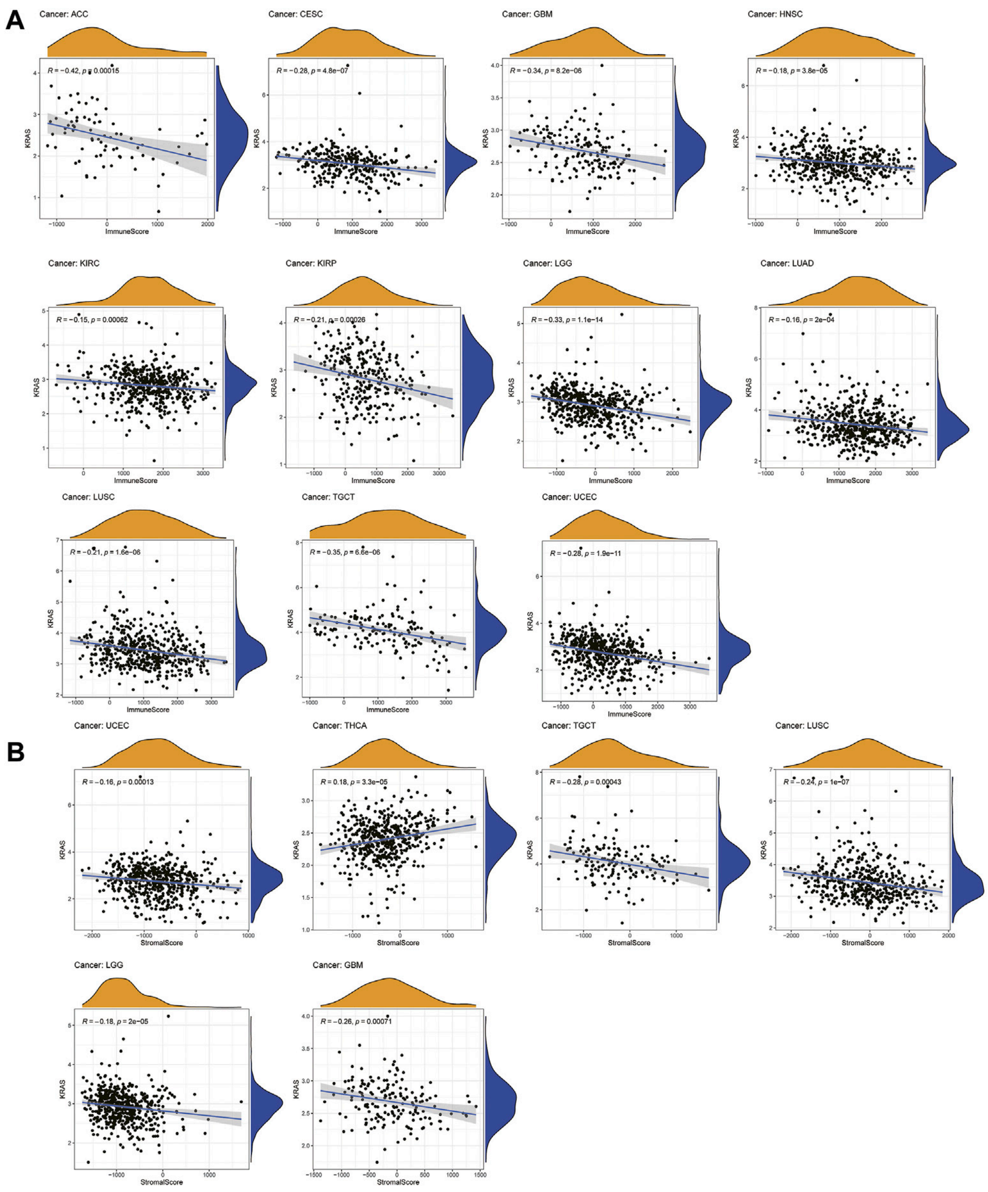


FIGURE 4
Relationships between *KRAS* expression and tumor microenvironment. **(A)** Immune score, **(B)** Stromal score.

levels of memory CD4 T cells resting in PAAD. High *KRAS* expression levels were negatively correlated with the infiltration levels of plasma cells in PAAD. High *KRAS* expression levels were positively correlated with the infiltration levels of naïve B cells, M1 macrophages and resting memory CD4 T cells in PRAD. High KRAS expression levels were negatively correlated with the infiltration levels of memory B cells, activated NK cells and CD8 T cells in PRAD. High KRAS expression levels were negatively correlated with the infiltration levels of dendritic cells resting in READ. High KRAS expression levels were negatively correlated with the infiltration levels of resting mast cells in SARC. High *KRAS* expression levels were positively correlated with the infiltration levels of activated memory CD4 T cells and resting CD4 T cells in SKCM. High *KRAS* expression levels were negatively correlated with the infiltration levels of regulatory T cells (Tregs) in SKCM. High *KRAS* expression levels were negatively correlated with the infiltration levels of Mococytes in STAD. High KRAS expression levels were positively correlated with the infiltration levels of naïve B cells and memory CD4 T cells activated in THCA. High *KRAS* expression levels were negatively correlated with the infiltration levels of M2 macrophages and NK cells activated in THCA. High *KRAS* expression levels were

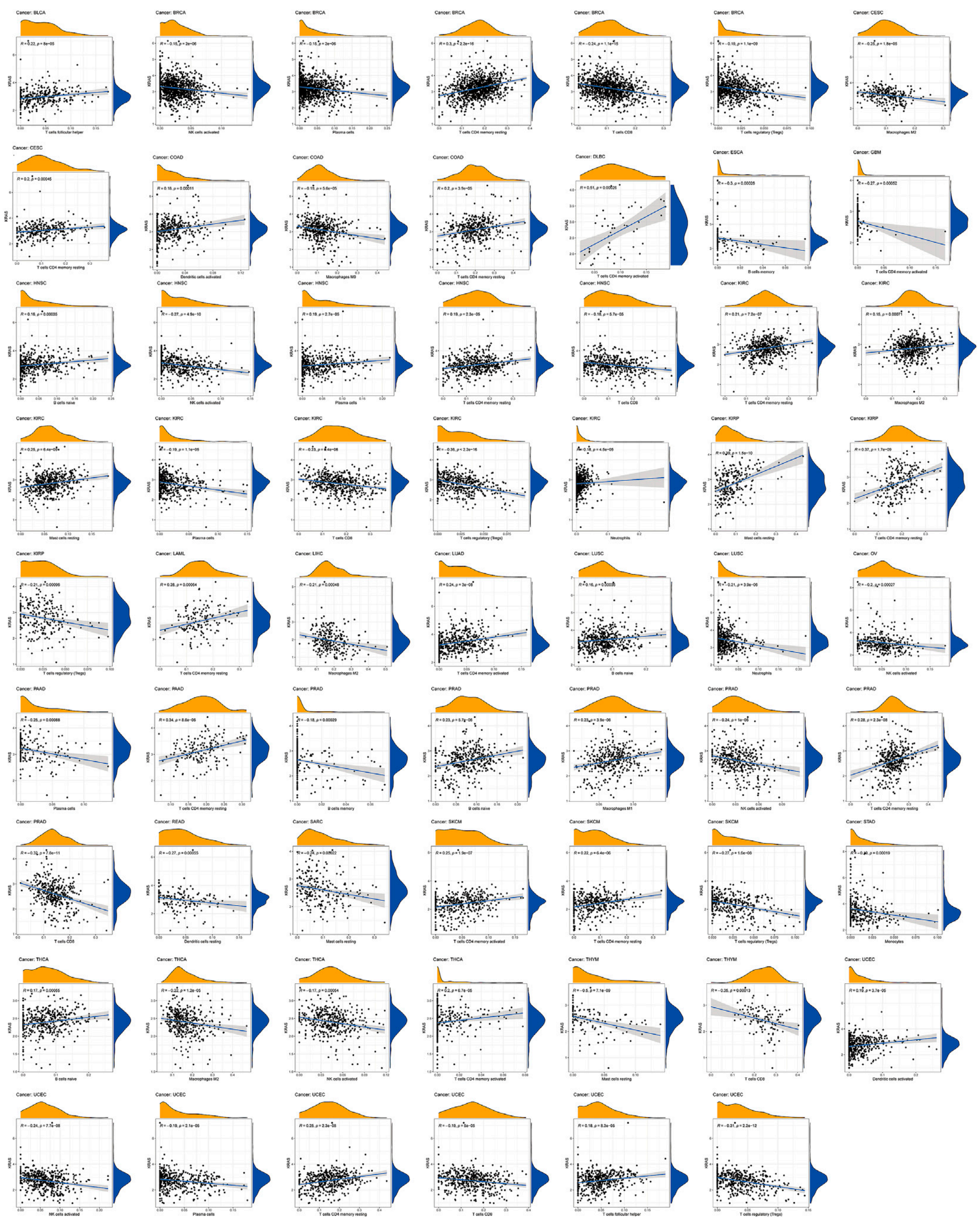


FIGURE 5
Relationships between *KRAS* expression and different types of immune cells infiltration level in tumors.

negatively correlated with the infiltration levels of resting mast cells and CD8 T cells in THYM. High KRAS expression levels were positively correlated with the infiltration levels of activated dendritic cells, resting memory CD4 T cells, and follicular helper T cells in UCEC. High *KRAS* expression levels were negatively correlated with the infiltration levels of activated NK cells, plasma cells, CD8 T cells and regulatory T cells (Tregs).

3.6 Functional enrichment analysis of KRAS

KRAS is a signal transducer protein that binds to GTP in the MAPK pathway, and mutations have been found in a quarter of cancers. We fabricated the PPI network using the STRING database based on *KRAS* and *KRAS*-related genes (Figure 6A).

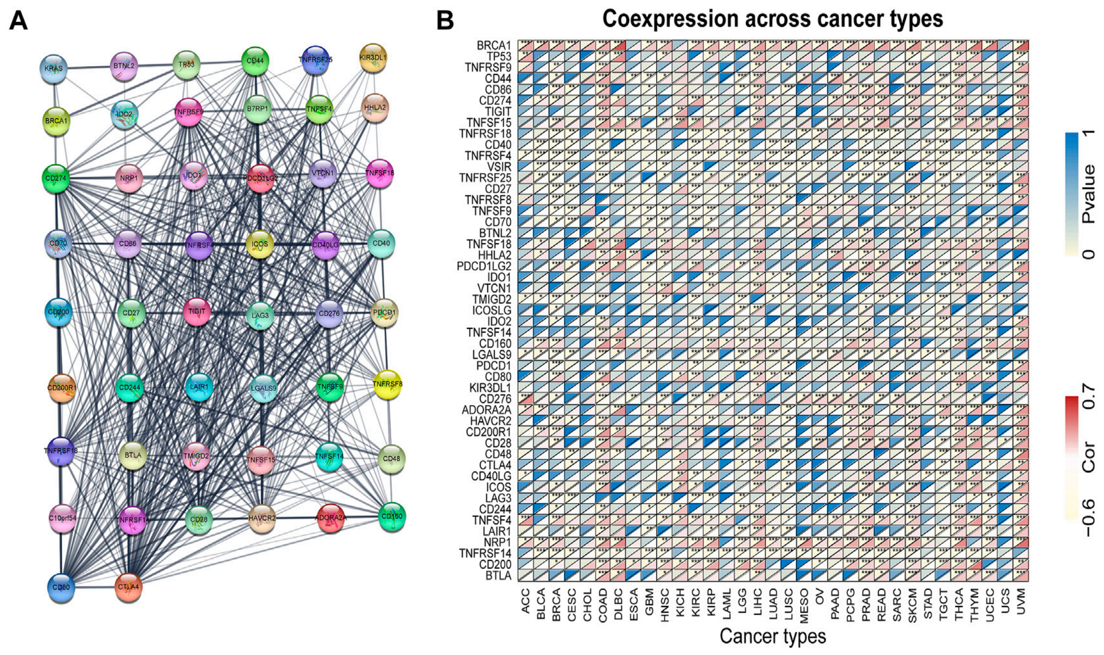


FIGURE 6
KRAS protein-protein network and expression relationships between *KRAS* and related genes. **(A)** PPI network for *KRAS*-interaction genes. **(B)** Correlation between *KRAS* expression and related genes (immune checkpoint genes and interacted related genes) expression.

Furthermore, we utilized the genes from the PPI network to analyze the association with *KRAS* expression. The results indicate that the great mass of immune checkpoint genes and *KRAS*-related genes were correlated with 33 kinds of tumor cancers (Figure 6B).

We utilized GSEA to discuss the molecular circadian mechanism in multiple cancers. *KRAS* expression was involved in more than 96 kinds of GO (Gene Ontology) pathways in 33 tumor cancers (Figure 7A). In ACC, high expression levels of *KRAS* proteins were involved in epidermal development, neural signal response and sensory perception of a smell. In BLAC, high *KRAS* expression levels affect the cellular amide metabolic process, mRNA binding sites, cell migration and cell keratinization. Furthermore, high *KRAS* gene expression levels were related to the detection of chemical stimuli in COAD, DLBC, ESCA, LIHC, LUAD, LUSC, PCPG, READ, SKCM, STAD and THYM. *KRAS* proteins could be involved in mRNA binding in LUAD, UCEC, DLBC, ESCA, LIHC, LUSC, PRAD and SKCM. High *KRAS* expression levels were involved in recognizing and characterizing the olfactory stimulus signal in BRCA, COAD, DLBC, ESCA, LUAD, LUSC, READ, SKCM, STAD and THYM. On the other hand, KRAS is an essential member of keratinocyte differentiation in LIHC, OV and PAAD. The production of the immunoglobulin complex was due to the high *KRAS* expression in CHOL, PRAD, UVM, HNSC and UVM. Intermediate filament formation was related to high *KRAS* expression in CESC, LIHC, PAAD and OV. The result demonstrated that *KRAS* genes were involved in the formation of amyloid fibrils in LAML and MESO.

Interestingly, KEGG analysis promoted revealed the expression levels of *KRAS* genes involved in more than 20 kinds of pathways in 7 tumors. Increased *KRAS* expression is involved in the calcium signaling pathway in LUSC, OV, READ and STAD. In LUSC and READ, antigen processing and presentation seemed to have a correlation with high *KRAS* expression levels. Neuroactive ligand–receptor interactions are also effected in LGG, OV and STAD. High *KRAS* expression levels promoted ascorbate and alternate metabolism in HNSC and LGG (Figure 7B). There are many other GO ontology and KEGG pathways related to *KRAS* expression.

4 Discussion

The RAS gene family is one of the most widely studied families of cancer-related genes. Previous studies have shown that *KRAS* is the most common mutation of the three altered genes (Wennerberg et al., 2005). KRAS-related carcinogenic mutations are prevalent in human cancer, occurring in 17%–25% of all cancers. The results indicated that cancer could lead to the abnormal expression of *KRAS* in tumor tissues. In addition,

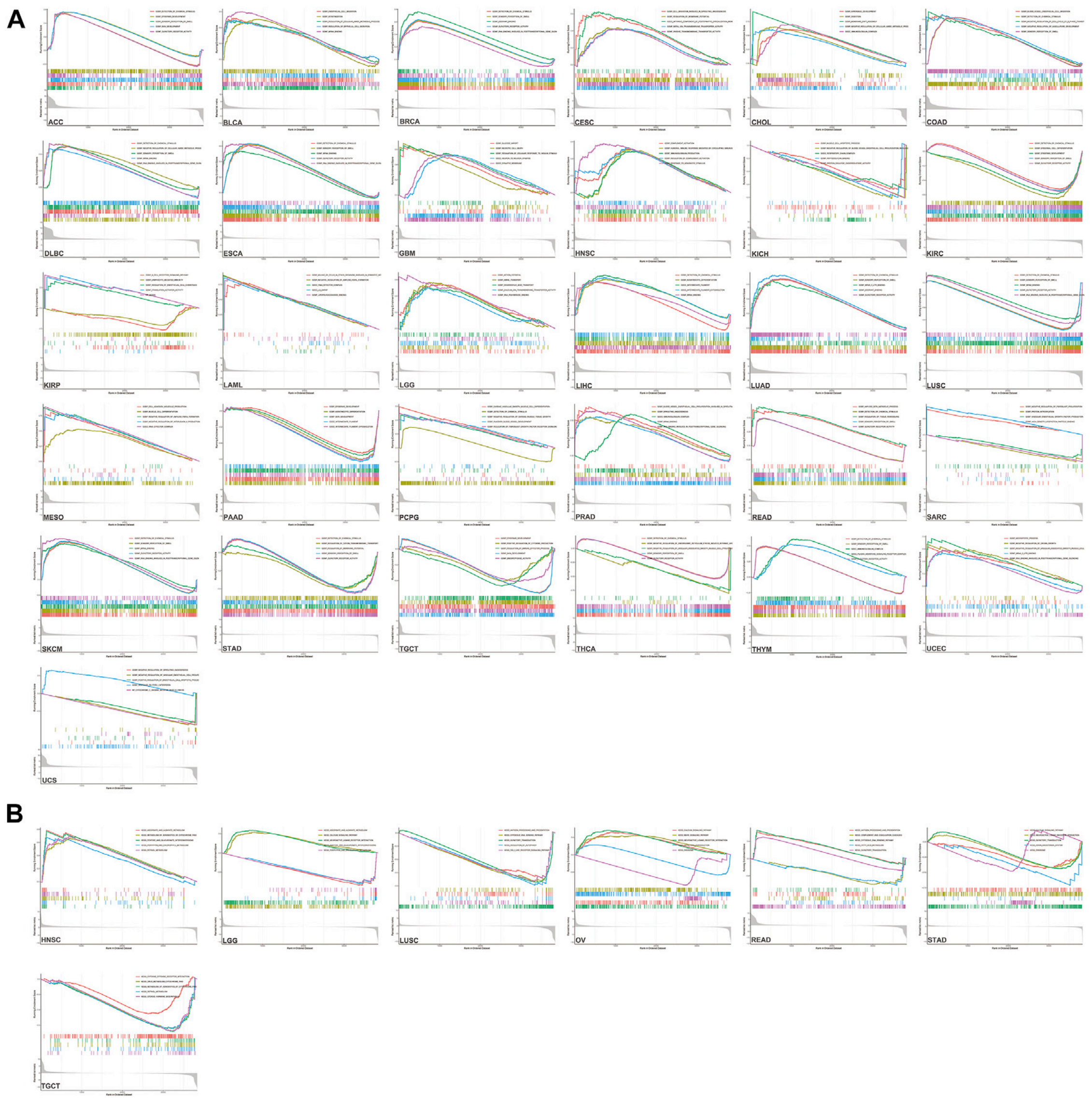


FIGURE 7
GSEA enrichment analysis of *KRAS* **(A)** Go enrichment analysis in various tumors. **(B)** KEGG enrichment analysis in various tumors.

we analyzed the *KRAS* expression levels on different pathology grades; high expression levels of *KRAS* genes affected the degree of tumor deterioration at serious stages, and correlation analysis showed in ACC, COAD, ESCA, KIRC, KIRP, LIHC, MESO, SKCM and STAD. Many studies have shown that *KRAS* has an essential relationship with the occurrence of cancer (Roberts and Stinchcombe, 2013). Therefore, *KRAS* has the potential to be introduced as a prognostic factor in many tumors.

TMB was calculated by the number of noninherited mutations, which are considered to be an important genetic feature to evaluate tumor tissue (Merino et al., 2020). As a genomic biomarker that predicts a good response to an immune checkpoint inhibitor (Kim et al., 2019). MSI is generated from impaired DNA mismatch repair, which occurs during gene duplication (Buecher et al., 2013). MSI p roduction is not random, and different target genes will lead to different phenotypes and pathologies and affect the pathogenesis of many kinds of cancer (Imai and Yamamoto, 2008). In addition, high *KRAS* expression levels showed a correlation with TMB in 14 tumors: BLCA, COAD, HNSC, LIHC, LUAD, LUSC, LUSC, PAAD, OV, PRAD, SARC, STAD, THYM, UCEC and UVM; with MSI in BRCA, COAD, DLBC, READ, SKCM, STAD, TGCT and UCEC. TMB and MSI are relatively new biomarkers, and there is still a need to perform more studies.

RAS mutations participating in the production of human cancer cells have been studied in many types of research.

Nevertheless, its potential mechanism and molecular regulatory mechanism need to be further clarified (Hu et al., 2022b). *KRAS* mutation is considered to be one of the most common genome variation in non-small cell lung cancer and is associated with a clinical background and pathological features (Suda et al., 2010). The pro-tumor inflammation caused by *KRAS* is associated with immune regulation, which leads to immune escape in the TME (Hamarsheh et al., 2020). The immune and stromal scores reflected the proportion of cancer cells in tumor tissue; we established 22 kinds of expression datasets by immune-infiltrating cells. Our research suggested that *KRAS* was associated with immunotherapeutic effects and cell viability in multiple tumors and had excellent potential as a cancer-targeted drug.

KRAS has been reported as a biomarker in multiple cancers (Petrelli et al., 2013; Siddiqui and Piperdi, 2010; Yang et al., 2013). In our study, the KRAS expression levels in various tumor cells were associated with prognosis and immune cell infiltration. From the GO ontology and KEGG pathways, the expression of *KRAS* plays an essential role in different stages of cancers with multiple functions. We found that *KRAS* interacts with several cancer-related genes from the PPI network, but their mechanism needs further study. Based on our study, *KRAS* may be a reliable prognostic biomarker for cancer patients in the course of diagnosis and treatment.

5 Conclusion

We studied the novel cancer-related gene *KRAS*, which belongs to the *RAS* gene family, in 33 kinds of tumors. This study has shown that *KRAS* were significantly different in 12 tumor tissues compared to normal tissues. Furthermore, the second significant finding was that the prognostic value of OS, DSS and DFI and DSS correlated with KRAS expression levels in 4, 3, 4 and 6 kinds of tumors, separately. In addition, *KRAS* expression levels were associated with tumor mutation burden (TMB) and microsatellite instability (MSI) in 14 and 8 tumors. As for 22 immune infiltrating cells, immune and stromal scores showed 11 and 6 kinds of tumors correlated with *KRAS* expression levels based on the tumor purity. The PPI network and functional enrichment participate in different biological metabolic pathways. Go and KEGG enrichment analysis revealed that *KRAS* was connected with more than 96 GO pathways in 33 tumor cancer cells and more than 20 kinds of KEGG pathways in 7 tumors, indicated that KRAS expression was involved in epidermal development, neural signal response, sensory perception of a smell, metabolic process, mRNA binding sites, cell migration and cell keratinization in multiple tumors. Our study suggested that *KRAS* may be a promising prognostic biomarker for cancer diagnosis and treatment.

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

LX and LZ designed the research; DZ and LW pergormed the reseatch, ZC modified figures; DZ wrote the manuscript. All authors read and approved the manuscript.

Funding

This work was supported by the National Natural Science Foundation of China (No. 61902259, No. 62001311), China Postdoctoral Science Foundation (No. 2022T150095), the National Science Foundation and Research Funding of Shenzhen Polytechnic (No. 6022310029K, No. 6020330001K).

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

References

Ando, Y., Sawada, M., Kawakami, T., Morita, M., and Aoki, Y. (2021). A patient with noonan syndrome with a KRAS mutation who presented severe nerve root hypertrophy. *Case Rep. Neurol.* 13 (1), 108–118. doi:10.1159/000512265

Andreyev, H. J., Norman, A. R., Cunningham, D., Oates, J., Dix, B. R., Iacopetta, B. J., et al. (2001). Kirsten ras mutations in patients with colorectal cancer: The 'RASCAL II' study. *Br. J. Cancer* 85 (5), 692–696. doi:10.1054/bjoc.2001.1964

Ao, C., Yu, L., and Zou, Q. (2021). Prediction of bio-sequence modifications and the associations with diseases. *Brief. Funct. Genomics* 20 (1), 1–18. doi:10.1093/bfgp/elaa023

Arrington, A. K., Heinrich, E. L., Lee, W., Duldulao, M., Patel, S., Sanchez, J., et al. (2012). Prognostic and predictive roles of KRAS mutation in colorectal cancer. *Int. J. Mol. Sci.* 13 (10), 12153–12168. doi:10.3390/ijms131012153

Buecher, B., Cacheux, W., Rouleau, E., Dieumegard, B., Mitry, E., and Lievre, A. (2013). Role of microsatellite instability in the management of colorectal cancers. *Dig. Liver Dis.* 45 (6), 441–449. doi:10.1016/j.dld.2012.10.006

Cao, C., Wang, J., Kwok, D., Cui, F., Zhang, Z., Zhao, D., et al. (2021). webTWAS: a resource for disease candidate susceptibility genes identified by transcriptome-wide association study. *Nucleic Acids Res.* 50 (D1), D1123–D1130. doi:10.1093/nar/gkab957

Capella, G., Cronauer-Mitra, S., Pienado, M. A., and PeruchoM. (1991). Frequency and spectrum of mutations at codons 12 and 13 of the c-K-ras gene in human tumors. *Environ. Health Perspect.* 93, 125–131. doi:10.1289/ehp.9193125

Chang, Z., Ju, H., Ling, J., Zhuang, Z., Li, Z., Wang, H., et al. (2014). Cooperativity of oncogenic K-ras and downregulated p16/INK4A in human pancreatic tumorigenesis. *PLoS One* 9 (7), e101452. doi:10.1371/journal.pone.0101452

Drosten, M., and Barbacid, M. (2020). Targeting the MAPK pathway in KRAS-driven tumors. *Cancer Cell.* 37 (4), 543–550. doi:10.1016/j.ccell.2020.03.013

Forbes, S. A., Bindal, N., Bamford, S., Cole, C., Kok, C. Y., Beare, D., et al. (2011). Cosmic: Mining complete cancer genomes in the catalogue of somatic mutations in cancer. *Nucleic Acids Res.* 39, D945–D950. (Database issue). doi:10.1093/nar/gkq929

Gu, Y., Gao, Y., Tang, X., Xia, H., and Shi, K. (2021). Bioinformatics analysis identifies CPZ as a tumor immunology biomarker for gastric cancer. *Curr. Bioinform.* 16 (1), 98–105. doi:10.2174/1574893615999200707145643

Hamarsheh, S., GroB, O., Brummer, T., and Zeiser, R. (2020). Immune modulatory effects of oncogenic KRAS in cancer. *Nat. Commun.* 11 (1), 5439. doi:10.1038/s41467-020-19288-6

Hartman, D. J., Davison, J. M., Foxwell, T. J., Nikiforova, M. N., and Chiosea, S. I. (2012). Mutant allele-specific imbalance modulates prognostic impact of KRAS mutations in colorectal adenocarcinoma and is associated with worse overall survival. *Int. J. Cancer* 131 (8), 1810–1817. doi:10.1002/ijc.27461

Hu, Y., Zhang, Y., Zhang, H., Gao, S., Wang, L., Wang, T., et al. (2022). Mendelian randomization highlights causal association between genetically increased C-reactive protein levels and reduced Alzheimer's disease risk. *Alzheimer's & Dementia : The Journal of the Alzheimer's Association.* doi:10.1002/alz.12687

Hu, Y., Sun, J. Y., Zhang, Y., Zhang, H., Gao, S., Wang, T., et al. (1990). rs1990622 variant associates with Alzheimer's disease and regulates TMEM106B expression in human brain tissues. *BMC Med.* 19 (1), 11. doi:10.1186/s12916-020-01883-5

Hu, Y., Zhang, H., Liu, B., Gao, S., Wang, T., Han, Z., et al. (2020). rs34331204 regulates TSPAN13 expression and contributes to Alzheimer's disease with sex differences. *Brain* 143 (11), e95. doi:10.1093/brain/awaa302

Hu, Y., Zhang, Y., Zhang, H., Gao, S., Wang, L., Wang, T., et al. (2022). Cognitive performance protects against Alzheimer's disease independently of educational attainment and intelligence. *Mol. Psychiatry.* doi:10.1038/s41380-022-01695-4

Imai, K., and Yamamoto, H. (2008). Carcinogenesis and microsatellite instability: The interrelationship between genetics and epigenetics. *Carcinogenesis* 29 (4), 673–680. doi:10.1093/carcin/bgm228

Kim, J. Y., Kronbichler, A., Eisenhut, M., Hong, S. H., van der Vliet, H. J., Kang, J., et al. (2019). Tumor mutational burden and efficacy of immune checkpoint inhibitors: A systematic review and meta-analysis. *Cancers (Basel)* 11 (11), E1798. doi:10.3390/cancers11111798

Li, J., Tao, H., Shan, J., Liu, F., Deng, X., et al. (2021). Differential hippocampal protein expression between normal mice and mice with the perioperative neurocognitive disorder: A proteomic analysis. *Eur. J. Med. Res.* 16 (1), 130–138. doi:10.1186/s40001-021-00599-3

Luo, Y., Wang, X., Li, L., Wang, Q., Hu, Y., He, C., et al. (2021). Bioinformatics analysis reveals centromere protein K can serve as potential prognostic biomarker and therapeutic target for non-small cell lung cancer. *Curr. Bioinform.* 16 (1), 106–119. doi:10.2174/1574893615999200728100730

Merino, D. M., McShane, L. M., Fabrizio, D., Funari, V., Chen, S. J., White, J. R., et al. (2020). Establishing guidelines to harmonize tumor mutational burden (TMB): In silico assessment of variation in TMB quantification across diagnostic platforms: phase I of the friends of cancer research TMB harmonization project. *J. Immunother. Cancer* 8 (1), e000147. doi:10.1136/jitc-2019-000147

Newman, A. M., Liu, C. L., Green, M. R., Gentles, A. J., Feng, W., Xu, Y., et al. (2015). Robust enumeration of cell subsets from tissue expression profiles. *Nat. Methods* 12 (5), 453–457. doi:10.1038/nmeth.3337

Niihori, T., Aoki, Y., Narumi, Y., Neri, G., Cave, H., Verloes, A., et al. (2006). Germline KRAS and BRAF mutations in cardio-facio-cutaneous syndrome. *Nat. Genet.* 38 (3), 294–296. doi:10.1038/ng1749

Pantsar, T. (2020). The current understanding of KRAS protein structure and dynamics. *Comput. Struct. Biotechnol. J.* 18, 189–198. doi:10.1016/j.csbj.2019.12.004

Petrelli, F., Coinu, A., Cabiddu, M., Ghilardi, M., and Barni, S. (2013). KRAS as prognostic biomarker in metastatic colorectal cancer patients treated with bevacizumab: A pooled analysis of 12 published trials. *Med. Oncol.* 30 (3), 650. doi:10.1007/s12032-013-0650-4

Roberts, P. J., and Stinchcombe, T. E. (2013). KRAS mutation: Should we test for it, and does it matter? *J. Clin. Oncol.* 31 (8), 1112–1121. doi:10.1200/JCO.2012.43.0454

Scolnick, E. M., Papageorge, A. G., and Shih, T. Y. (1979). Guanine nucleotide-binding activity as an assay for src protein of rat-derived murine sarcoma viruses. *Proc. Natl. Acad. Sci. U. S. A.* 76 (10), 5355–5359. doi:10.1073/pnas.76.10.5355

Shields, J. M., Pruitt, K., McFAll, A., ShAub, A., and Der, C. J. (2000). Understanding ras: 'it ain't over 'til it's over'. *Trends Cell. Biol.* 10 (4), 147–154. doi:10.1016/s0962-8924(00)01740-2

Siddiqui, A. D., and Piperdi, B. (2010). KRAS mutation in colon cancer: A marker of resistance to egfr-I therapy. *Ann. Surg. Oncol.* 17 (4), 1168–1176. doi:10.1245/s10434-009-0811-z

Singh, K., Gollapudi, S., Mittal, S., Small, C., Kumar, J., and Ohgami, R. S. (2021). Point mutation specific antibodies in B-cell and T-cell lymphomas and leukemias: Targeting IDH2, KRAS, BRAF and other biomarkers RHOA, IRF8, MYD88, ID3, NRAS, SF3B1 and EZH2. *Diagn. (Basel)* 11 (4), 600. doi:10.3390/diagnostics11040600

Suda, K., Tomizawa, K., and Mitsudomi, T. (2010). Biological and clinical significance of KRAS mutations in lung cancer: An oncogenic driver that contrasts with EGFR mutation. *Cancer Metastasis Rev.* 29 (1), 49–60. doi:10.1007/s10555-010-9209-4

Szklarczyk, D., Gable, A. L., Lyon, D., Junge, A., Wyder, S., Huerta-Cepas, J., et al. (2019). STRING v11: Protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Res.* 47 (D1), D607-D613–D613. doi:10.1093/nar/gky1131

Tang, W., Wan, S., Yang, Z., Teschendorff, A. E., and Zou, Q. (2018). Tumor origin detection with tissue-specific miRNA and DNA methylation markers. *Bioinformatics* 34 (3), 398–406. doi:10.1093/bioinformatics/btx622

Tsuchida, N., Ryder, T., and Ohtsubo, E. (1982). Nucleotide sequence of the oncogene encoding the p21 transforming protein of Kirsten murine sarcoma virus. *Science* 217 (4563), 937–939. doi:10.1126/science.6287573

Vogelstein, B., and Kinzler, K. W. (2004). Cancer genes and the pathways they control. *Nat. Med.* 10 (8), 789–799. doi:10.1038/nm1087

Wang, S., Zhang, Y., Mu, H., and Pang, S. (2021). Identification of cancer trait genes and association analysis under pan-cancer. *Curr. Bioinform.* 16 (8), 1101–1114. doi:10.2174/1574893616666210601151306

Welman, A., Burger, M. M., and Hagmann, J. (2000). Structure and function of the C-terminal hypervariable region of K-Ras4B in plasma membrane targetting and transformation. *Oncogene* 19 (40), 4582–4591. doi:10.1038/sj.onc.1203818

Wennerberg, K., Rossman, K. L., and Der, C. J. (2005). The Ras superfamily at a glance. *J. Cell. Sci.* 118 (5), 843–846. doi:10.1242/jcs.01660

Westcott, P. M., Halliwill, K. D., To, M. D., Rashid, M., Rust, A. G., Keane, T. M., et al. (2015). The mutational landscapes of genetic and chemical models of Kras-driven lung cancer. *Nature* 517 (7535), 489–492. doi:10.1038/nature13898

Xu, Q., Zhang, G., Liu, Q., Li, S., and Zhang, Y. (2022). Inhibitors of the GTPase KRAS^{G12C} in cancer: A patent review (2019–2021). *Expert Opin. Ther. Pat.* 32 (5), 475–505. doi:10.1080/13543776.2022.2032648

Yang, Z.-Y., Wu, X. Y., Huang, Y. F., Di, M. Y., Zheng, D. Y., Chen, J. Z., et al. (2013). Promising biomarkers for predicting the outcomes of patients with KRAS wild-type metastatic colorectal cancer treated with anti-epidermal growth factor receptor monoclonal antibodies: A systematic review with meta-analysis. *Int. J. Cancer* 133 (8), 1914–1925. doi:10.1002/ijc.28153

Yoshihara, K., Shahmoradgoli, M., Martinez, E., Vegesna, R., Kim, H., Torres-Garcia, W., et al. (2013). Inferring tumour purity and stromal and immune cell admixture from expression data. *Nat. Commun.* 4, 2612. doi:10.1038/ncomms3612

Yu, L., et al. (2020). Prediction of drug response in multilayer networks based on fusion of multiomics data. *Methods* 192, 85. doi:10.1016/j.ymeth.2020.08.006

Yu, L., Wang, M., Yang, Y., Xu, F., Zhang, X., Xie, F., et al. (2021). Predicting therapeutic drugs for hepatocellular carcinoma based on tissue-specific pathways. *PLoS Comput. Biol.* 17 (2), e1008696. doi:10.1371/journal.pcbi.1008696

Yu, L., Zhao, J., and Gao, L. (2018). Predicting potential drugs for breast cancer based on miRNA and tissue specificity. *Int. J. Biol. Sci.* 14 (8), 971–982. doi:10.7150/ijbs.23350

Zarredar, H., Pashapour, S., Ansarin, K., Khalili, M., Baghban, R., and Farajnia, S. (2019). Combination therapy with KRAS siRNA and EGFR inhibitor AZD8931 suppresses lung cancer cell growth *in vitro*. *J. Cell. Physiol.* 234 (2), 1560–1566. doi:10.1002/jcp.27021

TYPE Original Research
PUBLISHED 09 November 2022
DOI 10.3389/fgene.2022.1053772

Check for updates

OPEN ACCESS

EDITED AND REVIEWED BY
Quan Zou,
University of Electronic Science and Technology of China, China
EDITED AND REVIEWED BY
Jing Yang,
ShanghaiTech University, China
Gele Aori,
University of Toyama, Japan

*CORRESPONDENCE
Zhenbing Zeng,
zbzeng@shu.edu.cn
Tao Huang,
tohuangtao@126.com
Yu-Dong Cai,
cai_yud@126.com

†These authors have contributed equally to this work

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 26 September 2022
ACCEPTED 27 October 2022
PUBLISHED 09 November 2022

CITATION
Lu J, Meng M, Zhou X, Ding S, Feng K, Zeng Z, Huang T and Cai Y-D (2022), Identification of COVID-19 severity biomarkers based on feature selection on single-cell RNA-Seq data of CD8+ T cells.
Front. Genet. 13:1053772.
doi: 10.3389/fgene.2022.1053772

Identification of COVID-19 severity biomarkers based on feature selection on single-cell RNA-Seq data of CD8$^+$ T cells

Jian Lu[1,2†], Mei Meng[3†], XianChao Zhou[3], Shijian Ding[4], KaiYan Feng[5], Zhenbing Zeng[1*], Tao Huang[2,6*] and Yu-Dong Cai[4*]

[1]Department of Mathematics, School of Sciences, Shanghai University, Shanghai, China, [2]CAS Key Laboratory of Computational Biology, Bio-Med Big Data Center, Shanghai Institute of Nutrition and Health, University of Chinese Academy of Sciences, Chinese Academy of Science, Shanghai, China, [3]State Key Laboratory of Oncogenes and Related Genes, Center for Single-Cell Omics, School of Public Health, Shanghai Jiao Tong University School of Medicine, Shanghai, China, [4]School of Life Sciences, Shanghai University, Shanghai, China, [5]Department of Computer Science, Guangdong AIB Polytechnic College, Guangzhou, China, [6]CAS Key Laboratory of Tissue Microenvironment and Tumor, Shanghai Institute of Nutrition and Health, University of Chinese Academy of Sciences, Chinese Academy of Sciences, Shanghai, China

The global outbreak of the COVID-19 epidemic has become a major public health problem. COVID-19 virus infection triggers a complex immune response. CD8$^+$ T cells, in particular, play an essential role in controlling the severity of the disease. However, the mechanism of the regulatory role of CD8$^+$ T cells on COVID-19 remains poorly investigated. In this study, single-cell gene expression profiles from three CD8$^+$ T cell subtypes (effector, memory, and naive T cells) were downloaded. Each cell subtype included three disease states, namely, acute COVID-19, convalescent COVID-19, and unexposed individuals. The profiles on each cell subtype were individually analyzed in the same way. Irrelevant features in the profiles were first excluded by the Boruta method. The remaining features for each CD8$^+$ T cells subtype were further analyzed by Max-Relevance and Min-Redundancy, Monte Carlo feature selection, and light gradient boosting machine methods to obtain three feature lists. These lists were then brought into the incremental feature selection method to determine the optimal features for each cell subtype. Their corresponding genes may be latent biomarkers to determine COVID-19 severity. Genes, such as ZFP36, DUSP1, TCR, and IL7R, can be confirmed to play an immune regulatory role in COVID-19 infection and recovery. The results of functional enrichment analysis revealed that these important genes may be associated with immune functions, such as response to cAMP, response to virus, T cell receptor complex, T cell activation, and T cell differentiation. This study further set up different gene expression pattens, represented by classification rules, on three states of COVID-19 and constructed several efficient classifiers to distinguish COVID-19 severity. The findings of this study provided new insights into the biological processes of CD8$^+$ T cells in regulating the immune response.

KEYWORDS

COVID-19, severity, CD8$^+$ T cell, single-cell, feature selection

1 Introduction

Caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), coronavirus disease 2019 (COVID-19) has cumulatively infected more than 400 million people. It is mainly transmitted in the population through close contact, and typical clinical symptoms are fever and cough (Sanyal, 2020). SARS-CoV-2 enters host cells through endocytosis by binding to angiotensin-converting enzyme 2 (ACE2) receptor on the cell surface (Samudrala et al., 2020). Several variants have emerged, and the main ones are Alpha, Beta, Gamma, Delta, Lambda, and Omicron (Araf et al., 2022; Fiolet et al., 2022).

Viral infections involve a complex immune response process, in which T lymphocytes, especially CD8$^+$ T cells, are crucial to the control and clearance of acute infections. CD8$^+$ T lymphocytes can selectively kill infected cells by mediating adaptive cytotoxic T cell responses, thereby eliminating the virus (Westmeier et al., 2020). CD8$^+$ T cells exert cytotoxic effects mainly through target cell lysis and cytokine release (Slifka and Whitton, 2000). In the target cell lysis pathway, target cells are lysed through the Fas/FasL pathway or perforin, whereas the cytokine pathway is associated with IFNγ and TNFα. Strong CD8$^+$ T cell responses specific to SARS-CoV-2 are associated with worse disease severity; SARS-CoV-2 infection results in a decrease in CD8$^+$ T cell frequency, which becomes more pronounced with increasing infection severity (Chen et al., 2020). SARS-CoV-2-specific CD8$^+$ T-cell responses are rarely detected in patients with fatal COVID-19 (Dan et al., 2021) because of CD8$^+$ T-cell depletion after overactivation, which ultimately reduces the host cellular immune response to the virus (Zheng et al., 2020; Gong et al., 2021).

Cellular immunity involves the transformation of naive, effector, and memory T cells. The proportion of CD8$^+$ T cell subsets correlates with COVID-19 severity (Westmeier et al., 2020). Patients with moderate COVID-19 have a significantly increased proportion of effector CD8$^+$ T cells and effector memory CD8$^+$ T cells than healthy subjects and severely infected patients (Fenoglio et al., 2021), whereas naïve CD8$^+$ T cells are reduced in old people and negatively correlated with patient age (Westmeier et al., 2020). Naïve CD8$^+$ T cells correlated with age and differed across infection status (unexposed, acute, and recovering patients) (Grifoni et al., 2020). In contrast, studies between groups of COVID-19 patients showed that those with severe infection exhibited higher levels of naive CD8$^+$ T cells and lower levels of effector CD8$^+$ T cells and effector memory CD8$^+$ T cells compared with patients with mild infection (Fenoglio et al., 2021), which may imply a defective cytotoxic lymphocyte response in severe infections. In addition, in COVID-19 patients, the dominant effector CD8$^+$ T cells were GzmA, GzmB, and perforin triple-positive cells, compared with uninfected individuals; patients expressing effector CD8$^+$ T cells that produce multiple virulence molecules exhibited milder symptoms (Westmeier et al., 2020), which may indicate a potential protective mechanism.

As the viral infection subsides, some T cells differentiate into memory T cells. Memory T cells can persist in patients for long periods of time; thus, they play a protective role in preventing viral reinfection (Nguyen et al., 2019). Compared with non-hospitalized patients, hospitalized patients did not have a higher frequency of memory CD8$^+$ T cells, and the proportion tended to be stable over time (Grifoni et al., 2020). SARS-CoV-2-specific memory CD8$^+$ T cells were related to less severe COVID-19 during infection, because SARS-CoV-2 memory T cells can limit the accumulation of SARS- CoV-2 and viral load, thereby reducing COVID-19 disease severity (Kotturi et al., 2007; Francis et al., 2022). As CD8$^+$ T cells are crucial to the infection of SARS-CoV-2, studying the characteristics of different types of CD8$^+$ T cells in different infection states provides a useful reference for finding potential targets for treatment.

In this study, several computational methods were used to investigate the gene expression profiles of three subtypes of CD8$^+$ T cells (effector, memory and naïve T cells) related to COVID-19. Three disease states: unexposed, acute, and convalescent, were included in the profiles on each cell subtype. The profiles on each cell subtype were individually analyzed in the same way. First, the profiles were analyzed by Boruta feature selection method (Kursa and Rudnicki, 2010) to exclude irrelevant gene features. Then, three feature ranking algorithms: Max-Relevance and Min-Redundancy (mRMR) (Peng et al., 2005), Monte Carlo feature selection (MCFS) (Draminski et al., 2008), and light gradient boosting machine (LightGBM) (Ke et al., 2017), were used to examine remaining features, resulting in three feature lists. Each list was fed into the incremental feature selection (IFS) method (Liu and Setiono, 1998) to extract essential gene features, construct efficient classifiers and set up classification rules. The essential genes can be latent biomarkers and the rules can indicate different expression patterns on three COVID-19 states, deepening our understanding on COVID-19.

2 Materials and methods

2.1 Datasets

The gene expression profiles of three subtypes of CD8$^+$ T cells related to COVID-19, including effector, memory, and naïve

TABLE 1 Sample size for the different categories under the datasets for three cell subtypes.

Cell subtype	Acute COVID-19	Convalescent COVID-19	Unexposed individuals	Total
Effector T cells	4832	23542	21288	49662
Memory T cells	6527	22031	28257	56815
Naïve T cells	5204	21504	12108	38816

T cells, were obtained from the GEO database by accessing a number of GSE188429 (Francis et al., 2022). These expression profiles were obtained by isolating CD8$^+$ T cells from individual peripheral blood mononuclear cells (PBMCs) and quantifying mRNA expression in the cells by single-cell transcriptome sequencing techniques. CD8$^+$ T cell responses in PBMCs from three cohorts were studied, as follows: acute COVID-19, convalescent COVID-19, and unexposed individuals. A total of 145,293 cell samples were included in these profiles and the number of samples under different cohorts for each CD8$^+$ T cell subtype is shown in Table 1. After filtering low expression and low variance genes, 1046 genes were kept and deemed as features in this study. We used the processed data in h5ad file acquired from https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE188429 for detailed analysis. For the next round of machine learning computations, the datasets for each of the three different cell subtypes were studied independently.

2.2 Boruta feature selection

Lots of gene features were used to represent each cell in three subtypes of CD8$^+$ T cells. Evidently, only a few of them are highly related to distinguish the states of COVID-19. It is essential to discover them. This task can be completed by some feature analysis methods. Here, the Boruta feature selection method (Kursa and Rudnicki, 2010) was adopted first to exclude irrelevant features.

The Boruta feature selection method is a feature selection wrapper algorithm, which can be used to assess the importance of features using a tree classifier (e.g., random forest (RF) (Breiman, 2001)) and hence reject irrelevant features. The approach particularly creates a shadow feature at random for each original feature and then compares them with the original features in terms of their importance generated by RF. An original feature is selected when it is statistically more important than the shadow features. Selected features are removed from the current dataset and the dataset containing remaining features is processed in the next round. Above procedures repeat several times until the number of rounds reaches the predefined value.

The present study used the Boruta program available at https://github.com/scikit-learn-contrib/boruta_py to analyze the datasets individually for three cell subtypes. It was run with default parameters.

2.3 Feature ranking methods

Important features can be extracted through Boruta. However, their importance was not clear. Three feature analysis methods followed to investigate selected features, including mRMR (Peng et al., 2005), MCFS (Draminski et al., 2008) and LightGBM (Ke et al., 2017).

2.3.1 mRMR

The mRMR uses mutual information as a metric to achieve the maximum correlation between features and class labels as well as the minimum redundancy between features. After mRMR analysis, features are ranked in a list. The list is produced by repeatedly selecting a feature with maximum correlation to class labels and minimum redundancy to already-selected features. For convenience, this list was called the mRMR feature list.

2.3.2 MCFS

The MCFS method is another effective feature selection method in machine learning. The method evaluates the importance of features by constructing a number of decision trees. Trees are set up on some randomly generated feature groups and sample sets. According to the occurrence of each feature in all trees, a relative importance (RI) score is computed and assigned to the feature to indicate its importance. With the decreasing order of RI scores, features are ranked in a list, named MCFS feature list.

2.3.3 LightGBM

The LightGBM represents ensemble learning algorithms and is a distributed gradient-boosting framework based on decision tree algorithm. As the algorithm is based on a tree classifier, it can be used to evaluate the importance of a feature by counting its frequency in all trees. Likewise, features are ranked in a list with the decreasing order of their frequencies. Such list was termed as LightGBM feature list.

In this investigation, the mRMR program used is obtained from http://home.penglab.com/proj/mRMR/. As for the MCFS program, the software developed by Draminski et al. (2008) was adopted, which can be accessed at http://www.ipipan.eu/staff/m.

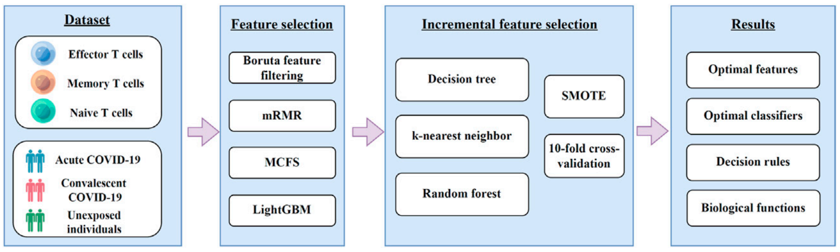


FIGURE 1
A diagram of the computational framework used in this study. We first analyzed the 3 T cell expression profiles of COVID-19 by different feature selection methods in machine learning. Then, we used the incremental feature selection method to determine the optimal features, build the optimal classifiers, and extract the important classification rules. The critical features obtained were enriched by GO and KEGG analysis to uncover their biological implications.

draminski/mcfs.html. The LightGBM program was implemented using the LightGBM library in python, which is available at https://lightgbm.readthedocs.io/en/latest/. The default parameters were used in all above three programs.

2.4 Incremental feature selection

After feature ranking, three feature lists for one subtype of $CD8^{+}$ T cells were obtained. However, it was not easy to determine the optimal features from these feature lists. In this step, the IFS method (Liu and Setiono, 1998) was employed to determine the optimal features in each list for a given classification algorithm. The procedures were described as below. When the step size was set to 1, the IFS method first generated a succession of feature subsets in a way that the first feature subset contained the first feature in the list, and the second feature subset included the top two features, and so on. For each feature subset, a classifier was built based on samples represented by features in this subset. All classifiers' performance was tested using 10-fold cross-validation (Kohavi, 1995). Finally, based on the performance indicators of each classifier, the classifier with the best performance can be obtained. Such classifier was called the optimal classifier and features used in this classifier were termed as the optimal features.

2.5 SMOTE

As shown in Table 1, the sizes of different categories in the gene expression profiles were of great differences, i.e., the profiles were imbalanced, which may lead to the unstable performance of the classifier on different categories. Therefore, the synthetic minority oversampling technique (SMOTE) algorithm (Chawla et al., 2002) was adopted to tackle this problem. It works by linearly synthesizing new samples for minority categories using the k-nearest neighbors concept, thereby ensuring that the

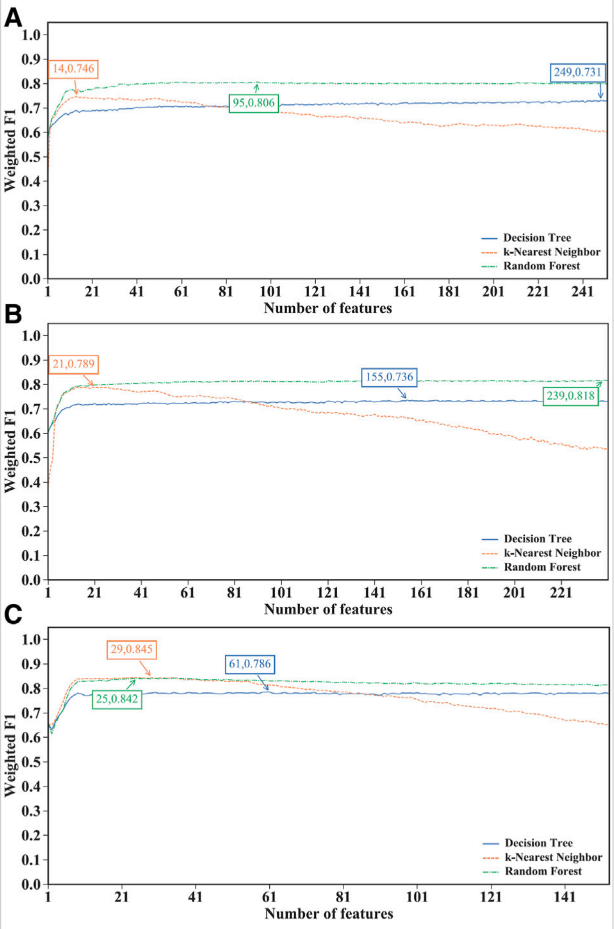


FIGURE 2
Performance of different classification algorithms with different number of features under the mRMR feature lists. **(A)** effector T cells, **(B)** memory T cells, **(C)** naïve T cells. Random forest provided the best performance on effector and memory T cells, whereas k-nearest neighbor yielded the best performance on naïve T cells.

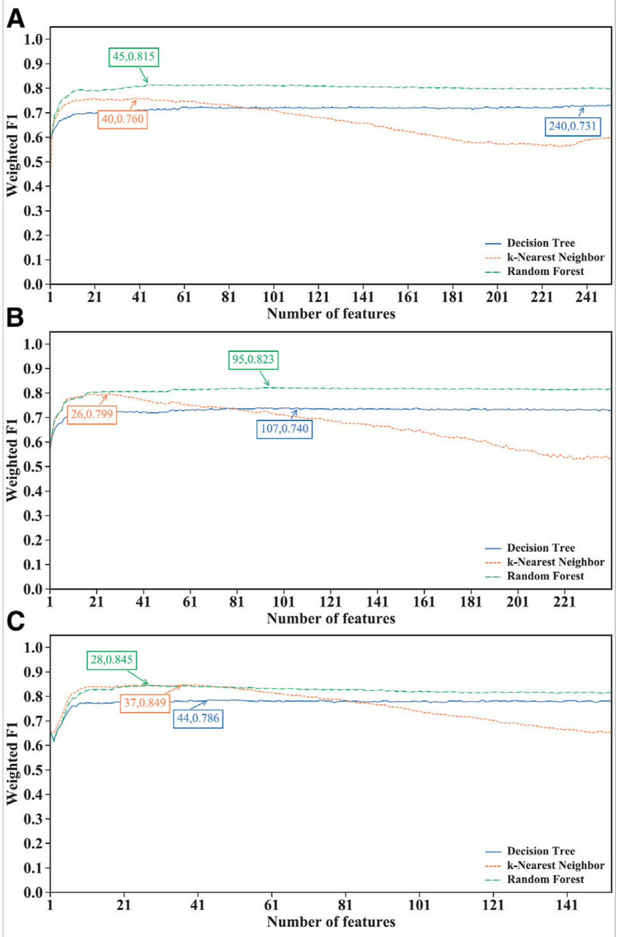


FIGURE 3
Performance of different classification algorithms with different number of features under the MCFS feature lists. **(A)** effector T cells, **(B)** memory T cells, **(C)** naïve T cells. Random forest provided the best performance on effector and memory T cells, whereas k-nearest neighbor yielded the best performance on naïve T cells.

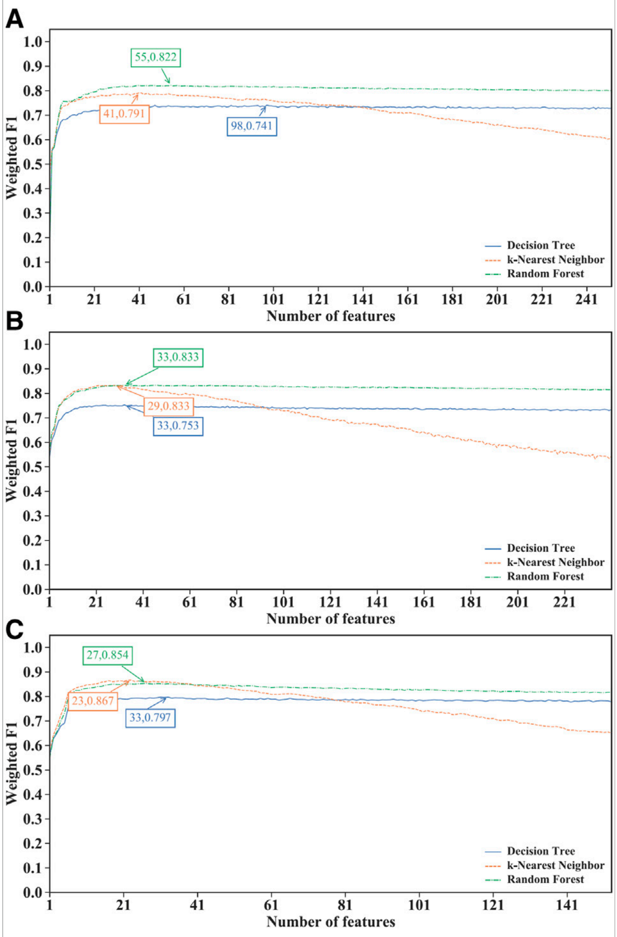


FIGURE 4
Performance of different classification algorithms with different number of features under the LightGBM feature list. **(A)** effector T cells, **(B)** memory T cells, **(C)** naïve T cells. Random forest provided the best performance on effector and memory T cells, whereas k-nearest neighbor yielded the best performance on naïve T cells.

quantity of samples from different categories is almost equal. This study used the SMOTE program available at https://github.com/scikit-learn-contrib/imbalanced-learn and executed it with default parameters.

2.6 Classification algorithms

As mentioned above, the IFS method needs a classification algorithm. For wide tests, three classification algorithms: k-nearest neighbors (kNN) (Cover and Hart, 1967), RF (Breiman, 2001), and decision tree (DT) (Safavian and Landgrebe, 1991), were attempted. These algorithms were widely used to tackling various medical problems (Chen et al., 2021; Chen et al., 2022; Ding et al., 2022; Li et al., 2022; Ran et al., 2022; Tang and Chen, 2022; Wu and Chen, 2022; Zhou et al., 2022; Wu and Chen, 2023).

2.6.1 kNN

This algorithm is one of the most classic classification algorithms in machine learning. For a test sample, kNN calculates its distance to all training samples and finds *k* nearest training samples. According to the classes of these training samples, the class of the test sample is determined. Generally, the majority voting is adopted to make the decision.

2.6.2 RF

RF is a classic algorithm in ensemble learning that first resamples *N* subsets from the original dataset based on the

TABLE 2 Detailed performance of the optimal classifiers obtained by using the mRMR, MCFS, and LightGBM methods for three cell subtypes.

Cell subtype	Feature ranking method	Classification algorithm	Number of features	ACC	MCC	Macro F1	Weighted F1
Effector T cells	mRMR	RF	95	0.809	0.670	0.748	0.806
	MCFS	RF	45	0.815	0.688	0.776	0.815
	LightGBM	RF	55	0.823	0.698	0.787	0.822
Memory T cells	mRMR	RF	239	0.821	0.694	0.786	0.818
	MCFS	RF	95	0.824	0.704	0.798	0.823
	LightGBM	RF	33	0.834	0.724	0.815	0.833
Naïve T cells	mRMR	kNN	29	0.841	0.755	0.826	0.845
	MCFS	kNN	37	0.844	0.761	0.828	0.849
	LightGBM	kNN	23	0.863	0.787	0.847	0.867

bagging strategy and uses each subset to train a decision tree classifier. Each tree is constructed by randomly selecting features. For a test sample, each tree gives its prediction. RF integrates these predictions with majority voting. Compared with decision trees, RF is more accurate and has a high generalization capability.

2.6.3 DT

This algorithm is quite different from kNN and RF. Although above two algorithms can provide high performance, their principles are hard to be understood. In this regard, DT has its special merits. The classification procedures of DT are completely open. In this case, it is possible for us to understand its classification principle. Besides the tree form, DT can also be represented by a set of if-then rules, each of which contains a group of conditions and one result. The conditions may indicate a special pattern for the result, giving insights to understand essential differences of various categories.

In this study, all three abovementioned algorithms were implemented *via* the scikit-learn library. These programs were performed by using their default parameters.

2.7 Performance measurement

For multi-class classification, overall accuracy is the most widely used measurement. It is defined as the proportion of corrected predicted samples among all samples. However, such measurement is not perfect when the dataset is imbalanced. In this case, Mathews Correlation Coefficient (MCC) (Matthews, 1975; Jurman et al., 2012; Liu et al., 2021; Pan et al., 2022; Wang and Chen, 2022; Yang and Chen, 2022) is more accurate to evaluate the performance of classifiers. It can be computed by

$$MCC = \frac{cov(X,Y)}{\sqrt{cov(X,X)cov(Y,Y)}} \tag{1}$$

where X indicates the binary matrix of the true classes of all samples, Y represents the binary matrix of the predicted classes of all samples, and $cov(.)$ denotes the correlation between two matrices.

Besides, F1 score was used to evaluate the performance of classifiers on each category in this study. The F1 score for one category can be computed by

$$F1\ score = \frac{2 \times TP}{2 \times TP + FN + FP} \tag{2}$$

where TP, FN and FP stand for the true positive, false negative and false positive of such category. In detail, TP is the number of accurately predicted samples in this category, FN is the number of wrongly predicted samples in this category and FP is the number of samples that belong to other categories but are predicted to be in this category. The F1 scores on all categories can be integrated to give an overall evaluation on classifiers' performance. Generally, there are two forms to make integrations. The first one is the direct mean of F1 scores on all categories. Such measurement is called macro F1. The second one further considers the weights of categories, i.e., the weighted mean of F1 scores on all categories. It is called weighted F1.

As different measurements can induce different results, a major measurement should be determined in advance. Here, we selected weighted F1 as the major measurement.

2.8 Biological function enrichment

Through the above computational analysis, some important genes can be discovered from the profiles on each subtype of CD8$^+$ T cells. To uncover the biological meanings behind these genes, the gene ontology (GO) and KEGG enrichment analysis was employed. The clusterProfiler 4.0 tool (Wu et al., 2021) was adopted to conduct the enrichment analysis. The threshold on p-value was set to 0.05 for selecting enriched GO terms and KEGG pathways.

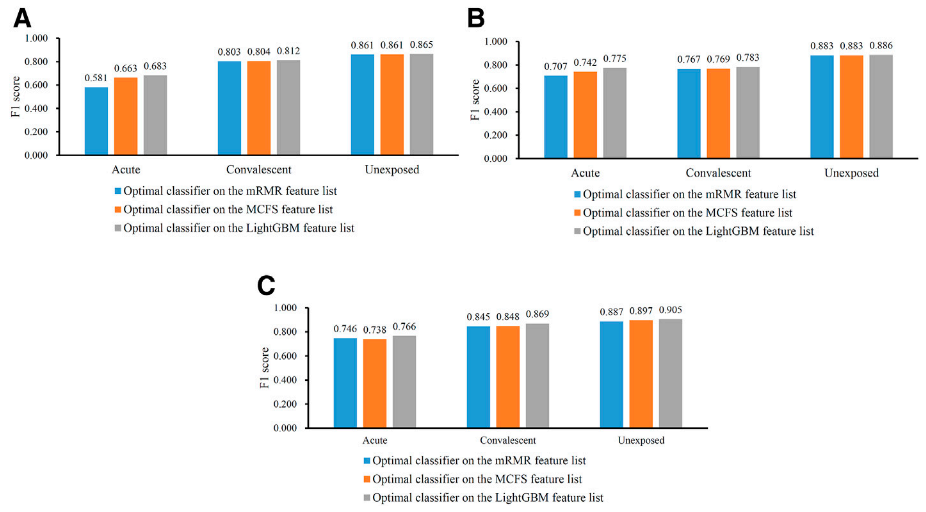


FIGURE 5
Performance of the optimal classifiers for three CD8$^+$ T cells subtypes on three categories. **(A)** effector T cells, **(B)** memory T cells, **(C)** naïve T cells. The optimal classifiers on the LightGBM feature lists were better than those on other two feature lists.

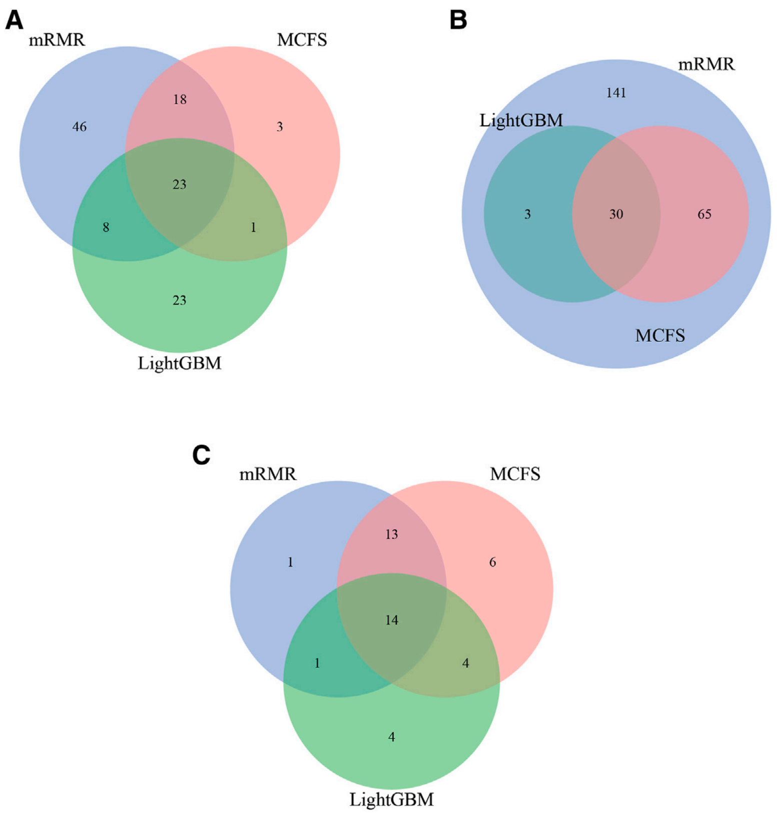


FIGURE 6
Intersection results of the optimal feature sets based on different feature lists yielded by three feature ranking methods for three CD8$^+$ T cells subtypes. **(A)** effector T cells, **(B)** memory T cells, **(C)** naïve T cells.

TABLE 3 Details of the optimal DT classifiers obtained for each cell subtype under different feature ranking methods and the number of rules extracted.

Cell subtype	Feature ranking method	Number of features	Weighted F1	Number of rules
Effector T cells	mRMR	249	0.731	5394
	MCFS	240	0.731	5412
	LightGBM	98	0.741	5810
Memory T cells	mRMR	155	0.736	6371
	MCFS	107	0.740	6404
	LightGBM	33	0.753	6959
Naïve T cells	mRMR	61	0.786	3930
	MCFS	44	0.786	4045
	LightGBM	33	0.797	3931

3 Results

In this study, we first downloaded COVID-19 expression profiles for three CD8$^+$ T cells subtypes, including effector, memory, and naïve T cells from GEO. Irrelevant features in the dataset on each CD8$^+$ T cells subtype were excluded using Boruta, and the retained features were ranked by using mRMR, MCFS, and LightGBM in three feature ranking lists. These feature lists were then used to identify the optimal features and extract classification rules using the IFS method. The entire computational framework is shown in Figure 1.

3.1 Results of feature selection on CD8$^+$ T cells expression profiles

For the expression profiles on each CD8$^+$ T cells subtype, the Boruta method was first adopted to remove irrelevant features. 252 features remained for effector T cells. For memory and naïve T cells, 241 and 153 features were kept, respectively. These selected features were analyzed by mRMR, MCFS, and LightGBM, respectively, resulting in three feature lists for each CD8$^+$ T cells subtype. These lists are provided in Supplementary Tables S1–S3.

3.2 Recognition of key features to distinguish COVID-19 severity on CD8$^+$ T cells with the IFS method

Through the above step, three feature lists (mRMR, MCFS and LightGBM feature lists) were obtained for each CD8$^+$ T cells subtype. However, important features for the classification task are still difficult to determine. Therefore, the IFS method was used to find the optimal features and construct the optimal classifiers, which constructed a series of classifiers and calculated their performance metrics. The IFS results for the three CD8$^+$ T cells subtypes using different feature lists are provided in Supplementary Tables S4–S6. The IFS curves were plotted to observe the trend of the classifiers' performance, measured by weighted F1, under the changing of feature numbers, as shown in Figures 2–4.

For the IFS results on the mRMR feature lists of three CD8$^+$ T cells subtypes (Supplementary Table S4), the IFS curves are shown in Figure 2. For the effector T cells, DT, kNN and RF reached the highest performance when the first 249, 14 and 95 features were used with weighted F1 values of 0.731, 0.746 and 0.806 (Figure 2A). For the memory T cells, three classification algorithms yielded the maximum weighted F1 values of 0.736, 0.789 and 0.818 when first 155, 21 and 239 features were adopted (Figure 2B). As for the naïve T cells, the highest weighted F1 values for three classification algorithms were 0.786, 0.845, and 0.842 (Figure 2C), which were obtained by using top 61, 29 and 25 features in the list. Clearly, for effector and memory T cells, RF provided better performance than DT and kNN, whereas kNN was best for the naïve T cells. Accordingly, we can construct the optimal RF classifiers for effector and memory T cells, and the optimal kNN classifier for the naïve T cells based on the mRMR feature lists. The overall performance of the above optimal classifiers, measured by ACC, MCC and macro F1, is listed in Table 2. ACC and MCC values were all no less than 0.8 and 0.67, respectively, indicating the good performance of these classifiers.

Of the IFS results on the MCFS feature lists of three CD8$^+$ T cells subtypes (Supplementary Table S5), Figure 3 shows the IFS curves. For the effector T cells, RF achieved the highest weighted F1 of 0.815 using the first 45 features (Figure 3A). Other two classification algorithms provided the highest weighted F1 values of 0.731 and 0.760 when top 240 and 40 features were adopted. For the memory T cells, the IFS curves of three classification algorithms reached the highest points with the top 107, 26 and 95 features with weighted F1 values of 0.740, 0.799 and 0.823 (Figure 3B). For the naïve T cells, kNN obtained the highest weighted F1 of 0.849 using the first

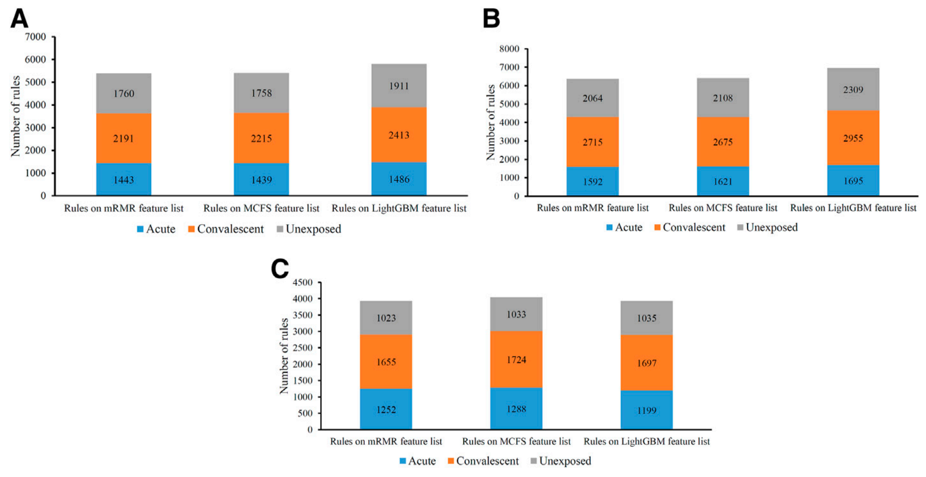


FIGURE 7
Distribution of rules yielded by decision trees on three categories in three CD8$^+$ T cells subtypes. **(A)** effector T cells, **(B)** memory T cells, **(C)** naïve T cells.

37 features (Figure 3C). DT and RF yielded the highest weighted F1 values of 0.786 and 0.845 when top 44 and 28 features were used. It was interesting that the performance of three classification algorithms on the MCFS feature lists was similar to that on the mRMR feature lists. RF was best on effector and memory T cells, whereas kNN was best on the naïve T cells. Likewise, three optimal classifiers can be built on three CD8$^+$ T cells subtypes based on the MCFS feature lists. Their detailed overall performance is also listed in Table 2. ACC and MCC values were all higher than 0.81 and 0.68, respectively, suggesting high performance of these classifiers.

For the IFS results on the LightGBM feature lists of three CD8$^+$ T cells subtypes (Supplementary Table S6), IFS curves are illustrated in Figure 4. For the effector T cells, DT/kNN/RF achieved the maximum weighted F1 of 0.741/0.791/0.822 when the first 98/41/55 features were used (Figure 4A). For the memory T cells, DT/kNN/RF peaked at 33/29/33 features with a weighted F1 value of 0.753/0.833/0.833 (Figure 4B). For the naïve T cells, DT/kNN/RF gained the maximum weighted F1 value of 0.797/0.867/0.854 when the first 33/23/27 features were used (Figure 4C). It was surprising that RF was still better than DT and kNN on effector and memory T cells, and kNN was still better than DT and RF on the naïve T cells, similar to the results on mRMR and MCFS feature lists. This also increased the reliability of our results. Likewise, three optimal classifiers on three CD8$^+$ T cells subtypes can be set up based on the LightGBM feature lists. Table 2 lists the detailed overall performance of these classifiers. ACC and MCC values were all higher than 0.82 and 0.69, respectively, indicating their high performance.

In Table 2, the overall performance of nine optimal classifiers on different feature lists and cell subtypes is provided. We further extracted their performance on three categories (acute, convalescent and unexposed), measured by F1 score, which are shown in Figure 5. It can be observed that on each cell subtype, optimal classifier on the LightGBM feature list always provided the highest performance on all categories, generally followed by the optimal classifiers on the MCFS and mRMR feature lists. Such results also conformed to their overall performance (Table 2). Furthermore, all classifiers generally yielded best performance on unexposed individuals, followed by convalescent and acute COVID-19.

For each CD8$^+$ T cells subtype, three optimal classifiers were constructed based on three feature lists. The features used in these classifiers (i.e., optimal features) can be obtained, comprising three optimal feature sets. It is interesting to investigate the intersection of these three optimal feature sets using Venn diagrams. The Venn diagrams are provided in Figure 6. The detailed intersection results are shown in Supplementary Table S7. It can be observed that there were 23 important features in three optimal feature sets for effector T cells (Figure 5A). For the memory T cells, 30 important features were included in three optimal feature sets (Figure 5B). The three optimal feature subsets under the naïve T cells had 14 essential features intersected (Figure 5C). The biological mechanisms of these important feature genes are described in the Section 4.

3.3 Classification rules for important features in the CD8$^+$ T cells profiles

On each CD8$^+$ T cells subtype, DT always provided the lowest performance under a given feature list. The performance

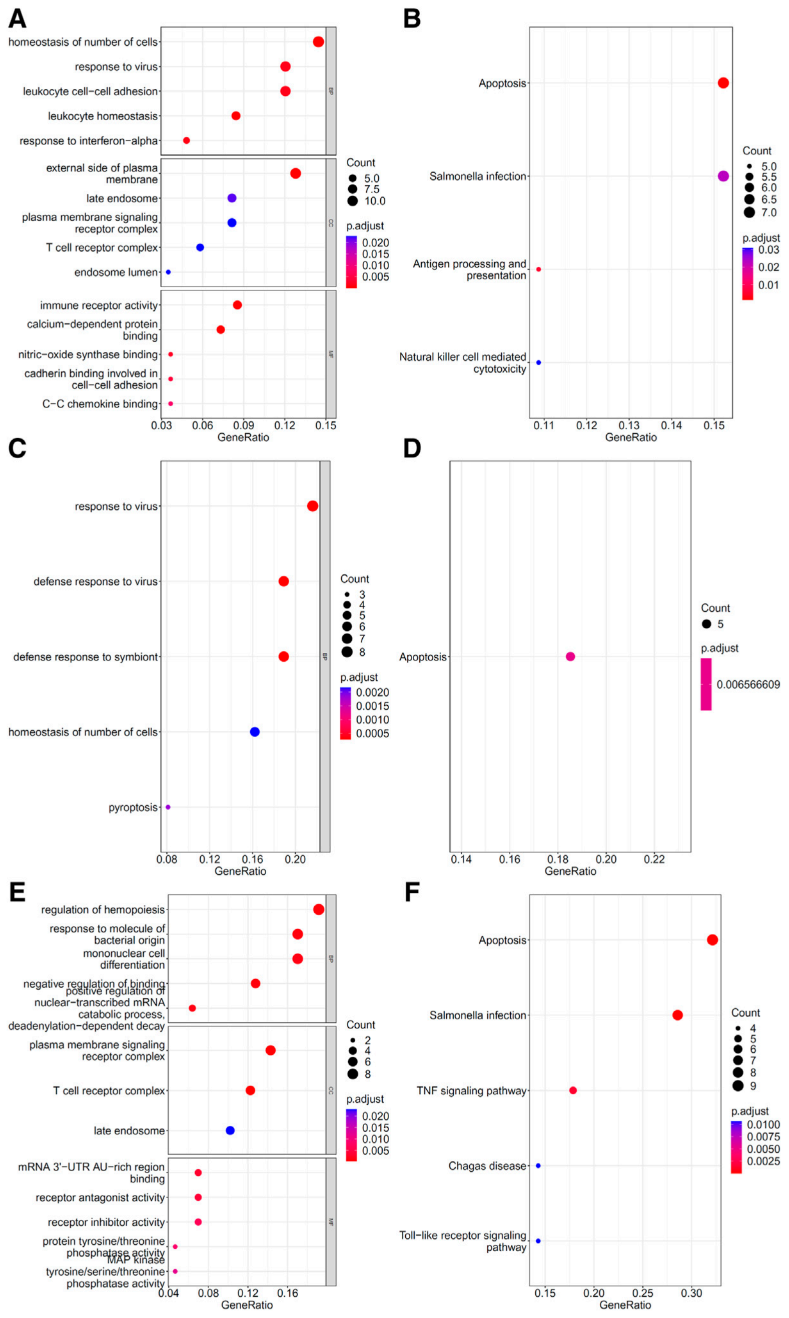


FIGURE 8
Results of the functional enrichment analysis on the optimal genes for different feature lists in the effector T cells. Top GO terms (**(A)**: mRMR, **(C)**: MCFS, and **(E)**: LightGBM) and KEGG pathways (**(B)**: mRMR, **(D)**: MCFS, **(F)**: LightGBM) are shown.

is listed in Table 3. However, it has the special merit that is not shared by kNN and RF. From the constructed DT, several classification rules can be obtained, which implies the special patterns on each category. Thus, we further employed DT to investigate profiles on three CD8$^+$ T cells subtypes. As mentioned in Section 3.2, the optimal features for DT can be found by executing IFS method on different feature lists of three CD8$^+$ T cells subtypes. With these optimal features, DT was applied on all samples to learn a large tree, from which a group of classification rules were obtained. These classification rules on three CD8$^+$ T cells subtypes and three feature lists are shown in Supplementary Table S8. The number of rules on each CD8$^+$

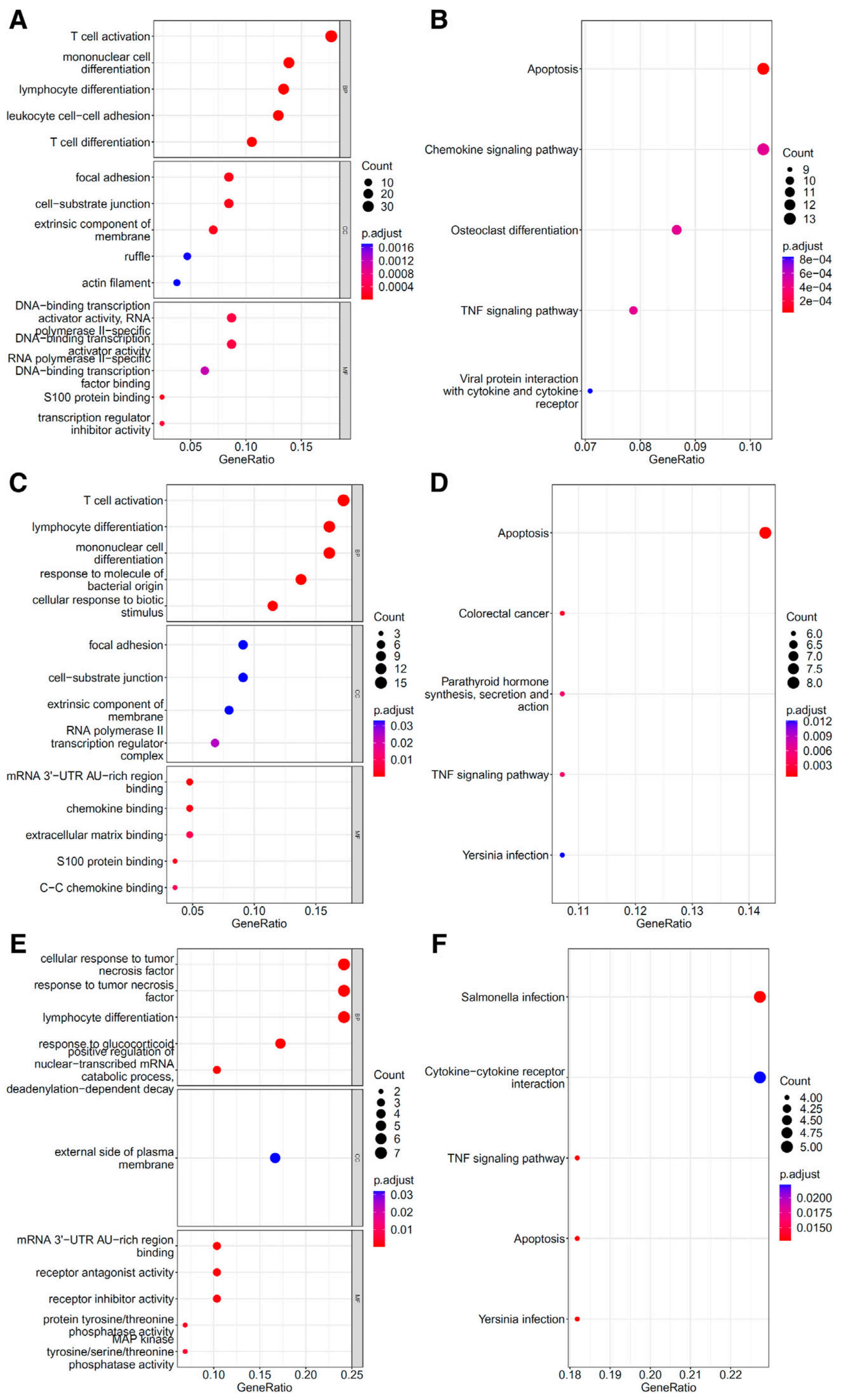


FIGURE 9
Results of the functional enrichment analysis on the optimal genes for the different feature lists in the memory T cells. Top GO terms (**(A)**: mRMR, **(C)**: MCFS, and **(E)**: LightGBM) and KEGG pathways (**(B)**: mRMR, **(D)**: MCFS, **(F)**: lightGBM) are shown.

T cells subtype and feature list is listed in Table 3. A fair number of rules were obtained, which provides informative reference for revealing the relationships between expression patterns of key feature genes and three categories. For each rule set, some rules were for acute COVID-19, whereas others were for convalescent COVID-19 or unexposed individuals. The

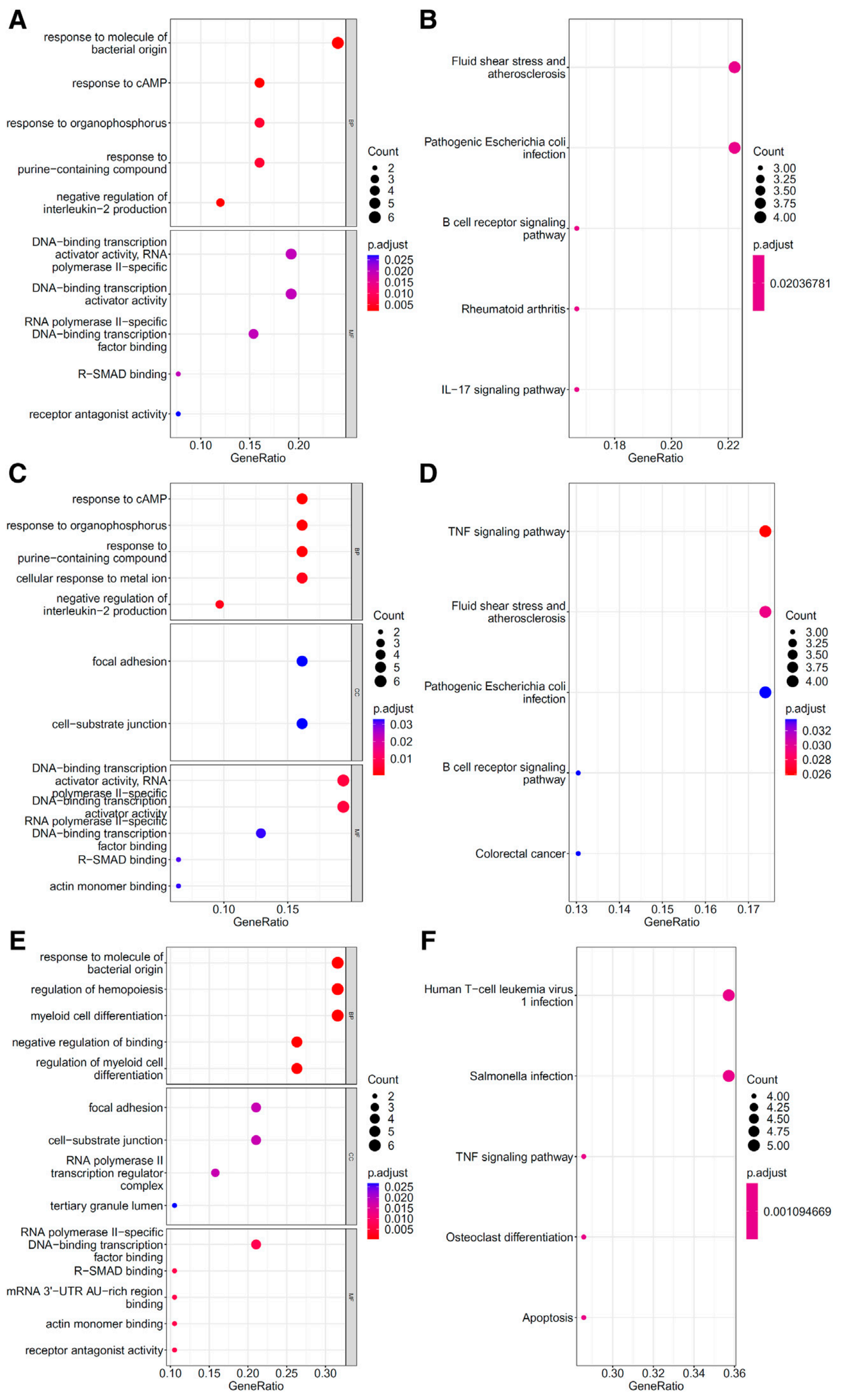


FIGURE 10
Results of the functional enrichment analysis on the optimal genes for the different feature lists in the naïve T cells. Top GO terms (**(A)**: mRMR, **(C)**: MCFS, and **(E)**: LightGBM) and KEGG pathways (**(B)**: mRMR, **(D)**: MCFS, **(F)**: lightGBM) are shown.

number of rules for each category is illustrated in Figure 7. It can be observed that convalescent COVID-19 was always assigned most rules, whereas the rules on unexposed individuals were the second most on effector and memory T cells, and acute COVID-19 was assigned the second most rules on Naïve T cells.

3.4 Immune functions for genes identified in the optimal feature sets

To explore the biological functions and pathways involved in the essential genes for each CD8$^+$ T cells subtype, we performed GO and KEGG enrichment analyses on the genes in the optimal feature sets obtained under each feature ranking list for each subtype of CD8$^+$ T cells. The results are provided in Supplementary Tables S9–S11. The top five GO terms and KEGG pathways from the enrichment results are shown in Figures 8–10. For the effector T cells, the main biological functions enriched are response to virus, homeostasis of number of cells, T cell receptor complex, and signaling pathways, including apoptosis and *salmonella* infection (Figure 8). For the memory T cells, the enrichment results contain T cell activation, lymphocyte differentiation, mononuclear cell differentiation, and signaling pathways, such as apoptosis and TNF signaling pathway (Figure 9). For the naïve T cells, the enrichment results were for GO terms, such as response to cAMP, response to organophosphorus, and signaling pathways, e.g., B-cell receptor signaling pathway and TNF signaling pathway (Figure 10). These critical biological functions and signaling pathways are developed in the Section 4.

4 Discussion

For each CD8$^+$ T cell subtype, we obtained three sets of features that are important to distinguish the disease state of patients with COVID-19 through three feature ranking algorithms and IFS method. Next, we conducted GO and KEGG enrichment analyses for all the genes in the three groups of features to facilitate our interpretation of these key genes. We discussed genes to confirm their important roles in COVID-19 according to existing studies. The main discussion results of each cell subtype were organized as follows.

4.1 Functional analysis of the key features of CD8$^+$ effector T cells

The effector CD8$^+$ T cells that respond to antigen stimulation proliferate and differentiate. Some will eventually differentiate into memory CD8$^+$ T cells. Among the features that can distinguish CD8$^+$ Effector T cells in different stages of COVID-19 infection, we found that they mainly contain cytotoxic genes (GZMA, GZMK, and PRF1), T cell receptor (TCR)-related genes (TRBV4.2 and TRBV7.2), cytokine-related genes (IFITM2, IL7R, and IL32), and others. At the same time, our functional enrichment results also showed the relationship between these genes and immune killing, as follows: GO:0009615 (response to virus), GO:0042101 (T cell receptor complex), GO: 0140375 (immune receptor activity), and hsa04210 (apoptosis).

CD8$^+$ effector T cells are essential for adaptive immunity against COVID-19 virus infection, and the cytotoxic response intensity of CD8$^+$ Effector T cells also corresponds to different stages of antiviral immunity. Patients with COVID-19 infection have higher levels of GZMs and PRF1 than healthy controls; they also have characteristic expression changes during infection recovery (Wen et al., 2020; Westmeier et al., 2020). During viral infection, TCRs recombine to generate a functional and highly diverse TCR repertoire crucial for CD8 effector T cells to identify and kill infected cells (Luo et al., 2021). Therefore, the dynamic changes of genes, such as TRBV4.2, TRBV7.2 (TCR components), and IL7R (associated with V(D)J recombination), may be related to different periods of infection. Other genes, such as FITM2, reportedly restrict the entry of COVID-19 virus into cells (Winstone et al., 2021), but its expression in CD8$^+$ Effector T cells and its dynamic characteristics at different stages of infection have not been studied.

4.2 Functional analysis of the key features of CD8$^+$ memory T cells

There are two types of CD8$^+$ memory T cells, namely, effector memory T cells (Tem) and central memory T cells (Tcm). CD8$^+$ Tcm mainly reside in secondary lymphoid organs and can rapidly be converted into effector cells upon antigen stimulation, whereas CD8$^+$ Tem is mainly distributed in peripheral tissues and can respond rapidly to stimulation by producing effector cytokines. In different stages of infection, CD8$^+$ Memory T cells have different activation, proliferation, and secretion states (Tavukcuoglu et al., 2021). GO/KEGG enrichment analysis for the features in our results also revealed that many genes were associated with T cell differentiation and effector activity, such as: GO:0042110 (T cell activation), GO:0030217 (T cell differentiation), hsa04062 (chemokine signaling pathway), and hsa04061 (viral protein interaction with cytokine and cytokine receptor). In these genes, features associated with cellular activation and differentiation (B2M, IL7R, ZFP36, ZFP36L1, ZFP36L2, CD8A, KLF6, and LGALS1) may be related to the function of CD8$^+$ memory T cells at different stages of infection, whereas features associated with cell chemotaxis (SELL, CCL5, CXCR4, and NFKBIA) could be linked to the recruitment of CD8$^+$ memory cells (Xiong et al., 2020). The COVID-19 virus may also escape the immune system through chemokines (Khalil et al., 2021), suggesting that the expression of chemokines may be associated with the different stages of infection.

4.3 Functional analysis of the key features of CD8$^+$ naïve T cells

The GO enrichment results of key genes show that they are related to the response to multiple stimuli, such as: GO:0071216 (cellular response to biotic stimulus) and GO:0051591 (response to cAMP). cAMP has been shown to play an important role in the initial activation and effector differentiation of naïve CD8$^+$ T cells (Linnemann et al., 2009). At different stages of infection in

COVID-19 patients, naïve CD8$^+$ T cells responded to different levels of antigenic stimulation (Wen et al., 2020; Fenoglio et al., 2021), which resulted in different proportions and activation states of naïve CD8$^+$ T cells; this phenomenon may help distinguish different disease states. In addition, a correlation was found between the proportion of naive CD8$^+$ T cells and infection severity (Moderbacher et al., 2020).

At the gene expression level, some genes showed important roles in differentiating infection stages and were identified by all three feature ranking algorithms. Among these genes, the protein product of the ZFP36 gene belongs to the zinc finger family and has been linked to the regulation of gene expression and cellular response to growth factor stimulation. Studies on COVID-19 showed that ZFP36 inhibited T cell activation, and proliferation during viral infection and the expression level of ZFP36 changed dramatically during infection (Xiong et al., 2020). DUSP1 was downregulated in COVID-19 infection and may be associated with enhanced MAPK pathway activation and steroid resistance (Sharif-Askari et al., 2021). Our features also contained some inflammatory genes (FOS, JUN, and KLF6), which may be related to the inflammatory state at different disease stages; these genes have different expression levels during COVID-19 infection and recovery (Wen et al., 2020).

5 Conclusion

In this study, the single-cell RNA-Seq datasets under three subtypes of CD8$^+$ T cells (effector, memory, and naïve T cells) related to COVID-19 infection, convalescent, and unexposed were deeply investigated. Several advanced computational methods were applied on these datasets. Essential genes, interpretable classification rules and efficient classifiers were obtained. The former two results can deepen our understanding on the mechanism of the regulatory role of CD8$^+$ T cells on COVID-19. The last one can be useful tools to distinguish patients' COVID-19 severity in terms of CD8$^+$ T cells.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found here: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE188429.

Author contributions

ZZ, TH and Y-DC designed the study. JL, SD and KF performed the experiments. MM and XZ analyzed the results. JL and MM wrote the manuscript. All authors contributed to the research and reviewed the manuscript.

Funding

This work was supported by the Shanghai Municipal Science and Technology Major Project (2017SHZDZX01), Strategic Priority Research Program of Chinese Academy of Sciences [XDB38050200, XDA26040304], the Fund of the Key Laboratory of Tissue Microenvironment and Tumor of Chinese Academy of Sciences [202002].

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

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2022.1053772/full#supplementary-material

References

Araf, Y., Akter, F., Tang, Y. D., Fatemi, R., Parvez, M. S. A., Zheng, C., et al. (2022). Omicron variant of SARS-CoV-2: Genomics, transmissibility, and responses to current COVID-19 vaccines. *J. Med. Virol.* 94, 1825–1832. doi:10.1002/jmv.27588

Breiman, L. (2001). Random forests. *Mach. Learn.* 45, 5–32. doi:10.1023/a:1010933404324

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* 16, 321–357. doi:10.1613/jair.953

Chen, G., Wu, D., Guo, W., Cao, Y., Huang, D., Wang, H., et al. (2020). Clinical and immunological features of severe and moderate coronavirus disease 2019. *J. Clin. Invest.* 130, 2620–2629. doi:10.1172/JCI137244

Chen, W., Chen, L., and Dai, Q. (2021). iMPT-FDNPL: identification of membrane protein types with functional domains and a natural language processing approach. *Comput. Math. Methods Med.* 2021, 7681497. doi:10.1155/2021/7681497

Chen, L., Li, Z., Zhang, S., Zhang, Y.-H., Huang, T., and Cai, Y.-D. (2022). Predicting RNA 5-methylcytosine sites by using essential sequence features and distributions. *Biomed. Res. Int.* 2022, 4035462. doi:10.1155/2022/4035462

Cover, T., and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* 13, 21–27. doi:10.1109/tit.1967.1053964

Dan, J. M., Mateus, J., Kato, Y., Hastie, K. M., Yu, E. D., Faliti, C. E., et al. (2021). Immunological memory to SARS-CoV-2 assessed for up to 8 months after infection. *Science* 371, eabf4063. doi:10.1126/science.abf4063

Ding, S., Wang, D., Zhou, X., Chen, L., Feng, K., Xu, X., et al. (2022). Predicting heart cell types by using transcriptome profiles and a machine learning method. *Life* 12, 228. doi:10.3390/life12020228

Draminski, M., Rada-Iglesias, A., Enroth, S., Wadelius, C., Koronacki, J., and Komorowski, J. (2008). Monte Carlo feature selection for supervised classification. *Bioinformatics* 24, 110–117. doi:10.1093/bioinformatics/btm486

Fenoglio, D., Dentone, C., Parodi, A., Di Biagio, A., Bozzano, F., Vena, A., et al. (2021). Characterization of T lymphocytes in severe COVID-19 patients. *J. Med. Virol.* 93, 5608–5613. doi:10.1002/jmv.27037

Fiolet, T., Kherabi, Y., Macdonald, C. J., Ghosn, J., and Peiffer-Smadja, N. (2022). Comparing COVID-19 vaccines for their characteristics, efficacy and effectiveness against SARS-CoV-2 and variants of concern: a narrative review. *Clin. Microbiol. Infect.* 28, 202–221. doi:10.1016/j.cmi.2021.10.005

Francis, J. M., Leistritz-Edwards, D., Dunn, A., Tarr, C., Lehman, J., Dempsey, C., et al. (2022). Allelic variation in class I HLA determines CD8(+) T cell repertoire shape and cross-reactive memory responses to SARS-CoV-2. *Sci. Immunol.* 7, eabk3070. doi:10.1126/sciimmunol.abk3070

Gong, J., Zhan, H., Liang, Y., He, Q., and Cui, D. (2021). Role of Th22 cells in human viral diseases. *Front. Med.* 8, 708140. doi:10.3389/fmed.2021.708140

Grifoni, A., Weiskopf, D., Ramirez, S. I., Mateus, J., Dan, J. M., Moderbacher, C. R., et al. (2020). Targets of T Cell responses to SARS-CoV-2 coronavirus in humans with COVID-19 disease and unexposed individuals. *Cell* 181, 1489–1501. doi:10.1016/j.cell.2020.05.015

Jurman, G., Riccadonna, S., and Furlanello, C. (2012). A comparison of MCC and CEN error measures in multi-class prediction. *PLoS ONE* 7, e41882. doi:10.1371/journal.pone.0041882

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., et al. (2017). "LightGBM: A highly efficient gradient boosting decision tree," in 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, December 4–9, 2017

Khalil, B. A., Elemam, N. M., and Maghazachi, A. A. (2021). Chemokines and chemokine receptors during COVID-19 infection. *Comput. Struct. Biotechnol. J.* 19, 976–988. doi:10.1016/j.csbj.2021.01.034

Kohavi, R. (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection," in International joint Conference on artificial intelligence, Montreal, QC, August 20–25, 1995 (Lawrence Erlbaum Associates), 1137–1145.

Kotturi, M. F., Peters, B., Buendia-Laysa, F., Jr., Sidney, J., Oseroff, C., Botten, J., et al. (2007). The CD8+ T-cell response to lymphocytic choriomeningitis virus involves the L antigen: uncovering new tricks for an old virus. *J. Virol.* 81, 4928–4940. doi:10.1128/JVI.02632-06

Kursa, M. B., and Rudnicki, W. R. (2010). Feature selection with the Boruta package. *J. Stat. Softw.* 36, 1–13. doi:10.18637/jss.v036.i11

Li, X., Lu, L., and Chen, L. (2022). Identification of protein functions in mouse with a label space partition method. *Math. Biosci. Eng.* 19, 3820–3842. doi:10.3934/mbe.2022176

Linnemann, C., Schildberg, F. A., Schurich, A., Diehl, L., Hegenbarth, S. I., Endl, E., et al. (2009). Adenosine regulates CD8 T-cell priming by inhibition of membrane-proximal T-cell receptor signalling. *Immunology* 128, e728–e737. doi:10.1111/j.1365-2567.2009.03075.x

Liu, H. A., and Setiono, R. (1998). Incremental feature selection. *Appl. Intell.* 9, 217–230. doi:10.1023/a:1008363719778

Liu, H., Hu, B., Chen, L., and Lu, L. (2021). Identifying protein subcellular location with embedding features learned from networks. *Curr. Proteomics* 18, 646–660. doi:10.2174/15701646mtex2nzc51

Luo, L., Liang, W., Pang, J., Xu, G., Chen, Y., Guo, X., et al. (2021). Dynamics of TCR repertoire and T cell function in COVID-19 convalescent individuals. *Cell Discov.* 7, 89. doi:10.1038/s41421-021-00321-x

Matthews, B. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim. Biophys. Acta* 405, 442–451. doi:10.1016/0005-2795(75)90109-9

Moderbacher, C. R., Ramirez, S. I., Dan, J. M., Grifoni, A., Hastie, K. M., Weiskopf, D., et al. (2020). Antigen-specific adaptive immunity to SARS-CoV-2 in acute COVID-19 and associations with age and disease severity. *Cell* 183, 996–1012. doi:10.1016/j.cell.2020.09.038

Nguyen, Q. P., Deng, T. Z., Witherden, D. A., and Goldrath, A. W. (2019). Origins of CD4(+) circulating and tissue-resident memory T-cells. *Immunology* 157, 3–12. doi:10.1111/imm.13059

Pan, X., Chen, L., Liu, I., Niu, Z., Huang, T., and Cai, Y. D. (2022). Identifying protein subcellular locations with embeddings-based node2loc. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 19, 666–675. doi:10.1109/TCBB.2021.3080386

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 1226–1238. doi:10.1109/TPAMI.2005.159

Ran, B., Chen, L., Li, M., Han, Y., and Dai, Q. (2022). Drug-Drug interactions prediction using fingerprint only. *Comput. Math. Methods Med.* 2022, 7818480. doi:10.1155/2022/7818480

Safavian, S. R., and Landgrebe, D. (1991). A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man. Cybern.* 21, 660–674. doi:10.1109/21.97458

Samudrala, P. K., Kumar, P., Choudhary, K., Thakur, N., Wadekar, G. S., Dayaramani, R., et al. (2020). Virology, pathogenesis, diagnosis and in-line treatment of COVID-19. *Eur. J. Pharmacol.* 883, 173375. doi:10.1016/j.ejphar.2020.173375

Sanyal, S. (2020). How SARS-CoV-2 (COVID-19) spreads within infected hosts - what we know so far. *Emerg. Top. Life Sci.* 4, 371–378. doi:10.1042/ETLS20200165

Sharif-Askari, F. S., Sharif-Askari, N. S., Goel, S., Hafezi, S., Assiri, R., Al-Muhsen, S., et al. (2021). SARS-CoV-2 attenuates corticosteroid sensitivity by suppressing DUSP1 expression and activating p38 MAPK pathway. *Eur. J. Pharmacol.* 908, 174374. doi:10.1016/j.ejphar.2021.174374

Slifka, M. K., and Whitton, J. L. (2000). Antigen-specific regulation of T cell-mediated cytokine production. *Immunity* 12, 451–457. doi:10.1016/s1074-7613(00)80197-1

Tang, S., and Chen, L. (2022). iATC-NFMLP: Identifying classes of anatomical therapeutic chemicals based on drug networks, fingerprints and multilayer perceptron. *Curr. Bioinform.* 17, 814–824. doi:10.2174/1574893617666220318093000

Tavukcuoglu, E., Horzum, U., Inkaya, A. C., Unal, S., and Esendagli, G. (2021). Functional responsiveness of memory T cells from COVID-19 patients. *Cell. Immunol.* 365, 104363. doi:10.1016/j.cellimm.2021.104363

Wang, R., and Chen, L. (2022). Identification of human protein subcellular location with multiple networks. *Curr. Proteomics* 19, 344–356. doi:10.2174/1570164619666220531113704

Wen, W., Su, W., Tang, H., Le, W., Zhang, X., Zheng, Y., et al. (2020). Erratum: Author Correction: Immune cell profiling of COVID-19 patients in the recovery stage by single-cell sequencing. *Cell Discov.* 6, 41. doi:10.1038/s41421-020-00187-5

Westmeier, J., Paniskaki, K., Karaköse, Z., Werner, T., Sutter, K., Dolff, S., et al (2020). Impaired cytotoxic CD8+ T cell response in elderly COVID-19 patients. *MBio* 11, e02243–e02220. doi:10.1128/mBio.02243-20

Winstone, H., Lista, M. J., Reid, A. C., Bouton, C., Pickering, S., Galao, R. P., et al. (2021). The polybasic cleavage site in SARS-CoV-2 spike modulates viral sensitivity to type I interferon and IFITM2. *J. Virol.* 95, e02422–e02420. doi:10.1128/JVI.02422-20

Wu, Z., and Chen, L. (2022). Similarity-based method with multiple-feature sampling for predicting drug side effects. *Comput. Math. Methods Med.* 2022, 9547317. doi:10.1155/2022/9547317

Wu, C., and Chen, L. (2023). A model with deep analysis on a large drug network for drug classification. *Math. Biosci. Eng.* 20, 383–401. doi:10.3934/mbe.2023018

Wu, T., Hu, E., Xu, S., Chen, M., Guo, P., Dai, Z., et al. (2021). clusterProfiler 4.0: A universal enrichment tool for interpreting omics data. *Innovation.* 2, 100141. doi:10.1016/j.xinn.2021.100141

Xiong, Q., Peng, C., Yan, X., Yan, X., Chen, L., Sun, B., et al. (2020). Characteristics of SARS-CoV-2-specific cytotoxic T cells revealed by single-cell immune profiling of longitudinal COVID-19 blood samples. *Signal Transduct. Target. Ther.* 5, 285. doi:10.1038/s41392-020-00425-y

Yang, Y., and Chen, L. (2022). Identification of drug–disease associations by using multiple drug and disease networks. *Curr. Bioinform.* 17, 48–59. doi:10.2174/1574893616666210825115406

Zheng, H. Y., Zhang, M., Yang, C. X., Zhang, N., Wang, X. C., Yang, X. P., et al. (2020). Elevated exhaustion levels and reduced functional diversity of T cells in peripheral blood may predict severe progression in COVID-19 patients. *Cell. Mol. Immunol.* 17, 541–543. doi:10.1038/s41423-020-0401-3

Zhou, X., Ding, S., Wang, D., Chen, L., Feng, K., Huang, T., et al. (2022). Identification of cell markers and their expression patterns in skin based on single-cell RNA-sequencing profiles. *Life* 12, 550. doi:10.3390/life12040550

TYPE Original Research
PUBLISHED 17 November 2022
DOI 10.3389/fgene.2022.1069558

Antimicrobial Peptides Prediction method based on sequence multidimensional feature embedding

Benzhi Dong[1], Mengna Li[1], Bei Jiang[2], Bo Gao[3], Dan Li[1]* and Tianjiao Zhang[1]*

[1]College of Information and Computer Engineering, Northeast Forestry University, Harbin, China, [2]Tianjin Second People's Hospital, Tianjin Institute of Hepatology, Tianjin, China, [3]Department of Radiology, The Second Affiliated Hospital of Harbin Medical University, Harbin, China

Check for updates

OPEN ACCESS

EDITED BY
Quan Zou,
University of Electronic Science and Technology of China, China

REVIEWED BY
Liang Yu,
Xidian University, China
Leyi Wei,
Shandong University, China

*CORRESPONDENCE
Dan Li,
ld725725@126.com
Tianjiao Zhang,
tianjiaozhang@nefu.edu.cn

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 14 October 2022
ACCEPTED 02 November 2022
PUBLISHED 17 November 2022

CITATION
Dong B, Li M, Jiang B, Gao B, Li D and Zhang T (2022), Antimicrobial Peptides Prediction method based on sequence multidimensional feature embedding.
Front. Genet. 13:1069558.
doi: 10.3389/fgene.2022.1069558

Antimicrobial peptides (AMPs) are alkaline substances with efficient bactericidal activity produced in living organisms. As the best substitute for antibiotics, they have been paid more and more attention in scientific research and clinical application. AMPs can be produced from almost all organisms and are capable of killing a wide variety of pathogenic microorganisms. In addition to being antibacterial, natural AMPs have many other therapeutically important activities, such as wound healing, antioxidant and immunomodulatory effects. To discover new AMPs, the use of wet experimental methods is expensive and difficult, and bioinformatics technology can effectively solve this problem. Recently, some deep learning methods have been applied to the prediction of AMPs and achieved good results. To further improve the prediction accuracy of AMPs, this paper designs a new deep learning method based on sequence multidimensional representation. By encoding and embedding sequence features, and then inputting the model to identify AMPs, high-precision classification of AMPs and Non-AMPs with lengths of 10–200 is achieved. The results show that our method improved accuracy by 1.05% compared to the most advanced model in independent data validation without decreasing other indicators.

KEYWORDS
deep learning, feature encoding, feature embedding, N-gram encoding, antimicrobial peptides

1 Introduction

Antimicrobial peptides (AMPs) are host defense molecules produced by the innate immune system in a variety of organisms and have many advantages, such as rapid killing, low toxicity, and broad activity (Fjell et al., 2009), and their drug resistance is relatively low. About 50% of the amino acids in AMP are hydrophobic, and they can adopt an amphiphilic structure, which enables them to interact with and penetrate cell membranes, which then lead to disruption of membrane potential, changes in membrane permeability, and permeation of metabolites leakage, eventually leading to bacterial cell death (Kumar

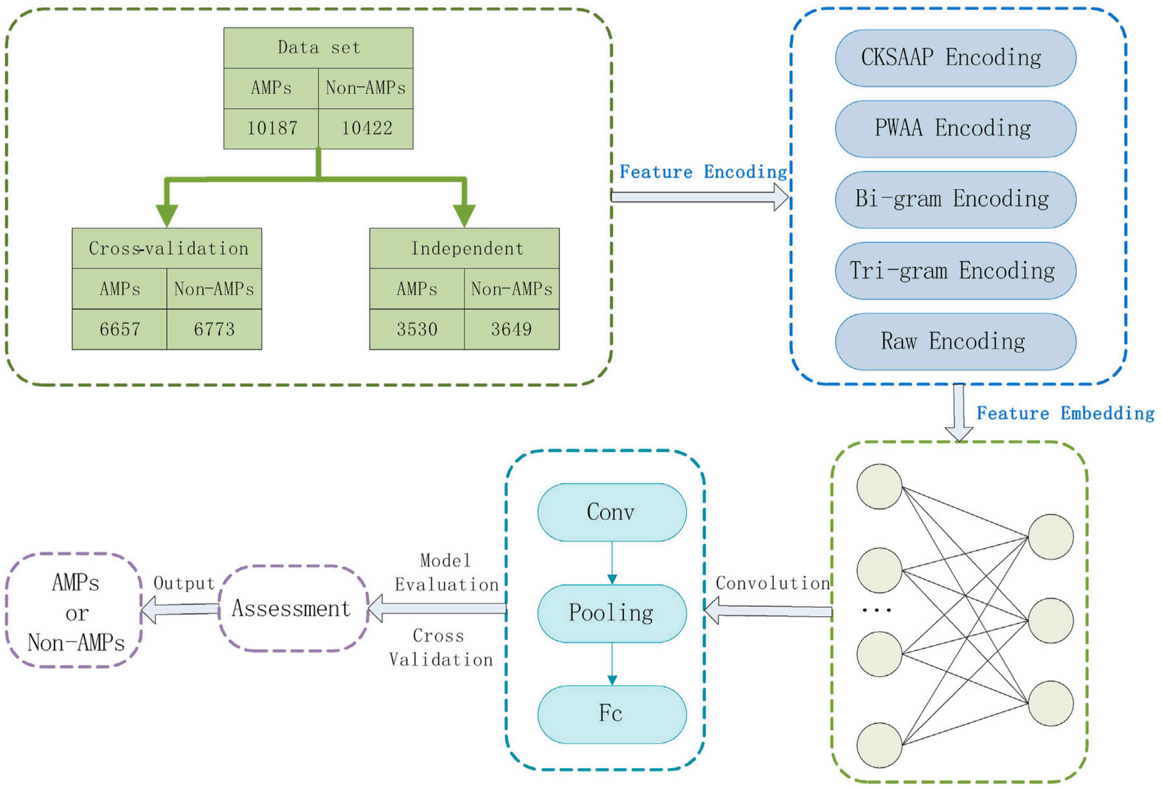


FIGURE 1
The workflow.

et al., 2018). AMPs not only exhibit synergy with antibiotics, but may also synergize with the immune system (Pasupuleti et al., 2012). At present, there are corresponding drug-resistant pathogenic strains of conventional antibiotics, and the drug-resistant problem of pathogenic bacteria has increasingly threatened people's health. Finding new antibiotics is an effective way to solve the drug-resistant problem. The characteristics of high antibacterial activity, broad antibacterial spectrum, and wide selection range are considered to be an effective way to solve the problem of drug resistance (Hancock and Sahl, 2006). Given the multiple advantages of AMPs, there is an urgent need to identify new AMPs.

In recent years, the rapid development of bioinformatics has provided a rational design method for the acquisition of AMPs. We can predict AMPs based on their sequence information. At present, the research on sequence classification algorithms mainly focuses on the combination of classification algorithms and biological sequence features. Various applied machine learning models have also been applied in AMPs prediction, for example, support vector machines (SVM) (Lata et al., 2010; Meher et al., 2017; Agrawal et al., 2018; Gong et al., 2021; Zou et al., 2021; Zhang Q. et al., 2022), random forest (RF) (Bhadra et al., 2018; Veltri, 2015; Nakayama et al., 2021; Yang et al., 2021; Ao et al., 2022; Lv et al., 2022a), discriminant analysis (DA) (Thomas et al., 2010; Waghu et al., 2016), Hidden Markov (Fjell et al., 2009), k-nearest neighbors (Xiao et al., 2013), *etc.* The core problem of such methods is how to perform feature extraction on protein sequences, which is greatly affected by the feature extraction method, which limits the maximum performance of the model. In addition, artificial feature engineering is often required when machine learning builds a classification model. In this process, important information is likely to be lost. Deep learning methods that have developed rapidly in recent years can effectively solve this problem.

Deep learning methods can automatically learn features from the raw data through convolution operations, avoiding the loss of data features. Various deep learning methods have been applied in protein sequence classification, such as bidirectional long short-term memory network (Bi-LSTM) (Tng et al., 2021; Zhang Y. et al., 2022; Zhang et al., 2022c; Li et al., 2022; Qiao et al., 2022; Wang et al., 2022), two-dimensional convolutional neural network (2D CNN) (Le et al., 2021), deep residual network (ResNet) (Xu et al., 2021), graph convolutional network (GCN) (Chen et al., 2021), deep neural network (DNN) (Gao et al., 2019; Han et al., 2019; Le et al., 2019; Hathaway et al., 2021), and Recurrent Neural Network (RNN)

TABLE 1 Statistics for datasets.

	Total	Cross-validation	Independent
AMPs	10,187	6,657	3,530
Non-AMPs	10,422	6,773	3,649

(Zheng et al., 2020; Yun et al., 2021). These research methods have generally achieved good classification results and have attracted increasing attention. In the prediction of AMPs, deep learning methods have also received attention, such as deep neural network (DNN) (Veltri et al., 2018; Su et al., 2019; Fu et al., 2020; Yan et al., 2020), bidirectional long short-term memory network (Bi-LSTM) (Sharma et al., 2021a; Xiao et al., 2021; Sharma et al., 2022), and Transformer (Zhang et al., 2021). These models all demonstrate the superiority of deep learning in AMPs prediction.

Whether it is a machine learning method or a deep learning method, the first step of these methods is to represent protein sequences as machine-readable and to encode biological sequences with features, that is, to map biological sequences to digital sequences using digital signal processing methods. It is widely used in biological sequence classification. As an important biological sequence analysis method, biological sequence encoding has been studied by many scholars, for example, the interaction of protein sequences (Moretta et al., 2020; Khabbaz et al., 2021; Wani et al., 2021; Söylemez et al., 2022), sparse coding (binary coding) (Spänig and Heider, 2019; Akbar et al., 2021; Jain et al., 2021; Ren et al., 2022). In addition, pre-trained models in natural language processing (NLP) have been used in protein sequence analysis, for example, the word2vec method (Zhang et al., 2019; Dao et al., 2021) and the N-gram method (Li et al., 2018; Wu and Yu, 2021) showed excellent performance in prediction.

The AMPs classification methods are usually based on machine learning or deep learning consider the interaction between protein sequences and represents the sequences as a matrix, ignoring the upstream and downstream information of the sequences, and the prediction accuracy will be reduced during the classification process. In this paper, deep learning-based feature combinations of N-gram encoding, K-space amino acid pair composition (CKSAAP), position-weighted amino acid composition (PWAA), and raw sequence number encoding were selected to predict AMPs. The CKSAAP encoding effectively describes the short-range interactions between amino acids, the PWAA encoding determines the positional information of amino acids in the protein sequence, and considers the upstream and downstream information of the protein sequence, and the N-gram encoding enhances the expression of the protein sequence and reduces the training process. Information is lost. It not only considers the interaction and positional weight of amino acids in the protein sequence but also combines the upstream and downstream information in the sequence and enhances the expression of the AMPs sequence, avoiding the above problems and improving the prediction performance. To evaluate the model, we use a 10-fold cross-validation method. Figure 1 shows our workflow.

2 Materials and methods

2.1 Baseline datasets

In this study, we used the dataset of (Sharma et al., 2021b), which collected AMPs data belonging to 13 phyla and 41 kingdoms (animal kingdom) categories from NCBI and StarPepDB databases and obtained Non-AMPs data from the UniProt database. This dataset considers all AMPs of suitable

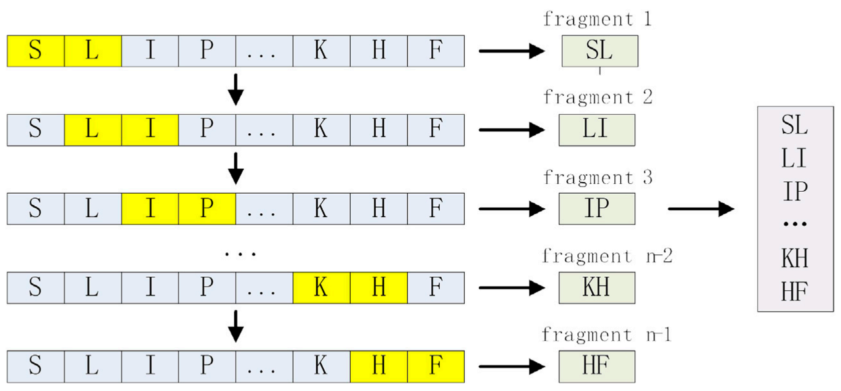


FIGURE 2
Bi-gram encoding process.

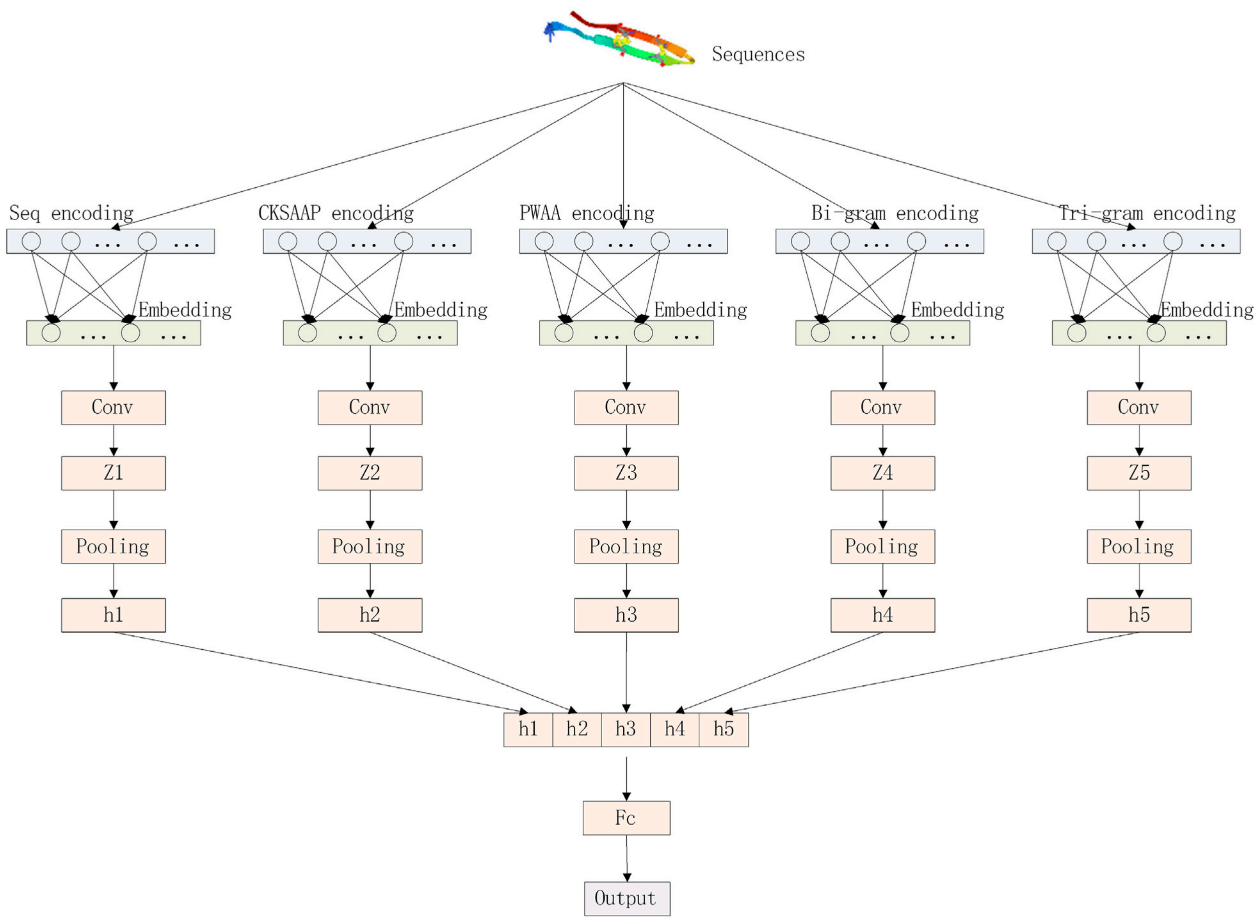


FIGURE 3
Model architecture diagram.

length in the animal kingdom to train the model. After the data is de-redundant, the dataset finally consists of 10,187 AMPs and 10,422 Non-AMPs, shown in supplementary materialthe, which contains about 65% of AMPs and non-AMPs. AMPs were used as the cross-validation dataset to train our model, and the rest contained about 35% of AMPs and non-AMPs as independent datasets for evaluating model performance, whose composition is shown in Table 1.

2.2 Encoding method of sequence

2.2.1 Raw sequence encoding

Protein is composed of 20 kinds of amino acids, each amino acid is represented by a character, and the sequences represented by these 20 kinds of characters contain important biological genetic information. The raw sequence encoding, that is, mapping the sequence to a set of numbers, reflects the selection bias of the AMPs sequence at each amino acid position. If given a protein sequence of length n, $S = (s_1, s_2, \ldots, s_n)$, where $s_i \in \{A, R, N, D, C, Q, E, G, H, I, L, K, M, F, P, S, T, W, Y, V\}$, $i = 1, 2, \ldots, n$, then the sequence S can be expressed as a one-dimensional vector of length n. For example, a protein sequence FLPKLFAKITKKNMAHIRC with a length of 19 can be used as a vector $[5, 10, 13, 9, 10, 5, 1, 9, 8, 17, 9, 9, 12, 11, 1, 7, 8, 15, 2]_{19}$. The maximum length of protein sequences in the dataset used in this paper is 200, so we set the sequence coding dimension to 200, and all sequences shorter than 200 are filled with 0 at the end.

2.2.2 Composition of k-space amino acid pairs (CKSAAP) encoding

CKSAAP is a coding scheme based on the interaction between amino acid pairs, which has been widely used in protein prediction (Yuan et al., 2022). CKSAAP can represent amino acids as a combination of multiple amino acid pairs with spacing K (Chen et al., 2011), reflecting the short-range

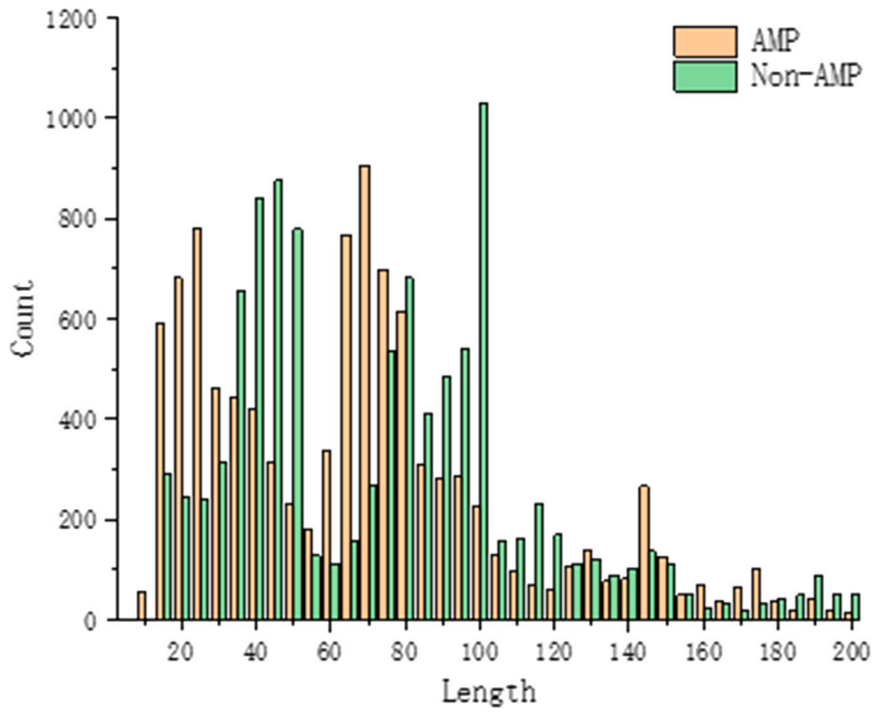


FIGURE 4
Benchmark dataset protein sequence length statistics.

interaction between amino acid pairs. If K = 0, there are 400 residue pairs with spacing 0 (AA, AC, AD, AE, . . ., YY). The eigenvector can be calculated by Eq. 1:

$$\left(\frac{N_{AA}}{N_{Total}}, \frac{N_{AC}}{N_{Total}}, \frac{N_{AD}}{N_{Total}}, \frac{N_{AE}}{N_{Total}}, \ldots, \frac{N_{YY}}{N_{Total}}\right)_{400} \tag{1}$$

Where, N_{Total} = L-K-1, N_{Total} represents the total number of residue pairs in the protein sequence, L represents the sequence length, and K represents the amino acid spacing. For example, when the sequence length is 200 and K = 0, 1, 2, 3, the values of N_{Total} are 199, 198, 197, and 196. In this paper, we take K as 0, 1, 2, 3, 4, and 5, so the total dimension of this feature is 2,400.

2.2.3 Position weighted amino acid composition (PWAA) encoding

To determine the position information of amino acids in the protein sequences, we used the PWAA method for encoding. Given amino acid residue a_i (i = 1, 2, 3,..., 20), we can calculate the positional information of a_i in a protein sequence by Eq. 2:

$$C_i = \frac{1}{L(L+1)} \sum_{j=-L}^{L} x_{i,j}\left(j + \frac{|j|}{L}\right) (j = -L, \ldots, q, \ldots, L) \tag{2}$$

Where L represents the data of upstream residue or downstream residue at the central site of the protein sequence fragment, if a_i is the residue at the *j*th position of the protein sequence fragment, then x (i, j) = 1, otherwise x (i, j) = 0. Generally, the closer a_i is to the center position (position 0), The smaller the absolute value of C_i. The PWAA encoding involves 20 kinds of amino acid residues, so this method encodes a dimension of 20.

2.2.4 N-gram encoding

N-gram is a statistical language model, which can be applied to protein sequence analysis to enhance the expression of protein sequences (Sharma and Srivastava, 2021). We treat each amino acid residue of a protein sequence of length L-N+1 as a word and each sequence as a sentence. In this study, our data length is short, and the Bi-gram (binary model) and tri-gram (ternary model) we used are enough to enhance the expression of AMPs sequences. For an raw sequence of length n S= (s_1, s_2, . . . s_n), Bi-gram can be expressed as S_2=(s_1s_2,s_2s_3, . . . ,$s_{(n-1)}s_n$), whose length is n-1, and the coding process is shown in Figure 2. Similarly, Tri-gram can be expressed as S_3=($s_1s_2s_3$,$s_2s_3s_4$, . . . ,$s_{(n-2)}s_{(n-1)}s_n$), whose length is n-2. To align the encoding length of the N-gram, we set the encoding length of the N-gram to 200, and the encodings shorter than 200 are padded with 0 at the end, so the dimensions of the Bi-gram and Tri-gram are 200 respectively.

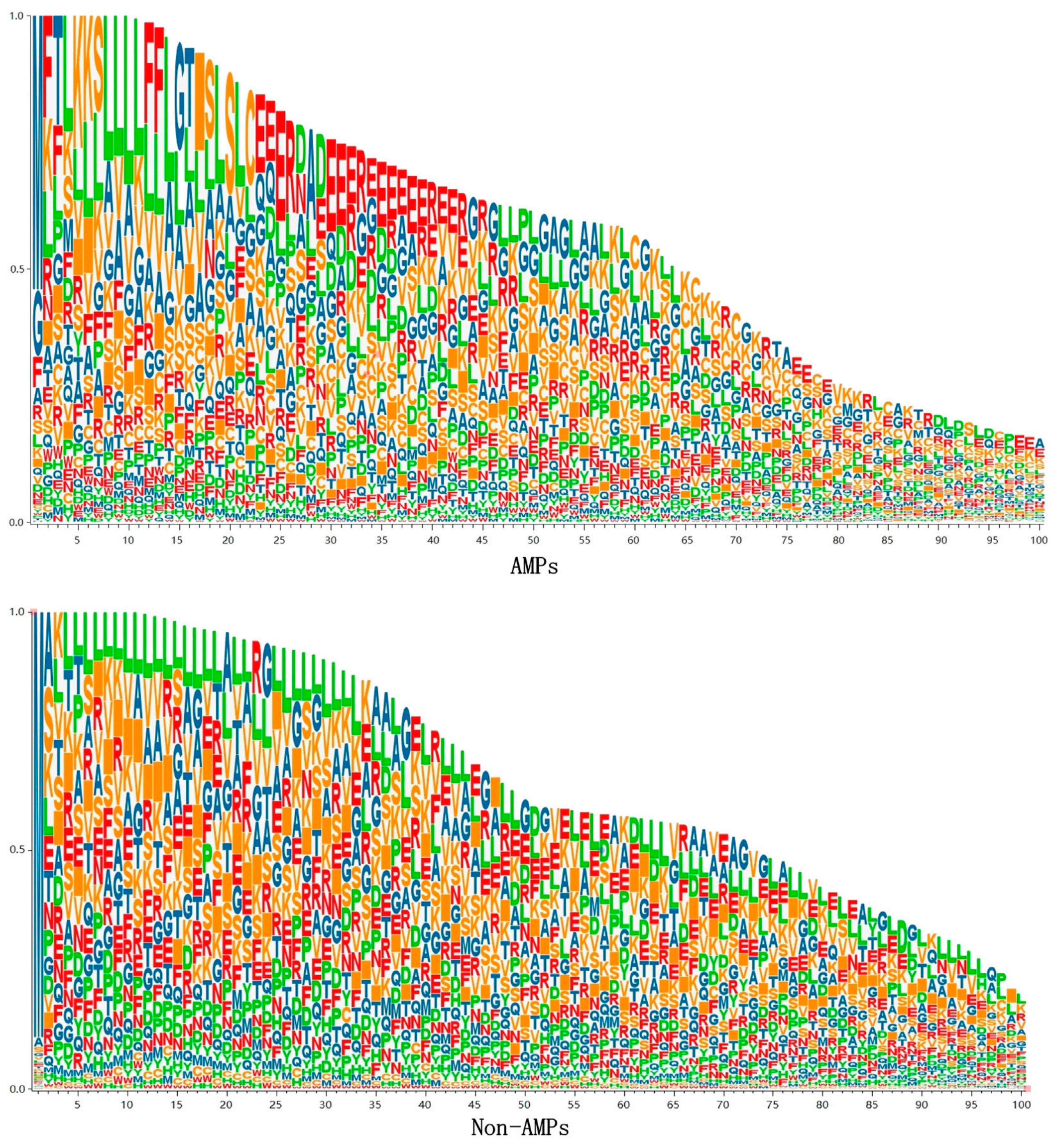


FIGURE 5
sequence logo diagram.

2.3 Deep learning model

Our deep learning model consists of three parts: encoding layer, embedding layer, and convolutional layer. The model architecture is shown in Figure 3.

We convert protein sequences into numerical vectors using CKSAAP, PWAA, N-grams, and the numerical encoding of the raw sequence and then pass these vectors into the embedding layer. The embedding layer converts the sparse vector into a dense vector and reduces the dimension of the vector to facilitate the processing of the upper neural network. The processing process of the embedding layer can be represented by the following matrix operations. The first matrix represents the input feature matrix, the middle matrix represents the weight of this layer, and the multiplied result matrix represents the dimension-reduced feature matrix.

$$[0\ 1\ 0\ 0] \times \begin{bmatrix} 2 & 5 & 7 \\ 4 & 2 & 1 \\ 2 & 8 & 5 \\ 3 & 6 & 8 \end{bmatrix} = [4\ 2\ 1]$$

The convolution layer convolutes the embedded matrix E with N parallel convolution blocks, which can be composed of a set of triples $\{(s_k, q_k, r_k)\}_{(k=1,\ldots,N)}$, where s_K represents the size of the convolution filter, q_k represents the number of convolution filters in the convolution block, and r_k represents the activation

TABLE 2 Comparison of different combination feature coding methods.

Coding	Acc(%)	Sn(%)	Pr (%)	Sp(%)	Fs(%)	Ba (%)	AUROC(%)
Seq + CKSAAP	96.36	97.34	95.36	95.42	96.34	96.38	99.38
Seq + PWAA	95.10	98.20	92.66	91.86	95.35	95.03	99.17
Seq + Bi-gram	97.57	98.48	96.83	96.62	97.65	97.55	99.64
Seq + Tri-gram	96.94	98.11	95.73	95.83	96.90	96.97	99.49
Seq + CKSAAP + PWAA + Bi-gram + Tri-gram	**98.11**	**99.15**	**97.21**	**97.02**	**98.17**	**98.08**	**99.74**

Note: the best performance on a metric is marked in bold.

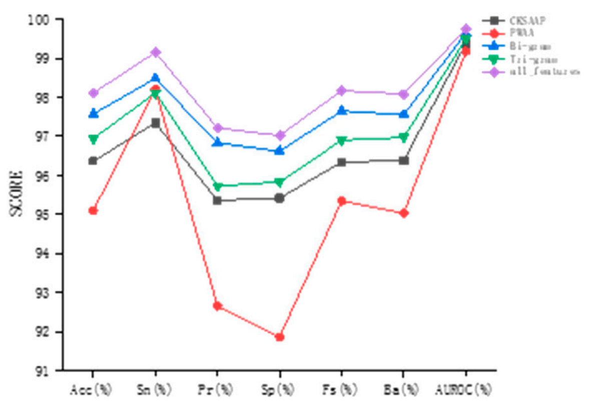


FIGURE 6
Comparison of different combination feature coding methods.

function corresponding to the convolution block. The convolution direction is one-dimensional convolution along the direction of the sequence, and the convolution block will output a set of feature maps $\{Z_k \in R^{(l-s_k+1)\times q_k}\}_{k=1,\ldots,N}$, the convolution block k can be expressed by Eq. 3:

$$Z_k(m, q) = a_k\left(\sum_{i=0}^{e}\sum_{j=0}^{s_k} C(i,j,k) \times E(i, m+j)\right) \quad (3)$$

Where, q = 1, . . . , q_k, C∈ $R^{e\times s_k\times q_k}$ contains the weight tensors of all q_k convolution filters in this convolution block. a_k is the activation function, and we use Rectified Linear Unit (ReLU) as the activation. Z_k (m, q) is the feature map Z_k of the (m, q)th element in the training phase.

Global average pooling integrates global spatial information, while CKSAAP and PWAA codes encode protein sequences as sparse matrices (with many 0s). Choosing global average pooling may reduce the accuracy of prediction, while global pooling can preserve more Boundary information. Therefore, after obtaining each feature map, we perform a global maximum pooling operation to reduce the number of features in the training phase to prevent overfitting. The vector h_k can be calculated by Eq. 4:

$$h_k = [max\, Z_k(:, 1);\ max\, Z_k(:, 2); \ldots;\ max\, Z_k(:, q_k)] \quad (4)$$

Finally, the vector h = [h_1; h_2; . . .; h_N] is obtained by fully connecting all h_k, and the prediction results are output.

Because the learning rate is greatly affected by the output error, the cross-entropy loss function has a larger parameter adjustment range in the early stage of model training, which can make the model training converge faster. To improve the classification efficiency, we use the binary cross-entropy function as our loss function, which can be expressed by Eq. 5:

$$Loss = -\frac{1}{N}\sum_{i=1}^{N} y_i \times log(p(y_i)) + (1-y_i) \times log(1-p(y_i)) \quad (5)$$

Where, y represents the binary label 0 or 1, and p(y) represents the probability that the output belongs to the y label. If the predicted value p(y) approaches 1, then the value of the loss function should approach 0. Conversely, if the predicted value p(y) approaches 0 at this point, the value of the loss function should be very large.

2.4 Model evaluation

To objectively evaluate the performance of this method, we train the model using a 10-fold cross-validation method, which randomly divides the negative and positive samples into k (k = 10) equal-sized subsamples. Among the k subsamples, one sub-sample is reserved as validation data for testing the model, and the remaining k-1 subsamples are used as training data (Lv et al., 2022b; Zhang et al., 2022d). Then repeat the cross-validation process for K (k = 10) times (folds), and each sub-sample is used only once as validation data.

To evaluate the precision of the results, we use 7 metrics of accuracy (A_{cc}), sensitivity (S_n), precision (P_r), specificity (S_p), F1 score (F_s), balance accuracy (B_a), and area under the curve (AUROC) on independent datasets, as shown in Formulas 6 to 12.

$$A_{cc} = \frac{TP+TN}{TP+FN+TN+FP} \quad (6)$$

$$S_n = \frac{TP}{TP+FN} \quad (7)$$

$$P_r = \frac{TP}{TP+FP} \quad (8)$$

$$S_p = \frac{TN}{TN+FP} \quad (9)$$

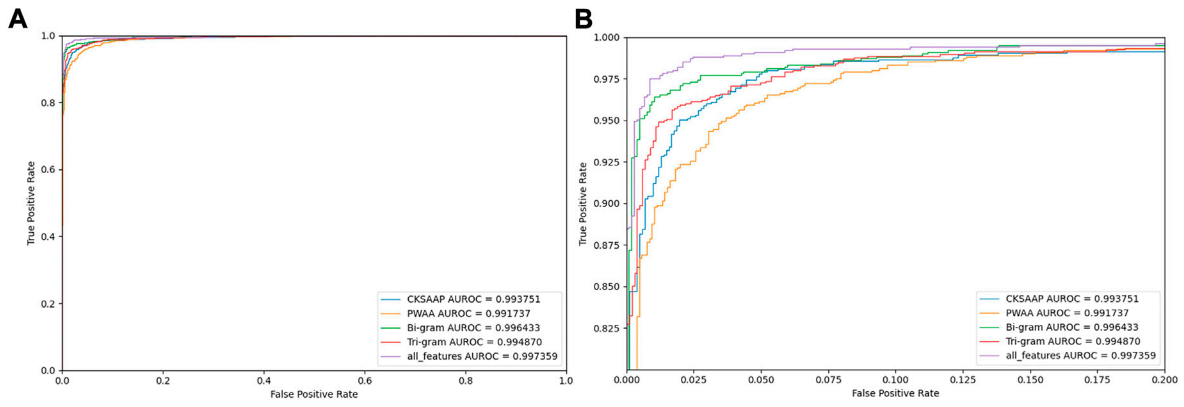


FIGURE 7
Comparing ROC curves with different feature codes. Note: **(B)** is a partially enlarged view of **(A)**.

TABLE 3 Performance comparison of different models.

Methods	Acc(%)	Sn(%)	Pr (%)	Sp(%)	Fs(%)	Ba (%)	AUROC(%)
AMPFUN	54.76	53.85	54.01	55.63	53.93	54.74	64.26
AMP Scanner vr.2	81.71	90.40	76.61	73.31	82.94	81.85	89.37
CAMPR3-ANN	71.64	63.71	74.87	79.31	68.84	71.51	71.51
CAMPR3-RF	70.20	70.40	69.43	70.02	69.91	70.21	74.15
CAMPR3-SVM	74.45	75.98	73.12	72.98	74.52	74.48	76.60
CAMPR3-DA	68.85	67.28	68.72	70.38	67.99	68.83	72.75
ADAM	74.15	67.85	76.86	80.24	72.07	74.04	74.04
ANIAMPpred	96.82	94.99	**98.50**	**98.60**	96.71	96.79	99.30
Our model	**97.87**	**98.39**	97.46	97.32	**97.92**	**97.85**	**99.73**

Note: performance values of other methods come from Sharma. The best performance on a metric is marked in bold.

$$F_s = \frac{2 \times S_n \times P_r}{TN + FP} \tag{10}$$

$$B_a = \frac{S_n + P_r}{2} \tag{11}$$

$$AUROC = \int TPR\, d\,(FPR) \tag{12}$$

Where, TP is the true positive, FP is the false positive, TN is the true negative, FN is the false negative, TPR is the true positive and FPR is the false positive.

3 Results and discussion

3.1 Sequence composition analysis based on benchmark datasets

All proteins are made up of 20 amino acid residues, but the frequency of amino acid residues in each protein varies and the lengths of the amino acid sequences that make up the protein vary. During model training, the composition of peptides in the benchmark dataset is very important to analyze the properties of antimicrobial peptides. By counting the centralized peptide lengths of the AMPs and Non-AMPs data, the peptide lengths of our AMPs and Non-AMPs data sets are between 10 and 200, and most of the peptides are below 100 in length, as shown in Figure 4.

To analyze the sequence consisting of the benchmark dataset, we counted the occurrence frequency of different amino acids at each sequence position. Since the length of AMPs sequences is mainly concentrated in 10–100, we only draw the sequence logo diagram of the first 100 positions, as shown in Figure 5. It can be seen from the figure that specific amino acids belonging to AMPs and Non-AMPs have different positional preferences. In the AMPs sequence, the positions 22–42 are often occupied by glutamic acid (E), and in the Non-AMPs sequence, the positions 22–42 are often occupied by glutamic acid (E). The positions 4–33 are often occupied by leucine (L), and this difference may be due to their belonging to different protein families.

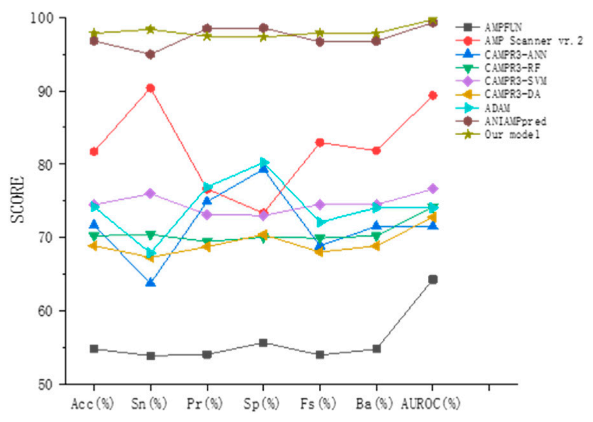


FIGURE 8
Performance comparison of different models.

3.2 Comparison of feature coding methods for different combinations

To study the prediction effect of different feature encodings, we conducted experiments on the combination of these three feature encodings with the original sequences based on the verification set. We treat the Bi-gram and Tri-gram encodings as independent feature encoding methods, and finally, combine all the features for experiments, so we did five sets of comparative experiments. CKSAAP encoding and PWAA encoding only extract amino acid combination and position information. The feature encoding is a sparse matrix with many 0 elements. When it is used alone, the prediction accuracy is relatively low, so the original sequence encoding is added to the experiment to make up. The experimental results are shown in Table 2.

It can be found by observation that in the combination with the original sequence, Bi-gram encoding has the best prediction effect, and the sizes of various indicators after combination are most similar to Bi-gram encoding. Bi-gram encoding combines two adjacent amino acids to enhance sequence expression. Compared with Tri-gram encoding, Bi-gram encoding has stronger local association expression. PWAA encoding has the worst prediction effect and the various indicators are not as balanced as the other three encoding methods. This encoding method considers the upstream and downstream information of the sequence and does not consider the interaction between amino acids. It has only 20 dimensions and is a sparse matrix, which contains data Relatively few, even if there is a supplementary prediction effect encoded by the raw sequence, the effect is not good enough. CKSAAP encoding describes short-range interactions between amino acids. Although its form is also a sparse encoding, it has higher dimensions and more information, so the prediction effect is better than PWAA encoding. The prediction results of this study are most affected by Bi-gram encoding and less affected by PWAA encoding. After we combine these kinds of codes, the prediction effect is improved. As can be seen from Figure 6, this feature combination combines the advantages of these kinds of feature codes and considers the interaction of amino acids in protein sequences, position weights, and upstream and downstream information. And it is not affected by the imbalance of PWAA encoding indicators.

To judge the recognition ability of various encoding combinations for AMPs, we plotted the ROC curves of various combinations, as shown in Figure 7.

3.3 Comparison with other methods

To prove the effectiveness of our method, we compared the prediction results of the method proposed in this paper with other most advanced models (AMPFUN (Chung et al., 2020), AMP Scanner vr.2 (Veltri et al., 2018), CAMPR3 (Waghu et al., 2016), ADAM (Lee et al., 2015), ANIAMPpred (Sharma et al., 2021b)) based on independent test sets. The results are shown in Table 3 and Figure 8. It can be seen from the figure that the performance of ANIAMPpred and the method proposed in this paper is far superior to other models. In terms of PR and SP indicators, ANIAMPpred is slightly higher than our method, but we are the highest in other indicators. The accuracy of our model is 1.05% higher than that of the most advanced method.

4 Discussion

In this paper, we combine CKSAAP, PWAA, N-gram, and raw sequence encoding and apply a deep learning approach to predict AMPs. First, we analyzed the benchmark dataset and compared the differences. Then, we separately evaluated and analyzed the prediction effects of CKSAAP, PWAA, N-gram encoding, and raw sequence encoding combination. Finally, we compare state-of-the-art methods, and the results show that this method has the best performance. We combined CKSAAP, PWAA, N-gram encoding, and original sequence encoding, which not only considered the interaction between amino acids commonly used by other methods, but also considered the upstream and downstream information ignored by other methods, and enhanced the AMPs sequence. Therefore, this method has better performance.

Our method achieves high-precision classification of AMPs based on protein sequence information and yields good performance. But AMPs may have undesirable properties as a drug, including instability and toxicity. In studies of synthesizing and modifying AMPs, even small changes can alter the function of AMPs. This method can only identify AMPs and does not consider the functional characteristics of AMPs. Further research can be carried out according to the functions of AMPs, which will help to better understand the mode of action of AMPs and predict their activities.

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 authors.

Author contributions

Writing—Original Draft, BD; Writing—Review ML, BJ; Writing—Editing, BG, DL,TZ; Funding Acquisition, TZ.

Funding

This work was supported by National Natural Science Foundation of China (62272095, 62172087, 62172129); the Fundamental Research Funds for the Central Universities (2572021BH01);

Acknowledgments

We would like to thank the reviewers for valuable suggestions.

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

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2022.1069558/full#supplementary-material

References

Agrawal, P., Bhalla, S., Chaudhary, K., Kumar, R., Sharma, M., and Raghava, G. P. (2018). *In silico* approach for prediction of antifungal peptides. *Front. Microbiol.* 9, 323. doi:10.3389/fmicb.2018.00323

Akbar, S., Ahmad, A., Hayat, M., Rehman, A. U., Khan, S., and Ali, F. (2021). iAtbP-Hyb-EnC: Prediction of antitubercular peptides via heterogeneous feature representation and genetic algorithm-based ensemble learning model. *Comput. Biol. Med.* 137, 104778. doi:10.1016/j.compbiomed.2021.104778

Ao, C., Zou, Q., and Yu, L. (2022). NmRF: Identification of multispecies RNA 2'-O-methylation modification sites from RNA sequences. *Brief. Bioinform.* 23 (1), bbab480. doi:10.1093/bib/bbab480

Bhadra, P., Yan, J., Li, J., Fong, S., and Siu, S. W. (2018). AmPEP: Sequence-based prediction of antimicrobial peptides using distribution patterns of amino acid properties and random forest. *Sci. Rep.* 8 (1), 1697–1710. doi:10.1038/s41598-018-19752-w

Chen, J., Zheng, S., Zhao, H., and Yang, Y. (2021). Structure-aware protein solubility prediction from sequence through graph convolutional network and predicted contact map. *J. Cheminform.* 13 (1), 7–10. doi:10.1186/s13321-021-00488-1

Chen, Z., Chen, Y.-Z., Wang, X.-F., Wang, C., Yan, R.-X., and Zhang, Z. (2011). Prediction of ubiquitination sites by using the composition of k-spaced amino acid pairs. *PloS one* 6 (7), e22930. doi:10.1371/journal.pone.0022930

Chung, C.-R., Kuo, T.-R., Wu, L.-C., Lee, T.-Y., and Horng, J.-T. (2020). Characterization and identification of antimicrobial peptides with different functional activities. *Briefings Bioinforma.* 21 (3), 1098–1114. doi:10.1093/bib/bbz043

Dao, F.-Y., Lv, H., Zhang, D., Zhang, Z.-M., Liu, L., and Lin, H. (2021). DeepYY1: A deep learning approach to identify YY1-mediated chromatin loops. *Brief. Bioinform.* 22 (4), bbaa356. doi:10.1093/bib/bbaa356

Fjell, C. D., Jenssen, H., Hilpert, K., Cheung, W. A., Panté, N., Hancock, R. E., et al. (2009). Identification of novel antibacterial peptides by chemoinformatics and machine learning. *J. Med. Chem.* 52 (7), 2006–2015. doi:10.1021/jm8015365

Fu, H., Cao, Z., Li, M., and Wang, S. (2020). Acep: Improving antimicrobial peptides recognition through automatic feature fusion and amino acid embedding. *BMC genomics* 21 (1), 597–614. doi:10.1186/s12864-020-06978-0

Gao, R., Wang, M., Zhou, J., Fu, Y., Liang, M., Guo, D., et al. (2019). Prediction of enzyme function based on three parallel deep CNN and amino acid mutation. *Int. J. Mol. Sci.* 20 (11), 2845. doi:10.3390/ijms20112845

Gong, Y., Liao, B., Wang, P., and Zou, Q. (2021). DrugHybrid_BS: Using hybrid feature combined with bagging-SVM to predict potentially druggable proteins. *Front. Pharmacol.* 12, 771808. doi:10.3389/fphar.2021.771808

Han, X., Zhang, L., Zhou, K., and Wang, X. (2019). ProGAN: Protein solubility generative adversarial nets for data augmentation in DNN framework. *Comput. Chem. Eng.* 131, 106533. doi:10.1016/j.compchemeng.2019.106533

Hancock, R. E., and Sahl, H.-G. (2006). Antimicrobial and host-defense peptides as new anti-infective therapeutic strategies. *Nat. Biotechnol.* 24, 1551–1557. doi:10.1038/nbt1267

Hathaway, Q. A., Yanamala, N., Budoff, M. J., Sengupta, P. P., and Zeb, I. (2021). Deep neural survival networks for cardiovascular risk prediction: The Multi-Ethnic Study of Atherosclerosis (MESA). *Comput. Biol. Med.* 139, 104983. doi:10.1016/j.compbiomed.2021.104983

Jain, P., Tiwari, A. K., and Som, T. (2021). Enhanced prediction of anti-tubercular peptides from sequence information using divergence measure-based intuitionistic fuzzy-rough feature selection. *Soft Comput.* 25 (4), 3065–3086. doi:10.1007/s00500-020-05363-z

Khabbaz, H., Karimi-Jafari, M. H., Saboury, A. A., and BabaAli, B. (2021). Prediction of antimicrobial peptides toxicity based on their physico-chemical properties using machine learning techniques. *BMC Bioinforma.* 22 (1), 549–611. doi:10.1186/s12859-021-04468-y

Kumar, P., Kizhakkedathu, J. N., and Straus, S. K. (2018). Antimicrobial peptides: Diversity, mechanism of action and strategies to improve the activity and biocompatibility *in vivo*. *Biomolecules* 8, 4. doi:10.3390/biom8010004

Lata, S., Mishra, N. K., and Raghava, G. P. (2010). AntiBP2: Improved version of antibacterial peptide prediction. *BMC Bioinforma.* 11 (1), 199–S27. doi:10.1186/1471-2105-11-S1-S19

Le, N. Q. K., Ho, Q.-T., Nguyen, T.-T.-D., and Ou, Y.-Y. (2021). A transformer architecture based on BERT and 2D convolutional neural network to identify DNA enhancers from sequence information. *Brief. Bioinform.* 22 (5), bbab005. doi:10.1093/bib/bbab005

Le, N. Q. K., Yapp, E. K. Y., Nagasundaram, N., and Yeh, H.-Y. (2019). Classifying promoters by interpreting the hidden information of DNA sequences via deep learning and combination of continuous fasttext N-grams. *Front. Bioeng. Biotechnol.* 7, 305. doi:10.3389/fbioe.2019.00305

Lee, H.-T., Lee, C.-C., Yang, J.-R., Lai, J. Z., and Chang, K. Y. (2015). A large-scale structural classification of antimicrobial peptides. *Biomed. Res. Int.* 2015, 475062. doi:10.1155/2015/475062

Li, M., Ling, C., Xu, Q., and Gao, J. (2018). Classification of G-protein coupled receptors based on a rich generation of convolutional neural network, N-gram transformation and multiple sequence alignments. *Amino acids* 50 (2), 255–266. doi:10.1007/s00726-017-2512-4

Li, Z., Fang, J., Wang, S., Zhang, L., Chen, Y., and Pian, C. (2022). Adapt-kcr: A novel deep learning framework for accurate prediction of lysine crotonylation sites based on learning embedding features and attention architecture. *Brief. Bioinform.* 23 (2), bbac037. doi:10.1093/bib/bbac037

Lv, H., Dao, F. Y., and Lin, H. (2022a). DeepKla: An attention mechanism-based deep neural network for protein lysine lactylation site prediction. *iMeta* 1 (1), e11. doi:10.1002/imt2.11

Lv, H., Zhang, Y., Wang, J.-S., Yuan, S.-S., Sun, Z.-J., Dao, F.-Y., et al. (2022b). iRice-MS: an integrated XGBoost model for detecting multitype post-translational modification sites in rice. *Brief. Bioinform.* 23 (1), bbab486. doi:10.1093/bib/bbab486

Meher, P. K., Sahu, T. K., Saini, V., and Rao, A. R. (2017). Predicting antimicrobial peptides with improved accuracy by incorporating the compositional, physico-chemical and structural features into Chou's general PseAAC. *Sci. Rep.* 7 (1), 42362–42412. doi:10.1038/srep42362

Moretta, A., Salvia, R., Scieuzo, C., Di Somma, A., Vogel, H., Pucci, P., et al. (2020). A bioinformatic study of antimicrobial peptides identified in the Black Soldier Fly (BSF) Hermetia illucens (Diptera: Stratiomyidae). *Sci. Rep.* 10 (1), 16875–16914. doi:10.1038/s41598-020-74017-9

Nakayama, J. Y., Ho, J., Cartwright, E., Simpson, R., and Hertzberg, V. S. (2021). Predictors of progression through the cascade of care to a cure for hepatitis C patients using decision trees and random forests. *Comput. Biol. Med.* 134, 104461. doi:10.1016/j.compbiomed.2021.104461

Pasupuleti, M., Schmidtchen, A., and Malmsten, M. (2012). Antimicrobial peptides: Key components of the innate immune system. *Crit. Rev. Biotechnol.* 32 (2), 143–171. doi:10.3109/07388551.2011.594423

Qiao, Y., Zhu, X., and Gong, H. (2022). BERT-kcr: Prediction of lysine crotonylation sites by a transfer learning method with pre-trained BERT models. *Bioinformatics* 38 (3), 648–654. doi:10.1093/bioinformatics/btab712

Ren, Y., Chakraborty, T., Doijad, S., Falgenhauer, L., Falgenhauer, J., Goesmann, A., et al. (2022). Prediction of antimicrobial resistance based on whole-genome sequencing and machine learning. *Bioinformatics* 38 (2), 325–334. doi:10.1093/bioinformatics/btab681

Sharma, A. K., and Srivastava, R. (2021). Protein secondary structure prediction using character bi-gram embedding and bi-LSTM. *Curr. Bioinform.* 16 (2), 333–338. doi:10.2174/15748936mta3imdeu1

Sharma, R., Shrivastava, S., Kumar Singh, S., Kumar, A., Saxena, S., and Kumar Singh, R. (2021b). AniAMPpred: Artificial intelligence guided discovery of novel antimicrobial peptides in animal kingdom. *Brief. Bioinform.* 22 (6), bbab242. doi:10.1093/bib/bbab242

Sharma, R., Shrivastava, S., Kumar Singh, S., Kumar, A., Saxena, S., and Kumar Singh, R. (2021a). Deep-ABPpred: Identifying antibacterial peptides in protein sequences using bidirectional LSTM with word2vec. *Brief. Bioinform.* 22 (5), bbab065. doi:10.1093/bib/bbab065

Sharma, R., Shrivastava, S., Kumar Singh, S., Kumar, A., Saxena, S., and Kumar Singh, R. (2022). Deep-AFPpred: Identifying novel antifungal peptides using pretrained embeddings from seq2vec with 1DCNN-BiLSTM. *Brief. Bioinform.* 23 (1), bbab422. doi:10.1093/bib/bbab422

Söylemez, Ü. G., Yousef, M., Kesmen, Z., Büyükkiraz, M. E., and Bakir-Gungor, B. (2022). Prediction of linear cationic antimicrobial peptides active against gram-negative and gram-positive bacteria based on machine learning models. *Appl. Sci.* 12 (7), 3631. doi:10.3390/app12073631

Spänig, S., and Heider, D. (2019). Encodings and models for antimicrobial peptide classification for multi-resistant pathogens. *BioData Min.* 12 (1), 7–29. doi:10.1186/s13040-019-0196-x

Su, X., Xu, J., Yin, Y., Quan, X., and Zhang, H. (2019). Antimicrobial peptide identification using multi-scale convolutional network. *BMC Bioinforma.* 20 (1), 730–810. doi:10.1186/s12859-019-3327-y

Thomas, S., Karnik, S., Barai, R. S., Jayaraman, V. K., and Idicula-Thomas, S. (2010). Camp: A useful resource for research on antimicrobial peptides. *Nucleic Acids Res.* 38 (1), D774–D780. doi:10.1093/nar/gkp1021

Tng, S. S., Le, N. Q. K., Yeh, H.-Y., and Chua, M. C. H. (2021). Improved prediction model of protein lysine Crotonylation sites using bidirectional recurrent neural networks. *J. Proteome Res.* 21 (1), 265–273. doi:10.1021/acs.jproteome.1c00848

Veltri, D., Kamath, U., and Shehu, A. (2018). Deep learning improves antimicrobial peptide recognition. *Bioinformatics* 34 (16), 2740–2747. doi:10.1093/bioinformatics/bty179

Veltri, D. P. (2015). *A computational and statistical framework for screening novel antimicrobial peptides*. George Mason University.

Waghu, F. H., Barai, R. S., Gurung, P., and Idicula-Thomas, S. (2016). CAMPR3: A database on sequences, structures and signatures of antimicrobial peptides. *Nucleic Acids Res.* 44 (D1), D1094–D1097. doi:10.1093/nar/gkv1051

Wang, Y., Xu, L., Zou, Q., and Lin, C. (2022). prPred-DRLF: Plant R protein predictor using deep representation learning features. *Proteomics* 22 (1-2), 2100161. doi:10.1002/pmic.202100161

Wani, M. A., Garg, P., and Roy, K. K. (2021). Machine learning-enabled predictive modeling to precisely identify the antimicrobial peptides. *Med. Biol. Eng. Comput.* 59 (11), 2397–2408. doi:10.1007/s11517-021-02443-6

Wu, X., and Yu, L. (2021). Epsol: Sequence-based protein solubility prediction using multidimensional embedding. *Bioinformatics* 37 (23), 4314–4320. doi:10.1093/bioinformatics/btab463

Xiao, X., Shao, Y.-T., Cheng, X., and Stamatovic, B. (2021). iAMP-CA2L: a new CNN-BiLSTM-SVM classifier based on cellular automata image for identifying antimicrobial peptides and their functional types. *Brief. Bioinform.* 22 (6), bbab209. doi:10.1093/bib/bbab209

Xiao, X., Wang, P., Lin, W.-Z., Jia, J.-H., and Chou, K.-C. (2013). iAMP-2L: a two-level multi-label classifier for identifying antimicrobial peptides and their functional types. *Anal. Biochem.* 436 (2), 168–177. doi:10.1016/j.ab.2013.01.019

Xu, Z., Luo, M., Lin, W., Xue, G., Wang, P., Jin, X., et al. (2021). DLpTCR: An ensemble deep learning framework for predicting immunogenic peptide recognized by T cell receptor. *Brief. Bioinform.* 22 (6), bbab335. doi:10.1093/bib/bbab335

Yan, J., Bhadra, P., Li, A., Sethiya, P., Qin, L., Tai, H. K., et al. (2020). Deep-AmPEP30: Improve short antimicrobial peptides prediction with deep learning. *Mol. Ther. Nucleic Acids* 20, 882–894. doi:10.1016/j.omtn.2020.05.006

Yang, H., Luo, Y., Ren, X., Wu, M., He, X., Peng, B., et al. (2021). Risk prediction of diabetes: Big data mining with fusion of multifarious physical examination indicators. *Inf. Fusion* 75, 140–149. doi:10.1016/j.inffus.2021.02.015

Yuan, S.-S., Gao, D., Xie, X.-Q., Ma, C.-Y., Su, W., Zhang, Z.-Y., et al. (2022). IBPred: A sequence-based predictor for identifying ion binding protein in phage. *Comput. Struct. Biotechnol. J.* 20, 4942–4951. doi:10.1016/j.csbj.2022.08.053

Yun, H.-R., Lee, G., Jeon, M. J., Kim, H. W., Joo, Y. S., Kim, H., et al. (2021). Erythropoiesis stimulating agent recommendation model using recurrent neural networks for patient with kidney failure with replacement therapy. *Comput. Biol. Med.* 137, 104718. doi:10.1016/j.compbiomed.2021.104718

Zhang, H., Liao, L., Cai, Y., Hu, Y., and Wang, H. (2019). IVS2vec: A tool of inverse virtual screening based on word2vec and deep learning techniques. *Methods* 166, 57–65. doi:10.1016/j.ymeth.2019.03.012

Zhang, Q., Li, H., Liu, Y., Li, J., Wu, C., and Tang, H. (2022a). Exosomal non-coding RNAs: New insights into the biology of hepatocellular carcinoma. *Curr. Oncol.* 29 (8), 5383–5406. doi:10.3390/curroncol29080427

Zhang, Y., Lin, J., Zhao, L., Zeng, X., and Liu, X. (2021). A novel antibacterial peptide recognition algorithm based on BERT. *Brief. Bioinform.* 22 (6), bbab200. doi:10.1093/bib/bbab200

Zhang, Y., Zhu, G., Li, K., Li, F., Huang, L., Duan, M., et al. (2022b). Hlab: Learning the BiLSTM features from the ProtBert-encoded proteins for the class I HLA-peptide binding prediction. *Briefings Bioinforma.* 23. doi:10.1093/bib/bbac173

Zhang, Z.-Y., Ning, L., Ye, X., Yang, Y.-H., Futamura, Y., Sakurai, T., et al. (2022c). iLoc-miRNA: extracellular/intracellular miRNA prediction using deep BiLSTM with attention mechanism. *Brief. Bioinform.* 23 (5), bbac395. doi:10.1093/bib/bbac395

Zhang, Z.-Y., Sun, Z.-J., Yang, Y.-H., and Lin, H. (2022d). Towards a better prediction of subcellular location of long non-coding RNA. *Front. Comput. Sci.* 16 (5), 165903–165907. doi:10.1007/s11704-021-1015-3

Zheng, X., Fu, X., Wang, K., and Wang, M. (2020). Deep neural networks for human microRNA precursor detection. *BMC Bioinforma.* 21 (1), 17–7. doi:10.1186/s12859-020-3339-7

Zou, Y., Wu, H., Guo, X., Peng, L., Ding, Y., Tang, J., et al. (2021). MK-FSVM-SVDD: A multiple kernel-based fuzzy SVM model for predicting DNA-binding proteins via support vector data description. *Curr. Bioinform.* 16 (2), 274–283. doi:10.2174/2212392xmta3jmtydy

frontiers | Frontiers in Genetics

TYPE Original Research
PUBLISHED 29 November 2022
DOI 10.3389/fgene.2022.1067562

TSSNote-CyaPromBERT: Development of an integrated platform for highly accurate promoter prediction and visualization of *Synechococcus* sp. and *Synechocystis* sp. through a state-of-the-art natural language processing model BERT

Dung Hoang Anh Mai, Linh Thanh Nguyen and Eun Yeol Lee*

Department of Chemical Engineering (BK21 FOUR Integrated Engineering Program), Kyung Hee University, Yongin-si, South Korea

Check for updates

OPEN ACCESS

EDITED BY
Quan Zou,
University of Electronic Science and Technology of China, China

REVIEWED BY
Wang Xi,
Harvard University, United States
Qiong Zhang,
Affiliated Hospital of Nantong University, China

*CORRESPONDENCE
Eun Yeol Lee,
eunylee@khu.ac.kr

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 12 October 2022
ACCEPTED 17 November 2022
PUBLISHED 29 November 2022

CITATION
Mai DHA, Nguyen LT and Lee EY (2022), TSSNote-CyaPromBERT: Development of an integrated platform for highly accurate promoter prediction and visualization of *Synechococcus* sp. and *Synechocystis* sp. through a state-of-the-art natural language processing model BERT.
Front. Genet. 13:1067562.
doi: 10.3389/fgene.2022.1067562

Since the introduction of the first transformer model with a unique self-attention mechanism, natural language processing (NLP) models have attained state-of-the-art (SOTA) performance on various tasks. As DNA is the blueprint of life, it can be viewed as an unusual language, with its characteristic lexicon and grammar. Therefore, NLP models may provide insights into the meaning of the sequential structure of DNA. In the current study, we employed and compared the performance of popular SOTA NLP models (i.e., XLNET, BERT, and a variant DNABERT trained on the human genome) to predict and analyze the promoters in freshwater cyanobacterium *Synechocystis* sp. PCC 6803 and the fastest growing cyanobacterium *Synechococcus elongatus* sp. UTEX 2973. These freshwater cyanobacteria are promising hosts for phototrophically producing value-added compounds from CO_2. Through a custom pipeline, promoters and non-promoters from *Synechococcus elongatus* sp. UTEX 2973 were used to train the model. The trained model achieved an AUROC score of 0.97 and F1 score of 0.92. During cross-validation with promoters from *Synechocystis* sp. PCC 6803, the model achieved an AUROC score of 0.96 and F1 score of 0.91. To increase accessibility, we developed an integrated platform (TSSNote-CyaPromBERT) to facilitate large dataset extraction, model training, and promoter prediction from public dRNA-seq datasets. Furthermore, various visualization tools have been incorporated to address the "black box" issue of deep learning and feature analysis. The learning transfer ability of large language models may help identify and analyze promoter regions for newly isolated strains with similar lineages.

KEYWORDS

deep learning, natural language processing, transformer, promoter prediction, dRNA-Seq, differential RNA sequencing

Introduction

A classic problem in bioinformatics is the challenge of predicting promoters (Zhang et al., 2022). Promoter regions are DNA regions where RNA polymerase binds to initiate the transcription process, the first step in the central dogma of molecular biology (Butler and Kadonaga, 2002). Owing to their essential role in regulating and determining the timing and expression levels of genes needed for vital functions, the prediction and in-depth functional analysis of promoters have been of interest to biologists. Previously, owing to the complexity of cis-regulation networks and lack of data, attempts at developing promoter prediction tools were inadequate (Bhandari et al., 2021). However, recent advancements in machine learning and deep learning have successfully leveraged genomic data. To date, many groups have successfully constructed promoter prediction tools using traditional machine learning methods, knowledge-based position matrix weight (Huerta and Collado-Vides, 2003; Burden et al., 2005; Rangannan and Bansal, 2010; Di Salvo et al., 2018) through support vector machines, and artificial neural networks for this logistic regression task (Gordon et al., 2003; da Silva et al., 2006; Mann et al., 2007; Towsey et al., 2008; He et al., 2018; Liu et al., 2018; Rahman et al., 2019; Xiao et al., 2019; Zhang et al., 2019; Li et al., 2021). Convolutional neural networks (CNN) and recurrent neural network (RNN)-based architectures (long short-term memory, gated recurrent units) have recently become the most popular choices for promoter classification (Nguyen et al., 2016; Le et al., 2019; Oubounyt et al., 2019; Amin et al., 2020; Zhu et al., 2021). CNN-based models depend on predetermined kernel size designs to extract and generalize local features; therefore, they might fail to capture long-range contexts. To overcome this limitation, some research groups have integrated RNN-based models to retrieve long-term dependencies. By design, LTSM computations from RNNs are processed sequentially and depend on the outputs of the previous hidden states for the next state to maintain the sentence structure and context; however, this, in turn, leads to the vanishing gradient problem. These limitations pose difficulties and may restrict the scalability and flexibility of constructed models when applied to other species.

Since its first appearance in 2017, the transformer architecture, with its unique self-attention mechanism, has revolutionized the natural language processing (NLP) field and achieved SOTA performance in various machine learning tasks (Vaswani et al., 2017). As these transformers perform well, they have made their way to other branches (e.g., computer vision) (Wu et al., 2020; Arnab et al., 2021; Zhou et al., 2021) that were previously dominated by CNNs, and they are now also used in multimodal learning for content generation (Tsai et al., 2019; Yu et al., 2019; Dzabraev et al., 2021). Transformer-based models are versatile and can be incorporated into different architectures owing to their robustness and flexibility through their learning-transfer capability. Considering the sequential nature of DNA, which can be regarded as a natural language with unique grammar and lexicon, transformer-based models are particularly well suited for supervised classification tasks.

Therefore, adopting a different approach in the current study, we employed and compared transformer-based models for the promoter prediction problem. To date, most of the currently constructed models have been designed for popular species with curated regulatory databases such as humans, fruit flies, mice, *Escherichia coli*, and yeasts (Oubounyt et al., 2019; Rahman M et al., 2019; Li et al., 2021). However, there is still considerable interest in integrating deep-learning techniques for promoter analysis in other (less popular) species. For example, cyanobacteria are an ancient and diverse group of photo-oxygenic prokaryotes with ample potential for the photosynthetic production of value-added chemical compounds from the greenhouse gas CO_2. Many cyanobacterial species with a high potential for valorizing CO_2 are still being isolated and characterized every year. Some of the most notable genera were *Synechocystis* and *Synechococcus*. These model organisms can convert CO_2 into various useful products (Luan et al., 2019; Sarnaik et al., 2019; Lin et al., 2020; Pattharaprachayakul et al., 2020; Qiao et al., 2020; Taylor and Heap, 2020; Kato and Hasunuma, 2021; Roh et al., 2021; Santos-Merino et al., 2021). Although they have been characterized and researched for a few decades, the application of deep learning for promoter prediction specifically in cyanobacteria is still lacking. Therefore, in this study, we used the promoters of *Synechococcus elongatus* sp. UTEX 2973, the fastest growing cyanobacterium for model training and testing (Song et al., 2016; Mueller et al., 2017). We further conducted cross-validation of the promoters of the model organism *Synechocystis* sp. PCC 6803 to test whether the models also work on related species (Ikeuchi and Tabata 2001). Combined with knowledge-based analysis, in-depth model characterization may help tackle the "black box" problem of deep-learning models.

To facilitate the development and incorporation of SOTA transformer-based promoter prediction tools, we reconstructed a pipeline (using TSSNote and PromBERT Google Colab notebooks) to compute and extract the promoters of public differential RNA-seq (dRNA-seq) datasets from the National Center for Biotechnology Information Sequence Read Archive (NCBI SRA) database and used them for model training. dRNA-seq is an RNA sequencing technique that allows the determination of TSS at 1 bp resolution by enriching primary

TABLE 1 Datasets employed in this study.

Species	SRA accession number	Condition	TEX treatment
Synechococcus elongatus sp. UTEX2973	SRR6334749, SRR6334750	Primary transcripts under normal condition	TEX (+)
	SRR6334747, SRR6334748	Control under normal condition	TEX (-)
Synechocystis sp. PCC 6803	SRR1019366, SRR1019365	Primary transcripts under exponential and stationary phase	TEX (+)
	SRR1019368, SRR1019367	Secondary reads from 10 different conditions	TEX (-)
Synechocystis sp. PCC 6714	SRR1019241	Primary reads from stationary phase	TEX (+)
	SRR1019242	Secondary reads from 10 different conditions	TEX (-)

transcripts (Bischler et al., 2015). In contrast to conventional differential expression RNA-seq (RNA-seq), dRNA-seq requires additional treatments and more expensive and complex procedures, making these datasets rather limited. Transfer learning is a core advantage of large-parameter language models. We expect that, with fine-tuning, transformer-based promoter models can be good approximators for other related species. To improve the accessibility to researchers with and without expertise in machine learning, separate modules of the pipeline for promoter extraction, model training, promoter prediction, and visualization were ported into the cloud-based platform Google Colab. We demonstrated that, even without the advantage of the pre-training phase, transformer-based models, such as bidirectional encoder representations from transformers (BERT) and XLNET, are capable of highly accurate promoter prediction for *Synechocystis* and *Synechococcus* species solely through a context-wise self-attention mechanism (Devlin et al., 2018; Yang et al., 2019).

Materials and methods

Datasets

Raw dRNA-seq datasets for *Synechocystis* sp. PCC 6803 and *Synechococcus elongatus* sp. UTEX 2973 and for *Synechocystis* sp. PCC 6714 were downloaded from the NCBI SRA database, and genomic DNA sequence assemblies were downloaded from the NCBI RefSeq database (Table 1).

Independent *E. coli* promoter datasets for benchmarking were obtained from https://github.com/chenli-bioinfo/promoter.

Available data and local and Google Colab versions of TSSNote-CyaPromBERT are available at https://github.com/hanepira/TSSnote-CyaPromBert.

Constructing promoter extracting module from dRNA-seq datasets

Because one of the objectives of the current work is to create a cloud-computing-based pipeline that can be applied without strong hardware requirements, we implemented algorithms in a Colab notebook for TSS prediction based on changes in read coverage, in a similar manner to TSSpredator (Dugar et al., 2013) but with more flexibility for customizations. This promoter extracting module (TSSNote) takes SRA ids for TEX (+) and TEX (-) treatments and fasta from NCBI as inputs and conducts alignment by HISAT2 and read coverage extraction through SAMTools. HISAT2 enables soft-clipping alignment, through which adapters do not interfere with the read alignment. SAMTools are then used to extract read coverage from the plus and minus strands for later computations. The read coverage files from both TEX (+) enrichment and TEX (-) were used to locate and compute the potential TSSs enriched by TEX treatment. Because the quality of dRNAseq datasets is dependent on experimental procedures, after calculating potential TSSs, users can filter TSSs based on the read coverage cut-off or coverage change cut-off. BAM files can be downloaded into local drives for manual observation and curation using NGS genome browsers. The overall design is illustrated in Figure 1, and the detailed workflow of the TSSNote is shown in Figure 2.

Read coverage change at a specific location is calculated by the following function:

$$\triangle xi = \frac{x_{i+1} + c}{x_i + c}$$

Where: x_i = coverage depth at position i $x_i + 1$ = coverage depth at position $i + 1$ Δx_i = change factor from xi to $x_i + 1$ c = calibration constant to prevent division zero (0.01).

Promoter and non-promoter sequences extraction

Promoters were extracted directly upstream from the predicted TSSs. For promoter sequences, ribosomal RNA depletion in dRNAseq experiments may not be 100%; therefore, further trimming methods were implemented. We tested the TSSs identified by TSSNote based on the wildtype dataset with the TSSs proposed in the original publication (Tan et al., 2018). Even though the

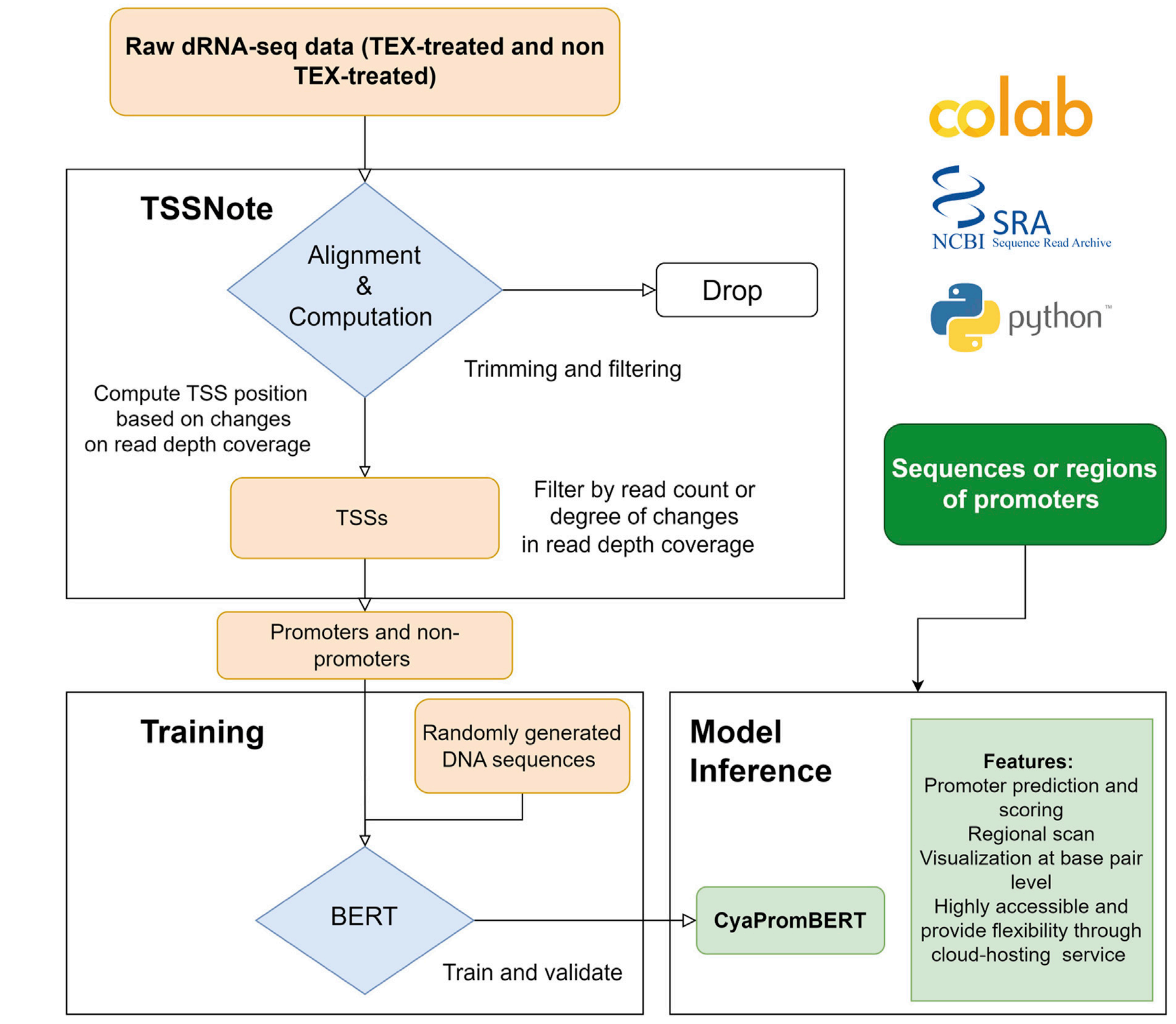

Raw dRNA-seq data (TEX-treated and non TEX-treated)
TSSNote
Alignment & Computation
Drop
Trimming and filtering
Compute TSS position based on changes on read depth coverage
TSSs
Filter by read count or degree of changes in read depth coverage
Promoters and non-promoters
Training
Randomly generated DNA sequences
BERT
Train and validate
CyaPromBERT
Model Inference
Features: Promoter prediction and scoring Regional scan Visualization at base pair level Highly accessible and provide flexibility through cloud-hosting service
Sequences or regions of promoters
colab
NCBI SRA Sequence Read Archive
python

FIGURE 1
Overall scheme for constructing and developing TSSNote and CyaPromBERT. TSSNote facilitates downloading raw dRNA-seq datasets from NCBI SRA database and conducts alignment, sorting, and filtering for extracting promoters and non-promoters. These sequences are later used to train a BERT model for the task of promoter prediction. Randomly generated DNA sequences with similar size to promoter length are added to reduce biases, and overfitting is used to improve the model's robustness. The trained model is capable of promoter prediction, regional scanning, and visualization at base-pair level.

implementation method was different, many of the predicted TSSs were consistent. By setting constraints more stringent, through expression strength and degree of changes, more than 90% of the TSSs identified in the wildtype dataset were also found in the original proposed TSSs concatenated from multiple conditions. Therefore, filtered promoter datasets extracted from strongly expressed and enriched TSSs should be sufficiently reliable. As deep-learning models require a large amount of data for accurate generalization, we believe that the flexibility offered by TSSNote can be crucial. Furthermore, read counts and fold-changes in read coverage can provide more information to group and filter promoters based on promoter strength. It can be used independently or together with existing tools for better analysis. In the current work we lowered the constraints to take into account the potential spurious transcriptional events and weak promoters of other sigma factor groups which would be filtered by the method used in the original publication. The good performance on cross validation and clear pattern enrichment indicate that the model has successfully learned key features from the extracted promoters for promoter recognition task.

The non-promoter sequences were extracted from the "non-promoter" regions. Specifically, Non-promoter sequences were

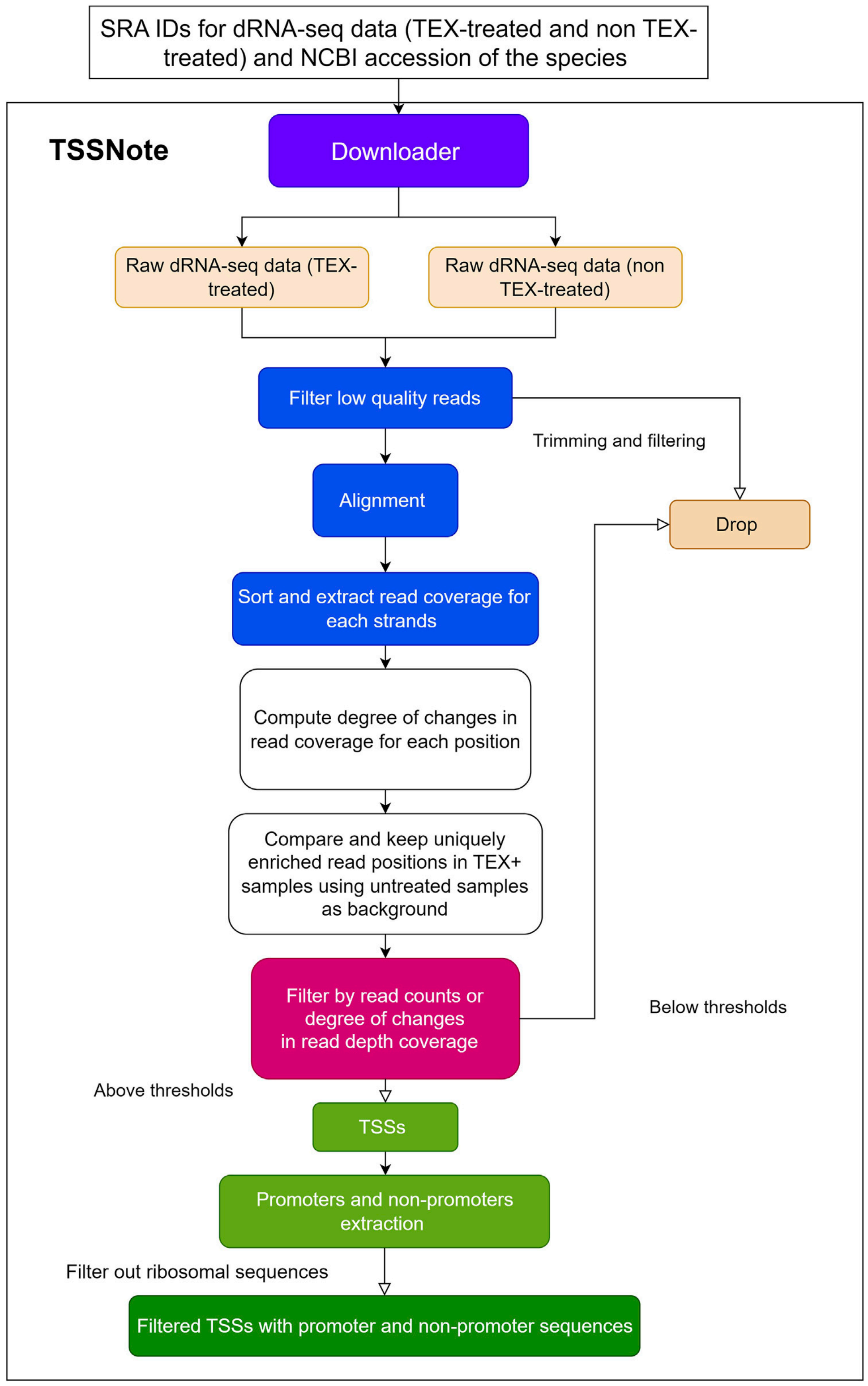

SRA IDs for dRNA-seq data (TEX-treated and non TEX-treated) and NCBI accession of the species
TSSNote
Downloader
Raw dRNA-seq data (TEX-treated)
Raw dRNA-seq data (non TEX-treated)
Filter low quality reads
Trimming and filtering
Alignment
Drop
Sort and extract read coverage for each strands
Compute degree of changes in read coverage for each position
Compare and keep uniquely enriched read positions in TEX+ samples using untreated samples as background
Filter by read counts or degree of changes in read depth coverage
Below thresholds
Above thresholds
TSSs
Promoters and non-promoters extraction
Filter out ribosomal sequences
Filtered TSSs with promoter and non-promoter sequences

FIGURE 2
Detailed flowchart of TSSNote operation to extract TSSs, promoters, and non-promoter sequences.

sampled from the downstream of TSSs. If the distance between two neighboring TSSs is larger than 2 times the sequence length, that interval region is marked and used for sampling non promoter sequences. We further added 10% randomly generated sequences to increase noise and reduce overfitting. The non-promoter sequences then are shuffled, and a portion of the non-promoter sequences was used at the ratio 1: 1 promoter–non-promoter for model training.

Model training

The TSSs of each species from different datasets was extracted and concatenated for model construction using Python wrapper TSSNote, which was written in Python 3.9 as a user-friendly pipeline to conduct raw data gathering using SRA toolkits 3.0 (https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=software) and Entrez-direct (Kans 2022), sequence indexing, read alignment by HISAT2 (Kim et al., 2019), strand sorting, and read coverage calculation by SAMtools (Danecek et al., 2021). Promoter sequences were extracted from the calculated TSSs using the Biopython package (Cock et al., 2009).

To construct CyaPromBert and evaluate the performance of different transformer-based models, Pytorch 1.11.0 and Pytorch-lightning 1.6.4 (Paszke et al., 2019). Transformer-based models were constructed using base models from huggingface's transformer library 4.18.0 (Wolf et al., 2020).

The probability was calculated by the sigmoid function:

$$S(x) = \frac{1}{1 + e - x}$$

The performance of the models was evaluated by precision, recall, F-1, and AUPRC, AUROC scores.

$$Precision = \frac{tp}{tp + fp}$$

$$Recall = \frac{tp}{tp + fn}$$

$$F1 = \frac{2tp}{2tp + fp + fn}$$

Where: tp = true positive fp = false positive fn = false negative

The area under the precision-recall curve (AUPRC) is calculated from the average precision score and AUROC is the area under the receiver operating characteristics.

Binary cross entropy was used as the loss function.

$$BCELoss = -\frac{1}{N}\sum_{i=1}^{N}\left(yi^{*}\log\ (pi) + (1 - yi)^{*}\log\left(1 - \mathrm{pi}\right)\right)$$

Attention weight visualization libraries, BERTviz 1.4.0, and Captum 0.5.0, were implemented to improve visualization and interpretability (Vig 2019; Kokhlikyan et al., 2020). Both TSSNote and the models were first developed and trained on a local workstation equipped with an NVIDIA RTX 3070 before porting and testing on the Google Colab cloud computing service.

Results and discussion

Selecting the best performing SOTA transformer-based model for promoter prediction

The transformer-based architecture has demonstrated that, with sufficient data, matrix multiplications, linear layers, and layer normalization, the deep-learning model can achieve SOTA machine translation tasks without relying on CNN and RNN (Vaswani et al., 2017). BERT and XLNET are two of the most popular transformer-based language models (Devlin et al., 2018; Yang et al., 2019). Fundamentally, these large-language models are stacks of encoding modules from the original transformer model. However, they are pre-trained differently and use different tokenizers. BERT is an autoencoding-based model, whereas XLNet employs an autoregressive method similar to the famous GPT models from OpenAI (Floridi and Chiriatti 2020). These differences reflect the capability to capture the semantic context for prediction in masked language prediction pretraining, and thus they can affect the performance of the model. However, the corpora, on which both BERT and XLNet were trained, are far different from the genomic DNA sequences; therefore, they might not have pretraining advantages. Thus, we also compared a different variant of BERT (DNABERT) pretrained on human genomic DNA at different kmer lengths (from three to five nucleotides) (Ji et al., 2021). The DNABERT models outperformed previous CNN-based models for TATA and non-TATA promoter prediction tasks in eukaryotes. To improve the resolution, we trained a byte-level byte-pair-encoding (BPE) tokenizer at a length of one nucleotide (or kmer 1). The operating mechanism is illustrated in Figure 3 and the performance results are listed in Table 2 and Figure 4.

For this particular promoter prediction task (using binary cross entropy as the loss function and F1 score as the key determinants to evaluate model performance), both XLNet-base and BERT-base using a one kmer length byte-level BPE tokenizer had the best performance compared to the default tokenizers or tokenizer at different lengths. Both XLNet+1bp tokenizer and BERT+1bp tokenizer achieved AUROC scores of 0.97 and 0.977, and F1 scores of 0.92 and 0.93 respectively. These two models exhibited comparable performance. However, during training and testing, XLNet used more computing resources than BERT; therefore, we selected the BERT-base + 1bp tokenizer for further investigation. The corpora in which these two base models were pretrained did not contain genomic databases. They should not benefit from the pre-training process for the

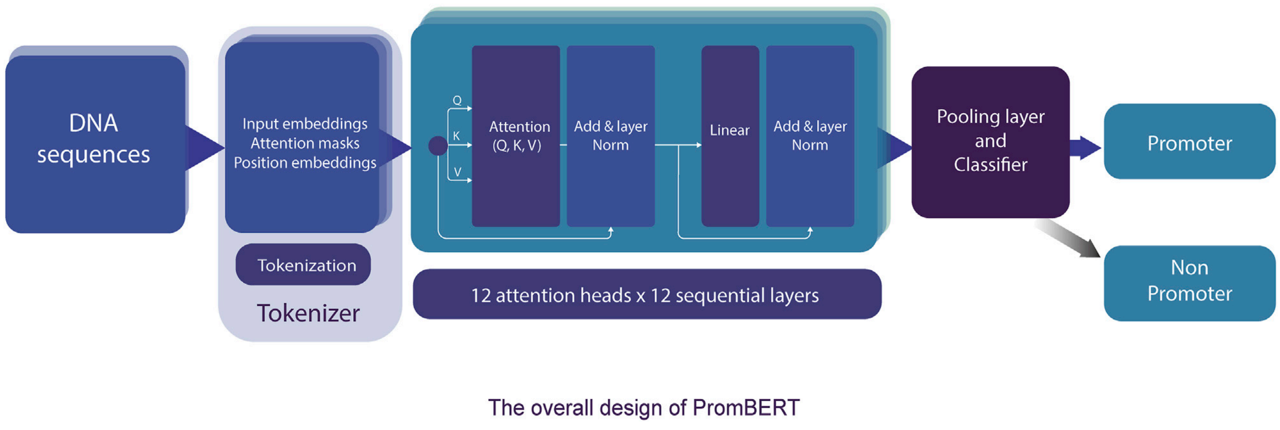


FIGURE 3
The detailed model architecture of PromBERT for promoter prediction. DNA sequences of fixed length are tokenized using a custom 1bp tokenizer and fed into 144 attention modules. Based on the final tensors in the pooling layer, the classifier calculates the probabilities of promoter and non-promoter using the sigmoid function. Backpropagation was conducted using binary cross entropy loss.

TABLE 2 Performance of popular transformer-based NLP models for promoter prediction.

Model and tokenizer	AUROC		Precision		F1 score		Support	
	Promoter	Non-promoter	Promoter	Non-promoter	Promoter	Non-promoter	Promoter	Non-promoter
XLNET	0.926	0.925	0.85	0.84	0.85	0.85	1018	1019
XLNET + 1bp tokenizer	0.97	0.97	0.92	0.92	0.92	0.92	1018	1019
BERT-base	0.941	0.942	0.84	0.89	0.87	0.87	1001	1036
BERT-base + 1bp tokenizer	0.977	0.977	0.92	0.95	0.93	0.93	1001	1036
DNABERT3 + kmer 3	0.944	0.944	0.9	0.84	0.86	0.88	1008	1029
DNABERT4 + kmer 4	0.944	0.944	0.88	0.86	0.87	0.87	1028	1009
DNABERT5+ kmer 5	0.956	0.956	0.9	0.89	0.89	0.89	1031	1006

promoter prediction task. The high performance can be attributed to context awareness (context-based embedding) of the position and composition of the tokens (nucleotides) through the self-attention mechanism. We further tested the performance of the BERT-base + 1bp tokenizer and DNABERT5 + 1bp tokenizer. The results further show that there are no differences in performance. These findings also confirmed that, during training for promoter prediction tasks using BERT, the choice of tokenizer influenced the performance.

Surprisingly, the DNABERT variants trained in the genomic context performed worse than the BERT-base + 1bp tokenizer. Longer kmer lengths might provide a better context and have more meaningful biological values for interpretation (Ji et al., 2021); however, the F1 scores of the pretrained DNABERT 3, 4, and five were lower than those of BERT-base and XLNet with the 1bp tokenizer. One possible explanation for this finding is that the 1bp tokenizer better captured nuances at the single-nucleotide level interactions in the training dataset. As the promoter datasets in the current study were extracted solely from TSSs and were not grouped in transcriptional factor classes, less information is required to make decisions. This model may significantly favor specific nucleotides at certain fixed positions. Using tokenizers with longer kmer lengths (for the case of DNABERT) might be better for other genomic applications or designs that require larger curated datasets with expected long-range interactions within those genomic sequences. This is particularly relevant if the model is pre-trained or fine-tuned by permutation and masked language modeling first on the genomic data of the target species. We further tested the influence of promoter length on model performance; however, increasing the promoter length to 200bp did not change the performance of any of the tested models (data not shown).

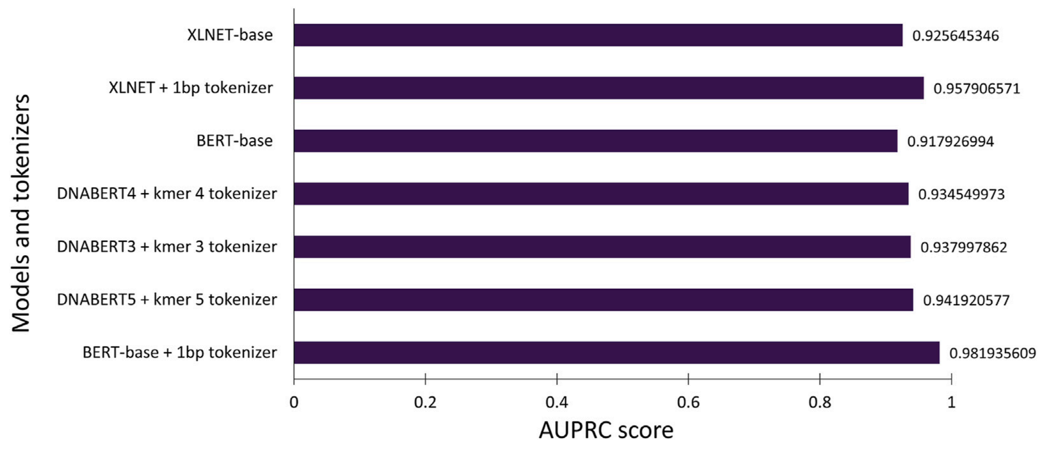


FIGURE 4
Average precision scores of the tested transformer-based models.

TABLE 3 Performance of Eco70PromBERT and popular promoter prediction models for *E.coli* using an independent dataset (σ70 promoters and non-promoters).

Model and tokenizer	AUROC		Precision		F1 score		Support	
	Promoter	Non-promoter	Promoter	Non-promoter	Promoter	Non-promoter	Promoter	Non-promoter
Eco70PromBERT (BERT-base + 1bp tokenizer)	0.92	0.90	0.91	0.91	0.91	0.91	110	108
iPro70-FMWin	0.90	0.90	0.93	0.88	0.90	0.91	110	108
iPromoter-2L2.0	0.91	0.91	0.90	0.92	0.91	0.91	110	108

Evaluating model performance compared to existing promoter prediction models using independent datasets from *E. coli*

To evaluate the robustness of the proposed BERT-base +1bp tokenizer for promoter prediction task, we conducted model training using an independent dataset for σ70 promoters for model benchmarking from a previous study (Zhang et al., 2022).

We compared the performance of our model with two promoter prediction webservers iPro70-FMWin (Rahman et al., 2019) and iPromoter-2L2.0 (Liu et al., 2018) which were reported to have very high accuracy for σ70 promoters. The results showed that those three models performed equally well on the benchmarking dataset with F1 scores around 91%. Our model performed slightly better across promoter and non-promoter tag (Table 3). Since iPro70-FMWin also provides probability scores, we compared the AUPRC scores of this model with our Eco70PromBERT-1bp (Figure 5). Our model had a better AUPRC score of 0.967 compared to 0.953 from iPro70-FMWin.

The results illustrated the robustness of BERT-base + 1bp tokenizer for promoter prediction task in general. Considering that both iPro70-FMWin and iPromoter-2L2.0 were designed specifically to extract sequence features with various customizations for promoter classification to achieve SOTA performance. The plug-and-play characteristic of large language models like BERT would be better for scalability and broader applications.

Interpreting the model's behavior through Monte Carlo sampling and attention score visualization

Interpreting deep-learning (DL) models is another important aspect of model validation. One of the main issues concerning deep-learning models is the "black box" problem, where users might not know how DL models process and compute the outputs for reverse engineering and understanding. This problem is particularly difficult for large parameter models

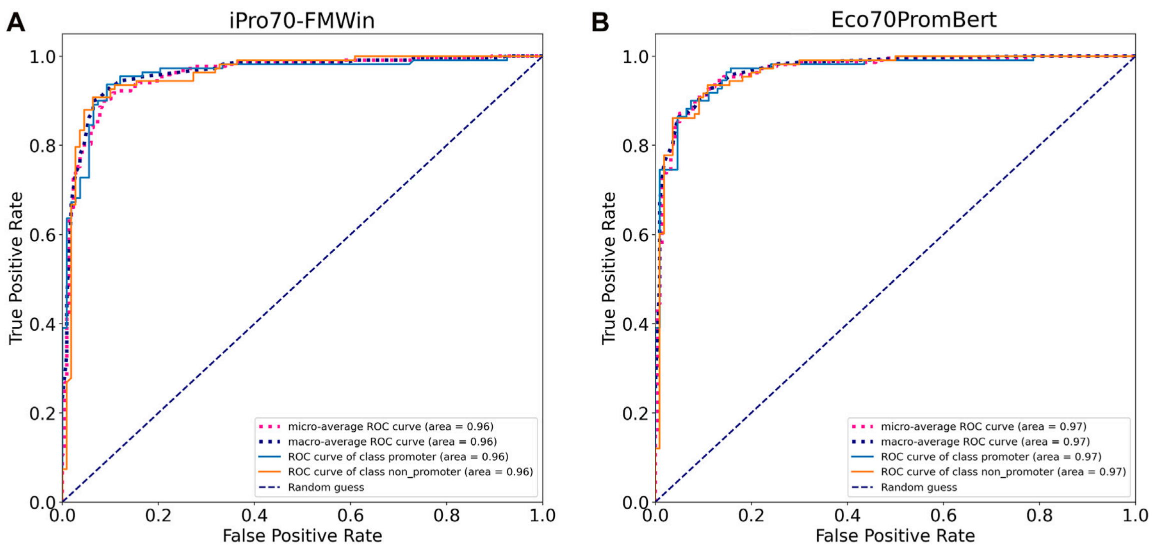


FIGURE 5
Model performance based on receiver operating characteristic curves tested on an independent promoter datasets from *E. coli*. **(A)** iPro70-FMWin **(B)** Eco70PromBERT- (BERT-base + 1bp tokenizer trained on promoter datasets from *E. coli*).

such as NLP models (e.g., BERT). Specifically, the BERT-base model used in this study consists of 86.8 million trainable parameters from 144 attention modules (12 layers × 12 heads). The use of attention scores to visualize token weights is a commonly used method for improving model understanding. We employed integrated libraries for interpretability, namely BERTviz and Captum, to gain more insight into CyaPromBERT behavior and key features determining true promoters or non-promoters.

From the BERTviz model view and Captum, it appeared that a large number of self-attention modules focused on -10 element and occasionally on -35 element for sequences classified as promoters (Figure 6 and Figure 7). This is understandable, as the training dataset consists of all promoters from different sigma factor groups. In prokaryotes, the promoter regions are AT-rich and depend on the differences between their local structural properties and flanking sequences. The AT-rich -10 element plays a conserved role in DNA unwinding and facilitates transcription. Therefore, the constructed model could capture this local interaction context for promoter classification. Not all attention modules were utilized in the trained model; non-operational modes were observed in several layers and attention heads (cross-attention pointing to <s> and </s > tokens).

To estimate the closeness of the classifier to the real consensus of the -10 element, we defined a simple Monte Carlo generator using the constructed CyaPromBERT model as the discriminator. The pseudo-random generator generated fixed-length DNA sequences (50 nucleotides) until an expected number of sequences (500 sequences) passed the discriminator (cutoff value ≥0.99). Using this enrichment method, a recognition motif of the GnTAAAATT region was identified with a strong emphasis on thymine at the -11 and adenine at the -10 and -9 positions followed by two thymine bases at -6 and -5 (Figure 7C), which is similar to the consensus motif of the extended -10 element GnTATAAT of the extended -10 element previously reported in *E. coli* (Feklistov and Darst 2011). Further stretching of GGG was similar to that of the discriminator element in *E. coli*. Reversed enrichment using Monte Carlo sampling did not yield any motifs for non-promoter sequences. Promoters recognized by sigma factor groups have preferred motifs; however, crosstalk between groups does occur due to similarity of the transcriptional factors (Figure 7B). Group 1 (SigA), from the model cyanobacterium *Synechocystis* sp. PCC 6803 has consensus motifs similar to RpoD from *E. coli* (-35 element TTCACA and -10 element TATAAT), whereas the promoters recognized by sigma factor group 2 (SigB,C,D,E,F) have only a consensus motif of TATAAT for the -10 element. Group 3 (sigF,G,H,I) has dissimilar motifs of the -32 element TAGGC and -12 element GGTAA (Imamura and Asayama 2009). Therefore, the trained model CyaPromBERT potentially learned and gave better attention scores to nucleotide matching the enriched motif to distinguish promoter-like and non-promoter sequences.

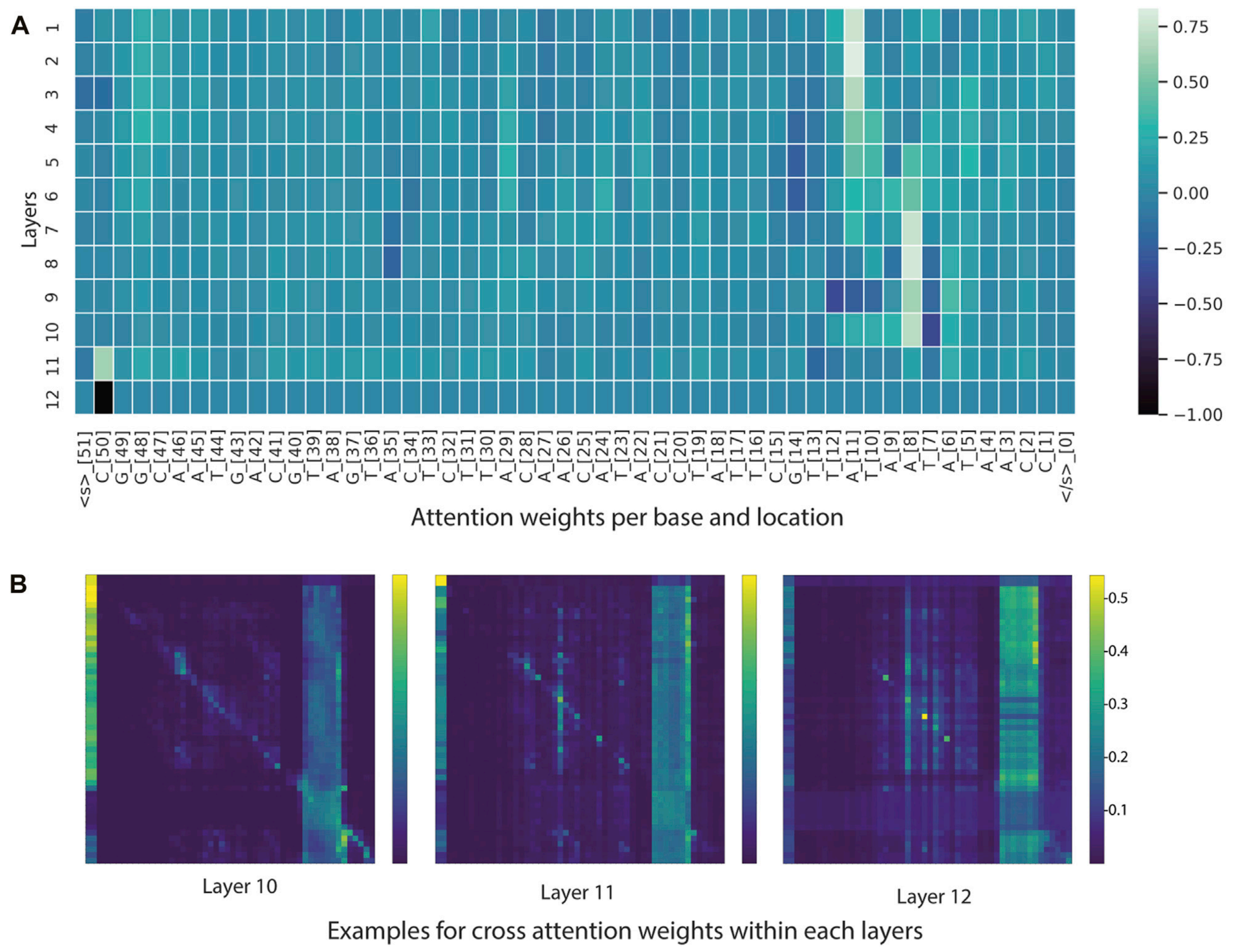
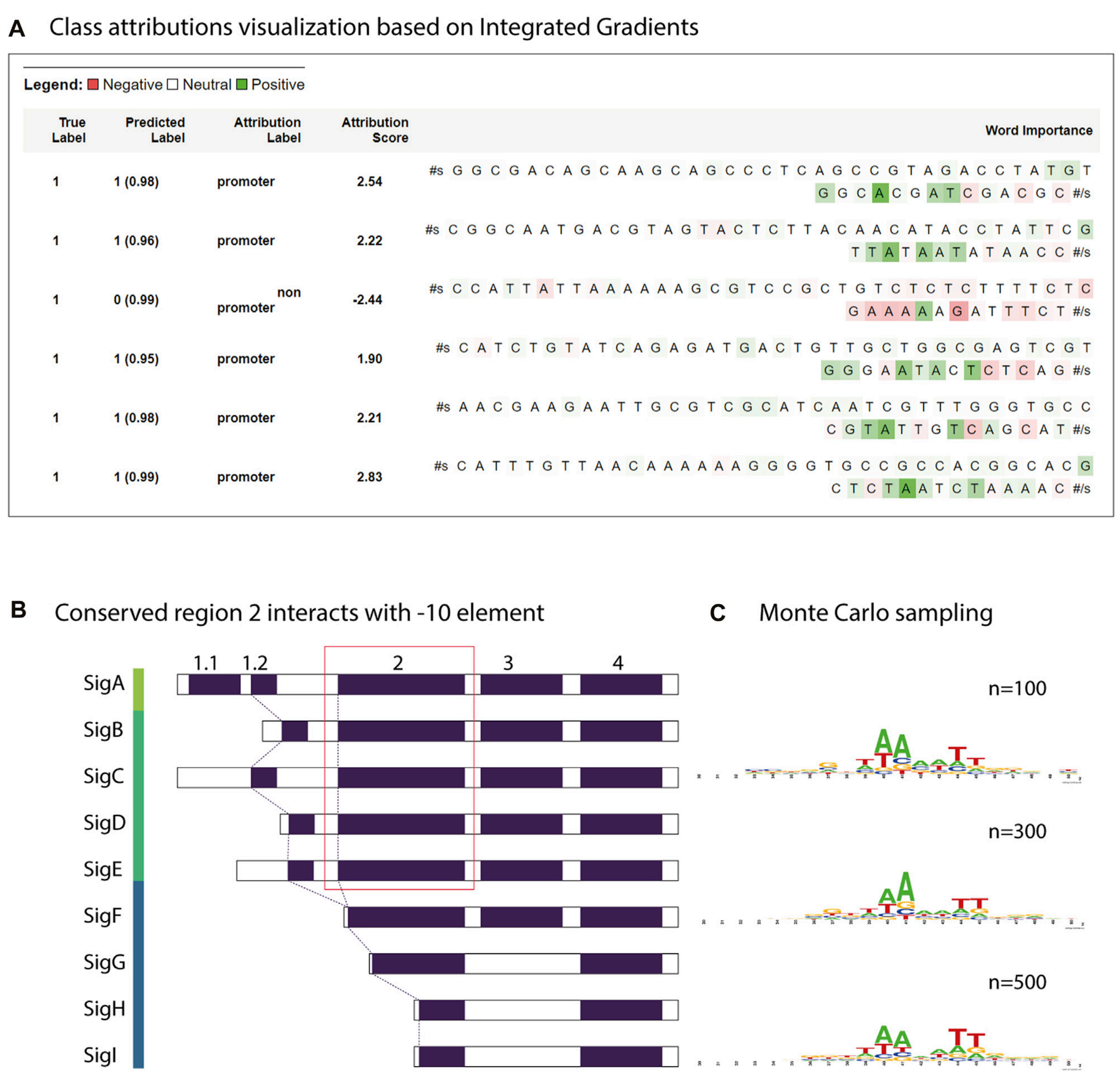


FIGURE 6
Visualization tools for model interpretability. **(A)** Heatmap based on attention scores of nucleotides (tokens) across 12 layers. **(B)** Heatmap illustrating cross-attention scores of nucleotides (tokens) in the last three layers. In the example heatmap, the self-attention modules focused on -10 element and some positions in the -35 element.

Cross-species validation through *Synechocystis* sp. PCC 6803 and *Synechocystis* sp. PCC 6714 datasets

As stated above, one of the main objectives of the current work was to use the limited dRNA-seq datasets of some model organisms that are closely related to the organisms of interest to construct curated models capable of high-performance inferencing for species with similar lineages by taking advantage of the learning transferability of deep-learning models. Therefore, we further validated the trained model using promoter and non-promoter datasets prepared from *Synechocystis* sp. PCC 6803 using TSSNote. They were from a different genus than *Synechococcus elongatus* sp. UTEX 2973. The trained model performed well on promoter prediction tasks using datasets consisting of 2840 sequences from *Synechocystis* sp. PCC 6803, with an AUROC score of 0.961 and F1 score of 0.91 (Table 4). A slight reduction in performance compared with that of *Synechococcus elongatus* sp. UTEX 2973 may be due to overfitting or differences in genomic preferences between the two species. Additionally, we trained similarly a promoter prediction model from *Synechocystis* sp. PCC 6803 and cross validated it with a closely related species *Synechocystis* sp. PCC 6714. The performance was similar but F1 scores of 0.89 were lower than those from *Synechocystis* sp. PCC 6803 (Table 5). However, it should be noted that the quality of datasets for *Synechocystis* sp. PCC 6714 was not high, leading to more noisy data. Regardless, the results still demonstrated the capability of maintaining good performance in cross-species promoter prediction from similar lineages.


A Class attributions visualization based on Integrated Gradients
B Conserved region 2 interacts with -10 element
1.1
1.2
2
3
4
SigA
SigB
SigC
SigD
SigE
SigF
SigG
SigH
SigI
C Monte Carlo sampling
n=100
n=300
n=500

Legend: 🟥 Negative ☐ Neutral 🟩 Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
1	1 (0.98)	promoter	2.54	#s G G C G A C A G C A A G C A G C C C T C A G C C G T A G A C C T A T G T G G C A C G A T C G A C G C #/s
1	1 (0.96)	promoter	2.22	#s C G G C A A T G A C G T A G T A C T C T T A C A A C A T A C C T A T T C G T T A T A A T A T A A C C #/s
1	0 (0.99)	non promoter	-2.44	#s C C A T T A T T A A A A A A G C G T C C G C T G T C T C T C T T T T C T C G A A A A A G A T T T C T #/s
1	1 (0.95)	promoter	1.90	#s C A T C T G T A T C A G A G A T G A C T G T T G C T G G C G A G T C G T G G G A A T A C T C T C A G #/s
1	1 (0.98)	promoter	2.21	#s A A C G A A G A A T T G C G T C G C A T C A A T C G T T T G G G T G C C C G T A T T G T C A G C A T #/s
1	1 (0.99)	promoter	2.83	#s C A T T T G T T A A C A A A A A A G G G G T G C C G C C A C G G C A C G C T C T A A T C T A A A A C #/s

FIGURE 7
Motif analysis using attribution weights and reverse enrichment through Monte Carlo sampling. **(A)** Class attributions visualization of a few strong promoters in *Synechococcus elongatus* sp. UTEX 2973 and a non-promoter sequence. **(B)** Transcription factor groups in *Synechocystis sp.* PCC 6803. The relatively conserved region two in group 1 and group 2 retains a motif similar to the consensus -10 element TATAAT. **(C)** The motif learned by the trained model discovered by Monte Carlo sampling.

The limitations of the pipeline and the trained model

Despite the fast construction and relatively high performance, a few limitations were present in the current work. First, for TSSNote, the quality and accuracy of promoter extraction depend on the quality of raw dRNAseq datasets and their experimental designs. The quality and performance of the trained model also depend on the quality of the inputs; therefore, selecting suitable parameters and preparing good datasets are the most important part of this pipeline. We tested the pipeline on datasets of the model acetogen *Eubacterium limosum* (Song et al., 2018). The pipeline produced a model with F1 scores of 0.88 and AUROC scores of 0.89. However, when we tested the pipeline on more dated datasets of other species, the trained models did not perform well. Second,

TABLE 4 Cross validation the performance of CyaPromBERT trained on *Synechococcus elongatus* sp. UTEX 2973 for a distantly related species Synechocystis sp. PCC 6803.

Species	AUROC		Precision		F1 score		Support	
	Promoter	Non-promoter	Promoter	Non-promoter	Promoter	Non-promoter	Promoter	Non-promoter
Synechococcus sp. UTEX 2973	0.98	0.98	0.92	0.95	0.93	0.93	1001	1036
Synechococcus sp. PCC 6803	0.96	0.96	0.88	0.94	0.91	0.91	1407	1433

TABLE 5 Cross validation the performance of CyaPromBERT trained on *Synechocystis* sp. PCC 6803 for a closely related species *Synechocystis* sp. PCC 6714.

Species	AUROC		Precision		F1 score		Support	
	Promoter	Non-promoter	Promoter	Non-promoter	Promoter	Non-promoter	Promoter	Non-promoter
Synechococcus sp. PCC 6803	0.97	0.97	0.91	0.92	0.91	0.92	364	378
Synechococcus sp. PCC 6714	0.96	0.96	0.91	0.88	0.89	0.89	330	330

despite the high performance of the test datasets and cross-validation, the trained model still suffers from false positives in the regional scanning mode. Thus, the results should be interpreted as the most potential locations, and further analyses for decision-making should be conducted. There are several possible explanations for this finding. To capture most promoters of the genera *Synechocystis* and *Synechococcus* through the learned pattern, the model focused solely on the interrelationship and composition of nucleotides in the -10 element. Therefore, the model may be confused with AT-rich promoter-like sequences. Another explanation is that transcription is a complex biological process, which is influenced by multiple factors, such as protein–DNA interactions and protein–protein interactions (DNA-binding proteins, transcription factors, enhancers, competition of sigma factors for the holoenzyme RNA polymerase), and the topographical state of the genome (chromosome folding states). The tertiary structures of chromosomes can greatly influence functional DNA-related processes, such as transcription and DNA replication (Dorman, 2019; Szabo et al., 2019). Such interactions cannot be fully captured with sequential information, which is another limitation of the current work. Regardless, the transformer architecture is a powerful building block for the construction of multimodal models; therefore, future incorporation of additional data reflecting cis/trans interactions and/or other neural networks may improve the accuracy and reduce false positives to make the model more deterministic. The pipeline and model in the current work may be used for constructing a fast and accessible promoter prediction and screening tool using a deep-learning approach, which can help reduce the time needed for downstream analyses.

Conclusion

With the rapid evolution and continuous development of next-generation sequencing techniques, an unprecedented vast amount of high-quality biological data has become increasingly accessible to researchers. This ever-expanding source of genomic data is a valuable, yet underexplored, reservoir of knowledge that can provide valuable insights into the mystery of life. Recently, methodological and computational advancements have enabled systematic and high-throughput approaches to elucidate the biological meanings of DNA sequences, in addition to traditional knowledge-based analysis. The traditional method for promoter identification involves dRNA-seq or 5′-CAGE experiments. However, despite the growing number of high-quality RNA-seq datasets, dRNA-seq experiments are still limited and expensive. In the current study, we applied and compared the performance of various

SOTA transformer-based models for promoter prediction of *Synechococcus elongatus* sp. UTEX 2973 *and Synechocystis* sp. PCC 6803. The model achieved an AUROC score of 97% and an F1 score of 92% in the validation dataset of the promoters extracted from *Synechococcus elongatus* sp. UTEX 2973 and had an AUROC score of 96% and F1 score of 91% when cross-validated using 7000 promoters from *Synechocystis* sp. PCC 6803. This finding illustrated that core promoter features are conserved in related species, and the dRNA-seq dataset of one model organism is sufficient to construct a curated promoter prediction model.

Precise promoter prediction is essential to understand the regulatory mechanisms of genes and operons. A key advantage of this study is that it can rapidly identify potential promoter sequences and regions from genomic data with high precision. The model is integrated with the visualization libraries BERTviz and Captum to visualize cross-attention weights, allowing closer observation of base-pair interactions. To increase accessibility to other researchers, both the models and pipeline were ported to the cloud-computing service Google Colab. The pipeline developed (TSSNote and PromBERT) in this study can be applied to other species and lineages to develop fast promoter prediction tools. As transformer architecture has become increasingly popular for multimodal learning, the implementation and analysis of BERT behavior in the context of genomics is another case study for developing more robust implementations of transformers for biological application.

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found in the article/supplementary material.

Author contributions

DM, Conceptualization, Methodology, Investigation, Writing—review and editing. LN, Review and editing. EL, Funding acquisition, Project administration, Supervision, Writing—review and editing.

Funding

This research was supported by the C1 Gas Refinery Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (2015M3D3A1A01064882).

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

References

Amin, R., Rahman, C. R., Ahmed, S., Sifat, M. H. R., Liton, M. N. K., Rahman, M. M., et al. (2020). iPromoter-BnCNN: a novel branched CNN-based predictor for identifying and classifying sigma promoters. *Bioinformatics* 36 (19), 4869–4875. doi:10.1093/bioinformatics/btaa609

Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., and Schmid, C. (2021). Vivit: A video vision transformer. *Proc. IEEE Int. Conf. Comput. Vis.* 30, 1811–1820. doi:10.48550/arXiv.2103.15691

Bhandari, N., Khare, S., Walambe, R., and Kotecha, K. (2021). Comparison of machine learning and deep learning techniques in promoter prediction across diverse species. *PeerJ. Comput. Sci.* 7, e365. doi:10.7717/peerj-cs.365

Bischler, T., Tan, H. S., Nieselt, K., and Sharma, C. M. (2015). Differential RNA-seq (dRNA-seq) for annotation of transcriptional start sites and small RNAs in *Helicobacter pylori. Methods* 86, 89–101. doi:10.1016/j.ymeth.2015.06.012

Burden, S., Lin, Y.-X., and Zhang, R. (2005). Improving promoter prediction for the NNPP2.2 algorithm: A case study using *Escherichia coli* DNA sequences. *Bioinformatics* 21 (5), 601–607. doi:10.1093/bioinformatics/bti047

Butler, J. E., and Kadonaga, J. T. (2002). The RNA polymerase II core promoter: A key component in the regulation of gene expression. *Genes Dev.* 16 (20), 2583–2592. doi:10.1101/gad.1026202

Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., et al. (2009). Biopython: Freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* 25 (11), 1422–1423. doi:10.1093/bioinformatics/btp163

da Silva, K. P., Monteiro, M. I., and de Souto, M. C. P. (2006). "*In silico* prediction of promoter sequences of Bacillus species," in The 2006 IEEE Proc. Int. Jt. Conf. Neural Netw., Vancouver, BC, Canada, 16-21 July 2006 (IEEE), 1–5.

Danecek, P., Bonfield, J. K., Liddle, J., Marshall, J., Ohan, V., Pollard, M. O., et al. (2021). Twelve years of SAMtools and BCFtools. *Gigascience* 10 (2), giab008. doi:10.1093/gigascience/giab008

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv: 1810.04805*.

Di Salvo, M., Pinatel, E., Talà, A., Fondi, M., Peano, C., and Alifano, P. (2018). G4PromFinder: An algorithm for predicting transcription promoters in GC-rich bacterial genomes based on AT-rich elements and G-quadruplex motifs. *BMC Bioinforma.* 19 (1), 36–11. doi:10.1186/s12859-018-2049-x

Dorman, C. J. (2019). DNA supercoiling and transcription in bacteria: A two-way street. *BMC Mol. Cell Biol.* 20 (1), 26–29. doi:10.1186/s12860-019-0211-6

Dugar, G., Herbig, A., Förstner, K. U., Heidrich, N., Reinhardt, R., Nieselt, K., et al. (2013). High-resolution transcriptome maps reveal strain-specific regulatory features of multiple Campylobacter jejuni isolates. *PLoS Genet.* 9 (5), e1003495. doi:10.1371/journal.pgen.1003495

Dzabraev, M., Kalashnikov, M., Komkov, S., and Petiushko, A. (2021). "Mdmmt: Multidomain multimodal transformer for video retrieval," in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., Nashville, TN, USA, 19-25 June 2021 (IEEE), 1–5.

Feklistov, A., and Darst, S. A. (2011). Structural basis for promoter– 10 element recognition by the bacterial RNA polymerase σ subunit. *Cell* 147 (6), 1257–1269. doi:10.1016/j.cell.2011.10.041

Floridi, L., and Chiriatti, M. (2020). GPT-3: Its nature, scope, limits, and consequences. *Minds Mach. (Dordr).* 30 (4), 681–694. doi:10.1007/s11023-020-09548-1

Gordon, L., Chervonenkis, A. Y., Gammerman, A. J., Shahmuradov, I. A., and Solovyev, V. V. (2003). Sequence alignment kernel for recognition of promoter regions. *Bioinformatics* 19 (15), 1964–1971. doi:10.1093/bioinformatics/btg265

He, W., Jia, C., Duan, Y., and Zou, Q. (2018). 70ProPred: A predictor for discovering sigma70 promoters based on combining multiple features. *BMC Syst. Biol.* 12 (4), 44–107. doi:10.1186/s12918-018-0570-1

Huerta, A. M., and Collado-Vides, J. (2003). Sigma70 promoters in *Escherichia coli*: Specific transcription in dense regions of overlapping promoter-like signals. *J. Mol. Biol.* 333 (2), 261–278. doi:10.1016/j.jmb.2003.07.017

Ikeuchi, M., and Tabata, S. (2001). Synechocystis sp. PCC 6803—A useful tool in the study of the genetics of cyanobacteria. *Photosynth. Res.* 70 (1), 73–83. doi:10.1023/A:1013887908680

Imamura, S., and Asayama, M. (2009). Sigma factors for cyanobacterial transcription. *Gene Regul. Syst. Bio.* 3, 65–87. doi:10.4137/grsb.s2090

Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. (2021). Dnabert: Pre-trained bidirectional encoder representations from transformers model for DNA-language in genome. *Bioinformatics* 37 (15), 2112–2120. doi:10.1093/bioinformatics/btab083

Kans, J. (2022). *Entrez direct: E-Utilities on the UNIX command line. Entrez* programming *utilities help [internet]*. Bethesda, Maryland, United States: National Center for Biotechnology Information.

Kato, Y., and Hasunuma, T. (2021). *Metabolic engineering for carotenoid production using eukaryotic microalgae and prokaryotic cyanobacteria. Carotenoids: Biosynthetic and Biofunctional Approaches.* Berlin, Germany: Springer, 121–135.

Kim, D., Paggi, J. M., Park, C., Bennett, C., and Salzberg, S. L. (2019). Graph-based genome alignment and genotyping with HISAT2 and HISAT-genotype. *Nat. Biotechnol.* 37 (8), 907–915. doi:10.1038/s41587-019-0201-4

Kokhlikyan, N., Miglani, V., Martin, M., Wang, E., Alsallakh, B., Reynolds, J., et al. (2020). "Captum: A unified and generic model interpretability library for pytorch." *arXiv preprint arXiv:2009.07896.*

Le, N. Q. K., Yapp, E. K. Y., Nagasundaram, N., and Yeh, H.-Y. (2019). Classifying promoters by interpreting the hidden information of DNA sequences via deep learning and combination of continuous fasttext N-grams. *Front. Bioeng. Biotechnol.* 305, 305. doi:10.3389/fbioe.2019.00305

Li, F., Chen, J., Ge, Z., Wen, Y., Yue, Y., Hayashida, M., et al. (2021). Computational prediction and interpretation of both general and specific types of promoters in *Escherichia coli* by exploiting a stacked ensemble-learning framework. *Brief. Bioinform.* 22 (2), 2126–2140. doi:10.1093/bib/bbaa049

Lin, P.-C., Zhang, F., and Pakrasi, H. B. (2020). Enhanced production of sucrose in the fast-growing cyanobacterium Synechococcus elongatus UTEX 2973. *Sci. Rep.* 10 (1), 390–398. doi:10.1038/s41598-019-57319-5

Liu, B., Yang, F., Huang, D.-S., and Chou, K.-C. (2018). iPromoter-2L: a two-layer predictor for identifying promoters and their types by multi-window-based PseKNC. *Bioinformatics* 34 (1), 33–40. doi:10.1093/bioinformatics/btx579

Luan, G., Zhang, S., Wang, M., and Lu, X. (2019). Progress and perspective on cyanobacterial glycogen metabolism engineering. *Biotechnol. Adv.* 37 (5), 771–786. doi:10.1016/j.biotechadv.2019.04.005

Mann, S., Li, J., and Chen, Y.-P. P. (2007). A pHMM-ANN based discriminative approach to promoter identification in prokaryote genomic contexts. *Nucleic Acids Res.* 35 (2), e12. doi:10.1093/nar/gkl1024

Mueller, T. J., Ungerer, J. L., Pakrasi, H. B., and Maranas, C. D. (2017). Identifying the metabolic differences of a fast-growth phenotype in Synechococcus UTEX 2973. *Sci. Rep.* 7 (1), 41569–41578. doi:10.1038/srep41569

Nguyen, N. G., Tran, V. A., Phan, D., Lumbanraja, F. R., Faisal, M. R., Abapihi, B., et al. (2016). DNA sequence classification by convolutional neural network. *J. Biomed. Sci. Eng.* 9 (5), 280–286. doi:10.4236/jbise.2016.95021

Oubounyt, M., Louadi, Z., Tayara, H., and Chong, K. T. (2019). DeePromoter: Robust promoter predictor using deep learning. *Front. Genet.* 10, 286. doi:10.3389/fgene.2019.00286

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process Syst.* 32, 8026–8037.

Pattharaprachayakul, N., Choi, J.-i., Incharoensakdi, A., and Woo, H. M. (2020). Metabolic engineering and synthetic biology of cyanobacteria for carbon capture and utilization. *Biotechnol. Bioprocess Eng.* 25 (6), 829–847. doi:10.1007/s12257-019-0447-1

Qiao, Y., Wang, W., and Lu, X. (2020). Engineering cyanobacteria as cell factories for direct trehalose production from CO2. *Metab. Eng.* 62, 161–171. doi:10.1016/j.ymben.2020.08.014

Rahman, M., Aktar, U., Jani, M. R., and Shatabda, S. (2019). iPro70-FMWin: identifying Sigma70 promoters using multiple windowing and minimal features. *Mol. Genet. Genomics* 294 (1), 69–84. doi:10.1007/s00438-018-1487-5

Rahman, M. S., Aktar, U., Jani, M. R., and Shatabda, S. (2019). iPromoter-FSEn: Identification of bacterial σ70 promoter sequences using feature subspace based ensemble classifier. *Genomics* 111 (5), 1160–1166. doi:10.1016/j.ygeno.2018.07.011

Rangannan, V., and Bansal, M. (2010). High-quality annotation of promoter regions for 913 bacterial genomes. *Bioinformatics* 26 (24), 3043–3050. doi:10.1093/bioinformatics/btq577

Roh, H., Lee, J. S., Choi, H. I., Sung, Y. J., Choi, S. Y., Woo, H. M., et al. (2021). Improved CO2-derived polyhydroxybutyrate (PHB) production by engineering fast-growing cyanobacterium Synechococcus elongatus UTEX 2973 for potential utilization of flue gas. *Bioresour. Technol.* 327, 124789. doi:10.1016/j.biortech.2021.124789

Santos-Merino, M., Torrado, A., Davis, G. A., Röttig, A., Bibby, T. S., Kramer, D. M., et al. (2021). Improved photosynthetic capacity and photosystem I oxidation via heterologous metabolism engineering in cyanobacteria. *Proc. Natl. Acad. Sci. U. S. A.* 118 (11), e2021523118. doi:10.1073/pnas.2021523118

Sarnaik, A., Abernathy, M. H., Han, X., Ouyang, Y., Xia, K., Chen, Y., et al. (2019). Metabolic engineering of cyanobacteria for photoautotrophic production of heparosan, a pharmaceutical precursor of heparin. *Algal Res.* 37, 57–63. doi:10.1016/j.algal.2018.11.010

Song, K., Tan, X., Liang, Y., and Lu, X. (2016). The potential of Synechococcus elongatus UTEX 2973 for sugar feedstock production. *Appl. Microbiol. Biotechnol.* 100 (18), 7865–7875. doi:10.1007/s00253-016-7510-z

Song, Y., Shin, J., Jin, S., Lee, J.-K., Kim, D. R., Kim, S. C., et al. (2018). Genome-scale analysis of syngas fermenting acetogenic bacteria reveals the translational regulation for its autotrophic growth. *BMC Genomics* 19 (1), 837–915. doi:10.1186/s12864-018-5238-0

Szabo, Q., Bantignies, F., and Cavalli, G. (2019). Principles of genome folding into topologically associating domains. *Sci. Adv.* 5 (4), eaaw1668. doi:10.1126/sciadv.aaw1668

Tan, X., Hou, S., Song, K., Georg, J., Klähn, S., Lu, X., et al. (2018). The primary transcriptome of the fast-growing cyanobacterium Synechococcus elongatus UTEX 2973. *Biotechnol. Biofuels* 11 (1), 218–317. doi:10.1186/s13068-018-1215-8

Taylor, G. M., and Heap, J. T. (2020). "Combinatorial metabolic engineering platform enabling stable overproduction of lycopene from carbon dioxide by cyanobacteria." BioRxiv.

Towsey, M., Timms, P., Hogan, J., and Mathews, S. A. (2008). The cross-species prediction of bacterial promoters using a support vector machine. *Comput. Biol. Chem.* 32 (5), 359–366. doi:10.1016/j.compbiolchem.2008.07.009

Tsai, Y.-H. H., Bai, S., Liang, P. P., Kolter, J. Z., Morency, L.-P., and Salakhutdinov, R. (2019). "Multimodal transformer for unaligned multimodal language sequences," in Proceedings of The Conference Association for Computational Linguistics Meeting, 17 November 2019 (Italy: NIH Public Access), 6558–6569.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Adv. Neural Inf. Process Syst.* 30, 15.

Vig, J. (2019). BertViz: A tool for visualizing multihead self-attention in the BERT model. arXiv.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., et al. (2020). "Transformers: State-of-the-art natural language processing," in Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations, 16 October 2020 (Italy: Association for Computational Linguistics), 38–45.

Wu, B., Xu, C., Dai, X., Wan, A., Zhang, P., Yan, Z., et al. (2020). "Visual transformers: Token-based image representation and processing for computer vision." *arXiv preprint arXiv:2006.03677*.

Xiao, X., Xu, Z.-C., Qiu, W.-R., Wang, P., Ge, H.-T., and Chou, K.-C. (2019). iPSW (2L)-PseKNC: a two-layer predictor for identifying promoters and their strength by hybrid features via pseudo K-tuple nucleotide composition. *Genomics* 111 (6), 1785–1793. doi:10.1016/j.ygeno.2018.12.001

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *Adv. Neural Inf. Process Syst.* 32, 1–10.

Yu, J., Li, J., Yu, Z., and Huang, Q. (2019). Multimodal transformer with multi-view visual representation for image captioning. *IEEE Trans. Circuits Syst. Video Technol.* 30 (12), 4467–4480. doi:10.1109/tcsvt.2019.2947482

Zhang, M., Jia, C., Li, F., Li, C., Zhu, Y., Akutsu, T., et al. (2022). Critical assessment of computational tools for prokaryotic and eukaryotic promoter prediction. *Brief. Bioinform.* 23 (2), bbab551. doi:10.1093/bib/bbab551

Zhang, M., Li, F., Marquez-Lago, T. T., Leier, A., Fan, C., Kwoh, C. K., et al. (2019). MULTiPly: A novel multi-layer predictor for discovering general and specific types of promoters. *Bioinformatics* 35 (17), 2957–2965. doi:10.1093/bioinformatics/btz016

Zhou, D., Kang, B., Jin, X., Yang, L., Lian, X., Jiang, Z., et al. (2021). "Deepvit: Towards deeper vision transformer." *arXiv preprint arXiv:2103.11886*.

Zhu, Y., Li, F., Xiang, D., Akutsu, T., Song, J., and Jia, C. (2021). Computational identification of eukaryotic promoters based on cascaded deep capsule neural networks. *Brief. Bioinform.* 22 (4), bbaa299. doi:10.1093/bib/bbaa299

TYPE Original Research
PUBLISHED 14 December 2022
DOI 10.3389/fgene.2022.1081842

Predicting gene expression from histone modifications with self-attention based neural networks and transfer learning

Yuchi Chen, Minzhu Xie* and Jie Wen

College of Information Science and Engineering, Hunan Normal University, Changsha, China

Check for updates

OPEN ACCESS

EDITED BY
Quan Zou,
University of Electronic Science and Technology of China, China

REVIEWED BY
Wen Zhang,
Huazhong Agricultural University, China
Feng Fang,
University of Houston, United States
Qiong Zhang,
Affiliated Hospital of Nantong University, China

*CORRESPONDENCE
Minzhu Xie,
xieminzhu@hunnu.edu.cn

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 27 October 2022
ACCEPTED 28 November 2022
PUBLISHED 14 December 2022

CITATION
Chen Y, Xie M and Wen J (2022), Predicting gene expression from histone modifications with self-attention based neural networks and transfer learning.
Front. Genet. 13:1081842.
doi: 10.3389/fgene.2022.1081842

It is well known that histone modifications play an important part in various chromatin-dependent processes such as DNA replication, repair, and transcription. Using computational models to predict gene expression based on histone modifications has been intensively studied. However, the accuracy of the proposed models still has room for improvement, especially in cross-cell lines gene expression prediction. In the work, we proposed a new model TransferChrome to predict gene expression from histone modifications based on deep learning. The model uses a densely connected convolutional network to capture the features of histone modifications data and uses self-attention layers to aggregate global features of the data. For cross-cell lines gene expression prediction, TransferChrome adopts transfer learning to improve prediction accuracy. We trained and tested our model on 56 different cell lines from the REMC database. The experimental results show that our model achieved an average Area Under the Curve (AUC) score of 84.79%. Compared to three state-of-the-art models, TransferChrome improves the prediction performance on most cell lines. The experiments of cross-cell lines gene expression prediction show that TransferChrome performs best and is an efficient model for predicting cross-cell lines gene expression.

KEYWORDS

gene expression, histone modification, deep learning, transfer learning, convolutional neural network

Introduction

Understanding the patterns of gene regulation has been one of the major focuses of biological research. A variety of biological factors are thought to be involved in the regulation of gene expression. The regulatory factors usually include transcription factors, cis-regulatory elements, and epigenetic modifications. As a type of epigenetic modifications, histone modification plays an important role in gene expression regulation (Gibney and Nolan, 2010). Nucleosome is the building block of a chromosome, which consists of an octamer of histones and 147 base pair (bp) DNA wrapping around the octamer. Since histone is a core component of the nucleosome, histone modifications directly affect the structure of chromatin and control the expression intensity of nearby genes. Recently, a great number of researches have shown that histone modifications have a great impact on gene expression, chromosome inactivation, replication, and cell differentiation (Krajewski, 2022; Lin et al., 2022). There are a variety of histone modification marks at different chromosomal locations, and there may be a set of "codes" of histone modifications to control gene expression (Peterson and Laniel, 2004). Due to the high-throughput sequencing technologies, a huge amount of histone modifications data and gene expression data are available, and using computational algorithms to predict gene expression based on histone modifications is feasible.

To date, a variety of computational methods have been used to predict gene expression based on gene regulatory factors. For example, Beer and Tavazoie, (2004) used Bayesian networks to predict gene expression from DNA sequences. Ouyang et al. (2009) used a linear regression model to predict gene expression based on 12 transcription factors. Zeng et al. (2020) combined the information of proximal promoter and distal enhancer to predict gene expression. In 2010, Karlić et al.(2010) found histone modification levels and gene expression are well correlated, and derived quantitative models to predict gene expression from histone modifications. Li et al. (2015) used a machine learning method to predict gene expression in lung cancer from multiple epigenetic data such as CpG methylation, histone H3 methylation modification and nucleotide composition. In 2016, Singh et al. (Singh et al., 2016) used a convolutional neural network (CNN) DeepChrome to predict gene expression based on five critical histone modification marks. To improve prediction accuracy, they (Singh et al., 2017) integrated attention mechanism into a neural network and proposed a prediction model AttentiveChrome. Temporal Convolutional Network (Zhu et al., 2018; Kamal et al., 2020) is also utilized to predict the gene expression from histone modifications. In 2022, Hamdy et al. (2022) proposed three variations of CNN models called ConvChrome.

Though the above methods have achieved good performances, there are still room for improvement, and some recently emerging technologies have provided some ways. When models are trained and tested on different cell lines which is knowns as cross-cell lines prediction, the model performance is always compromised. For example, compared to training and testing on the same cell line dataset, the average prediction accuracy of DeepChrome trained on other cell lines is 2.3% lower. Because of the large variety of cell lines, it is difficult to obtain histone modification data and gene expression data for all types of cell lines. Therefore predicting gene expression using models trained on other cell lines is useful and in urgent need.

Transfer learning is a machine learning technique in which a model trained on a specific task is reused as part of the training process for another similar task (Tan et al., 2018). Transfer learning allows training and prediction using the dataset from different sources with similar characteristics and significantly reduces dataset bias. Transfer learning has achieved great success in prediction tasks that require learning transfer features (Sun et al., 2022; Zhu et al., 2022). In the field of bioinformatics, transfer learning enables existing trained models to efficiently work on similar datasets that are lack of labels, which reduces the cost of biological experiments.

In the paper, we propose a neural network model TransferChrome with self-attention mechanism and transfer learning to predict gene expression based on histone modifications data. TransferChrome uses neural network layers with self-attention mechanism to capture global contextual information of data. In order to correct the data bias of cross-cell lines gene expression prediction, we used transfer learning. The experimental results show that TransferChrome achieved an average Area Under the Curve (AUC) score of 84.79%, which is better than other 3 state-of-the-art similar models. The cross-cell lines prediction experiments also show that TransferChrome outperforms other models.

TABLE 1 Five core histone modification marks and their functional categories.

Histone mark	Associated regions	Functional category
H3K27me3	Polycomb repression	Repressor mark
H3K36me3	Transcribed regions	Structural mark
H3K4me1	Enhancer regions	Distal mark
H3K4me3	Promoter regions	Promoter mark
H3K9me3	Heterochromatin regions	Repressor mark

Materials and method

Data collection and processing

The experimental data comes from the Roadmap Epigenome Project (REMC) (Kundaje et al., 2015), which consists of 56 cell lines' histone modifications data and the corresponding normalized RPKM expression data of 17170 samples. Same as DeepChrome (Singh et al., 2016), five histone modification marks that play important roles in gene expression were selected for our experiments. These 5 marks include H3K4me3, H3K4me1, H3K36me3, H3K27me3, and H3K9me3. Their functional categories are summarized in Table 1. Each sample in the dataset represents a gene. The data of one sample include the five histone modification marks signal within 10000bp upstream and downstream of the transcription start sites (TSSs) of the corresponding gene.

According to DeepChrome (Singh et al., 2016), the 10000 bp is equally divided into 100 bins and the histone modifications data of one sample is encoded into an $n \times m$ matrix x, where n is

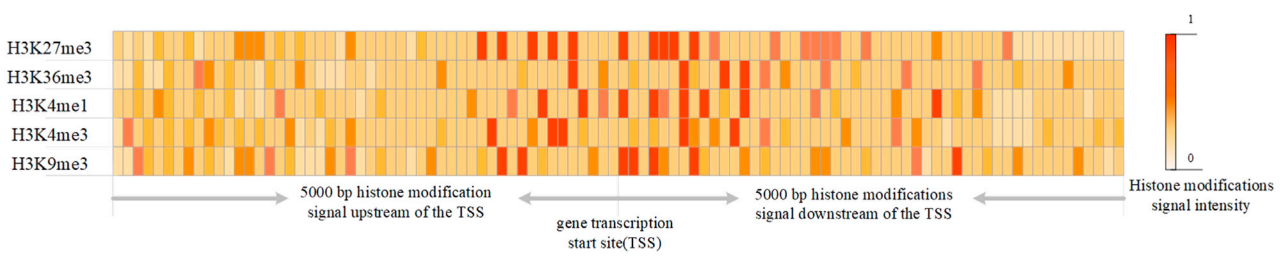


FIGURE 1
The data structure representing histone modifications.

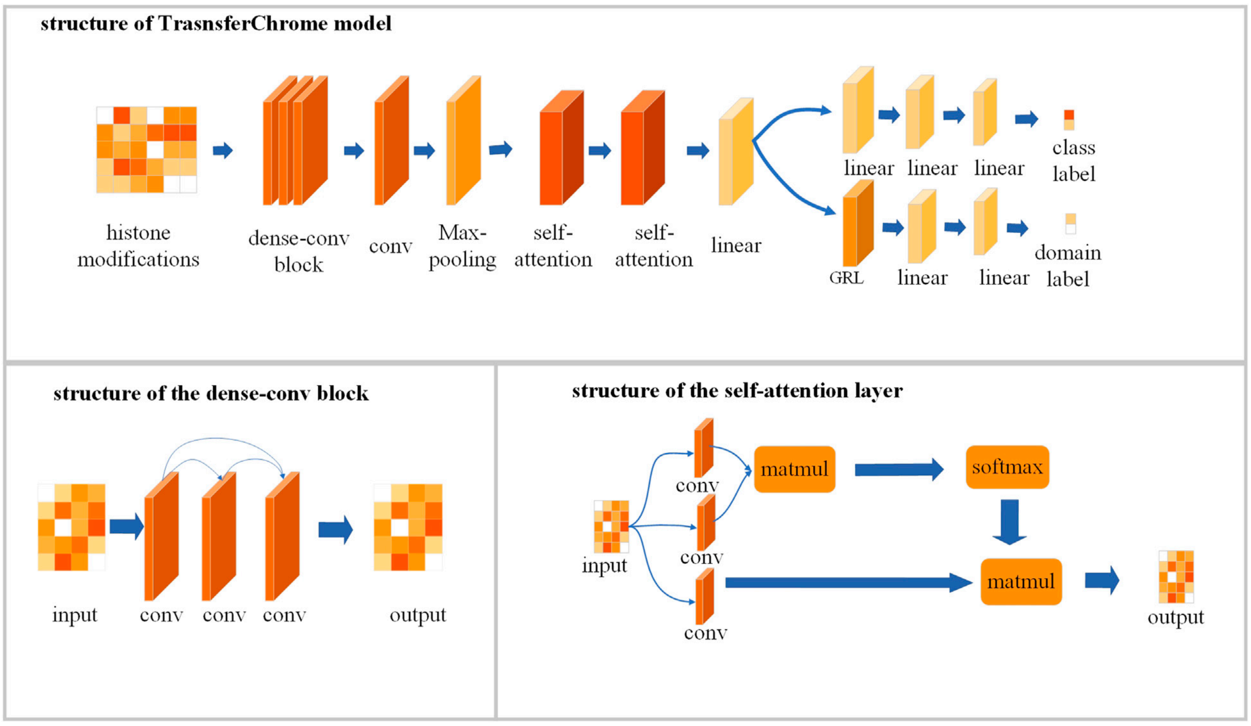


FIGURE 2
The model structure of TransferChrome.

the number of histone modification marks and m is the number of bins (see Figure 1). The histone sequencing data provided by REMC were quantified by BEDTools into histone modification signals. Therefore, $x_{i,j}$ represents the signal of the j-th histone modification mark in the i-th bin.

Since the normalization of training data can speed up the convergence of model training and allows the model to fit the data better (Singh and Singh, 2020), the z-score method is used to normalize the data for each histone modification mark as Eq. 1. In Eq. 1, $\hat{x}_{i,h}$ represents the normalized signal of h-th histone modification mark in the i-th bin. $\bar{x}_h$ and σ_h denote the mean and standard deviation of the signals of the h-th histone modification mark of all genes in a cell line.

$$\hat{x}_{i,h} = \frac{x_{i,h} - \bar{x}_h}{\sigma_h} \tag{1}$$

According to previous studies (Singh et al., 2016), each gene is assigned a label based on its expression value. The median of expression values of all genes in a given cell line is denoted as t. If the expression value of a gene is higher than or equal to t, it is labeled with 1; otherwise it is labeled with 0.

To avoid interfering from adjacent genes, those genes whose TSSs are within 5000 bp downstream of previous kept genes' TSSs are deleted. At last there are 17170 genes remained.

Design of neural network model

As shown in Figure 2, TransferChrome is composed of multiple modules: a feature extraction module, a label classification module and a domain classification module. The feature extraction module is used to calculate the latent features of data. It includes a dense-conv block, a 1D convolutional layer, a 1D max pooling layer, two self-attention layers, and a linear layer (also called fully connected layer or dense layer). The label classification module predicts a gene expression label. It includes three linear layers. The domain classification module predicts a domain label, which includes a gradient reversal layer (GRL) and two linear layers. It learns transfer features, which allows the model to achieve better performance in cross-cell lines gene expression prediction.

Dense-connected convolutional layer for extracting local features of data

To make the model better capture the features of the data, we optimized the convolutional neural network in the feature extraction module. The convolutional neural network uses feature detectors, also known as convolution kernals or filters to capture data's features. According to its size, a convolution kernal will aggregate all the information in the receptive field to extract a corresponding feature. By increasing the number of convolutional layers, a model can learn more complex features. However, the deepened network structure easily ignores the features captured by earlier convolutional layers, which usually represent the simple but also basic features of the data. Densely Connected Convolutional Networks (DenseNet) (Huang et al., 2017) uses a method called dense connectivity pattern enhances the reusability of features. Compared with the classic Convolutional Network, DenseNet connects convolutional layers densely so that the feature extracted by each layer could be used repeatedly. DenseNet performs a deep supervision and strengthen the weights of features captured by earlier convolutional layers. Inspired by DenseNet, a dense-conv block that contains several densely connected convolutional layers is used to extract features in our model. A dense-connected convolutional layer is a convolutional layer which connected to all other convolutional layers directly. It means that the input of a dense connected convolutional layer not only comes from its adjacent convolutional layer, but also from other preceding layers. The dense-conv block allows the model to learn the complex features of the data while also ensuring that the low-level convolutional layer retains a greater influence in the model's decision-making.

Let x_l be the output of the lth densely connected convolutional layer. The input of the lth dense connected convolutional layer contains the outputs of all the previous l–1 layers as Eq. 2 shows.

$$x_l = H([x_0, x_1, \ldots\ldots x_{l-1}]) \tag{2}$$

$[x_0, x_1, \ldots\ldots x_{l-1}]$ refers to the concatenation of the feature-maps output from preceding layers. The composite function H concludes a rectified linear unit (RELU) and 1D convolutional layer.

In TransferChrome, the dense-conv block consists of three dense-connected convolutional layers (kernal number = 32, 16, 8 and kernal length = 5, 5, 5). The dense-conv block is followed by a convolutional layer (kernal number = 50 and kernal length = 5) and a max-pooling layer (kernal length = 2). A dropout layer is added after each convolutional layer and the dropout rate is 0.4. The output of the max-pooling layer is input into a following self-attention layer.

Self-attention layer for aggregating global information

Regulatory factors at different locations may interact and act on gene expression together. Therefore effective integration of upstream and downstream information in the genome usually leads to better computational results (Ji et al., 2021). The Transformer (Vaswani et al., 2017) is an efficient neural network model which has achieved good results in many fields such as natural language processing (Devlin et al., 2019) and image recognition (Dosovitskiy et al., 2021). The self-attention mechanism used in Transformer can effectively integrate data's global features. The self-attention mechanism also has been widely adopted in the field of Bioinformatics (Avsec et al., 2021; Ji et al., 2021). For example, the researchers used this mechanism to significantly improve the regulatory elements prediction from genomic DNA sequences (Avsec et al., 2021). Previous experiments (Singh et al., 2017, 2016) have illustrated that histone modifications closer to gene's TSS have greater influence in the gene expression. We add self-attention mechanism to the model, and use a position encoding function to concatenate input data with relative distance information. The relative distance information contains the relative distance between each point and the TSS point, and the output of the position encoding function is denoted by x.

TransferChrome contains two self-attention layers to capture the long-distance dependence. The function of each self-attention layer is as Eqs 3–6.

$$Q = conv_q(x) \tag{3}$$

$$K = conv_k(x) \tag{4}$$

$$V = conv_v(x) \tag{5}$$

$$Attention(Q, K, V) = soft\max(QK^T)V \tag{6}$$

Each self-attention layer uses three one-dimensional convolutional layers (kernal length = 1) $conv_q$, $conv_k$, and $conv_v$ to calculate a query matrix Q, a key matrix K and a latent variable matrix V, respectively. The number of output channels of $conv_q$ and $conv_k$ is half of the number of input channels, and the number of output channels of $conv_v$ is equal to the number of input channels. Then the self-attention layer calculate data's attention score matrix QK^T by multiplying (matmul) Q and K. Finally, the attention score matrix will be normalized by a softmax function, and multiplied with V. At the end of the feature extraction module, there is a linear layer following the last self-attention layer. The feature extraction module outputs a low-dimensional feature vector. Then the feature vector is inputted into the label classification module and the domain classification module at the same time.

Label classification module and domain classification module

The label classification module predicts the gene expression label of the sample, which is the main task of our model. It is a binary classification task with 0 and 1 represent low expression and high expression, respectively. According to Long et al. (2015) and Ganin and Lempitsky (2015), domain adaption can improve prediction accuracy in transfer learning. Therefore, TransferChrome uses a domain classification module for cross-cell lines prediction. It contains a GRL and two linear layers. GRL acts as an identity transform in the forward propagation of the model. In the backward propagation, GRL takes the gradient from the subsequent layer and multiplies it by a parameter $-\lambda$ with $\lambda > 0$ and passes it to the preceding layer.

The domain classification module predicts whether the sample belongs to the target domain or the source domain. Source domains are the cell lines whose genes have gene expression labels, and the cell lines whose gene have no known gene expression information are called target domains. It is also a binary classification task with 0 and 1, where 0 indicates that the sample is from the target domain and 1 indicates that the sample is from the source domain. In cross-cell lines prediction, we try to extract those features that can not be used to discern the data domain.

Model training

For model training, we chose cross entropy as the loss function:

$$L = -[y\log\hat{y} + (1-y)\log(1-\hat{y})]. \tag{7}$$

Let G_f and θ_f be the function and the parameters of the feature extraction module, respectively. Let G_d (G_y) and θ_d (θ_y) be the function and the parameters of the domain (label) classification module, respectively. For the single cell line gene expression prediction task, the optimization goal of model training is to minimize the loss L_y of the label classification module without considering the domain classification module.

For the cross-cell lines gene expression prediction task, we train TransferChrome using the complete dataset from a source domain and a part of the dataset from a target domain to capture transfer features in different cell lines, and aim to minimize the objective function in Eq. 8.

$$\begin{aligned} E(\theta_f, \theta_y, \theta_d) &= \sum_{i=1}^{N} L_y\left(G_y\left(G_f\left(x_i;\theta_f\right);\theta_y\right), y_i\right) \\ &\quad -\lambda \sum_{i=1}^{N} L_d\left(G_d\left(G_f\left(x_i;\theta_f\right);\theta_d\right), d_i\right) \\ &= \sum_{i=1}^{N} L_y^i\left(\theta_f, \theta_y\right) - \lambda \sum_{i=1}^{N} L_d^i\left(\theta_f, \theta_d\right), \end{aligned} \tag{8}$$

where L_d is the loss function of the domain classification module.

In the training process, stochastic gradient descent (SGD) is used to update θ_y and θ_d to minimize the label classification loss L_y and L_d. In the backward propagation, the first layer GRL of the domain classification module reverses the gradient by multiplying a negative number $-\lambda$ and backward propagates it to the feature extraction module. After the model training, G_f is expected to extract transfer features in different cell lines. In the training process, the learning rate is set to 0.001, momentum is 0.85, and weight decay is 0.001. We set the max training epochs to 200 and adopted early stop strategy.

In the following single-cell line prediction experiments, each cell line data was partitioned into a training set, a validation set and a test set as DeepChrome (Singh et al., 2016). For cross-cell lines prediction, we used the source domain data and half of the target domain data to train our model, and used the other half of the target domain data as the test set.

Experiments

Comparison with other existing state-of-the-art methods

To evaluate the effectiveness of TransferChrome, we compared it with three state-of-the-art models (DeepChrome, AttentiveChrome, and ConvChrome_CNN1D). DeepChrome (Singh et al., 2016) is a convolutional neural network. It consists of a convolutional layer (convolution kernal size is 10, the number of convolution kernals is 50), a max pooling layer (convolution kernal size is 10), and two fully connected layers (the number of units is 900, 125). AttentiveChrome (Singh et al., 2017) is a recurrent neural network that uses two attention mechanisms. ConvChrome (Hamdy et al., 2022) includes three

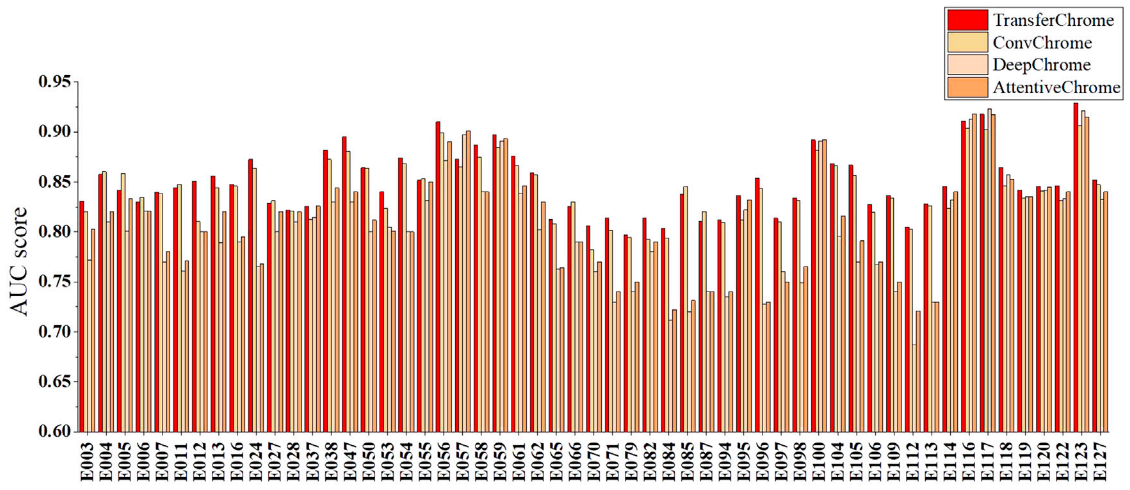


FIGURE 3
Single cell line gene expression prediction performance comparison on 56 cell lines of the models.

TABLE 2 The minimum, mean, maximum, and median of the AUC scores of single cell line gene expression prediction of the models on 56 cell lines.

	Min	Mean	Max	Median
TransferChrome	0.7972	0.8479	0.9289	0.8449
ConvChrome	0.7820	0.8399	0.9061	0.8386
DeepChrome	0.6871	0.8003	0.9236	0.8019
AttentiveChrome	0.7221	0.8093	0.9197	0.8216

variations of CNN models, among which ConvChrome_CNN1D achieved the best performance. In the experiments, all models used the same five types of core histone modifications from the REMC project to predict gene expression. In the experiments of single-cell gene expression prediction, we did not use the domain classification module of TransferChrome and only used the label classification module. DeepChrome was implemented and trained according to Singh's paper (Singh et al., 2016). For AttentiveChrome, we used the trained model downloaded from http://kipoi.org/models/AttentiveChrome/. Because

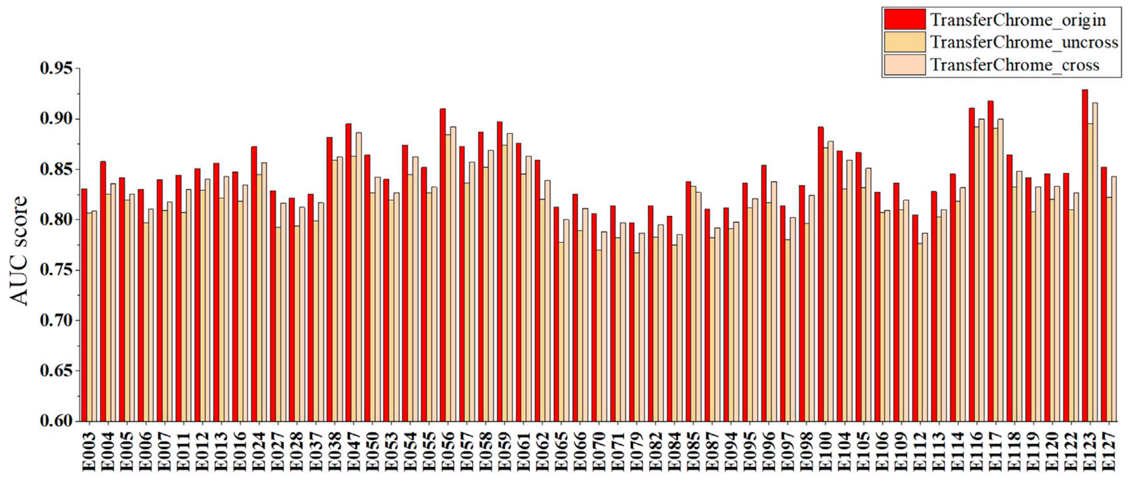


FIGURE 4
The performance comparison of different versions of TransferChrome: TransferChrome_cross, TransferChrome_uncross and TransferChrome_origin.

TABLE 3 Comparison of the minimum, mean, maximum, and median of the AUC scores of TransferChrome, DeepChrome and ConvChrome in single cell line and cross-cell lines gene expression predictions.

	Min	Mean	Max	Median
TransferChrome_origin	0.7972	0.8479	0.9289	0.8449
TransferChrome_uncross	0.7672	0.8185	0.8950	0.8182
TransferChrome_cross	0.7866	0.8330	0.9156	0.8300
DeepChrome	0.6871	0.8003	0.9236	0.8019
DeepChrome_uncross	0.6652	0.7737	0.8951	0.7710
ConvChrome	0.7820	0.8399	0.9061	0.8386
ConvChrome_uncross	0.7571	0.8111	0.8791	0.8076

ConvChrome's code and data are not available, We implement ConvChrome_CNN1D with PyTorch. We used the AUC score as our evaluation metric. Experimental results on 56 cell lines are shown in Figure 3 and Table 2, compared with the other models, TransferChrome improved the prediction accuracy on most cell lines. TransferChrome has a significant improvement in average AUC compared to DeepChrome and AttentiveChrome. Compared with ConvChrome, TransferChrome also has better performance.

Cross-cell lines gene expression prediction performance comparison

For the performance comparison in cross-cell lines gene expression prediction, we arbitrarily selected a cell line (E085) as the source domain and each one of other cell lines as the target domain. Figure 4 and Table 3 show the experimental results. In Figure 4 and Table 3, TransferChrome_cross means that TransferChrome was trained and tested on different cell lines. TransferChrome_uncross and TransferChrome_origin did not use the domain classification module, and TransferChrome_origin was trained and tested on the same cell lines, while TransferChrome_uncross was trained and tested on different cell lines. Table 3 shows the performance comparison of TransferChrome, DeepChrome and ConvChrome in cross-cell lines gene expression prediction. In Table 3, DeepChrome and ConvChrome indicate that the models were trained and tested on the same cell line, while DeepChrome_uncross and ConvChrome_uncross indicate that those models were trained with E085 cell line's data but were tested on other cell lines.

The results have shown that the average AUC of TransferChrome_uncross trained in a E085 cell line and tested on another dropped by 2.9% compared to those of TransferChrome_origin trained and tested on a same cell line. Similarly, the average AUCs of DeepChrome_uncross and ConvChrome_uncross dropped by 2.6% and 2.9% compared to those of DeepChrome and ConvChrome, respectively. Though TransferChrome_cross did not achieve the same effect as TransferChrome_origin, the average AUC drop is reduced to 1.5%, which showed that using domain classification module indeed improves the performance in cross-cell lines prediction.

Contributions of dense connectivity pattern and different position encoding functions

We carried experiments to see whether dense connectivity pattern and different position encoding functions have obvious impact on the performance of TransferChrome. A total of 9 cell lines out of 56 with worst (E079, E084, E112), median (E114, E120, E128), and best (E116, E117, E123) AUC scores were selected for ablation experiments.

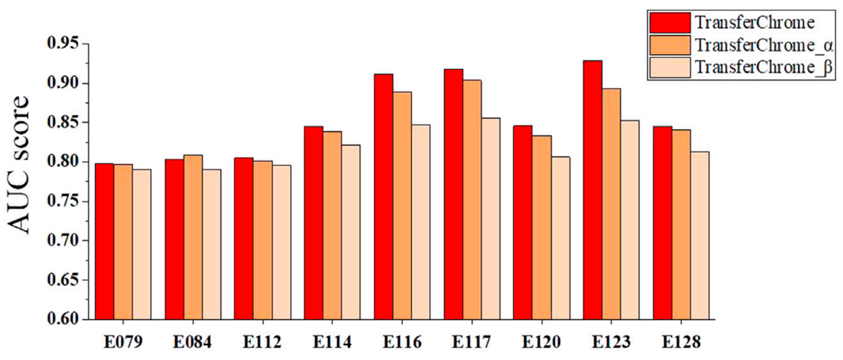


FIGURE 5
The performance comparison of different versions of TransferChrome: TransferChrome, TransferChrome_α and TransferChrome_β on 9 cell lines (E079, E084, E112, E114, E120, E128, E116, E117, E123).

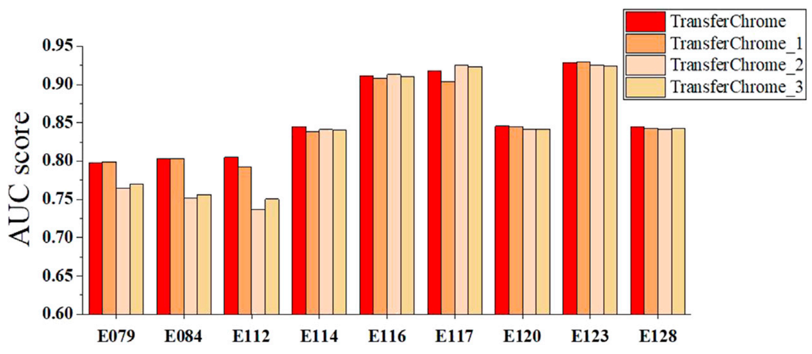


FIGURE 6
The performance comparison of different versions of TransferChrome: TransferChrome, TransferChrome_1, TransferChrome_2 and TransferChrome_3 on 9 cell lines (E079, E084, E112, E114, E120, E128, E116, E117, E123).

We compared three TransferChrome model variations to discuss the contribution of different position encoding functions. The position encoding function adopted by TransferChrome calculates the relative distance between TSS and bins. TransferChrome_α use sinusoidal position encoding (Vaswani et al., 2017) as the position function. Sinusoidal position encoding function calculates position information with a mix of sine and cosine functions. TransferChrome_β is the TransferChrome without adding position information. The experimental results are shown in Figure 5.

Histone modifications at different positions have different importance for gene expression prediction. Since TransferChrome_β ignores sequence position information, TransferChrome_β performance worser than TransferChrome and TransferChrome_α. Meanwhile, histone modifications which are close to TSS might have more significant effect on gene expression (Cheng et al., 2011). Accordingly, bins near to TSS should be assigned with higher weights for gene expression prediction (Singh et al., 2016). TransferChrome makes good use of relative distances between bins and TSS and performs better than TransferChrome_α.

We also conducted comparative experiments to discuss the contribution of the dense connectivity pattern and convolutional layer kernal numbers. As shown in Figure 6, four TransferChrome model variations are compared. TransferChrome has a dense-conv block, which has three dense-connected convolutional layers with different kernal numbers (32, 16, 8). TransferChrome_1 changes the structure of the dense-conv block. Dense-conv block of TransferChrome_1 uses three dense connected convolutional layers with 50 kernals. TransferChrome_2 and TransferChrome_3 do not use dense-conv block. TransferChrome_2 only has a convolutional layer with 50 kernals. TransferChrome_3 uses three convolutional layers with 50 kernals. Figure 6 shows the experimental results of above models. On 3 cell lines E079, E084 and E112, Transferchrome and TransferChrome_1 perform significantly better than others. TransferChrome uses fewer kernals than TransferChrome_1 but achieves a similar performance.

Conclusion and discussion

We proposed a new model called TransferChrome to predict gene expression levels based on histone modifications. TransferChrome uses self-attention mechanism to capture the long-distance dependence, and to learn hidden information features from the histone modifications data. Furthermore, TransferChrome adopts dense connectivity pattern to improve the feature exaction ability of convolutional neural network. Experimental results on the benchmark dataset of 56 cell lines showed that TransferChrome performed better than other 3 similar state-of-the-art models. To improve cross-cell lines gene expression prediction performance, TransferChrome uses transfer learning. Transfer learning makes the model capable of learning common features among different cell lines and reduces the data biases of different cell lines. Our experiments demonstrated that TransferChrome achieved the best accuracy in cross-cell lines gene expression prediction. We believe that it is useful to use transfer learning to improve cross-cell lines prediction accuracy. So far, gene expression prediction methods from histone modification data are mostly based on the five core histone modification marks. In future work, we will use more information from the histone modification data to predict gene expression. We also intend to increase the interpretability of the model in order to analyze the contribution of different histone modification marks on gene expression prediction.

Data availability statement

This study processed and analyzed publicly available data sets. These data can be found here: https://egg2.wustl.edu/roadmap/webportal/index.html.

Author contributions

YC and MX conceived the study and the conceptual design of the work. YC implemented the TransferChrome model and drafted the manuscript. JW collected the data and tested the model's performs. MX and YC polished the manuscript. All authors have read and approved the manuscript.

Funding

This work is supported by the National Natural Science Foundation of China under Grant 62172028 and Grant 61772197.

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

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2022.1081842/full#supplementary-material

References

Avsec, Ž., Agarwal, V., Visentin, D., Ledsam, J. R., Grabska-Barwinska, A., Taylor, K. R., et al. (2021). Effective gene expression prediction from sequence by integrating long-range interactions. *Nat. Methods* 18, 1196–1203. doi:10.1038/s41592-021-01252-x

Beer, M. A., and Tavazoie, S. (2004). Predicting gene expression from sequence. *Cell.* 117, 185–198. doi:10.1016/S0092-8674(04)00304-6

Cheng, C., Yan, K.-K., Yip, K. Y., Rozowsky, J., Alexander, R., Shou, C., et al. (2011). A statistical framework for modeling gene expression using chromatin features and application to modencode datasets. *Genome Biol.* 12, R15–R18. doi:10.1186/gb-2011-12-2-r15

Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. (2019). "Bert: Pre-training of deep bidirectional transformers for language understanding," in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (Stroudsburg, Pennsylvania, USA: Association for Computational Linguistics), 4171–4186.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., et al. (2021). "An image is worth 16x16 words: Transformers for image recognition at scale," in International Conference on Learning Representations.

Ganin, Y., and Lempitsky, V. (2015). "Unsupervised domain adaptation by backpropagation," in Proceedings of the 32nd International Conference on Machine Learning. Editors F. Bach and D. Blei (Cambridge, MA: PMLR), 1180–1189.

Gibney, E., and Nolan, C. (2010). Epigenetics and gene expression. *Heredity* 105, 4–13. doi:10.1038/hdy.2010.54

Hamdy, R., Maghraby, F. A., and Omar, Y. M. (2022). Convchrome: Predicting gene expression based on histone modifications using deep learning techniques. *Curr. Bioinform.* 17, 273–283. doi:10.2174/1574893616666211214110625

Huang, G., Liu, Z., Maaten, L. V. D., and Weinberger, K. Q. (2017). "Densely connected convolutional networks," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Honolulu, HI, USA: IEEE Computer Society), 2261–2269. doi:10.1109/CVPR.2017.243

Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. (2021). Dnabert: Pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinformatics* 37, 2112–2120. doi:10.1093/bioinformatics/btab083

Kamal, I. M., Wahid, N. A., and Bae, H. (2020). "Gene expression prediction using stacked temporal convolutional network," in 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), 402–405.

Karlić, R., Chung, H.-R., Lasserre, J., Vlahoviček, K., and Vingron, M. (2010). Histone modification levels are predictive for gene expression. *Proc. Natl. Acad. Sci. U. S. A.* 107, 2926–2931. doi:10.1073/pnas.0909344107

Krajewski, W. A. (2022). Histone modifications, internucleosome dynamics, and dna stresses: How they cooperate to "functionalize" nucleosomes. *Front. Genet.* 13, 873398. doi:10.3389/fgene.2022.873398

Kundaje, A., Meuleman, W., Ernst, J., Bilenky, M., Yen, A., Heravi-Moussavi, A., et al. (2015). Integrative analysis of 111 reference human epigenomes. *Nature* 518, 317–330. doi:10.1038/nature14248

Li, J., Ching, T., Huang, S., and Garmire, L. X. (2015). Using epigenomics data to predict gene expression in lung cancer. *BMC Bioinforma.* 16, S10–S12. doi:10.1186/1471-2105-16-S5-S10

Lin, W., Song, C., Meng, H., Li, N., and Geng, Q. (2022). Integrated analysis reveals the potential significance of hdac family genes in lung adenocarcinoma. *Front. Genet.* 13, 862977. doi:10.3389/fgene.2022.862977

Long, M., Cao, Y., Wang, J., and Jordan, M. (2015). "Learning transferable features with deep adaptation networks," in Proceedings of the 32nd International Conference on Machine Learning. Editors F. Bach and D. Blei (Cambridge, MA: PMLR), 97–105.

Ouyang, Z., Zhou, Q., and Wong, W. H. (2009). Chip-seq of transcription factors predicts absolute and differential gene expression in embryonic stem cells. *Proc. Natl. Acad. Sci. U. S. A.* 106, 21521–21526. doi:10.1073/pnas.0904863106

Peterson, C. L., and Laniel, M.-A. (2004). Histones and histone modifications. *Curr. Biol.* 14, R546–R551. doi:10.1016/j.cub.2004.07.007

Singh, D., and Singh, B. (2020). Investigating the impact of data normalization on classification performance. *Appl. Soft Comput.* 97, 105524. doi:10.1016/j.asoc.2019.105524

Singh, R., Lanchantin, J., Robins, G., and Qi, Y. (2016). Deepchrome: Deep-learning for predicting gene expression from histone modifications. *Bioinformatics* 32, i639–i648. doi:10.1093/bioinformatics/btw427

Singh, R., Lanchantin, J., Sekhon, A., and Qi, Y. (2017). "Attend and predict: Understanding gene regulation by selective attention on chromatin," in *Advances in neural information processing systems*. Editors I. Guyon, U. V. Luxburg, S. Bengio,

H. Wallach, R. Fergus, S. Vishwanathan, et al. (Red Hook, NY, USA: Curran Associates, Inc.), 30.

Sun, J., Dodlapati, S., and Jiang, Z. C. (2022). Completing single-cell dna methylome profiles via transfer learning together with kl-divergence. *Front. Genet.* 13, 910439. doi:10.3389/fgene.2022.910439

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., and Liu, C. (2018). "A survey on deep transfer learning," in *International conference on artificial neural networks*. Editors V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, and I. Maglogiannis (Berlin, Germany: Springer International Publishing), 270–279.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). "Attention is all you need," in *Advances in neural information processing systems*. Editors I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, et al. (Red Hook, NY, USA: Curran Associates, Inc.).

Zeng, W., Wang, Y., and Jiang, R. (2020). Integrating distal and proximal information to predict gene expression via a densely connected convolutional neural network. *Bioinformatics* 36, 496–503. doi:10.1093/bioinformatics/btz562

Zhu, L., Kesseli, J., Nykter, M., and Huttunen, H. (2018). "Predicting gene expression levels from histone modification signals with convolutional recurrent neural networks," in *EMBEC & NBC 2017*. Editors H. Eskola, O. Väisänen, J. Viik, and J. Hyttinen (Singapore: Springer Singapore), 555–558.

Zhu, X., Gu, Y., and Xiao, Z. (2022). Herbkg: Constructing a herbal-molecular medicine knowledge graph using a two-stage framework based on deep transfer learning. *Front. Genet.* 13, 799349. doi:10.3389/fgene.2022.799349

TYPE Original Research
PUBLISHED 06 January 2023
DOI 10.3389/fgene.2022.1096797

Check for updates

OPEN ACCESS

EDITED BY
Quan Zou,
University of Electronic Science and Technology of China, China

REVIEWED BY
Pu-Feng Du,
Tianjin University, China
Leyi Wei,
Shandong University, China

*CORRESPONDENCE
Juan Wang,
jwang@annoroad.com
Jiayin Wang,
wangjiayin@mail.xjtu.edu.cn

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 12 November 2022
ACCEPTED 14 December 2022
PUBLISHED 06 January 2023

CITATION
Wang S, Liu Y, Wang J, Zhu X, Shi Y, Wang X, Liu T, Xiao X and Wang J (2023), Is an SV caller compatible with sequencing data? An online recommendation tool to automatically recommend the optimal caller based on data features.
Front. Genet. 13:1096797.
doi: 10.3389/fgene.2022.1096797

Is an SV caller compatible with sequencing data? An online recommendation tool to automatically recommend the optimal caller based on data features

Shenjie Wang[1,2], Yuqian Liu[1,2], Juan Wang[1,2,3]*, Xiaoyan Zhu[1,2], Yuzhi Shi[3], Xuwen Wang[1,2], Tao Liu[3], Xiao Xiao[2,4] and Jiayin Wang[1,2]*

[1]School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, China, [2]Shaanxi Engineering Research Center of Medical and Health Big Data, Xi'an Jiaotong University, Xi'an, China, [3]Annoroad Gene Technology (Beijing) Co. Ltd, Beijing, China, [4]Geneplus Shenzhen, Shenzhen, China

A lot of bioinformatics tools were released to detect structural variants from the sequencing data during the past decade. For a data analyst, a natural question is about the selection of a tool fits for the data. Thus, this study presents an automatic tool recommendation method to facilitate data analysis. The optimal variant calling tool was recommended from a set of state-of-the-art bioinformatics tools by given a sequencing data. This recommendation method was implemented under a meta-learning framework, identifying the relationships between data features and the performance of tools. First, the meta-features were extracted to characterize the sequencing data and meta-targets were identified to pinpoint the optimal caller for the sequencing data. Second, a meta-model was constructed to bridge the meta-features and meta-targets. Finally, the recommendation was made according to the evaluation from the meta-model. A series of experiments were conducted to validate this recommendation method on both the simulated and real sequencing data. The results revealed that different SV callers often fit different sequencing data. The recommendation accuracy averaged more than 80% across all experimental configurations, outperforming the random- and fixed-pick strategy. To further facilitate the research community, we incorporated the recommendation method into an online cloud services for genomic data analysis, which is available at https://c.solargenomics.com/*via* a simple registration. In addition, the source code and a pre-trained model is available at https://github.com/hello-json/CallerRecommendation for academic usages only.

KEYWORDS

sequencing data analysis, bioinformatics tool, software recommendation, structural variant caller, meta-learning framework

1 Introduction

In genomics and bioinformatics, calling structural variants (SVs) from sequencing data is a somewhat straightforward topic (Handsakeret al., 2011; Northcott et al., 2012; English et al., 2014; Cao et al., 2014; Guan and Sung, 2016; Chiang et al., 2017). Tens of review papers (Seo et al., 2016; Fang et al., 2019; Kosugi et al., 2019; Amarasinghe et al., 2020; Luan et al., 2020; Zhao et al., 2020; Zook et al., 2020; De Coster et al., 2021; Guo et al., 2021) highlight SVs as important biomarkers and routinely identify them in various fields. Therefore, many SV callers have been developed to detect SVs (Stancu et al., 2017; Gong et al., 2018; Sedlazeck et al., 2018; Wenger et al., 2019; Jiang et al., 2020).

These callers used different strategies. Read pairs and depth approaches (Kosugi et al., 2019) primarily use the discordant alignment and depth features of paired-end reads that encompass or overlap an SV. The split read approach (English et al., 2014) primarily uses split alignment features of single- or paired-end reads that span an SV breakpoint. The assembly approach (Seo et al., 2016) detects SVs primarily by aligning assembled contigs with entire or unmapped sequencing reads to the reference sequence.

In summary, different strategies investigate various variant signals (values and/or distributions) in sequencing data and can deal with diverse sequencing data with different signals and their distributions. Furthermore, some empirical studies (Luan al., 2020; Kosugi et al., 2019; Guo et al., 2021) have been conducted to validate this phenomenon. A set of popular callers is compared on some benchmarking datasets in these studies, and the results showed that most callers have an edge for specific data.

In such instances, using the signal distributions in a given sequencing data to select the proper caller for diverse sequencing data makes sense. However, these signal distributions are usually ambiguous. When faced with a practice SV calling problem, it is difficult for users, especially non-experts, to decide which caller to use. Three simple approaches are commonly used in practice. First, choose one at random (random-pick strategy). Second, select one that is familiar or popular (fixed-pick strategy). Finally, consult an expert who will analyze the relationship between the sample and the SV caller's performance and make a recommendation based on their experience. The first two approaches are straightforward, but their efficacy cannot be guaranteed. The last one can sometimes boost effectiveness. However, there are very few such experts available to meet real-world demands.

Consequently, selecting appropriate SV callers becomes an urgent issue. As different SV callers explore different distributions (or patterns in some approaches) in sequencing data to make decisions, these distributions in sequencing data can affect the caller's performance. It is logical to use some signal distributions for SV caller selection. Thus, this study proposes an automatic SV caller recommendation method. The SV caller selection problem is established under a meta-learning framework in the method, with calling SVs from the sequencing data as the learning problem and caller selection as the meta-learning problem (Rendell and Cho, 1990; Ilchenkov and Pendryak, 2015; Morais et al., 2016; Sousa et al., 2016; Cruz et al., 2017; Vilalta and Drissi, 2022). The goal is to use meta-learning to improve the performance of the learning problem.

Specifically, the meta-features are collected to reflect the sequencing data's features, which attempt to reflect the hidden distributions or patterns in the sequencing data. The meta-targets are identified to indicate the most appropriate SV caller for the given data, and a meta-model is then built to mine the relationship between the meta-features and meta-target. When confronted with an SV caller section problem for a given sequencing data, the meta-features of the data are collected and fed into the constructed meta-model, and the meta-model specifies the final decision on the recommended a SV caller.

Since third-generation sequencing is becoming the major sequencing technology for SV detection (Luan al., 2020; Amarasinghe et al., 2020; Zook et al., 2020; Fang et al., 2019), this study focuses on the SV caller recommendation method on the third-generation sequencing data. A series of experiments are conducted on both simulated and real sequencing data to validate the performance of the recommendation. Compared to the random- and fixed-pick strategies, this recommendation method always selects a better caller, with an average recommendation accuracy of more than 80%. To the best of our knowledge, this study is one of the first automatic recommendation methods for bioinformatics tools for analyzing sequencing data. It can accurately recommend the best caller fits for the available sequencing data. This model is thought to be quite valuable for data analysts. To further facilitate the research community, we incorporated the recommendation method into an online cloud services for genomic data analysis, which is available at https://c.solargenomics.com/ *via* a simple registration. In addition, the source code and a pre-trained model is available at https://github.com/hello-json/CallerRecommendation for academic usages only.

2 Methods

2.1 Overview of the methods

The historical datasets in this specific meta-learning problem are the sequencing datasets with benchmarks, while the new dataset is the sequencing data to be detected. This meta-learning problem mines the potential relationship between meta-features and meta-target (appropriate callers) from sequencing datasets with benchmarks and recommends appropriate callers for the sequencing data to be detected based on this relationship.

A function **f** is created to map meta-features of the sequencing datasets to appropriate callers. The best function

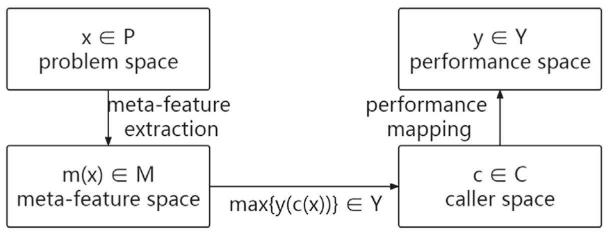


FIGURE 1
Abstract model of the SV caller selection problem.

$$f^* = argmin_{\{f\}}(L(f)) \quad (1)$$

is obtained by minimizing the loss function **L** based on caller performance on historical SVs calling problems. **L** is a function that measures the difference between the recommended and appropriate callers. Thus, the meta-learning function for the SV caller selection problem **P** can be formalized as follows: find the meta-learning function $\mathbf{f}(\mathbf{m}(\mathbf{x}))$ to the caller space **C** for a given sequencing data $\mathbf{x} \in \mathbf{P}$ with meta-features $\mathbf{m}(\mathbf{x}) \in \mathbf{M}$. The chosen caller c maximizes the performance mapping $\mathbf{y}(\mathbf{c}(\mathbf{x})) \in \mathbf{Y}$. That is,

$$f(m(\boldsymbol{x})) \rightarrow C: \ max\{y(c(\boldsymbol{x}))\} \in Y\,(\boldsymbol{x} \in P, m(\boldsymbol{x}) \in \boldsymbol{M}) \quad (2)$$

where **P**, **M**, **C**, and **Y** represent the problem space (sequencing dataset), meta-feature space (meta-feature set), caller space (SV caller set), and performance space (caller performance interval), respectively. Furthermore, $\mathbf{m}(\mathbf{x})$ and $\mathbf{c}(\mathbf{x})$ are the meta-features and the appropriate callers of **x**, respectively. Usually, the most important element is determining which caller outperforms the others. Thus, f can be improved further to map the features of **P** to the best caller. Figure 1 shows an abstract model of the SV caller selection problem.

According to the above analysis, a classification algorithm can build the meta-model in the SV caller recommendation method. This classification algorithm learns the relationship between the meta-features of each sequencing data in historical datasets and the optimal caller and then applies this relationship to map the detected sequencing data to its optimal caller. Consequently, the framework created in this study for the recommendation method is divided into three sections, namely, extracting data features and identifying the optimal caller, modeling the relationship between data features and the optimal caller, and recommending the optimal caller. Figure 2 shows the framework of this method.

2.2 Metadata collections

As shown in Figure 2, the first step of the SV caller recommendation method is metadata collection, which is

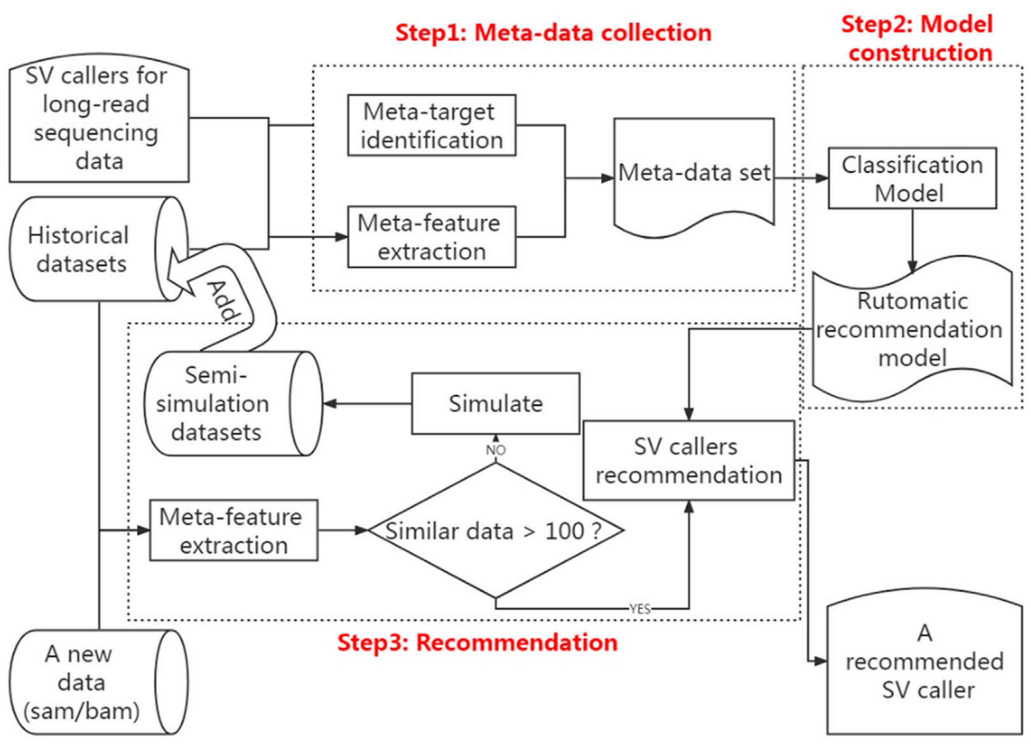


FIGURE 2
Computational pipeline for automatically recommending the optimal SV caller according to the features of sequencing data.

divided into two stages, *i.e.,* meta-feature extraction and meta-target identification.

2.2.1 Meta-feature extraction

The meta-features in this context are those sequence alignment and map (SAM)/binary alignment and map (BAM)files features that can effectively differentiate the performance of SV callers. The SAM/BAM file in sequencing data analysis is a sequence text file that contains the sequencing reads with information aligned to the reference genome. The statistical and information theory-based method is currently the most widely used meta-feature extraction method, which extracts meta-features such as dataset sizes, attribute types, numbers of attributes, mean, and variance (Brazdil et al., 2003; Pise, 2013; Ali et al., 2018; Wang et al., 2019). However, sequencing data is a unique type of data in that a single read or statistical information about reads contains little information, and the set of sequencing reads is mapped to a region that contains the most information. Therefore, this method does not perform well with sequencing data. For example, even if SAM/BAM files are very close in size or even have the same number of reads, the information they contain may be completely different due to the different bases of reads.

According to bioinformatics research, read length, sequencing depth, base quality, mapping quality, and insert size significantly impact caller performance (Kosugi et al., 2019; Wang et al., 2019; Zook et al., 2020; Chen et al., 2021). However, these features are used to call SVs, whereas meta-features are now required to effectively differentiate the performance of SV callers. As a result, there are several useless features here. According to this study's testing, some features, such as read length and sequencing depth, are useful, while others are not. Furthermore, some review studies have proposed some sequencing data features, such as the size of SVs and the proportion of SVs in tandem repeat regions, which have been shown to differentiate the performance of SV callers (Kosugi et al., 2019; Zook et al., 2020; Guo et al., 2021).

For example, Picky (Gong et al., 2018) uses an assembly approach to produce read alignment by stitching the segments from LAST with a greedy seed-and-extend strategy and can thus handle large SVs by assembling them as distinct contigs. However, when the sequencing depth is low, the assembly junctions are ambiguous, i.e., some of the haplotype sequences (particularly contigs of SV alleles) are missing, which may affect SV calling recall. Sniffles (Sedlazeck al., 2018) uses a split read approach to identify SVs by putative variant scoring using several features based on NGMLR alignment results and thus can identify SVs even when the sequencing depth is low. However, due to the lack of assembly, it is difficult to identify large SVs from ambiguous alignments for Sniffles.

Therefore, based on this study's experiments and review papers, features that can effectively differentiate the performance of SV callers while eight avoiding over-fittings were ultimately chosen. In this context, these are known as meta-features. They were distributed on the SAM/BAM file levels (datasets levels) and the variant signature levels (instance levels). The average length of reads and the average sequencing depth are SAM/BAM file-level meta-features. The proportion of SVs in tandem repeat regions, the proportion of short SVs (50–200 bp), the proportion of middle SVs (200–1,000 bp), the proportion of large SVs (>1,000 bp), read variant burden (the number of SVs that one read can traverse), and the proportion of regions with high read variant burden are among the variant signature level meta-features. Table 1 summarizes the selected meta-features.

TABLE 1 Meta-features extracted to characterize sequencing data.

Level	Meta-features
SAM/BAM file level	Average length of reads
	Average sequencing depth
Variant signature level	Proportion of SVs in tandem repeat regions
	Proportion of short SVs
	Proportion of middle SVs
	Proportion of large SVs
	Read variant burden
	Proportion of regions with high read variant burden

This study creates a new meta-feature extraction method, called the variational signature-based meta-feature estimation algorithm, to extract the above features from the sequencing datasets. This is a fast scanning algorithm. It simply needs to estimate the features described above rather than accurately detect the SVs. Thus, while the proposed algorithm may be slightly inaccurate, it has been experimentally proven to affect caller recommendations. The algorithm can extract information from the SAM/BAM file level and variant signature levels. Meta-features, such as sequencing depth, can be obtained from SAM files using samtools. Meta-features were extracted for variant signatures by setting a sliding window and grabbing softclip reads with breakpoint information. Specifically, the loci with variant signatures and the size of SVs can be estimated as follows:

1) Cluster all the reads from the input SAM/BAM file with softclips.
2) Divide the reads with softclips into two categories based on whether the softclip is at the beginning or at the end of the reads.
3) Determine the variant loci of each category according to the cigar value of each read.

4) Determine the distance between pairs of breakpoints as an estimated value of the size of SVs.

Algorithm 1 presents the pseudocode of meta-feature extraction. The algorithm's input was the SAM/BAM file F, while the output was the meta-feature, MF. The tandem repeat regions used in this study were annotated in the hg19 annotation file, which can be downloaded from UCSC Genome Browser (http://hgdownload.soe.ucsc.edu/goldenPath/hg19/database/rmsk.txt.gz) (Zook et al., 2016).

```
Require:
F is the sam/bam file
Ensure: MF
1: f = []
2: tr = []
3: softclips = []
4: breakpoints = []
5: readlen_sum = 0
6: f ← F//Sam/bam file F are read in and saved in array f
7: tr ← TR//tandem repeat regions file TR are read in and saved in array tr
8: depth = caculateDepth(f)
9: for i = 1 to n do
10:    readlen = f[i].len()
11:    readlen_sum = readlen_sum + readlen
12:    if f[i].include("s") then
13:       softclips.add(f[i])
14:    else
15:       continue
16:    end if
17: end for
18: ave_readlen = readlen_sum / n
19: for j = 1 to m do
20:    breakpoint = calculateBP(softclips[j])
21:    breakpoints.add(breakpoint)
22: end for
23: short_sv, middle_sv, large_sv = caculateSVSize(breakpoints)
24: rvb = caculateRVB(ave_readlen, breakpoints)
25: high_rvb = caculateHigh_rvb(softclips, rvb)
26: sv_tr = caculateSv_tr(breakpoints, tr)
27: MF.add(depth, ave_readlen, short_sv, middle_sv, large_sv, rvb, high_rvb, sv_tr)
28: Return MF
```

Algorithm 1. Meta-feature extraction

Lines one to six of Algorithm 1 initialize the f, tr, softclips, breakpoint arrays, and readlen_sum. The f array saves the reads in the SAM/BAM file, the tr array saves regions in the tandem repeat regions file, the softclips array saves softclip reads, the breakpoint array saves softclip reads breakpoints, and the readlen_sum is used to save the total length of reads. In pseudocode lines six to seven, SAM/BAM file F and the tandem repeat regions file TR are read in and saved in arrays f and tr, respectively.

In pseudocode line 8, the depth function calculates the sequencing a depth, which is then assigned to depth. The pseudocode lines 9–18 traverse each read length separately, update the readlen_sum value, save softclips in the softclips array, and calculate the average length of softclip reads. The pseudocode lines 19–22 traverse each softclip reads separately, calculate breakpoints, and save them in the breakpoint array. The pseudocode lines 23–27 calculate the short_sv, middle_sv, large_sv, rvb, high_rvb, and sv_tr using the caculateSVSize, caculateRVB, caculateHigh_rvb, and caculateSv_tr functions, respectively, where short_sv, middle_sv, large_sv, rvb, high_rvb, and sv_tr denote the proportion of short SVs, the proportion of middle SVs, the proportion of large SVs, read variant burden, the proportion of regions with high read variant burden, and the proportion of SVs in tandem repeat regions, respectively. Finally, the meta-feature, MF, is saved on line 27 and returned on line 28.

2.2.2 Meta-target identification

This step involves tagging meta-targets representing the best of the callers. In turn, each caller was run on each sequencing data and then ranked based on their performance, and the best was chosen as the meta-target for that data. Algorithm 2 provides the pseudocode of meta-target identification. The algorithm's inputs include the long-read sequencing dataset as **D**, the set of SV callers as **C**, and the caller performance evaluation metric M. The algorithm's output is the meta-target set as **T**.

```
Require:
D = { $d_i$ |i = 1, 2,...,n and $d_i$ is a long-read sequencing data}
C = { $c_i$ |j = 1, 2,...,q and $c_i$ is a SV caller}
M is an SV caller performance evaluation metric
Ensure: T
1: Eval_Metric = []
2: Rank = []
3: for i = 1 to n do
4:    $l_i = d_i$ .class_label;
5:    $d_i = d_i - l_i$;
6:    for j = 1 to q do
7:       Caller_Result = calling( $d_i$, $c_i$)
8:       value = evaluate(Caller_Result, $l_i$, M)
9:       Eval_Metric.add(value)
10:   end for
11:   Rank = Eval_Metric.sort("descending")
12:   max_index = Rank[1]
13:   max_SVcaller = S[max_index]
14:   T.add(max_SVcaller)
15: end for
16: Return T
```

Algorithm 2. Meta-target identification

In Algorithm 2, the Eval_Metric and rank arrays are used in lines 1 and 2. The Eval_Metric array is used to save the performance evaluation values, whereas the rank array is used to save the ranks of callers based on their performance evaluation values. In pseudocode lines 3–15, the meta-target T is identified for each long-read sequencing dataset in ***D***. For a long-read sequencing dataset d_i, its label is saved in l_i and then removed from d_i lines 4 and 5. In lines 6–10, each caller c_i, d_i is called using the SV caller c_i. And the calling results are evaluated in terms of the metric M. The evaluation result is added to Eval_Metric. Eval_Metric is sorted in descending order in line 11. Ranks of SV callers according to their performance in terms of Eval_Metric are saved in rank. The meta-target is then obtained and saved in T in lines 12–14. Finally, the meta-target set T is returned at line 16. The meta-feature and meta-target are saved after the above meta-feature extraction and meta-target identification. Users can specify different meta-targets based on their requirements.

2.2.3 Meta-model construction and recommendation

The features of each data were obtained, and the best caller for that data using the steps outlined above was determined. That is, the meta-features and meta-targets are available. In this case, one meta-feature is a vector p_i $(mf_1, mf_2, mf_3, \ldots mf_n)$, and corresponding to this meta-feature, there is a meta-target, where i = 1, 2,..., m, and m is the number of training samples. All samples constitute a dataset that can be used to train a classification model. Therefore, the classifier was used to learn the relationship between the meta-features and meta-targets, and then the meta-model was built as the recommendation model. Finally, the RandomForest algorithm was used to build the classifier after considering the relevance of the features and effectiveness of the model.

This step above results in the automatic recommendation model. When users need to make caller recommendations based on the new long-read sequencing data, they first extract the data's meta-features from the SAM/BAM file. They then determine whether the number of data in the historical dataset with meta-features similar to the new long-read sequencing data is greater than 100. If not, they generate 100 semi-simulated data based on the meta-features of the real new long-read sequencing data and add them to the historical dataset to retrain the recommendation model. If yes, the extracted meta-features are fed into the recommendation model. Finally, the model will output the recommendation results of the new long-read sequencing data based on the meta-targets that users have specified.

3 Results

This section conducts experiments to verify the necessity and efficacy of the proposed recommendation method:

Question 1. Necessity: Does the matching degree between the SV caller and the signal distributions significantly impact SV caller performance?

This is an important question. The influence of the matching degree between the detection strategy and the signal distributions determines the necessity of the research on the recommendation method. A fixed or randomly selected SV caller can be used if it exerts minimal influence on the SV caller performance.

Question 2. Effectiveness: How effective is the proposed caller recommender?

This is also an important question. Suppose the matching degree between the SV caller and the signal distributions exerts a non-negligible influence on the SV calling performance. In that case, the performance of the proposed SV caller recommendation method determines whether it can be used in practice.

3.1 Experiment setup

3.1.1 Benchmark datasets and candidate long-read sequencing data SV callers

The small number of real long-read sequencing datasets with benchmarks that can be analyzed is insufficient to construct a historical dataset. Using the PBSIM simulator, 768 simulated long-read sequencing datasets were generated (Yukiteru al., 2013) (https://github.com/yukiteruono/pbsim2). Specifically, various SVs were planted on chromosome 1, and reads ranging in lengths from 1,000 to 25,000 bps with varying sequencing depths were generated (10–150 X). For each sample, the density of the SVs was varied by varying the distance between the SVs. Furthermore, the proportion of variations in the tandem repeat region was altered by varying the number of SVs within the tandem repeat region of the genome. True called SVs are defined as the called SVs that significantly overlap with the reference SVs by proportions (≥80%).

Five state-of-the-art SV callers, namely, NanoSV (Stancu et al., 2017) (https://github.com/mroosmalen/nanosv), Picky (https://github.com/TheJacksonLaboratory/Picky), Sniffles (https://github.com/fritzsedlazeck/Sniffles), PbSV (Wenger et al., 2019) (https://github.com/PacificBiosciences/pbsv), and CuteSV (Jiang et al., 2020) (https://github.com/tjiangHIT/cuteSV), were implemented on the simulated datasets as the candidate callers. Each of these callers has advantages due to their different strategies. For example, Picky can handle large SVs well by assembling reads as distinct contigs due to the assembly approach it adopts, while the assembly approach performs poorly when the sequencing depth is too low due to the lack of reads. However, due to the split read approach (alignment-based approach), NanoSV, PbSV, Sniffles, and CuteSV can detect SVs even at a low sequencing depth, but they cannot handle large SVs due to lack of read assembling. As another example, Sniffles and CuteSV are appropriate for dense SVs and SVs in repeat-rich regions. Because the sequencing data contain many sequencing errors, particularly for long reads, they have also designed error event filtering mechanisms in their algorithms, which greatly improve the detection of SVs in repeat-rich regions and dense SVs or even nested SVs. However, Picky, NanoSV, and PbSV cannot handle these SVs due to the lack of an error event filtering mechanism. Each caller in the experiments used the default parameters and the alignment tool recommended by the caller developer.

3.1.2 Metrics to evaluate the performance of SV callers

F-measure, precision, and recall are important metrics for evaluating bioinformatics analysis methods, and they are frequently discussed in bioinformatics methodology studies (Kosugi et al., 2019). Therefore, these three metrics were chosen to evaluate the performance of SV callers. Precision, recall, and f-measure are calculated as follows:

$$precision = \frac{TP}{Call} \tag{3}$$

$$recall = \frac{TP}{Ref} \tag{4}$$

$$f-measure = \frac{2 \times precision \times recall}{precision + recall} \tag{5}$$

where TP, Call, and Ref are the numbers of true positives, called SVs, and the corresponding reference SVs, respectively.

3.1.3 Evaluating the performance of the recommendation method

The performance of the recommended SV caller is an important evaluation metric (Song et al., 2012). Therefore, recommendation accuracy (RA) is used to evaluate the RA of the proposed recommendation method, reflecting the difference in performance between the recommended optimal SV caller and the real optimal SV caller. During the implementation of the experiments, a leave-one-out cross-validation method was used to calculate RA values.

For a given long-read sequencing data s, let CallerR be the recommended SV caller, CallerO the most optimal SV caller, and CallerW the worst caller. RA is defined as follows:

$$RA(s) = \frac{P_{CallerR}(s) - P_{CallerW}(s)}{P_{CallerO}(s) - P_{CallerW}(s)} \tag{6}$$

Where $P_X(Y)$ denotes the performance of SV caller X on long-read sequencing data Y.

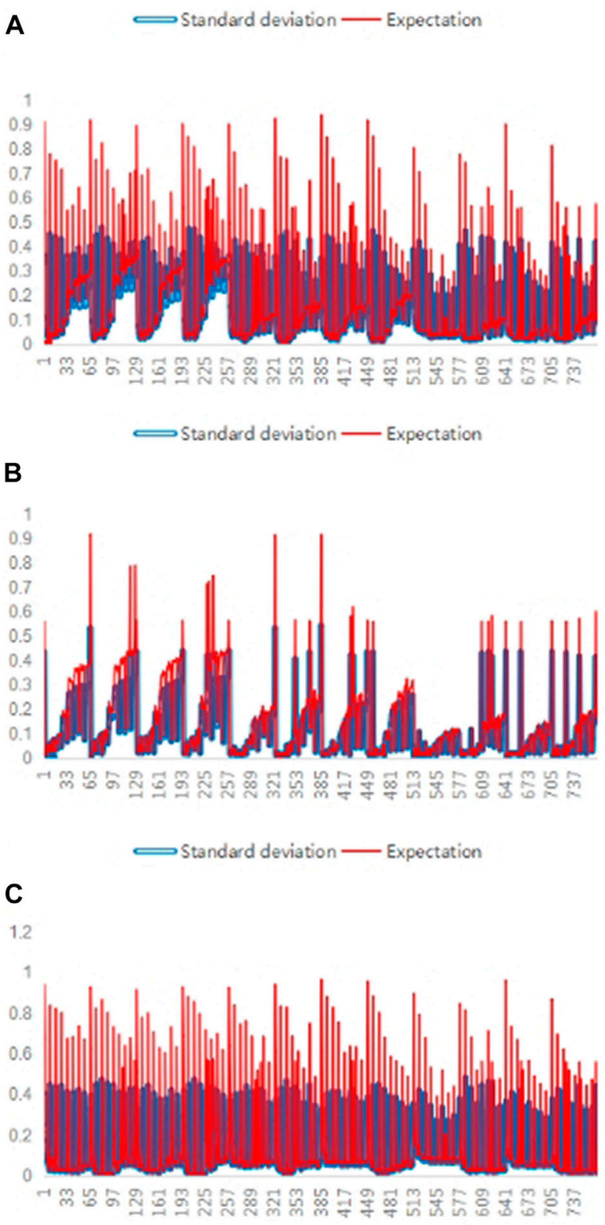


FIGURE 3
Differences between the various SV callers. In **(A)**, **(B)**, and **(C)**, the values of the three performance evaluation metrics, i.e., f-measure, precision, and recall, were calculated for each of the five callers on 768 long-read sequencing datasets. For each dataset, expectation values for one of the three performance evaluation metrics were calculated and shown with red lines, while the standard deviation values are shown with blue lines in the three subfigures, where the abscissa denotes the number of the long-read sequencing datasets and the ordinate denotes the corresponding performance evaluation metric value.

3.2 Necessity of the proposed recommendation method

The f-measure, precision, and recall values of the five SV callers were compared on different long-read sequencing data to evaluate the extent of performance differences between them, as shown in Figure 3.

As shown in Figure 3, the standard deviation values are non-negligible compared to the expected values for any meta-targets, indicating a significant difference in the performances of SV callers.

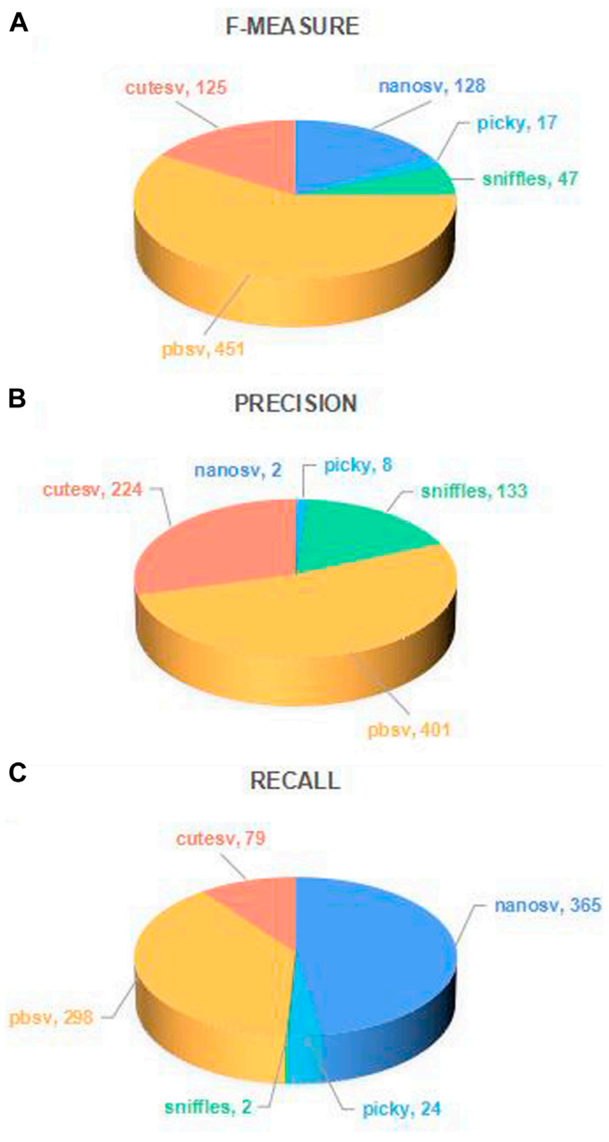


FIGURE 4
Number of long-read sequencing datasets that each SV caller ranks as top 1. The number of times each caller achieved the top one in the three performance metrics of f-measure, precision, and recall for 768 long-read sequencing datasets was calculated in **(A)**, **(B)**, and **(C)**. In each subfigure, the sectors of different colors represent different callers, and the sector's size indicates the proportion of different callers achieving the top 1. The number marked in each sector is the number of long-read sequencing datasets on which the performance of the SV caller in terms of the performance evaluation metric value ranks top 1.

Although the performance of SV callers varies significantly, this recommendation method of research is unnecessary if there is one SV caller who always has the best performance. Therefore, the number of long-read sequencing datasets that each SV caller ranks as top one was further compared, as shown in Figure 4.

As shown in Figure 4, the different SV callers rank top one on a certain number of long-read sequencing datasets, and the best

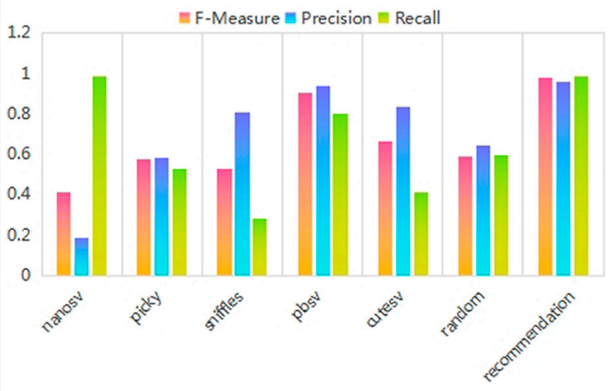


FIGURE 5
Recommendation accuracy values for three different performance evaluation metrics. The results of the three different performance evaluation metrics are displayed on three different colored bars. The abscissa indicates the five fixed SV callers, a randomly selected SV caller, and the recommended SV caller in order. The ordinate is the recommendation accuracy value.

performing SV caller can only account for about half of the overall, indicating that the optimal SV caller varies for different long-read sequencing datasets.

3.3 Effectiveness of the proposed recommendation method

This section presents the proposed method's recommendation results regarding RA. Furthermore, it also presents the recommendation performance on real long-read sequencing datasets.

3.3.1 Recommendation accuracy

In this subsection, the recommendation method's potential usefulness was demonstrated in real practice by comparing the RA values of the recommended SV caller with those of fixed and randomly selected SV callers. The RA describes the performance of the recommended SV caller compared with the best and worst SV caller, which is important in evaluating the usefulness of the SV caller recommendation method. Figure 5 shows the experimental results for the RA value.

As shown in Figure 5, the fixed-pick strategy performs differently for different SV callers. The random-pick strategy has poor performance. The recommended SV caller methods are better and more stable than the random-pick and fixed-pick strategies.

3.3.2 Hypothesis test for recommendation accuracy

The above analysis discovered that the proposed SV caller recommendation method improves the RA values of the random- and fixed-pick strategies. To test whether the

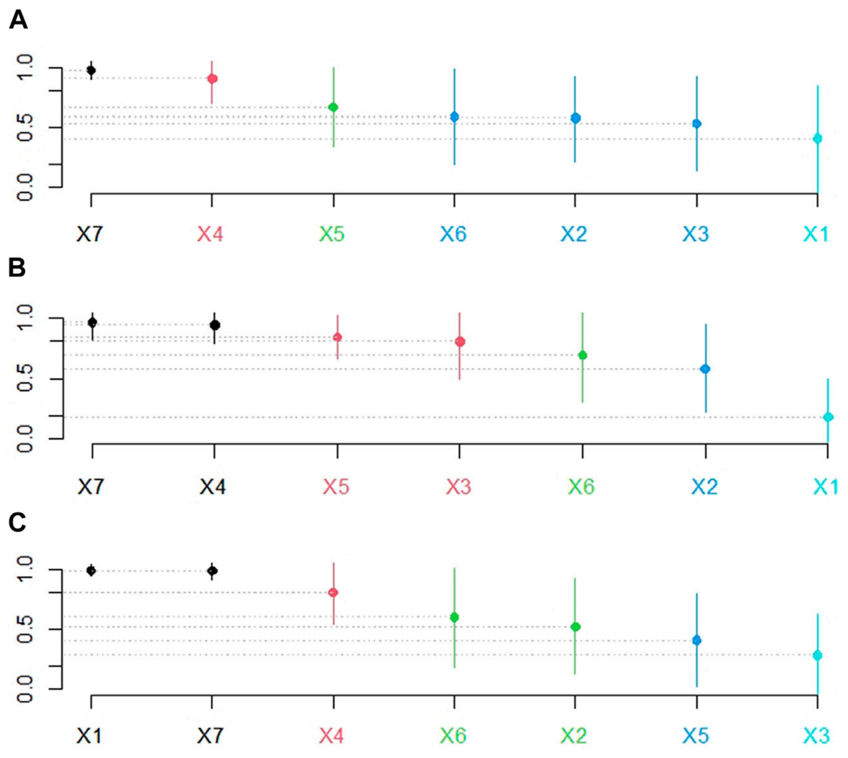


FIGURE 6
Hypothesis test results of RA values. In **(A)**, **(B)**, and **(C)**, X1-X7 denote the five fixed SV callers, a randomly selected SV caller, and the recommended SV caller in that order. The ordinate represents the recommendation accuracy values. Subfigures **(A)**, **(B)**, and **(C)** represent the recommendation schemes with f-measure, precision, and recall as meta-targets, respectively. The points on the bar graph give the average recommendation accuracy values. The length of the bars reflects the method's stability. The greater the average recommendation accuracy value, the better the recommendation method; the shorter the bars, the more stable the recommended method. Furthermore, no statistical difference existed between bars of the same color, whereas a significant difference existed between bars of different colors.

improvement is statistically significant, the Scott-Knott effect size difference test (Chakkrit al., 2017) was applied (https://github.com/klainfo/ScottKnottESD), allowing the RA values of different methods to be divided into different groups with non-negligible differences. Figure 6 shows the Scott-Knott effect size difference test results.

Figures 6A, B show that the SV caller recommendation method has superior and more stable performance advantages, whereas Figure 6C shows that in the recommendation scheme with recall as the meta-target, the difference between the recommendation method's performance and that of fixed with NanoSV is insignificant. Therefore, the following WinDrawLoss analysis experiments were conducted to compare the winners and losers between the different methods.

3.3.3 WinDrawLoss analysis for recommendation accuracy

From the above experiments, the difference in performance between the recommendation method and that of fixed with a specific SV caller is insignificant in the recommendation scheme with recall as the meta-target. Therefore, as presented in Table 2, the WinDrawLoss analysis of the different methods was conducted, showing the number of wins, draws, and losses of different methods on different datasets.

As presented in Table 2, in each case, the number of wins for the recommended method is much higher than the number of losses. In other words, the proposed SV caller recommendation method has significant advantages over other methods.

3.3.4 Evaluating the recommendation accuracy on real long-read sequencing datasets

To further test the performance of the proposed method on real data, all publicly available triple sequencing data were used with benchmarks. Specifically, the real long-read sequencing data from the well-studied NA12878 individual were used by the Ashkenazim Jewish and Chinese trios to assess the recommendation performance of the proposed SV caller recommendation method (Gong al., 2018; Zook al., 2016). Subreads datasets of the NA12878 individual (HG001), the Ashkenazim Jewish trio son (HG002), the Ashkenazim Jewish trio father (HG003), the Ashkenazim Jewish

TABLE 2 WinDrawLoss Analysis on RA Values.Subtables (A), (B), and (C) represent the recommendation schemes with f-measure, precision, and recall as meta-targets. In each subtable, each line represents the recommended SV caller being compared with the five fixed SV callers and a randomly selected SV caller, and each column represents the number of wins, draws, and losses, respectively.

(A)			
f1score	Win	Draw	Loss
fixed-pick_nanosv	596	134	38
fixed-pick_picky	735	6	27
fixed-pick_sniffles	701	35	32
fixed-pick_pbsv	233	491	44
fixed-pick_cutesv	590	137	41
random-pick	579	157	32
(B)			
Precision	Win	Draw	Loss
fixed-pick_nanosv	727	22	19
fixed-pick_picky	721	16	31
fixed-pick_sniffles	483	189	96
fixed-pick_pbsv	153	550	65
fixed-pick_cutesv	478	249	41
random-pick	533	185	50
(C)			
Recall	Win	Draw	Loss
fixed-pick_nanosv	71	654	43
fixed-pick_picky	669	77	22
fixed-pick_sniffles	736	28	4
fixed-pick_pbsv	385	371	12
fixed-pick_cutesv	672	93	3
random-pick	500	247	21

trio mother (HG004), the Chinese trio son (HG005), the Chinese trio father (HG006), and the Chinese trio mother (HG007) were downloaded from GIAB (https://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/).

The experimental results showed that the proposed SV caller recommendation method achieved the RA values of 86.05%, 64.28%, and 95.92% for meta-targets f-measure, precision, and recall, respectively, and the RA remained above 80% on average.

3.3.5 Threats to validity

A possible threat to the validity of this study lies in whether the simulated data used in the empirical study are representative of the broader datasets. Preferring to choose as many real datasets with benchmarks and well-known published simulator pbsim as possible is the primary way for this study to avoid sample bias.

3.4 The online recommendation tool

To further facilitate the research community, we incorporated the recommendation method into an online cloud services for genomic data analysis. This cloud system supports user-friendly online Web UI operation, eliminating the heavy work of setting up the running environment. More than 40 bioinformatics analysis tools are integrated on this, with functions covering eight categories including data statistics, data processing, format conversion, data comparison, visualization, table processing, plotting, and advanced tools to meet individual analysis needs.

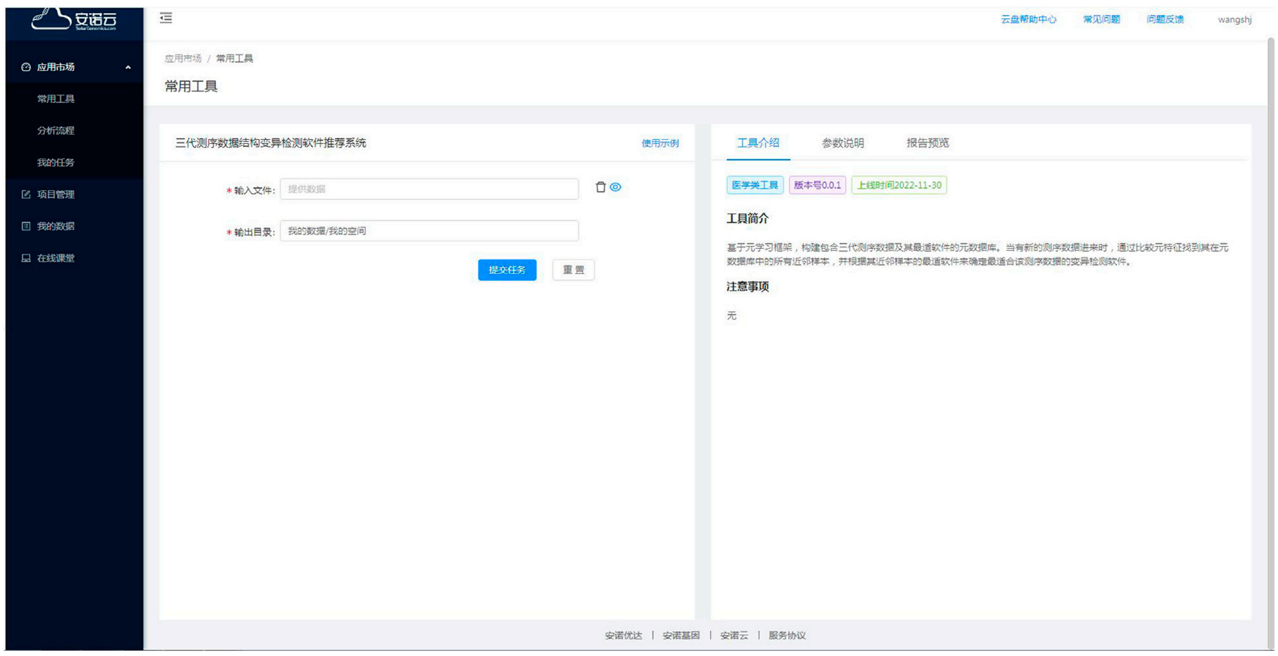
After logged into the cloud system at https://c.solargenomics.com/, users can search for this recommendation tool in the frequently used tools search box. Then, input the fastq file to be analyzed in the file input box and click the submit button, as shown in Figure 7. After the program is finished, you can see the recommended variant calling tool for that data in the task menu, as shown in Figure 8. Currently, the cloud system is collaborated with a PacBio Partner in China, and we are seeking for further collaborations on the cloud systems with English services. In addition, the source code and a pre-trained model is available at https://github.com/hello-json/CallerRecommendation for academic usages only.

4 Discussion

Other options for selecting meta-features and the classification algorithm may be available in the proposed recommendation method. Thus, these issues are discussed here. Two widely used classification evaluation metrics, i.e., f-measure and area under the receiver operating characteristic (AUC), were used to evaluate the classification accuracy of the method. Furthermore, a tenfold cross-validation method was used to fully utilize the dataset for the experiments.

First, the recommendation performance of recommendation models built from meta-features extracted using this study's variational signature-based meta-feature estimation algorithm was compared to the traditional meta-feature extraction method and a combination of the two meta-feature extraction methods, as shown in Figure 9. As shown in Figure 9, the performance of recommendation methods built with different meta-features varies with the recommendation model built with this study's variational signature-based meta-features having the best performance and being the most stable.

Next, the recommendation performance of recommendation models built using different classification algorithms was compared, as shown in Figure 10. As shown in Figure 10, the



常用工具

三代测序数据结构变异检测软件推荐系统

* 输入文件:	提供数据
* 输出目录:	我的数据/我的空间

医学类工具 版本号0.0.1 上线时间2022-11-30

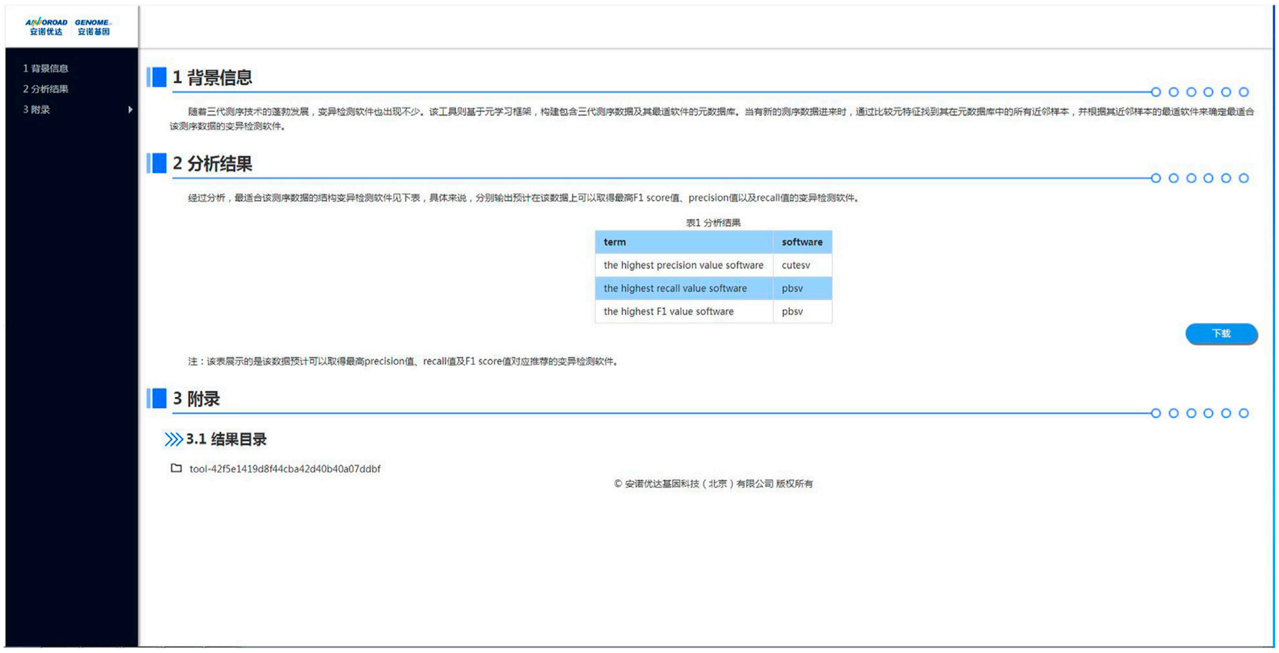
工具简介

基于元学习框架，构建包含三代测序数据及其最适软件的元数据库。当有新的测序数据进来时，通过比较元特征找到其在元数据库中的所有近邻样本，并根据其近邻样本的最适软件来确定最适合该测序数据的变异检测软件。

注意事项

无

FIGURE 7
The online recommendation tool input page.



1 背景信息

随着三代测序技术的蓬勃发展，变异检测软件也出现不少。该工具则基于元学习框架，构建包含三代测序数据及其最适软件的元数据库。当有新的测序数据进来时，通过比较元特征找到其在元数据库中的所有近邻样本，并根据其近邻样本的最适软件来确定最适合该测序数据的变异检测软件。

2 分析结果

经过分析，最适合该测序数据的结构变异检测软件见下表，具体来说，分别输出预计在该数据上可以取得最高F1 score值、precision值以及recall值的变异检测软件。

表1 分析结果

term	software
the highest precision value software	cutesv
the highest recall value software	pbsv
the highest F1 value software	pbsv

注：该表展示的是该数据预计可以取得最高precision值、recall值及F1 score值对应推荐的变异检测软件。

3 附录

3.1 结果目录

tool-42f5e1419d8f44cba42d40b40a07ddbf

© 安诺优达基因科技（北京）有限公司 版权所有

FIGURE 8
The online recommendation tool output page.

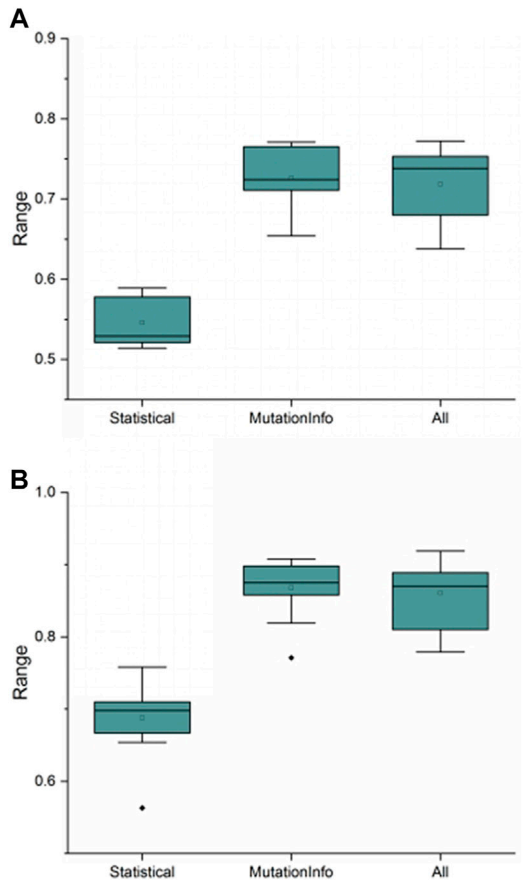


FIGURE 9
Recommendation performance of different recommendation models constructed with different meta-features. **(A)** and **(B)** show the comparison results of different meta-features on f-measure and AUC performance evaluation metrics, respectively, where "Statistical" denotes the traditional meta-features, "MutationInfo" denotes the variational signature-based meta-features, and "All" denotes the combination of two meta-features. The ordinate indicates the values of the corresponding performance evaluation metrics.

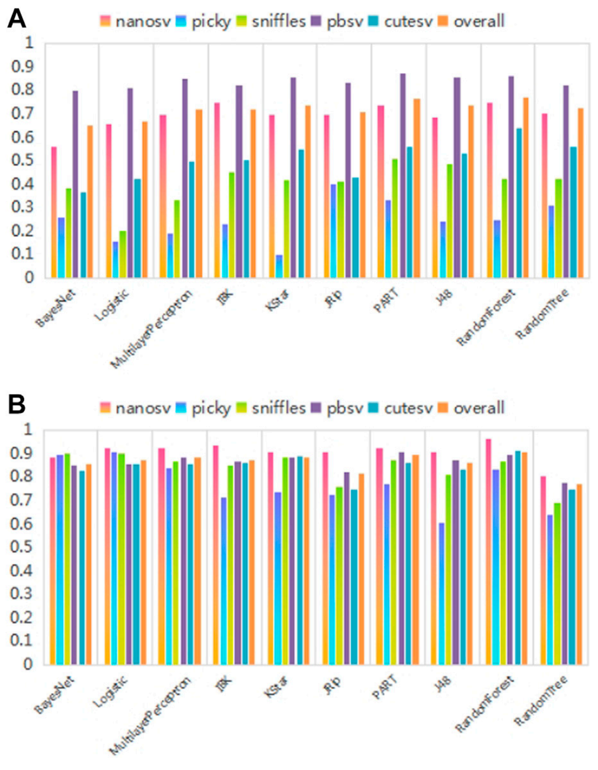


FIGURE 10
Recommendation performance of recommendation models constructed by different classification algorithms. **(A)** and **(B)** show the comparison results of different classification algorithms on f-measure and AUC performance evaluation metrics, respectively. The abscissa represents ten commonly used multi-classification algorithms based on different principles. The different colored bars for each multi-classification algorithm indicate their recommendation performance for each SV caller and overall. The ordinate is the performance evaluation metric values.

recommendation performance of the models built using different multi-classification algorithms differs significantly. Among these, the RandomForest algorithm achieves optimal values for f-measure and AUC performance evaluation metrics. The experimental results are consistent with the theoretical analysis in the method section.

5 Conclusion

Many bioinformatics approaches provide powerful algorithmic tools to investigate sequencing data in greater depth. However, quickly selecting the tool that best fits the data form among these state-of-the-art approaches becomes a real and practical level problem. An automatic recommendation method is designed and presented to facilitate the data analysts in selecting the best SV caller based on the sequencing data available. To the best of our knowledge, this is among the first recommendation methods for bioinformatics tools for analyzing sequencing data, and it has the potential to aid the research community.

The proposed method is designed under a meta-learning framework. This is acceptable because identifying the relationships between the data features and the performance of callers is a meta-learning problem. Eight data features distributed at the file and signature levels were selected. The relationship between the features and the optimal caller was then identified through a classification algorithm, RandomForest, and this relationship was used for the caller recommendation. A series of experiments validated the performance and advantages of the automatic recommendation, whatever it takes to recommend the optimal caller with the highest f-measure, precision, or recall. The experimental results also confirmed that different SV callers often fit different samples (sequencing data). The RA was maintained above 80% on average, which was much better than the random-pick and fixed-pick strategies.

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 authors.

Author contributions

JW and XZ conceived this research; SW, YL, XX, XW designed the algorithm and the method; SW implemented coding and designed the software; SW and XW collected the sequencing data; SW and XW performed the experiments and analyzed the data; SW and JW wrote the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding

This work was funded by the National Natural Science Foundation of China, grant numbers 72293581, 72293580, 72274152, 71872146; This work was also supported by the Natural Science Basic Research Program of Shaanxi, grant number 2020JC-01. The article processing charge was funded by the Natural Science Basic Research Program of Shaanxi, grant number 2020JC-01.

Acknowledgments

We thank all faculty members and graduate students who discussed the mathematical and statistical issues in seminars.

Conflict of interest

Authors JW, YS, and TL were employed by the company Annoroad Gene Technology Co. Ltd. Author XX was employed by GenePlus Shenzhen Institute.

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

References

English, A. C., Salerno, W. J., and Reid, J. G. (2014). PBHoney: Identifying genomic variants via long-read discordance and interrupted mapping. *BMC Bioinforma.* 15, 180. doi:10.1186/1471-2105-15-180

Ali, R., Khatak, A. M., and Chow, F., A case-based meta-learning and reasoning framework for classifiers. In Proceeding of the Selection//the 12th International Conference. 2018.

Alioto, T. S., Buchhalter, I., Derdak, S., Hutter, B., Eldridge, M. D., Hovig, E., et al. (2015). A comprehensive assessment of somatic mutation detection in cancer using whole-genome sequencing. *Nat. Commun.* 6, 10001. doi:10.1038/ncomms10001

Amarasinghe, S. L., Su, S., Dong, X., Zappia, L., Ritchie, M. E., and Gouil, Q. (2020). Opportunities and challenges in long-read sequencing data analysis. *Genome Biol.* 21, 30. doi:10.1186/s13059-020-1935-5

Brazdil, P. B., Soares, C., and da Costa, J. P. (2003). Ranking learning algorithms. *Mach. Learn* 50, 251–277. doi:10.1023/A:1021713901879

Cao, H., Hastie, A. R., Cao, D., Lam, E. T., Sun, Y., Huang, H., et al. (2014). Rapid detection of structural variation in a human genome using nanochannel-based genome mapping technology. *GigaScience* 3, 34. doi:10.1186/2047-217X-3-34

Chen, Y., Zhang, Y., Wang, A. Y., Gao, M., and Chong, Z. (2021). Accurate long-read de novo assembly evaluation with Inspector. *Genome Biol.* 22, 312. doi:10.1186/s13059-021-02527-4

Chiang, C., Scott, A. J., Davis, J. R., Tsang, E. K., Li, X., Kim, Y., et al. (2017). The impact of structural variation on human gene expression. *Nat. Genet.* 49, 692–699. doi:10.1038/ng.3834

Cruz, R. M. O., Sabourin, R., Cavalcanti, G. D. C., Meta, D., and Meta, D. (2017). META-DES.Oracle: Meta-learning and feature selection for dynamic ensemble selection. *Inf. Fusion* 38, 84–103. doi:10.1016/j.inffus.2017.02.010

De Coster, W. D., Weissensteiner, M. H., and Sedlazeck, F. J. (2021). Towards population-scale long-read sequencing. *Nat. Rev. Genet.* 22, 572–587. doi:10.1038/s41576-021-00367-3

Fang, L., Hu, J., Wang, D., and Wang, K. (2019). NextSV: A meta-caller for structural variants from low-coverage long-read sequencing data. *BMC Bioinforma.* 19, 180. doi:10.1186/s12859-018-2207-1

Fernandes, J. D., Zamudio-Hurtado, A., Clawson, H., Kent, W. J., Haussler, D., Salama, S. R., et al. (2020). The UCSC repeat browser allows discovery and visualization of evolutionary conflict across repeat families. *Mob. DNA* 11, 13. doi:10.1186/s13100-020-00208-w

Gong, L., Wong, C. H., Cheng, W. C., Tjong, H., Menghi, F., Ngan, C. Y., et al. (2018). Picky comprehensively detects high-resolution structural variants in nanopore long reads. *Nat. Methods* 15, 455–460. doi:10.1038/s41592-018-0002-6

Guan, P., and Sung, W. K. (2016). Structural variation detection using next-generation sequencing data: A comparative technical review. *Methods* 102, 36–49. doi:10.1016/j.ymeth.2016.01.020

Guo, M., Li, S., Zhou, Y., Li, M., and Wen, Z. (2021). Comparative analysis for the performance of long-read-based structural variation detection pipelines in tandem repeat regions. *Front. Pharmacol.* 12, 658072. doi:10.3389/fphar.2021.658072

Handsaker, R. E., Korn, J. M., Nemesh, J., and McCarroll, S. A. (2011). Discovery and genotyping of genome structural polymorphism by sequencing on a population scale. *Nat. Genet.* 43, 269–276. doi:10.1038/ng.768

Ilchenkov, F. A., and Pendryak, A. (2015). "Datasets meta-feature description for recommending feature selection algorithm Artificial intelligence & natural language & information extraction," in Proceeding of the social media & web search fruct conference (IEEE Publications), 11–18.

Jiang, T., Liu, Y., Jiang, Y., Li, J., Gao, Y., Cui, Z., et al. (2020). Long-read-based human genomic structural variation detection with cuteSV. *Genome Biol.* 21, 189. doi:10.1186/s13059-020-02107-y

Kosugi, S., Momozawa, Y., Liu, X., Terao, C., Kubo, M., Kamatani, Y., et al. (2019). Comprehensive evaluation of structural variation detection algorithms for whole genome sequencing. *Genome Biol.* 20, 117. doi:10.1186/s13059-019-1720-5

Luan, M. W., Zhang, X. M., Zhu, Z. B., Chen, Y., and Xie, S. Q. (2020). Evaluating structural variation detection tools for long-read sequencing datasets in *Saccharomyces cerevisiae*. *Front. Genet.* 11, 159. doi:10.3389/fgene.2020.00159

Morais, R., Miranda, P., and Silva, R. (2016). "Meta-learning A," in Proceeding of the Method to select under-sampling algorithms for imbalanced data sets/Braz Conference on Intelligent Systems (IEEE Computer Society).

Northcott, P. A., Shih, D. J., Peacock, J., Garzia, L., Morrissy, A. S., Zichner, T., et al. (2012). Subgroup-specific structural variation across 1, 000 medulloblastoma genomes. *Nature* 488, 49–56. doi:10.1038/nature11327

Pise, N. (2013). Dynamic algorithm selection for data mining classification. *Adapt. Learn. Methodol. Multi-Perspective Reason.*

Rendell, L., and Cho, H. (1990). Empirical learning as a function of concept character. *Mach. Learn* 5, 267–298. doi:10.1007/bf00117106

Sedlazeck, F. J., Rescheneder, P., Smolka, M., Fang, H., Nattestad, M., von Haeseler, A., et al. (2018). Accurate detection of complex structural variations using single-molecule sequencing. *Nat. Methods* 15, 461–468. doi:10.1038/s41592-018-0001-7

Seo, J. S., Rhie, A., Kim, J., Lee, S., Sohn, M. H., Kim, C. U., et al. (2016). De novo assembly and phasing of a Korean human genome. *Nature* 538, 243–247. doi:10.1038/nature20098

Song, Q., Wang, G., and Wang, C. (2012). Automatic recommendation of classification algorithms based on data set characteristics. *Pattern Recognit.* 45, 2672–2689. doi:10.1016/j.patcog.2011.12.025

Sousa, A. F. M., Prudêncio, R. B. C., Ludermir, T. B., and Soares, C. (2016). Active learning and data manipulation techniques for generating training examples in meta-learning. *Neurocomputing* 194, 45–55. doi:10.1016/j.neucom.2016.02.007

Stancu, M. C., Roosmalen, M. V., Renkens, I., Nieboer, M. M., Middelkamp, S., de Ligt, J., et al. (2017). Mapping and phasing of structural variation in patient genomes using nanopore sequencing. *Nat. Commun.* 8, 1326. doi:10.1038/s41467-017-01343-4

Tantithamthavorn, C. (2017). ScottKnottESD: The scott-knott effect size difference (ESD) test. R Package Version2. Available at: https://github.com/klainfo/ScottKnottESD.

Vilalta, R., and Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artif. Intell. Rev.* 18, 77–95. doi:10.1023/a:1019956318069

Wang, S., Wang, J., Xiao, X., Zhang, X., Wang, X., Zhu, X., et al. (2019). "GSDcreator: An efficient and comprehensive simulator for genarating NGS data with population genetic," in Proceeding of the Information[C]//IEEE International Conference on Bioinformatics and Biomedicine (BIBM).

Wenger, A. M., Peluso, P., Rowell, W. J., Chang, P. C., Hall, R. J., Concepcion, G. T., et al. (2019). Accurate circular consensus long-read sequencing improves variant detection and assembly of a human genome. *Nat. Biotechnol.* 37, 1155–1162. doi:10.1038/s41587-019-0217-9

Yukiteru, O., Asai, K., and Hamada, M. (2013). Pbsim: PacBio reads simulator—toward accurate genome assembly. *Bioinformatics* 29, 119–121. doi:10.1093/bioinformatics/bts649

Zhao, X., Collins, R. L., Lee, W. P., Weber, A. M., Jun, Y., Zhu, Q., et al. (2020). Expectations and blind spots for structural variation detection from short-read alignment and long-read assembly. *Cell Press* 108, 919–928. doi:10.1016/j.ajhg.2021.03.014

Zook, J. M., Catoe, D., Mcdaniel, J., Vang, L., Spies, N., Sidow, A., et al. (2016). Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Sci. Data* 3, 160025. doi:10.1038/sdata.2016.25

Zook, J. M., Hansen, N. F., Olson, N. D., Chapman, L., Mullikin, J. C., Xiao, C., et al. (2020). A robust benchmark for detection of germline large deletions and insertions. *Nat. Biotechnol.* 38, 1347–1355. doi:10.1038/s41587-020-0538-8

Frontiers in Genetics

TYPE Original Research
PUBLISHED 19 January 2023
DOI 10.3389/fgene.2023.1121694

OPEN ACCESS

EDITED BY
Quan Zou,
University of Electronic Science and Technology of China, China

REVIEWED BY
Chunyan Ao,
Xidian University, China
Lumeng Chao,
Inner Mongolia Agricultural University, China

*CORRESPONDENCE
Yuhui Zhu,
zhuyuh@jlu.edu.cn

SPECIALTY SECTION
This article was submitted to Computational Genomics, a section of the journal Frontiers in Genetics

RECEIVED 12 December 2022
ACCEPTED 09 January 2023
PUBLISHED 19 January 2023

CITATION
Zhang X, Wang S, Xie L and Zhu Y (2023), PseU-ST: A new stacked ensemble-learning method for identifying RNA pseudouridine sites.
Front. Genet. 14:1121694.
doi: 10.3389/fgene.2023.1121694

PseU-ST: A new stacked ensemble-learning method for identifying RNA pseudouridine sites

Xinru Zhang, Shutao Wang, Lina Xie and Yuhui Zhu*

Department of Pharmacy, The Second Hospital of Jilin University, Changchun, China

Background: Pseudouridine (Ψ) is one of the most abundant RNA modifications found in a variety of RNA types, and it plays a significant role in many biological processes. The key to studying the various biochemical functions and mechanisms of Ψ is to identify the Ψ sites. However, identifying Ψ sites using experimental methods is time-consuming and expensive. Therefore, it is necessary to develop computational methods that can accurately predict Ψ sites based on RNA sequence information.

Methods: In this study, we proposed a new model called PseU-ST to identify Ψ sites in *Homo sapiens (H. sapiens)*, *Saccharomyces cerevisiae (S. cerevisiae)*, and *Mus musculus (M. musculus)*. We selected the best six encoding schemes and four machine learning algorithms based on a comprehensive test of almost all of the RNA sequence encoding schemes available in the iLearnPlus software package, and selected the optimal features for each encoding scheme using chi-square and incremental feature selection algorithms. Then, we selected the optimal feature combination and the best base-classifier combination for each species through an extensive performance comparison and employed a stacking strategy to build the predictive model.

Results: The results demonstrated that PseU-ST achieved better prediction performance compared with other existing models. The PseU-ST accuracy scores were 93.64%, 87.74%, and 89.64% on H_990, S_628, and M_944, respectively, representing increments of 13.94%, 6.05%, and 0.26%, respectively, higher than the best existing methods on the same benchmark training datasets.

Conclusion: The data indicate that PseU-ST is a very competitive prediction model for identifying RNA Ψ sites in *H. sapiens*, *M. musculus*, and *S. cerevisiae*. In addition, we found that the Position-specific trinucleotide propensity based on single strand (PSTNPss) and Position-specific of three nucleotides (PS3) features play an important role in Ψ site identification. The source code for PseU-ST and the data are obtainable in our GitHub repository (https://github.com/jluzhangxinrubio/PseU-ST).

KEYWORDS

RNA pseudouridine site identification, sequence analysis, computational methods, machine learning, stacked ensemble-learning

Abbreviations: Accuracy, (ACC); Adaptive Boosting, (AdaBoost); Area under the receiver operating curve, (AUC); Convolutional neural network, (CNN); Enhanced nucleic acid composition, (ENAC); eXtreme Gradient Boosting, (XGBoost); Gaussian Naive Bayes, (GaNB); Incremental feature selection, (IFS); k-nearest neighbour, (KNN); Light gradient boosting machine, (lightGBM); Logistic regression, (LR); Matthew's Correlation Coefficient, (MCC); Naïve Bayes, (NB); Nucleotide chemical property, (NCP); Position-specific of three nucleotides, (PS3); Position-specific of two nucleotides, (PS2); Position-specific trinucleotide propensity based on single-strand, (PSTNPss); Random forest, (RF); Receiver Operating Characteristic, (ROC); Sensitivity, (Sn); Specificity, (Sp); Support vector machine, (SVM).

1 Introduction

Pseudouridine (Ψ) is one of the most abundant RNA modifications found in many RNAs, such as rRNA, mRNA, tRNA, and snRNA et al. (Charette and Gray, 2000). Research on Ψ has been developing since its discovery in 1957. Many studies have shown that Ψ plays a key role in several bioprocesses, including the maintenance of RNA construction stability (Boo and Kim, 2020), the metabolism of RNA (Carlile et al., 2014; Schwartz et al., 2014), and the RNA-protein or RNA-RNA interactions (Basak and Query, 2014). Previous studies also found that Ψ mutations are related to many cancers, such us lung and stomach cancer (Itoh et al., 1989; Penzo et al., 2017; Cao et al., 2021). The key to studying the various biochemical functions and mechanisms of Ψ is to identify the Ψ sites. However, identifying Ψ sites using experimental methods is time-consuming and expensive (Adachi et al., 2019). Therefore, it is necessary to develop computational methods which can accurately predict Ψ sites based on the RNA sequence information.

In recent years, many computational predictors of Ψ sites have been developed to complement experimental studies. Li et al. (2015) established the first computational model to predict Ψ sites in *S. cerevisiae* and *H. sapiens,* named PPUS, using support vector machine (SVM) algorithms. Similarly, Chen et al. (2016) established a SVM model called iRNA-PseU by combining the encoding schemes of pseudo-nucleotide composition and nucleotide chemical property (NCP) to predict Ψ sites in 2016. Subsequently, He et al. (2018) developed another SVM classifier called PseUI, which extracts RNA sequence features using five different encoding schemes. Tahir et al. (2019) established a convolutional neural network (CNN) model, named iPseU-CNN, which employs the binary encoding scheme. In 2020, Liu et al. (2020) proposed XG-PseU using eXtreme Gradient Boosting (XGBoost) algorithms to predict Ψ sites. In the same year, Bi et al. (2020) created an ensemble model called EnsemPseU, which integrates random forest (RF),SVM, Naïve Bayes (NB), XGBoost, and k-nearest neighbours (KNN). Lv et al. (2020) developed an RF-based method called RF-PseU, which applies a light gradient boosting machine (lightGBM) algorithms to identify Ψ sites. Mu et al. (2020) presented a layered ensemble model designated as iPseU-Layer, which applies classic RF to predict Ψ sites. Then, Li et al. (2021b) proposed a computational model called Porpoise, which selects four optimal types of features and fed them into a stacked model to predict Ψ sites. Zhuang et al. (2021) proposed PseUdeep, a deep learning framework, and Wang et al. (2021) proposed a feature fusion predictor named PsoEL-PseU in the same year; however, their performance are unsatisfactory. The accuracy scores of the best existing methods mentioned above are 79.70%, 81.69%, and 89.34% in *H. sapiens*, *S. cerevisiae*, and *M. musculus*, respectively, so there is still much opportunity for improvement.

In this study, we proposed a new model called PseU-ST to identify Ψ sites in *H. sapiens, S. cerevisiae*, and *M. musculus*. First, we thoroughly tested almost all of the available RNA sequence encoding schemes in the iLearnPlus software package with seven most popular machine learning algorithms and selected the best six types of encoding schemes and four machine learning algorithms (Cui et al., 2022). We then sorted the feature importance of the six encoding schemes separately using chi-square and selected the optimal features for each encoding scheme using incremental feature selection (IFS) algorithms. We used the cross-validation tests to evaluate and select the optimal feature and base-classifier combinations for each species. Next, we employed a stacking strategy to establish a predictive model. The results demonstrated that PseU-ST achieved better prediction performance compared with other existing models. Therefore, PseU-ST is a highly competitive prediction model for identifying RNA Ψ sites in *H. sapiens, S. cerevisiae*, and *M. musculus*.

2 Materials and methods

2.1 The framework of PseU-ST

The general framework design of PseU-ST is shown in Figure 1. The framework of PseU-ST had five major steps. Step 1, we saved the training datasets and the independent test datasets from online databases (Chen et al., 2016). Step 2, we thoroughly tested almost all of the available RNA sequence encoding schemes in the iLearnPlus software package with seven most popular machine learning algorithms and selected the best six encoding schemes and four algorithms. Step 3, we sorted the feature importance of the six encoding schemes separately using chi-square and selected the optimal features for each encoding scheme using IFS algorithms. We then built models using different combinations of optimal features and selected the optimal feature combinations for each species. Step 4, we built RF, SVM, Gaussian Naive Bayes (GaNB), and logistic regression (LR) models separately using the optimal feature combination selected in the forward step as the preliminary base-classifier; LR was used as the meta-classifier, and we built a series of stacked models by using different base-classifier combinations and selected the best base-classifier combination for each species. Step 5, we compared the predictive performance of the optimised stacked model in 5-fold cross-validation and independent tests with those of other existing models.

2.2 Dataset collection

Chen et al. (Chen et al., 2016) collected datasets from RMBase (Sun et al., 2016) to identify Ψ sites by machine learning methods. First, RNA fragments with uridine (U) in the center were collected by sliding the (2ξ + 1)-tuple nucleotide window along the RNA sequences; when the center of RNA sample is confirmed as Ψ site by experiment, it is considered positive, otherwise it is negative. Then, the samples with ≥60% paired sequence identity were screened out with any other samples in the same class using CD-HIT software, and the negative and positive subsets were made to have the same size using a random-picking procedure. The training datasets contained three datasets, they were H_990 (*H. sapiens*), M_944 (*M. musculus*), and S_628 (*S. cerevisiae*), while there were only two species, namely H_200 (*H. sapiens*) and S_200 (*S. cerevisiae*) in the independent testing datasets. Both the training and independent testing datasets had half-positive and half-negative samples. In addition, Chen et al. evaluated the performance of the predictor in identifying Ψ sites with different ξ values and found that when ξ = 10, the accuracy of *H. sapiens* or *M. musculus* reached a peak value, whereas that of *S. cerevisiae* reached a peak value when ξ = 15. Thus, the RNA sequence lengths in H_990 and M_944 were both 21 nt, and that in S_628 was 31 nt. The RNA sequence lengths in H_200 and S_200 were 21 and 31 nt, respectively. In recent years, the models mentioned in the introduction have all used the same

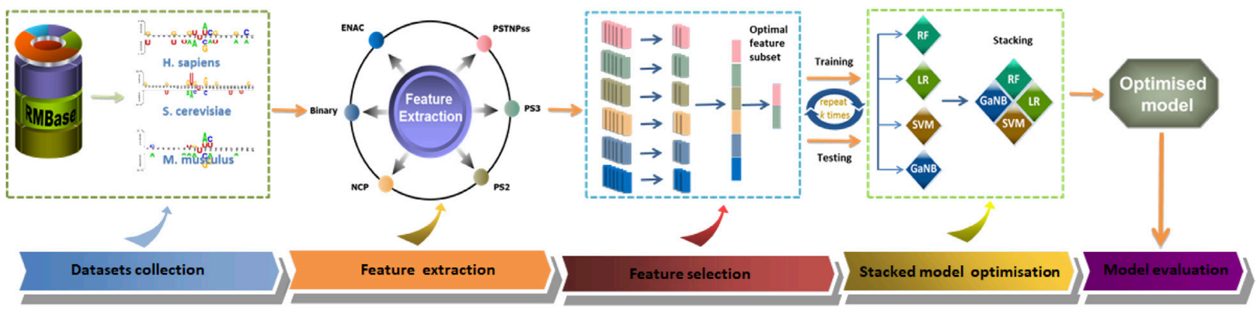


FIGURE 1
The overall framework of PseU-ST. There are five major steps, including dataset collection, feature extraction, feature selection, stacked model optimisation, and model evaluation.

TABLE 1 Training and independent dataset information.

Species	Datasets	Length (bp)	Positive samples	Negative samples
H. sapiens	H_990 (training)	21	495	495
	H_200 (testing)	21	100	100
S. cerevisiae	S_628 (training)	31	314	314
	S_200 (testing)	31	100	100
M. musculus	M_44 (training)	21	472	472
	—	—	—	—

datasets. In our study, we built the PseU-ST models using the same datasets. Detailed information on these datasets is presented in Table 1. Benchmark datasets were downloaded from http://lin-group.cn/server/iRNAPseu/data.

2.3 Feature extraction

In the computational model construction, feature extraction is a critical step. In our study, we thoroughly tested almost all of the available RNA sequence encoding schemes in the iLearnPlus software package (Chen et al., 2021). Then, according to their predictive performance, the best six encoding schemes were selected to determine the optimal feature combinations, including enhanced nucleic acid composition (ENAC), binary features, NCP, position-specific trinucleotide propensity based on single-strand (PSTNPss), position-specific of two nucleotides (PS2), and position-specific of three nucleotides (PS3) (Chen et al., 2017).

2.3.1 Enhanced nucleic acid composition

ENAC calculates the nucleic acid composition based on fixed length window (the default value is 5) of the sequence, the window slides from the 5′end of the RNA sequence to the 3′ end continuously, and encodes the RNA sequence into equal length feature vectors.

2.3.2 Binary feature (also called one-hot)

In binary encoding, four-dimensional binary vectors are used to represent nucleotides, for example, the A, C, G, and U in RNA are encoded to (1 0 0 0), (0 1 0 0), (0 0 1 0), and (0 0 0 1), respectively.

TABLE 2 Chemical structure of each nucleotide (Chen et al., 2015).

Chemical property	Class	Nucleotides
Ring Structure	Purine	A, G
	Pyrimidine	C, U
Functional Group	Amino	A, C
	Keto	G, U
Hydrogen Bond	Strong	C, G
	Weak	A, U

2.3.3 Nucleotide chemical property

According to the differences of chemical bonds and chemical structures, the four nucleotides of RNA sequences (ACGU) are classified into three different classes, as shown in Table 2.

Based on their different chemical properties, we can use three-dimensional coordinates to encode A, C, G, and U, they are encoded as (1,1,1), (0,0,1), (0,1,0), and (1,0,0), respectively.

2.3.4 Position-specific trinucleotide propensity based on single strand

The PSTNPss encodes DNA or RNA sequences using statistical rule. Generally, there were 4^3 (i.e. 64) trinucleotides, for example, AAA, AAC, AAG, UUU (TTT). Thus, for a given RNA sequence of

L-bp length, the position specificity of trinucleotide is defined as a 64 × (L-2) Matrix:

$$\mathbf{Z} = \begin{bmatrix} Z_{1,1} & Z_{1,2} & \cdots & Z_{1,L-2} \\ Z_{2,1} & Z_{2,2} & \cdots & Z_{2,L-2} \\ \vdots & \vdots & \ddots & \vdots \\ Z_{64,1} & Z_{64,2} & \cdots & Z_{64,L-2} \end{bmatrix} \quad (1)$$

where

$$Z_{i,j} = F^{+}\left(3mer_i \middle| j\right) - F^{-}\left(3mer_i \middle| j\right),\ i = 1, 2, \ldots, 64; j = 1, 2, \ldots, L-2 \quad (2)$$

F^{+}(3mer $_i$ |j) and F^{-} (3mer $_i$ |j) respectively indicate the occurrence frequency of the ith trinucleotide (3mer$_i$) at the jth position in the positive (S^{+}) and negative (S^{-}) datasets, and where 3mer$_1$ = AAA, 3mer$_2$ = AAC, and 3mer$_{64}$ = UUU. Thus, an L-bp-long RNA sequence is denoted as:

$$S = [\varnothing_1, \varnothing_2, \ldots, \varnothing_{L-2}]^T \quad (3)$$

where T is the transpose operator and $\varnothing_u$ is expressed as:

$$\varnothing_u = \begin{cases} Z_{1,u},\ when\ N_u N_{u+1} N_{u+2} = AAA \\ Z_{2,u},\ when\ N_u N_{u+1} N_{u+2} = AAG \\ \vdots \\ Z_{64,u},\ when\ N_u N_{u+1} N_{u+2} = UUU \end{cases} \quad (4)$$

Thus, in our study, the samples are denoted by 21−2 = 19 PSTNPss features in H_990 and M_944, and the samples are coded by 31−2 = 29 PSTNPss features in S_628.

2.3.5 Position-specific of two nucleotides (PS2) and position-specific of three nucleotides (PS3)

There are 16 (i.e. 4 × 4) pairs of adjacent paired nucleotides, e.g. AA/AT/AG ...; therefore, a single variable representing such a paired nucleotide can be encoded as 16 binary variables and becomes binary. For example, AA is expressed as (1000000000000000), AC is (0100000000000000) . . ., and AAC is (10000000000000001000000000000000). PS3 is encoded by three adjacent nucleotides (4 × 4 × 4 = 64) in a similar manner.

2.4 Feature selection

A helpful method to remove redundancy and avoid over-fitting in computational modelling is feature selection as it plays a crucial role in improving the model performance (Jones et al., 2021; Suresh et al., 2022). To effectively represent sequences, in this study, we first sorted the feature importance of the six encoding schemes separately using a chi-square test and selected the optimal feature set for each of them using IFS algorithms (Lv et al., 2020; Zhang et al., 2021). Subsequently, we determined the optimal feature combinations. We trained the optimal features of the six encoding schemes using the best four algorithms selected in the stacking ensemble learning model section and ranked them according to accuracy (ACC). Then, we used the first-ranked feature to build the PseU-ST model, added the second feature to build a new model, and then added the third feature until all obtained features were added. Finally, we selected the optimal feature combinations for each species.

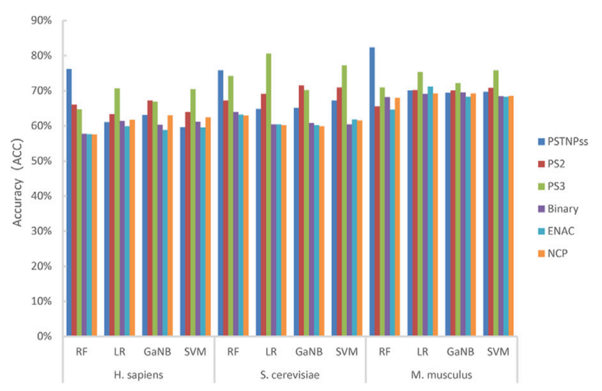


FIGURE 2
The accuracy of the four models trained using the best six encoding schemes for *H. sapiens*, *S. cerevisiae*, and *M. musculus*.

2.5 Stacking ensemble learning models

The stacking strategy can combine information from multiple classifiers to generate a more stable stacking model. It is a very useful integrated learning method that has been successfully applied to bioinformatics (Mishra et al., 2019; Li et al., 2021a). The "*mlxtend*" package in python (Raschka, 2018) provides a stacking cross-validation algorithm, which prepares input data for meta-level classifier by extending the standard stacking cross-validation algorithm. Moreover. The stacking strategy can be implemented using this algorithm. The stacking strategy can minimise the generalisation error rate of several predictive models (Su et al., 2020) and effectively avoids over-fitting (Sherwani et al., 2021). In this study, we employed a stacking strategy to establish a predictive model for RNAΨ sites. The stacking learning strategy has two major steps. Step 1, we built a series of classifiers, called base-classifiers. Step 2, we used the outputs obtained in the previous step of the base-classifiers as the input to train another classifier, called meta-classifiers.

In our study, we assessed the seven most popular algorithms: RF, LR, SVM, GaNB, Adaptive Boosting (AdaBoost), XGBoost, and Gradient Boosting Decision Tree (GBDT). RF is an integrated learning algorithm based on a decision tree. It can obtain accurate and stable predictions by building multiple decision trees and merging them. RF is one of the commonly used algorithms in bioinformatics (Lv et al., 2020; El Allali et al., 2021; Yin et al., 2021). LR is a generalised linear classification algorithm, it uses the *sigmod* function for non-linear mapping of all data to limit the prediction value to [0,1] and reduces the prediction range to classify samples. LR is a common machine learning method (Wei et al., 2020; Li and Wang, 2021; Zhu et al., 2021). SVM is another linear classification algorithm that is one of the most popular algorithms in computational biology (Chen et al., 2016; He et al., 2018). The decision boundary of SVM is to find an optimal separating hyperplane to segment samples. GaNB classifies sample data using probability and statistical methods based on the Bayesian theorem, assuming that the feature conditions are independent of each other. GaNB is also a commonly used algorithm (Yan et al., 2020; Shah et al., 2022). AdaBoost, XGBoost, and GBDT are all boosting models. They learn using different methods and form a strong classifier. They are widely used in bioinformatics

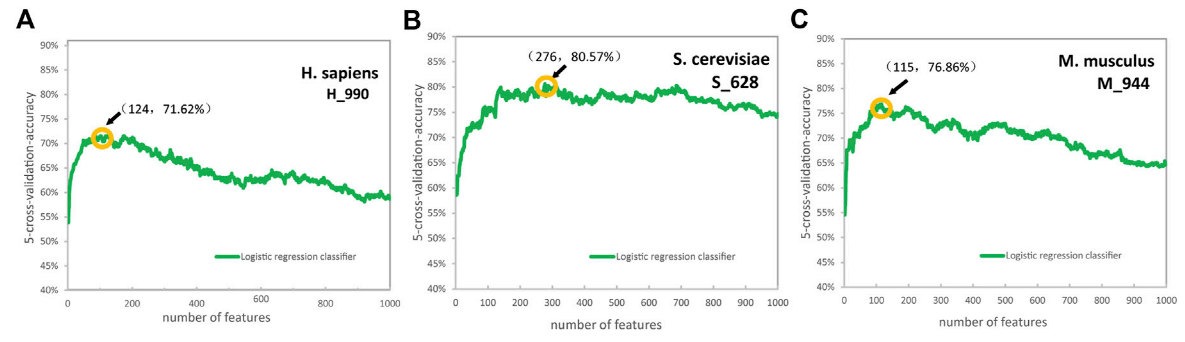


FIGURE 3
The accuracy curves for *H. sapiens* **(A)**, *S. cerevisiae* **(B)**, and *M. musculus* **(C)** of the position-specific of three nucleotides encoding schemes. (Due to the excessive dimensions of position-specific of three nucleotides features, 1000 features were selected for drawing for convenience).

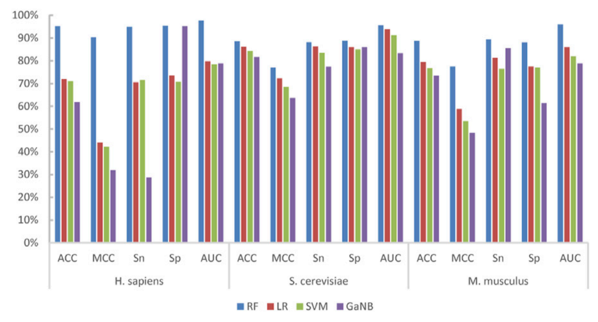


FIGURE 4
The performances of the four base-classifiers for *H. sapiens*, *S. cerevisiae*, and *M. musculus*.

(Liu et al., 2020; El Allali et al., 2021; Jayashree et al., 2022; Niu et al., 2022).

For each algorithm, we selected default parameters for training. For example, we set the tree numbers as 100 and the tree range as 100: 1000:100 for RF. For SVM, the kernel function selected rbf, the penalty parameter selected 1.0, and the penalty range and gamma range was 1.0:15.0:1.0 and −10.0:5.0:1.0, respectively. For XGBoost, the booster parameter selected gbtree, the max depth was set as 3, and the penalty range was 3:10:1. Based on these parameters, we selected the best four algorithms for training the stacked models through an extensive performance comparison. Subsequently, we trained the optimal feature combinations of the three species that were previously determined using the best four algorithms as the candidate base classifier. We trained the stacked models using LR as the meta-classifier, and we evaluated the different combinations of base classifiers to select the best base-classifier combination as the final model.

2.6 Evaluation metrics

We used several widely used performance metrics to evaluate and compare the function of PseU-ST and other existing methods. The metrics are sensitivity (Sn), specificity (Sp), accuracy (ACC), Matthew's Correlation Coefficient (MCC), and area under the receiver operating curve (AUC) (Mu et al., 2020; Li et al., 2021a; Zhuang et al., 2021). Sn, Sp, ACC, and MCC are defined as follows:

$$\mathbf{Sn} = \frac{TP}{TP + FN} \tag{5}$$

$$\mathbf{Sp} = \frac{TN}{FP + TN} \tag{6}$$

$$\mathbf{ACC} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

$$\mathbf{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}} \tag{8}$$

where TP, TN, FP, and FN represent the true positive, true negative, false positive, and false negative, respectively. We drew receiver operating characteristic (ROC) curves with 1-Sp as abscissa and Sn as ordinate and calculated AUC values.

3 Results and discussion

3.1 Determine the optimal feature combinations

First, we thoroughly tested almost all of the RNA sequence encoding schemes available in the iLearnPlus software package with seven widely used machine learning algorithms, and built models for each algorithm with default parameters. Then, the best six encoding schemes and four machine learning algorithms were selected to build the stacked models. The best six encoding schemes were ENAC, binary feature, NCP, PSTNPss, PS2, and PS3, and the best four algorithms were LR, RF, SVM, and GaNB. For each algorithm, we trained six separate classifier features and ranked them according to the ACC. The ACC of each model is shown in Figure 2.

As shown in Figure 2, RF achieved the highest ACC for H_990 and M_944, whereas LR reached the highest ACC for S_628. The PSTNPss and PS3 features formed more contributions to model than the other features. For H_990 and M_944, the RF model trained using PSTNPss features outperformed the other features. Whereas the LR model trained using PS3 features outperformed the other features for S_628. Overall, the contributions to the model performance of the six features were PSTNPss > PS3 > PS2 > binary > ENAC > NCP for *H. sapiens*, PS3 > PSTNPss > PS2 > binary > ENAC > NCP for *S.*

TABLE 3 The performances of the base-classifier combinations for the three species.

Species	Base classifiers combination	5-Fold cross- validation					Independent testing				
		ACC (%)	MCC (%)	Sn (%)	Sp (%)<	AUC (%)	ACC (%)	MCC (%)	Sn (%)	Sp (%)	AUC (%)
H. sapiens	RF + LR	**93.64**	**87.28**	**94.34**	**92.93**	**98.56**	**89.00**	**79.02**	**97.00**	**81.00**	**96.51**
	RF + LR + SVM	93.43	86.88	**94.34**	92.53	98.42	86.50	73.84	94.00	79.00	95.47
	RF + LR + SVM + GaNB	92.93	85.88	93.94	91.92	98.41	86.00	74.17	**97.00**	74.00	95.56
S. cerevisiae	RF + LR	87.74	75.49	**86.94**	88.54	**95.95**	**83.50**	**67.00**	**83.00**	**84.00**	**89.00**
	RF + LR + SVM	87.74	75.49	**86.94**	88.54	95.25	82.50	65.00	82.00	83.00	87.64
	RF + LR + SVM + GaNB	**88.06**	**76.13**	**86.94**	**89.17**	95.17	81.50	63.00	81.00	82.00	86.48
M. musculus	RF + LR	**89.60**	**79.21**	**90.66**	**88.54**	**96.20**					
	RF + LR + SVM	87.47	74.96	88.32	86.62	95.29					
	RF + LR + SVM + GaNB	87.37	74.74	88.11	86.62	95.28					

Notes: Bold values indicate the best performance in terms of the corresponding measure.

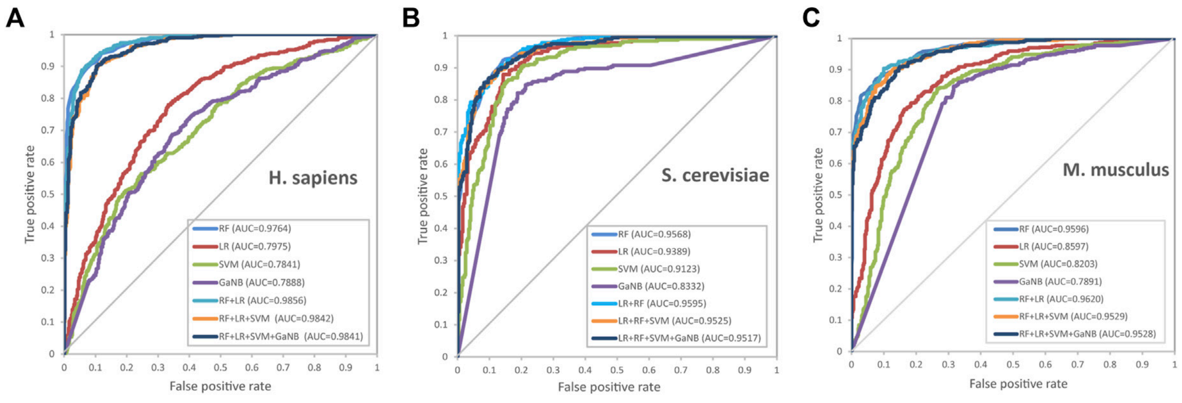


FIGURE 5
Receiver operating characteristic curves for the base classifiers and the stacked models of different base-classifier combinations during 5-fold cross-validation. The vertical coordinate is the true positive rate (sensitivity), while the horizontal coordinate is the false positive rate (1-specificity). [**(A)** *H. sapiens*, **(B)** *S. cerevisiae* and **(C)** *M. musculus*].

cerevisiae, and PSTNPss > PS3 > ENAC > PS2 > binary > NCP for *M. musculus*. However, no single type of feature consistently outperformed other features for any species, and no single algorithm consistently outperformed other algorithms for any species. We can see that a single model using a single feature is unsatisfactory; therefore, we may need to integrate learning strategies to improve model performance.

In the experiment, we found that the PS3 features made a considerable contribution to the model performance, and the feature vector dimensions of PS3 were particularly high, up to more than 1000 dimensions. In theory, the more features, the more likely it is to provide features with strong discrimination ability in limited training samples. However, too many features may cause redundancy and "dimension disaster" (Suresh et al., 2022), which will lead to a long training time of the model and the risk of over-fitting, and reduce the generalisation ability of the model. Feature selection can remove some redundant features, reduce training time, select truly relevant features, and enhance the prediction performance of the model (Jones et al., 2021; Zhang et al., 2021; Suresh et al., 2022).

Based on the LR algorithm, we employed a chi-square test and the IFS strategy to determine the optimal features (Dao et al., 2019; Lv et al., 2020; Zhang et al., 2021) were employed. We first ranked the feature importance of the six encoding schemes using a chi-square test separately, then set a whole ranked features set, named F: $F = \{f_1, f_2, ...f_{n-1}, f_n\}$, where *n* represent the features number. We tested the training dataset using the IFS by performing 5-fold cross-validation tests. In each iteration, IFS added a feature in F to the preliminary feature subset to build *n* feature subsets. When the highest ACC value was achieved, optimal feature subsets were obtained. The ACC curves for *H. sapiens*, *S. cerevisiae*, and *M. musculus* of PS3 encoding schemes are shown in Figure 3. When the number of features was the top 124, 276, and 115, we obtained the best predictive accuracies of 71.62%, 80.57%, and 76.86% for identifying Ψ sites in *H. sapiens*, *S. cerevisiae*, and *M. musculus*, respectively (Figure 3).

TABLE 4 Performance comparison of PseU-ST and other existing methods on the same benchmark training datasets.

Species	*H. sapiens*				*S. cerevisiae*				*M. musculus*			
Method	ACC (%)	MCC (%)	Sn (%)	Sp (%)	ACC (%)	MCC (%)	Sn (%)	Sp (%)	ACC (%)	MCC (%)	Sn (%)	Sp (%)
PseU-ST	**93.64**	**87.28**	**94.34**	**92.93**	**87.74**	**75.49**	**86.94**	**88.54**	**89.60**	**79.21**	**90.66**	**88.54**
PseUdeep	66.99	35.00	74.47	60.71	72.73	45.00	61.75	78.13	72.45	44.00	66.70	77.36
PsoEL-PseU	70.80	42.00	66.90	74.70	80.30	62.00	69.10	91.40	76.50	53.00	82.20	70.80
Porpoise	78.53	58.45	89.11	67.94	81.69	63.38	81.21	82.17	77.75	55.55	77.83	77.67
iPseU-Layer	79.70	60.00	71.18	88.22	80.08	60.00	77.92	81.82	89.34	79.00	84.68	93.76
RF-PseU (10-fold)	64.30	29.00	66.10	62.60	74.80	49.00	77.20	72.40	74.80	50.00	73.10	76.50
RF-PseU (LOO)	64.00	29.00	65.90	62.60	75.80	52.00	78.20	73.40	74.50	48.00	72.70	75.20
EnsemPseU	66.28	33.00	63.46	69.09	74.16	49.00	73.88	74.45	73.85	48.00	75.43	72.25
XG-PseU	65.44	31.00	63.64	67.24	68.15	37.00	66.84	69.45	72.03	45.00	76.48	67.57
iPseU-CNN	66.68	34.00	65.00	68.78	68.15	37.00	66.36	70.45	71.81	44.00	74.79	69.11
PseUI	64.24	28.00	64.85	63.64	65.13	30.00	62.74	67.52	70.44	41.00	74.58	66.31
iRNA-PseU	60.40	21.00	61.01	59.80	64.49	29.00	64.65	64.33	69.07	38.00	73.31	64.83

Notes: 10-fold–10-fold cross-validation; LOO—leave-one-out cross-validation. Bold values indicate the performance of PseU-ST.

TABLE 5 Performance comparison of PseU-ST and other existing methods on the same independent test datasets.

Species	*H. sapiens*				*S. cerevisiae*			
Method	ACC (%)	MCC (%)	Sn (%)	Sp (%)	ACC (%)	MCC (%)	Sn (%)	Sp (%)
PseU-ST	**89.00**	**79.02**	**97.00**	**81.00**	**83.50**	**67.00**	**83.00**	**84.00**
PseUdeep	66.18	33.00	73.53	58.82	80.88	62.00	77.45	84.31
PsoEL-PseU	75.50	51.00	76.00	75.00	82.00	64.00	83.00	81.00
Porpoise	77.35	55.13	82.30	72.40	83.50	67.27	88.00	79.00
iPseU-Layer	71.00	43.00	63.00	79.00	72.50	45.00	68.00	77.00
RF-PseU (10-fold)	75.00	50.00	78.00	72.00	77.00	54.00	75.00	79.00
RF-PseU (LOO)	74.00	48.00	74.00	74.00	74.50	49.00	70.00	79.00
EnsemPseU	69.50	39.00	73.00	66.00	75.00	51.00	85.00	65.00
XG-PseU	67.50	35.00	68.00	67.00	71.00	42.14	75.00	67.00
iPseU-CNN	69.00	40.00	77.72	60.81	73.50	47.00	68.76	77.82
PseUI	65.50	31.00	64.85	68.00	68.50	37.00	65.00	72.00
iRNA-PseU	61.50	23.00	58.00	65.00	60.00	20.00	63.00	57.00

Notes: 10-fold–10-fold cross-validation; LOO—leave-one-out cross-validation. Bold values indicate the performance of PseU-ST.

The ACC curves of the ENAC, binary, NCP, and PS2 encoding schemes are shown in Supplementary Figures S1–4. The optimal features are: the top 46 from 80 of ENAC, top 23 from 84 of binary, top 34 from 63 of NCP, and top 100 from 320 of PS2 for *H. sapiens*, the top 21 from 120 of ENAC, top 40 from 124 of binary, top 37 from 93 of NCP, and top 116 from 480 of PS2 for *S. cerevisiae*, and the top 17 from 80 of ENAC, top 49 from 84 of binary, top 44 from 63 of NCP, and top 63 from 320 of PS2 for *M. musculus*. The feature dimension of the PSTNPss is small; therefore, all PSTNPss features are selected.

Next, we examined the best combination of features. We used the first-ranked feature to build the PseU-ST model, added the second feature to build a new model, and then the third feature, until all of the obtained features were added. The performances of the feature combinations for *H. sapiens*, *S. cerevisiae*, and *M. musculus* are displayed in Supplementary Table S1. The optimal feature combination was PS3 + PSTNPss for *S. cerevisiae*, and that for *M. musculus* was PSTNPss + PS3, which both achieved the best performance of all metrics in either 5-fold cross-validation or independent testing (Supplementary Table S1). For *H.*

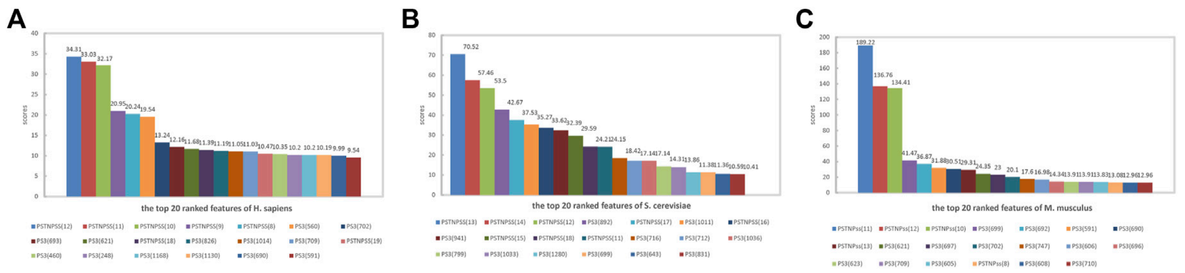


FIGURE 6
Top 20 features of PseU-ST ranked according to feature scores for predicting RNA Ψ sites of **(A)** *H. sapiens*, **(B)** *S. cerevisiae*, and **(C)** *M. musculus*.

sapiens, PSTNPss + PS3 achieved the best performance in 5-fold cross-validation, but the MCC and Sn of PSTNPss + PS3 + PS2 were better in independent testing, the ACC and Sp of PSTNPss + PS3 + PS2 + binary + ENAC were better in independent testing, but just 0.28%, 1.00%, 0.5%, and 7% higher, respectively. Therefore, PSTNPss + PS3 was selected as the optimal feature combination for *H. sapiens*.

3.2 Evaluation of the base-classifier combinations

We built integrated learning models using the stacking strategy. First, we built the RF, LR, SVM, and GaNB models separately as the candidate base classifier using the optimal feature combination selected in the forward step, namely, PSTNPss + PS3 for *H. sapiens*, PS3 + PSTNPss for *S. cerevisiae*, and PSTNPss + PS3 for *M. musculus*. We compared the performance of the four models for each species and ranked them according to ACC. The performances of the four models for each species are exhibited in Figure 4. The order of best performance the four models for each species was RF, LR, SVM, and GaNB (Figure 4). The performances of the RF models were good, but there was obvious over-fitting in *H. sapiens* and *S. cerevisiae*, so we employed the stacking strategy. We trained the stacked model using LR as the meta-classifier to determine the optimal base-classifiers. We assessed three different base-classifier combinations, which were RF + LR, RF + LR + SVM, and RF + LR + SVM + GaNB. The performances of the three combinations for each species is listed in Table 3. For *H. sapiens*, the combination of RF + LR achieved the best performance of all metrics in either cross validation or independent testing (Table 3). For *M. musculus*, the combination of RF + LR achieved the optimal performance of all metrics in cross validation too. For *S. cerevisiae*, the combination of RF + LR + SVM + GaNB achieved the best performance for almost all of the metrics in cross validation, but the performance of RF + LR had the best performance for all metrics in independent testing. Comparing the performance of the two combinations, it was found that in cross validation, the ACC, MCC, and Sp of RF + LR + SVM + GaNB were 0.32%, 0.64%, and 0.63% higher than those of RF + LR, but the AUC was lower by 0.78%, and the Sn was equal. In independent testing, the performance of RF + LR was better than that of RF + LR + SVM + GaNB in terms of all performance metrics, with ACC, MCC, Sn, Sp, and AUC being 2.00%, 4.00%, 2.00%, 2.00%, and 2.52% higher, respectively. Therefore, RF + LR was selected as the optimal base-classifier combination for *S. cerevisiae*.

We further drew ROC curves to assess the performance of base classifiers and stacked models of different combinations. As seen in Figure 5, in cross validation, the combination of RF + LR reached the optimal performance of the AUC in all three species, *H. sapiens*, *S. cerevisiae*, and *M. musculus*, which is 98.56%, 95.95%, and 96.20%, respectively. Taken together, we selected RF + LR as the optimal base-classifier combination for the stacked model and named this stacked model PseU-ST.

3.3 Comparison with the other existing methods

To further examine the performance of PseU-ST, we compared it with other existing methods using the same benchmark training, listed in Tables 4, 5. As shown in Table 4, compared with other existing methods using the same training datasets, PseU-ST performed best in three important measures across all three species, that is, ACC, MCC, and Sn. For H_990, the ACC and MCC of PseU-ST were 13.94% and 27.28% higher, respectively, than those of the second-best method, iPseU-Layer. The Sn of PseU-ST was 5.23% higher than that of the second-best method, Porpoise. For S_628, the ACC, MCC, and Sn of PseU-ST were 6.05%, 12.11%, and 5.73% higher, respectively, than those of the second-best method, Porpoise. For M_944, the ACC, MCC, and Sn of PseU-ST were 0.26%, 0.21%, and 5.98% higher, respectively, than those of the second-best method, iPseU-Layer. In addition, for H_990, the Sp of PseU-ST was 4.71% higher than that of the second-best method, iPseU-Layer.

To examine if PseU-ST models are subjected to over-fitting, we performed independent testing on independent test datasets to validate the models. The performance comparison of PseU-ST and other existing methods is presented in Table 5. As indicated, PseU-ST performed the best in all four measures for H_200. The ACC, MCC, and Sn of PseU-ST was 11.65%, 23.89%, and 14.70% higher, respectively, than those of the second-best method, Porpoise, and the Sp of PseU-ST was 2.00% higher than that of the second-best method, iPseU-Layer.

Besides, there was little difference between the prediction performance of independent and cross validation tests, for instance, the ACC and MCC of PseU-ST on H_200 was 89.00% and 79.02%, respectively, which is close to those of H_990 (93.64% and 87.28%, respectively). PseU-ST obtained an ACC of 83.5% and MCC of 67.00% on S_200, which are also very close to those of S_628 (87.74% and 75.49%, respectively), and there was no over-fitting.

In summary, compared with other existing models, PseU-ST achieved better prediction performance and had obvious advantages. PseU-ST is a highly competitive model for identifying RNA Ψ sites in *H. sapiens*, *S. cerevisiae*, and *M. musculus*.

3.4 The interpretation of model

To interpret the feature importance for the performance of the PseU-ST models. We ranked the features in the model of all three species according to feature scores and mapped the top 20 ranked features of each species in Figure 6. The PSTNPss features played an important role in the PseU-ST models; the top three important features for all three species models were PSTNPss features, and their scores were significantly higher than those of other features (Figure 6). This indicates that the PSTNP features plays a crucial role in PseU-ST models and makes more contributions to the performance of PseU-ST. Owing to the large proportion of PS3 features in the PseU-ST models, the contribution of these features to the prediction performance cannot be ignored.

4 Conclusion

In our study, a novel stacked ensemble-learning method named PseU-ST (available at https://github.com/jluzhangxinrubio/PseU-ST) was developed to identify RNA Ψ sites in *H. sapiens, S. cerevisiae*, and *M. musculus* with a more stable and accurate performance. We thoroughly evaluated almost all of the RNA sequence encoding schemes available in the iLearnPlus software package and tested seven most popular machine learning algorithms to determine the optimal feature and best base-classifier combinations. Finally, we developed an optimised model for each of the three species. Owing to the adoption of a stacking strategy and the employ of optimal feature selection algorithms, PseU-ST achieved better performance on either cross-validation or independent tests compared with the other existing models. In addition, we interpreted the feature importance for the PseU-ST models, in which PSTNPss features were shown to play an important role.

The strategies used in this study are universal and they can be employed to predict other DNA/RNA modification sites, such as DNA N4-methylcytosine and 5-methylcytosine sites. We believe PseU-ST will be a powerful tool for promoting a community-wide works for identifying Ψ sites and supplying high-quality identified Ψ sites for biological validation.

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

XZ and YZ designed this study. XZ collected datasets, built models, evaluated performance, and wrote the manuscript. SW and LX developed the software and compared performance. XZ, SW, and YZ revised the manuscript. All authors approved the final manuscript.

Funding

The Jilin Provincial Higher Education Research Project (JGJX 2020D8) supported this work.

Acknowledgments

The authors thank everyone who helped us.

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

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fgene.2023.1121694/full#supplementary-material

References

Adachi, H., De Zoysa, M. D., and Yu, Y. T. (2019). Post-transcriptional pseudouridylation in mRNA as well as in some major types of noncoding RNAs. *Biochim. Biophys. Acta Gene Regul. Mech.* 1862 (3), 230–239. doi:10.1016/j.bbagrm.2018.11.002

Basak, A., and Query, C. C. (2014). A pseudouridine residue in the spliceosome core is part of the filamentous growth program in yeast. *Cell Rep.* 8 (4), 966–973. doi:10.1016/j.celrep.2014.07.004

Bi, Y., Jin, D., and Jia, C. Z. (2020). EnsemPseU: Identifying pseudouridine sites with an ensemble approach. *Ieee Access* 8, 79376–79382. doi:10.1109/access.2020.2989469

Boo, S. H., and Kim, Y. K. (2020). The emerging role of RNA modifications in the regulation of mRNA stability. *Exp. Mol. Med.* 52 (3), 400–408. doi:10.1038/s12276-020-0407-z

Cao, C., Wang, J., Kwok, D., Cui, F., Zhang, Z., Zhao, D., et al. (2021). webTWAS: a resource for disease candidate susceptibility genes identified by transcriptome-wide association study. *Nucleic Acids Res.* 50 (D1), D1123–D1130. doi:10.1093/nar/gkab957

Carlile, T. M., Rojas-Duran, M. F., Zinshteyn, B., Shin, H., Bartoli, K. M., and Gilbert, W. V. (2014). Pseudouridine profiling reveals regulated mRNA pseudouridylation in yeast and human cells. *Nature* 515 (7525), 143–146. doi:10.1038/nature13802

Charette, M., and Gray, M. W. (2000). Pseudouridine in RNA: what, where, how, and why. *IUBMB Life* 49 (5), 341–351. doi:10.1080/152165400410182

Chen, W., Tang, H., Ye, J., Lin, H., and Chou, K. C. (2016). iRNA-PseU: Identifying RNA pseudouridine sites. *Mol. Ther. Nucleic Acids* 5 (7), e332. doi:10.1038/mtna.2016.37

Chen, W., Tran, H., Liang, Z., Lin, H., and Zhang, L. (2015). Identification and analysis of the N(6)-methyladenosine in the *Saccharomyces cerevisiae* transcriptome. *Sci. Rep.* 5, 13859. doi:10.1038/srep13859

Chen, W., Xing, P., and Zou, Q. (2017). Detecting N6-methyladenosine sites from RNA transcriptomes using ensemble Support Vector Machines. *Sci. REP-UK* 7 (1), 40242. doi:10.1038/srep40242

Chen, Z., Zhao, P., Li, C., Li, F., Xiang, D., Chen, Y. Z., et al. (2021). iLearnPlus: a comprehensive and automated machine-learning platform for nucleic acid and protein sequence analysis, prediction and visualization. *Nucleic Acids Res.* 49 (10), e60. doi:10.1093/nar/gkab122

Cui, F., Zhang, Z., Cao, C., Zou, Q., Chen, D., and Su, X. (2022). Protein–DNA/RNA interactions: Machine intelligence tools and approaches in the era of artificial intelligence and big data. *Proteomics* 22 (8), 2100197. doi:10.1002/pmic.202100197

Dao, F. Y., Lv, H., Wang, F., Feng, C. Q., Ding, H., Chen, W., et al. (2019). Identify origin of replication in *Saccharomyces cerevisiae* using two-step feature selection technique. *Bioinformatics* 35 (12), 2075–2083. doi:10.1093/bioinformatics/bty943

El Allali, A., Elhamraoui, Z., and Daoud, R. (2021). Machine learning applications in RNA modification sites prediction. *Comput. Struct. Biotechnol. J.* 19, 5510–5524. doi:10.1016/j.csbj.2021.09.025

He, J., Fang, T., Zhang, Z., Huang, B., Zhu, X., and Xiong, Y. (2018). PseUI: Pseudouridine sites identification based on RNA sequence information. *BMC Bioinforma.* 19 (1), 306. doi:10.1186/s12859-018-2321-0

Itoh, K., Mizugaki, M., and Ishida, N. (1989). Detection of elevated amounts of urinary pseudouridine in cancer patients by use of a monoclonal antibody. *Clin. Chim. Acta* 181 (3), 305–315. doi:10.1016/0009-8981(89)90236-2

Jayashree, P., Janaka Sudha, , G., Srinivasan, K. S., and Robert Wilson, S. (2022). Clinical decision support system for early detection of Alzheimer's disease using an enhanced gradient boosted decision tree classifier. *Health Inf. J.* 28 (1), 146045822210828. doi:10.1177/14604582221082868

Jones, P. J., Catt, M., Davies, M. J., Edwardson, C. L., Mirkes, E. M., Khunti, K., et al. (2021). Feature selection for unsupervised machine learning of accelerometer data physical activity clusters - a systematic review. *Gait Posture* 90, 120–128. doi:10.1016/j.gaitpost.2021.08.007

Li, F., Chen, J., Ge, Z., Wen, Y., Yue, Y., Hayashida, M., et al. (2021a). Computational prediction and interpretation of both general and specific types of promoters in *Escherichia coli* by exploiting a stacked ensemble-learning framework. *Brief. Bioinform.* 22 (2), 2126–2140. doi:10.1093/bib/bbaa049

Li, F., Guo, X., Jin, P., Chen, J., Xiang, D., Song, J., et al. (2021b). Porpoise: a new approach for accurate prediction of RNA pseudouridine sites. *Brief. Bioinform.* 22 (6), bbab245. doi:10.1093/bib/bbab245

Li, Y. H., Zhang, G., and Cui, Q. (2015). PPUS: a web server to predict PUS-specific pseudouridine sites. *Bioinformatics* 31 (20), 3362–3364. doi:10.1093/bioinformatics/btv366

Li, Y., and Wang, L. (2021). RNA coding potential prediction using alignment-free logistic regression model. *Methods Mol. Biol.* 2254, 27–39. doi:10.1007/978-1-0716-1158-6_3

Liu, K., Chen, W., and Lin, H. (2020). XG-PseU: an eXtreme gradient boosting based method for identifying pseudouridine sites. *Mol. Genet. Genomics* 295 (1), 13–21. doi:10.1007/s00438-019-01600-9

Lv, Z., Zhang, J., Ding, H., and Zou, Q. (2020). RF-PseU: A random forest predictor for RNA pseudouridine sites. *Front. Bioeng. Biotechnol.* 8, 134. doi:10.3389/fbioe.2020.00134

Mishra, A., Pokhrel, P., and Hoque, M. T. (2019). StackDPPred: a stacking based prediction of DNA-binding protein from sequence. *Bioinformatics* 35 (3), 433–441. doi:10.1093/bioinformatics/bty653

Mu, Y., Zhang, R., Wang, L., and Liu, X. (2020). iPseU-Layer: Identifying RNA pseudouridine sites using layered ensemble model. *Interdiscip. Sci.* 12 (2), 193–203. doi:10.1007/s12539-020-00362-y

Niu, M., Zou, Q., and Lin, C. (2022). CRBPDL: Identification of circRNA-RBP interaction sites using an ensemble neural network approach. *PLoS Comput. Biol.* 18 (1), e1009798. doi:10.1371/journal.pcbi.1009798

Penzo, M., Guerrieri, A. N., Zacchini, F., Treré, D., and Montanaro, L. (2017). RNA pseudouridylation in physiology and medicine: For better and for worse. *Genes (Basel)* 8 (11), 301. doi:10.3390/genes8110301

Raschka, S. (2018). MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack. *J. Open Source Softw.* 3 (24), 638. doi:10.21105/joss.00638

Schwartz, S., Bernstein, D. A., Mumbach, M. R., Jovanovic, M., Herbst, R. H., León-Ricardo, B. X., et al. (2014). Transcriptome-wide mapping reveals widespread dynamic-regulated pseudouridylation of ncRNA and mRNA. *Cell* 159 (1), 148–162. doi:10.1016/j.cell.2014.08.028

Shah, A. A., Malik, H. A. M., Mohammad, A., Khan, Y. D., and Alourani, A. (2022). Machine learning techniques for identification of carcinogenic mutations, which cause breast adenocarcinoma. *Sci. Rep.* 12 (1), 11738. doi:10.1038/s41598-022-15533-8

Sherwani, F., Ibrahim, B., and Asad, M. M. (2021). Hybridized classification algorithms for data classification applications: A review. *Egypt. Inf. J.* 22 (2), 185–192. doi:10.1016/j.eij.2020.07.004

Su, R., Liu, X., Xiao, G., and Wei, L. (2020). Meta-GDBP: a high-level stacked regression model to improve anticancer drug response prediction. *Brief. Bioinform.* 21 (3), 996–1005. doi:10.1093/bib/bbz022

Sun, W. J., Li, J. H., Liu, S., Wu, J., Zhou, H., Qu, L. H., et al. (2016). RMBase: a resource for decoding the landscape of RNA modifications from high-throughput sequencing data. *Nucleic Acids Res.* 44 (D1), D259–D265. doi:10.1093/nar/gkv1036

Suresh, S., Newton, D. T., Everett, T. H. t., Lin, G., and Duerstock, B. S. (2022). Feature selection techniques for a machine learning model to detect autonomic dysreflexia. *Front. Neuroinform.* 16, 901428. doi:10.3389/fninf.2022.901428

Tahir, M., Tayara, H., and Chong, K. T. (2019). iPseU-CNN: Identifying RNA pseudouridine sites using convolutional neural networks. *Mol. Ther. Nucleic Acids* 16, 463–470. doi:10.1016/j.omtn.2019.03.010

Wang, X., Lin, X., Wang, R., Han, N., Fan, K., Han, L., et al. (2021). A feature fusion predictor for RNA pseudouridine sites with particle swarm optimizer based feature selection and ensemble learning approach. *Curr. Issues Mol. Biol.* 43 (3), 1844–1858. doi:10.3390/cimb43030129

Wei, Z., Qi, X., Chen, Y., Xia, X., Zheng, B., Sun, X., et al. (2020). Bioinformatics method combined with logistic regression analysis reveal potentially important miRNAs in ischemic stroke. *Biosci. Rep.* 40 (8), BSR20201154. doi:10.1042/bsr20201154

Yan, C., Wu, F. X., Wang, J., and Duan, G. (2020). PESM: predicting the essentiality of miRNAs based on gradient boosting machines and sequences. *BMC Bioinforma.* 21 (1), 111. doi:10.1186/s12859-020-3426-9

Yin, S., Tian, X., Zhang, J., Sun, P., and Li, G. (2021). PCirc: random forest-based plant circRNA identification software. *BMC Bioinforma.* 22 (1), 10. doi:10.1186/s12859-020-03944-1

Zhang, Z. Y., Yang, Y. H., Ding, H., Wang, D., Chen, W., and Lin, H. (2021). Design powerful predictor for mRNA subcellular location prediction in *Homo sapiens*. *Brief. Bioinform.* 22 (1), 526–535. doi:10.1093/bib/bbz177

Zhu, F., Zuo, L., Hu, R., Wang, J., Yang, Z., Qi, X., et al. (2021). A ten-genes-based diagnostic signature for atherosclerosis. *BMC Cardiovasc. Disord.* 21 (1), 513. doi:10.1186/s12872-021-02323-9

Zhuang, J., Liu, D., Lin, M., Qiu, W., Liu, J., and Chen, S. (2021). PseUdeep: RNA pseudouridine site identification with deep learning algorithm. *Front. Genet.* 12, 773882. doi:10.3389/fgene.2021.773882