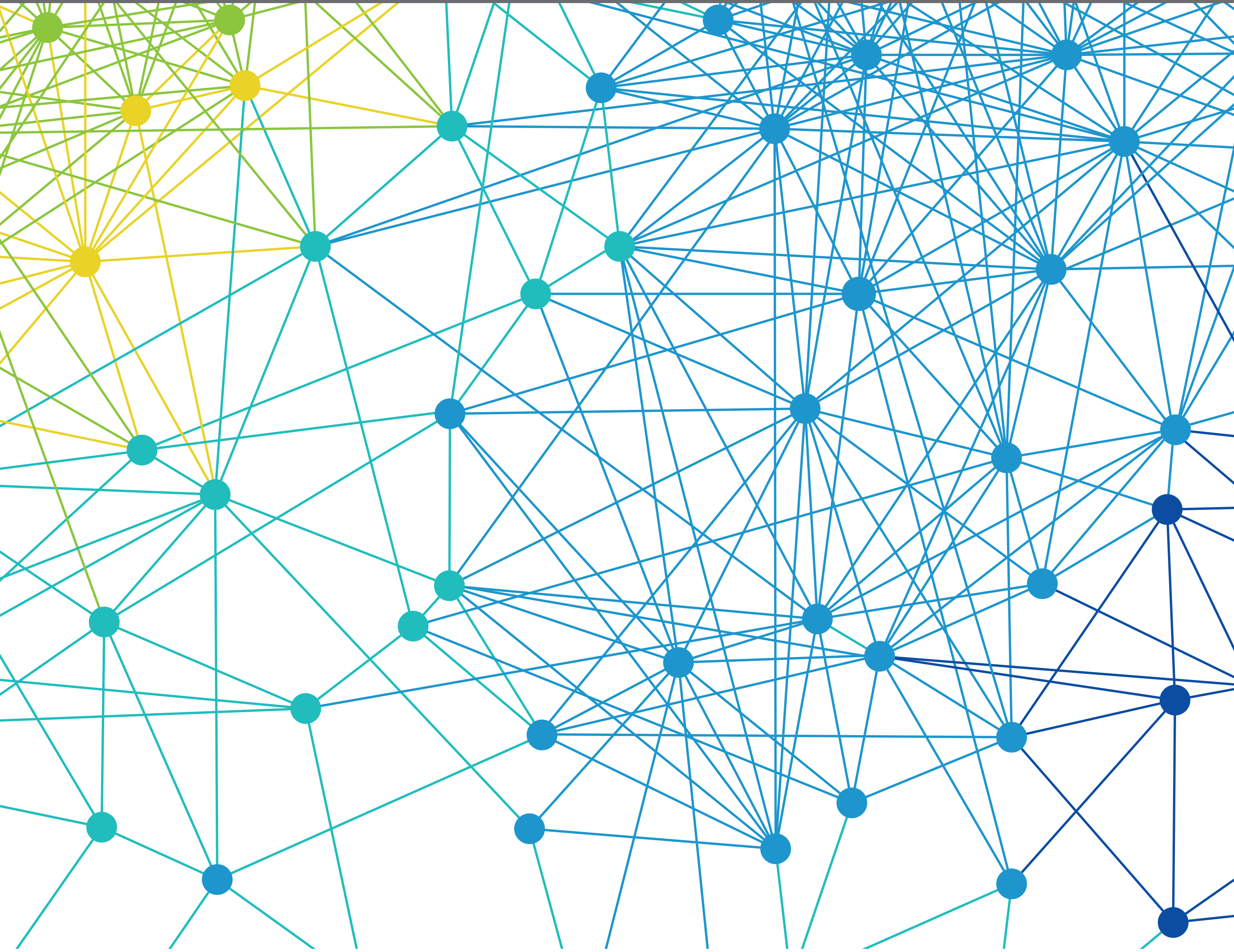
SAFE AND TRUSTWORTHY MACHINE LEARNING

EDITED BY: Bhavya Kailkhura, Xue Lin, Pin-Yu Chen and Bo Li

PUBLISHED IN: Frontiers in Big Data and Frontiers in Artificial Intelligence



frontiers Research Topics

About Frontiers

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

Frontiers Journal Series

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

Dedication to Quality

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

What are Frontiers Research Topics?

Frontiers Research Topics are very popular trademarks of the Frontiers Journals Series: they are collections of at least ten articles, all centered on a particular subject. With their unique mix of varied contributions from Original Research to Review Articles, Frontiers Research Topics unify the most influential researchers, the latest key findings and historical advances in a hot research area! Find out more on how to host your own Frontiers Research Topic or contribute to one as an author by contacting the Frontiers Editorial Office: frontiersin.org/about/contact

ISSN 1664-8714
ISBN 978-2-88971-414-8
DOI 10.3389/978-2-88971-414-8

SAFE AND TRUSTWORTHY MACHINE LEARNING

Topic Editors:
Bhavya Kailkhura, United States Department of Energy (DOE), United States
Xue Lin, Northeastern University, United States
Pin-Yu Chen, IBM Research, United States
Bo Li, University of Illinois at Urbana-Champaign, United States

Citation: Kailkhura, B., Lin, X., Chen, P.-Y., Li, B., eds. (2021). Safe and Trustworthy Machine Learning. Lausanne: Frontiers Media SA. doi: 10.3389/978-2-88971-414-8

Table of Contents

EDITORIAL
published: 06 August 2021
doi: 10.3389/fdata.2021.731605

Editorial: Safe and Trustworthy Machine Learning

Bhavya Kailkhura[1]*, *Pin-Yu Chen*[2], *Xue Lin*[3] *and Bo Li*[4]

[1] Lawrence Livermore National Laboratory, Livermore, CA, United States, [2] IBM Research, Yorktown Heights, NY, United States, [3] Electrical and Computer Engineering, Northeastern University, Boston, MA, United States, [4] Computer Science, University of Illinois Urbana-Champaign, Champaign, IL, United States

Keywords: robust machine learning, safe machine learning, trust, deep neural network, adversarial example defense

Editorial on the Research Topic

Safe and Trustworthy Machine Learning

Machine learning (ML) provides incredible opportunities to answer some of the most important and difficult questions in a wide range of applications. However, ML systems often face a major challenge when applied in the real world: the conditions under which the system was deployed can differ from those under which it was developed. Recent examples have shown that ML methods are highly susceptible to minor changes in image orientation, minute amounts of adversarial corruptions, or bias in the data. Susceptibility of ML methods to test-time shift is a major hurdle in a universal acceptance of these solutions in several high-regret applications. To overcome this challenge, in this research topic "Safe and Trustworthy Machine Learning", a wide range of solutions are contributed as potentially viable solutions to address trust, safety and security issues faced by ML methods.

OPEN ACCESS

Edited and reviewed by:
Sriraam Natarajan,
The University of Texas at Dallas,
United States

****Correspondence:***
Bhavya Kailkhura
kailkhura1@llnl.gov

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Big Data

Received: *27 June 2021*
Accepted: *27 July 2021*
Published: *06 August 2021*

Citation:
Kailkhura B, Chen P-Y, Lin X and Li B (2021) Editorial: Safe and Trustworthy Machine Learning. Front. Big Data 4:731605. doi: 10.3389/fdata.2021.731605

PAPERS INCLUDED IN THIS RESEARCH TOPIC

Song, et al., considered the problem of dataset shift detection for safety-critical graph applications. The authors proposed a practical two-sample test approach for shift detection in large-scale graph structured data.

Anirudh, et al., considered the problem of post-hoc interpretability tasks, such as, prediction explanation, noisy label detection, adversarial example detection. The authors introduced MARGIN, a simple yet general approach, that exploits ideas rooted in graph signal analysis to determine the most influential nodes in a graph to solve the aforementioned tasks.

Majumdar, et al., considered the problem of mitigation of bias arising due to unbalanced representation of sub-groups in the training data. The authors proposed a bias mitigation algorithm to generate Subgroup Invariant Perturbation (SIP) which when added the input dataset reduces the bias in model predictions.

Huang, et al., showed that seq2seq models, successful in natural language correction, is also applicable in programming language correction. Their results show that seq2seq models can provide suggestions to potential errors and have a decent correct rate in code auto-correction task.

Qayyum, et al., conducted a systematic evaluation of literature of cloud-hosted ML/DL models along both the important dimensions -- attacks and defenses -- related to their security. The authors identified the limitations and pitfalls of the analyzed papers and highlight open research issues that require further investigation.

Berghoff, et al., presented a comprehensive list of threats and possible mitigations of IT security of connectionist artificial intelligence (AI) applications. AI-specific vulnerabilities such as adversarial attacks and poisoning attacks as well as their AI-specific root causes are discussed in detail. The article concluded that single protective measures are not sufficient but rather multiple measures on different levels must be combined to achieve a minimum level of IT security for AI applications.

Kusters, et al., analyzed key challenges to interdisciplinary AI research, and deliver three broad conclusions: 1) future development of AI should not only impact other scientific domains but should also take inspiration and benefit from other fields of science, 2) AI research must be accompanied by decision explainability, dataset bias transparency as well as development of evaluation methodologies and creation of regulatory agencies to ensure responsibility, and 3) AI education should receive more attention, efforts and innovation from the educational and scientific communities.

CONCLUSIONS AND OUTLOOK

The papers included in this research topic "Safe and Trustworthy Machine Learning" discussed some promising solutions, highlighted open research issues, and offered visionary perspectives regarding trust, safety and security issues faced by machine learning. We hope that challenges and potential solutions presented here will help researchers better understand the current limitations of machine learning methods and motivate future work in the direction of developing trustworthy, safe, and robust machine learning methods, and their applications to high-regret application areas.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

Conflict of Interest: Author P-YC was employed by the company IBM Research.

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

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

frontiers in Big Data

REVIEW
published: 22 July 2020
doi: 10.3389/fdata.2020.00023

Vulnerabilities of Connectionist AI Applications: Evaluation and Defense

*Christian Berghoff**†, *Matthias Neu and Arndt von Twickel**†

Federal Office for Information Security, Bonn, Germany

This article deals with the IT security of connectionist artificial intelligence (AI) applications, focusing on threats to integrity, one of the three IT security goals. Such threats are for instance most relevant in prominent AI computer vision applications. In order to present a holistic view on the IT security goal integrity, many additional aspects, such as interpretability, robustness and documentation are taken into account. A comprehensive list of threats and possible mitigations is presented by reviewing the state-of-the-art literature. AI-specific vulnerabilities, such as adversarial attacks and poisoning attacks are discussed in detail, together with key factors underlying them. Additionally and in contrast to former reviews, the whole AI life cycle is analyzed with respect to vulnerabilities, including the planning, data acquisition, training, evaluation and operation phases. The discussion of mitigations is likewise not restricted to the level of the AI system itself but rather advocates viewing AI systems in the context of their life cycles and their embeddings in larger IT infrastructures and hardware devices. Based on this and the observation that adaptive attackers may circumvent any single published AI-specific defense to date, the article concludes that single protective measures are not sufficient but rather multiple measures on different levels have to be combined to achieve a minimum level of IT security for AI applications.

Keywords: artificial intelligence, neural network, IT security, interpretability, certification, adversarial attack, poisoning attack

OPEN ACCESS

Edited by:
Xue Lin,
Northeastern University, United States

Reviewed by:
Ping Yang,
Binghamton University, United States
Fuxun Yu,
George Mason University,
United States

****Correspondence:***
Christian Berghoff
christian.berghoff@bsi.bund.de
Arndt von Twickel
arndt.twickel@bsi.bund.de

†These authors have contributed equally to this work

Specialty section:
This article was submitted to Cybersecurity and Privacy, a section of the journal Frontiers in Big Data

Received: *20 March 2020*
Accepted: *10 June 2020*
Published: *22 July 2020*

Citation:
Berghoff C, Neu M and von Twickel A (2020) Vulnerabilities of Connectionist AI Applications: Evaluation and Defense. Front. Big Data 3:23. doi: 10.3389/fdata.2020.00023

1. INTRODUCTION

This article is concerned with the IT security aspects of artificial intelligence (AI) applications[1], namely their vulnerabilities and possible defenses. As any IT component, AI systems may not work as intended or may be targeted by attackers. Care must hence be taken to guarantee an appropriately high level of safety and security. This applies in particular whenever AI systems are used in applications where certain failures may have far-reaching and potentially disastrous impacts including the death of people. Examples commonly cited include computer vision tasks from biometric identification and authentication as well as driving on-road vehicles at higher levels of autonomy (ORAD Committee, 2018). Since the core problem of guaranteeing a secure and safe operation of AI systems lies at the intersection of the areas of AI and IT security, this article targets readers from both communities.

[1] AI is here defined as the capability of a machine to either autonomously take decisions or to support humans in making decisions. In order to distinguish AI from trivial functions, such as, for instance, a sensor that directly triggers an action using a threshold function, one might narrow the definition to non-trivial functions but since this term is not clearly defined, we refrain from doing so.

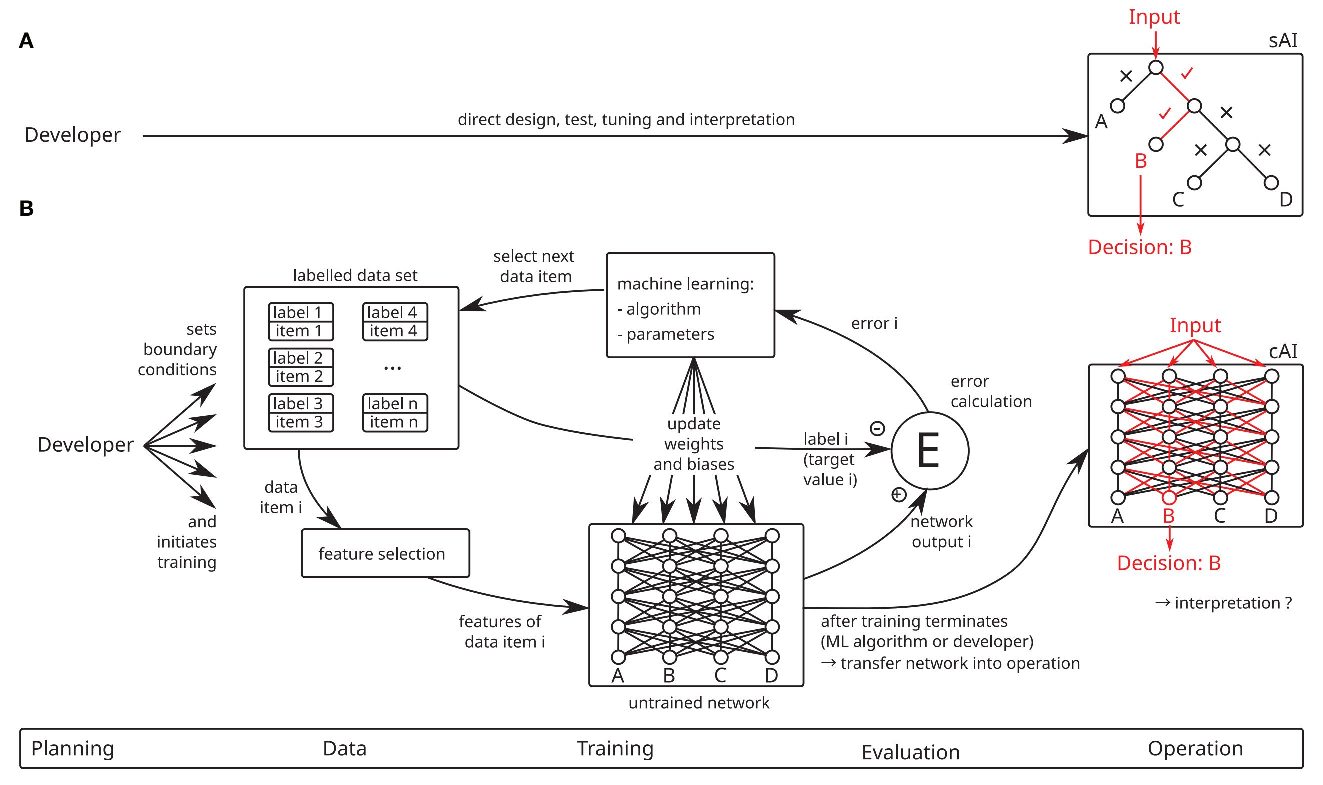


FIGURE 1 | Contrasting the development of **(A)** symbolic AI (sAI) and **(B)** connectionist AI (cAI) systems. Whereas sAI systems are directly designed by a human developer and are straightforward to interpret, cAI systems are trained by means of machine learning (ML) algorithms using large data sets (this figure shows supervised learning using a labeled data set). Due to their indirect design and their distributed decision-making, cAI systems are very hard to interpret.

1.1. Symbolic vs. Connectionist AI

AI systems are traditionally divided into two categories: symbolic AI (sAI) and non-symbolic (or connectionist) AI (cAI) systems. sAI has been a subject of research for many decades, starting from the 1960s (Lederberg, 1987). In sAI, problems are directly encoded in a human-readable model and the resulting sAI system is expected to take decisions based on this model. Examples of sAI include rule-based systems using decision trees (expert systems), planning systems and constraint solvers. In contrast, cAI systems consist of massively parallel interconnected systems of simple processing elements, similar in spirit to biological brains. cAI includes all variants of neural networks, such as deep neural networks (DNNs), convolutional neural networks (CNNs) and radial basis function networks (RBFNs) as well as support-vector machines (SVMs). Operational cAI models are created indirectly using training data and machine learning and are usually not human-readable. The basic ideas for cAI systems date back to as early as 1943 (McCulloch and Pitts, 1943). After a prolonged stagnation in the 1970s, cAI systems slowly started to gain traction again in the 1980s (Haykin, 1999). In recent years, starting from about 2009, due to significant improvements in processing power and the amount of example data available, the performance of cAI systems has tremendously improved. In many areas, cAI systems nowadays outperform sAI systems and even humans. For this reason, they are used in many applications, and new proposals for using them seem to be made on a daily basis. Besides pure cAI and sAI systems, hybrid systems exist. In this article, sAI is considered a traditional IT system and the focus is on cAI systems, especially due to their qualitatively new vulnerabilities that in turn require qualitatively new evaluation and defense methods. Unless otherwise noted, the terms AI and cAI will from now on be used interchangeably.

1.2. Life Cycle of AI Systems

In contrast to sAI and traditional IT systems, cAI systems are not directly constructed by a human programmer (cf. **Figure 1**). Instead, a developer determines the necessary boundary conditions, i.e., required performance[2], an untrained AI system, training data and a machine learning (ML) algorithm, and then starts a ML session, during which a ML algorithm trains the untrained AI system using the training data. This ML session consists of alternating training and validation phases (not shown in **Figure 1**) and is repeated until the required performance of the AI system is achieved. If the desired performance is not reached within a predefined number of iterations or if performance ceases to increase beforehand, the training session is canceled and a new one is started. Depending on the ML policy, the training session

[2] In contrast to narrowing the term performance to cover only accuracy, we use it in a broader sense, cf. subsection 2.1 for details.

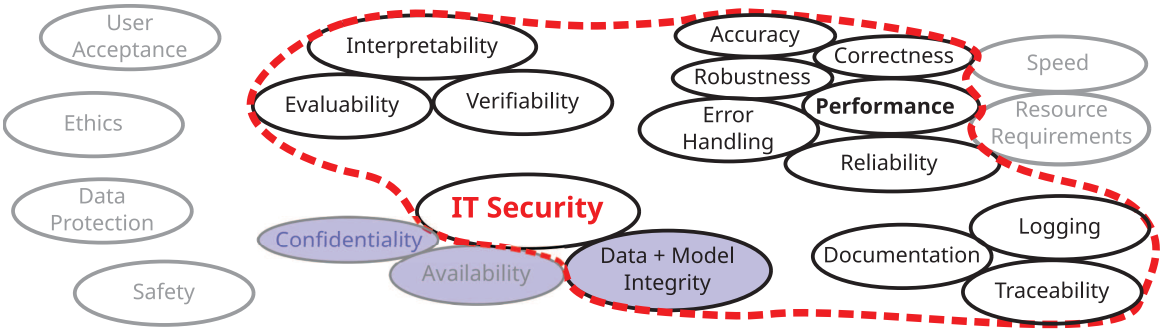


FIGURE 2 | Besides the three core properties confidentiality, integrity, and availability, a holistic view on the IT security of AI applications involves many additional aspects. This paper focuses on data and model integrity and important related aspects, especially robustness, interpretability and documentation, here depicted in the center and encircled with a red line. Note that due to a lack of common definitions and concepts across disciplines, this figure is neither complete nor are the terms used unambiguous.

is initialized anew using randomized starting conditions or the boundary conditions are manually adjusted by the developer. Once the desired performance is achieved, it is validated using the test data set, which must be independent from the training data set. Training can be performed in the setting of supervised learning, where the input data contain preassigned labels, which specify the correct corresponding output (as shown in **Figure 1**), or unsupervised learning, where no labels are given and the AI system learns some representation of the data, for instance by clustering similar data points. While this article takes the perspective of supervised learning, most of its results also apply to the setting of unsupervised learning. After successful training, the AI system can be used on new, i.e., previously unknown, input data to make predictions, which is called inference.

Due to this development process, cAI systems may often involve life cycles with complex supply chains of data, pre-trained systems and ML frameworks, all of which potentially impact security and, therefore, also safety. It is well-known that cAI systems exhibit vulnerabilities which are different in quality from those affecting classical software. One prominent instance are so-called adversarial examples, i.e., input data which are specially crafted for fooling the AI system (cf. subsection 2.5). This new vulnerability is aggravated by the fact that cAI systems are in most practical cases inherently difficult to interpret and evaluate (cf. subsection 3.2). Even if the system resulting from the training process yields good performance, it is usually not possible for a human to understand the reasons for the predictions the system provides. In combination with the complex life cycle as presented in section 2 this is highly problematic, since it implies that it is not possible to be entirely sure about the correct operation of the AI system even under normal circumstances, let alone in the presence of attacks. This is in analogy to human perception, memory and decision-making, which are error-prone, may be manipulated (Eagleman, 2001; Loftus, 2005; Wood et al., 2013, cf. also **Figure 6**) and are often hard to predict by other humans (Sun et al., 2018). As with human decision-making, a formal verification of cAI systems is at least extremely difficult, and user adoption of cAI systems may be hampered by a lack of trust.

1.3. IT Security Perspective on AI Systems

In order to assess a system from the perspective of IT security, the three main security goals[3] are used, which may all be targeted by attackers (Papernot et al., 2016d; Biggio and Roli, 2018):

1. Confidentiality, the protection of data against unauthorized access. A successful attack may for instance uncover training data in medical AI prognostics.
2. Availability, the guarantee that IT services or data can always be used as intended. A successful attack may for instance make AI-based spam filters block legitimate messages, thus hampering their normal operation.
3. Integrity, the guarantee that data are complete and correct and have not been tampered with. A successful attack may for instance make AI systems produce specific wrong outputs.

This article focuses on integrity, cf. **Figure 2**, since this is the most relevant threat in the computer vision applications cited above, which motivate our interest in the topic. Confidentiality and availability are thus largely out of scope. Nevertheless, further research in their direction is likewise required, since in other applications attacks on these security goals may also have far-reaching consequences, as can be seen by the short examples mentioned above.

Besides the three security goals, an AI system has to be assessed in terms of many additional aspects, cf. **Figure 2**. While this paper is focused on the integrity of the AI model and the data used, it also touches important related aspects, such as robustness, interpretability, and documentation.

1.4. Related Work

Although the broader AI community remains largely unaware of the security issues involved in the use of AI systems, this topic has been studied by experts for many years now. Seminal works, motivated by real-world incidents, were concerned

[3] We note that the concepts covered by the terms availability and integrity differ to some extent from the ones they usually denote. Indeed, prevalent attacks on availability are the result of a large-scale violation of integrity of the system's output data. However, this usage has widely been adopted in the research area.

with attacks and defenses for simple classifiers, notably for spam detection (Dalvi et al., 2004; Lowd and Meek, 2005; Barreno et al., 2006; Biggio et al., 2013). The field witnessed a sharp increase in popularity following the first publications on adversarial examples for deep neural networks (Szegedy et al., 2014; Goodfellow et al., 2015, cf. subsection 2.5). Since then, adversarial examples and data poisoning attacks (where an attacker manipulates the training data, cf. subsubsection 2.2.2) have been the focus of numerous publications. Several survey articles (Papernot et al., 2016d; Biggio and Roli, 2018; Liu Q. et al., 2018; Xu et al., 2020) provide a comprehensive overview of attacks and defenses on the AI level.

Research on verifying and proving the correct operation of AI systems has also been done, although it is much scarcer (Huang et al., 2017; Katz et al., 2017; Gehr et al., 2018; Singh et al., 2019). One approach to this problem is provided by the area of explainable AI (XAI, cf. subsection 4.3), which seeks to make decisions taken by an AI system comprehensible to humans and thus to mitigate an essential shortcoming of cAI systems.

Whereas previous survey articles like the ones cited above focus on attacks and immediate countermeasures on the level of the AI system itself, our publication takes into account the whole life cycle of an AI system (cf. section 2), including data and model supply chains, and the fact that the AI system is just part of a larger IT system. On the one hand, for doing so, we draw up a more complete list of attacks which might ultimately affect the AI system. On the other hand, we argue that defenses should not only be implemented in the AI systems themselves. Instead, more general technical and organizational measures must also be considered (as briefly noted in Gilmer et al., 2018) and in particular new AI-specific defenses have to be combined with classical IT security measures.

1.5. Outline

The outline of the paper is as follows: First, we inspect the life cycle of cAI systems in detail in section 2, identifying and analyzing vulnerabilities. AI-specific vulnerabilities are further analyzed in section 3 in order to give some intuition about the key factors underlying them which are not already familiar from other IT systems. Subsequently, section 4 sets out to present mitigations to the threats identified in section 2, focusing not only on the level of the AI system itself but taking a comprehensive approach. We conclude in section 5, where we touch on future developments and the crucial aspect of verifying correct operation of an AI system.

2. GENERALIZED AI LIFE CYCLE

In this section, we perform a detailed walk through the life cycle of cAI systems (cf. **Figure 3**), mostly adopting the point of view of functionality or IT security. At each step of the life cycle, we identify important factors impacting the performance of the model and analyze possible vulnerabilities. Since our objective is to provide a comprehensive overview, we discuss both classical vulnerabilities well-known from traditional IT systems as well as qualitatively new attacks which are specific to AI systems. Whereas classical vulnerabilities should be addressed using existing evaluation and defense methods, AI-specific attacks additionally require novel countermeasures, which are discussed in this section to some extent, but mostly in section 4.

The life cycle we consider for our analysis is that of a generalized AI application. This approach is useful in order to get the whole picture at a suitable level of abstraction. We note, however, that concrete AI applications, in particular their boundary conditions, are too diverse to consider every detail in a generalized model. For instance, AI systems can be used for making predictions from structured and tabular data, for computer vision tasks and for speech recognition but also for automatic translation or for finding optimal strategies under a certain set of rules (e.g., chess, go). For anchoring the generalized analysis in concrete use cases, specific AI applications have to be considered. It may hence be necessary to adapt the general analysis to the concrete setting in question or at least to the broader application class it belongs to. In the following, we use the example of traffic sign recognition several times for illustrating our abstract analysis.

2.1. Planning

The first step that is required in the development of an operational AI system is a thorough problem statement answering the question which task has to be solved under which boundary conditions. Initially, the expected inputs to the system as well as their distribution and specific corner cases are defined

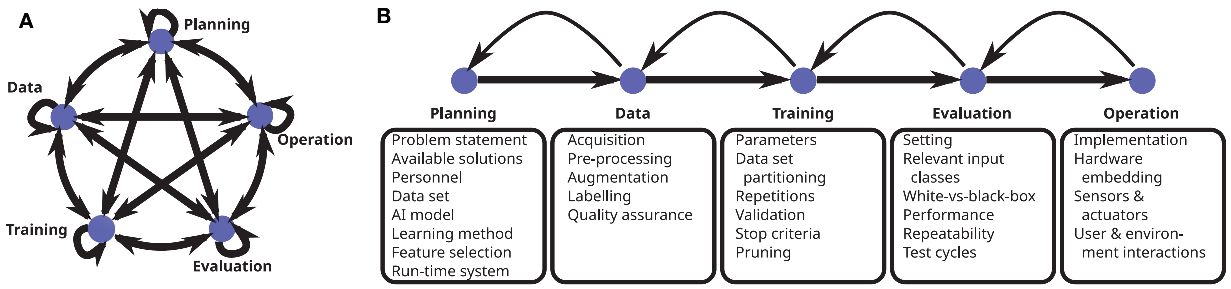


FIGURE 3 | The development of cAI applications may be broken down into phases. **(A)** In reality, the development process is non-sequential, often relies on intuition and experience and involves many feedback loops on different levels. The developer tries to find the quickest route to an operational AI system with the desired properties. **(B)** For a simplified presentation, sequential phases are depicted. Here prominent functional components are shown for each phase. Besides this functional perspective, the phases may be considered in terms of robustness, data protection, user acceptance or other aspects.

and the required performance of the system with respect to these inputs is estimated, including:

- The accuracy, or some other appropriate metric to assess the correctness of results of the system,
- The robustness, e.g., with respect to inputs from a data distribution not seen during training, or against maliciously crafted inputs,
- The restrictions on computing resources (e.g., the system should be able to run on a smartphone) and
- The runtime, i.e., combined execution time and latency.

Next, it might be helpful to analyze if the problem at hand can be broken down into smaller sub-tasks which could each be solved on their own. One may hope that the resulting modules are less complex compared to a monolithic end-to-end system and, therefore, are better accessible for interpretation and monitoring. Once the problem and the operational boundary conditions have been clearly defined, the state of the art of available solutions to related problems is assessed. Subsequently, one or several model classes and ML algorithms [e.g., back-propagation of error (Werbos, 1982)] for training the models are chosen which are assumed to be capable of solving the given task. In case a model class based on neural networks is chosen, a pre-trained network might be selected as a base model. Such a network has been trained beforehand on a possibly different task with a large data set [e.g., ImageNet (Stanford Vision Lab, 2016)] and is used as a starting point in order to train the model for solving the task at hand using transfer learning. Such pre-trained networks [e.g., BERT (Devlin et al., 2019) in the context of natural language processing] can pose a security threat to the AI system if they are modified or trained in a malicious way as described in sections 2.2, 2.3.

Based on the choices made before, the required resources in terms of quantity and quality (personnel, data set, computing resources, hardware, test facilities, etc.) are defined. This includes resources required for threat mitigation (cf. section 4). Appropriate preparations for this purpose are put into effect. This applies in particular to the documentation and cryptographic protection of intermediate data, which affects all phases up until operation.

In order to implement the model and the ML algorithm, software frameworks [e.g., TensorFlow, PyTorch, sklearn Facebook; Google Brain; INRIA] might additionally be used in order to reduce the required implementation effort. This adds an additional risk in the form of possible bugs or backdoors which might be contained in the frameworks used.

2.2. Data Acquisition and Pre-processing

After fixing the boundary conditions, appropriate data for training and testing the model need to be collected and pre-processed in a suitable way. To increase the size of the effective data set without increasing the resource demands, the data set may be augmented by both transformations of the data at hand and synthetic generation of suitable data. The acquisition can start from scratch or rely on an existing data set. In terms of efficiency and cost, the latter approach is likely to perform better. However, it also poses additional risks in terms of IT security, which need to be assessed and mitigated.

Several properties of the data can influence the performance of the model under normal and adverse circumstances. Using a sufficient quantity of data of good quality is key to ensuring the model's accuracy and its ability to generalize to inputs not seen during training. Important features related to the quality of data are, in a positive way, the correctness of their labels (in the setting of supervised learning) and, in a negative way, the existence of a bias. If the proportion of wrongly labeled data (also called noisy data) in the total data set is overly large, this can cripple the model's performance. If the training data contain a bias, i.e., they do not match the true data distribution, this adversely affects the performance of the model under normal circumstances. In special cases it might be necessary though to use a modified data distribution in the training data to adequately consider specific corner cases. Furthermore, one must ensure that the test set is independent from the training set in order to obtain reliable information on the model's performance. To trace back any problems that arise during training and operation, a sufficient documentation of the data acquisition and pre-processing phase is mandatory.

2.2.1. Collecting Data From Scratch

A developer choosing to build up his own data set has more control over the process, which can make attacks much more difficult. A fundamental question is whether the environment from which the data are acquired is itself controlled by the developer or not. For instance, if publicly available data are incorporated into the data set, the possibility of an attacker tampering with the data in a targeted way may be very small, but the extraction and transmission of the data must be protected using traditional measures of IT security. These should also be used to prevent subsequent manipulations in case an attacker gets access to the developer's environment. In addition, the data labeling process must be checked to avoid attacks. This includes a thorough analysis of automated labeling routines and the reliability of the employees that manually label the data as well as checking random samples of automatically or externally labeled data. Moreover, when building up the data set, care must be taken that it does not contain a bias.

2.2.2. Using Existing Data

If an existing data set is to be used, the possibilities for attacks are diverse. If the developer chooses to acquire the data set from a trusted source, the integrity and authenticity of the data must be secured to prevent tampering during transmission. This can be done using cryptographic schemes.

Even if the source is deemed trustworthy, it is impossible to be sure that the data set is actually correct and has not fallen prey to attacks beforehand. In addition, the data set may be biased, and a benign but prevalent issue may be data that were unintentionally assigned wrong labels [noise in the data set may be as high as 30% (Veit et al., 2017; Wang et al., 2018)]. The main problem in terms of IT security are so-called poisoning attacks though. In a poisoning attack, the attacker manipulates the training set in order to influence the model trained on this data set. Such attacks can be divided into two categories:

1. Attacks on availability: The attacker aims to maximize the generalization error of the model (Biggio et al., 2012; Xiao et al., 2014; Mei and Zhu, 2015) by poisoning the training set. This attack can be detected in the testing phase since it decreases the model's accuracy. A more focused attack might try to degrade the accuracy only on a subset of data. For instance, images of stop signs could be targeted in traffic sign recognition. Such an attack would only affect a small fraction of the test set and thus be more difficult to detect. The metrics used for testing should hence be selected with care.
2. Attacks on integrity: The attacker aims to introduce a backdoor into the model without affecting its overall accuracy (Chen et al., 2017; Turner et al., 2019; Saha et al., 2020) (cf. **Figure 4**), which makes it very hard to detect. The attack consists in injecting a special trigger pattern into the data and assigning it to a target output. A network trained on these data will produce the target output when processing data samples containing the trigger. Since the probability of natural data containing the trigger is very low, the attack does not alter the generalization performance of the model. In classification tasks, the trigger is associated with a target class. For instance, in biometric authentication the trigger may consist in placing a special pair of sunglasses upon the eyes in images of faces. The model would then classify persons wearing these sunglasses as the target class.

2.3. Training

In this phase, the model is trained using the training data set and subject to the boundary conditions fixed before. To this end, several hyperparameters (number of repetitions, stop criteria, learning rate etc.) have to be set either automatically by the ML algorithm or manually by the developer, and the data set has to be partitioned into training and test data in a suitable way. Attacks in this phase may be mounted by attackers getting access to the training procedure, especially if training is not done locally, but using an external source, e.g., in the cloud (Gu et al., 2017). Possible threats include augmenting the training data set with poisoned data to sabotage training, changing the hyperparameters of the training algorithm or directly changing the model's parameters (weights and biases). Furthermore, an attacker may manipulate already trained models. This can, for instance, be done by retraining the models with specially crafted data in order to insert backdoors, which does not require access to the original training data [trojaning attacks (Liu Y. et al., 2018; Ji et al., 2019)]. A common feature of these attacks is that they assume a rather powerful attacker having full access to the developer's IT infrastructure. They can be mitigated using measures from traditional IT security for protecting the IT environment. Particular countermeasures include, on the one hand, integrity protection schemes for preventing unwarranted tampering with intermediate results as well as comprehensive logging and documentation of the training process. On the other hand, the reliability of staff must be checked to avoid direct attacks by or indirect attacks via the developers.

2.4. Testing and Evaluation

After training, the performance of the model is tested using the validation data set and the metrics fixed in the planning phase. If it is below the desired level, training needs to be restarted and, if necessary, the boundary conditions need to be modified. This iterative process needs to be repeated until the desired level of performance is attained (cf. **Figures 1B**, **3A**). In order to check the performance of the model, the process of evaluation needs to be repeated after every iteration of training, every time that the model goes into operation as part of a more complex IT system, and every time that side conditions change.

After finishing the training and validation phase, the test set is used for measuring the model's final performance. It is important that using the test set only yields heuristic guarantees on the generalization performance of the model, but does not give any

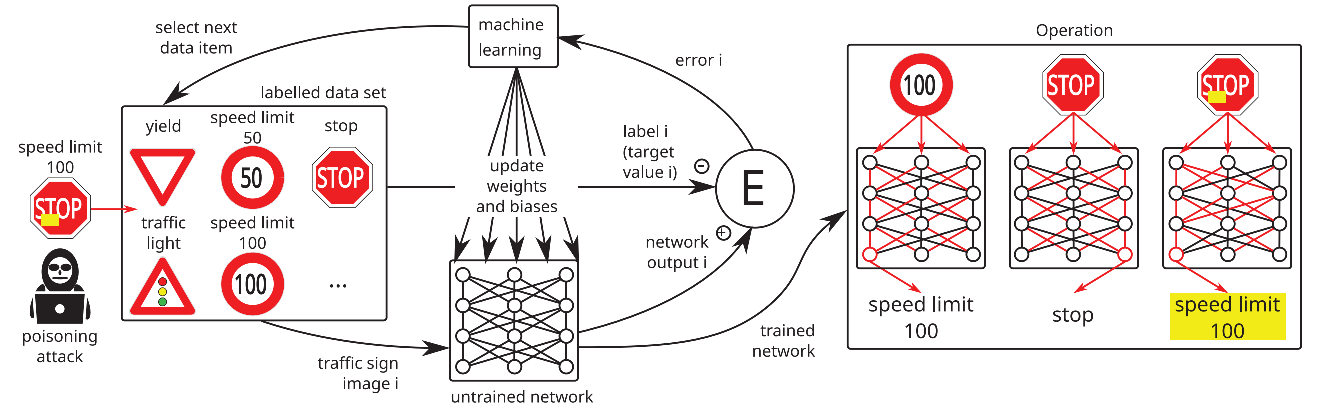


FIGURE 4 | A so-called poisoning or backdooring attack may be mounted by an attacker if he gets the chance to inject one or more manipulated data items into the training set: the manipulated data lead to undesired results but the usual training and test data still produce the desired results, making it extremely hard to detect backdoors in neural networks. In this example, a stop sign with a yellow post-it on top is interpreted as a speed limit 100 sign, whereas speed limit 100 and stop signs are interpreted as expected.

formal statements on the correctness or robustness of the model, nor does it allow understanding the decisions taken by the model if the structure of the model does not easily lend itself to human interpretation (black-box model). In particular, the model may perform well on the test set by having learnt only spurious correlations in the training data. Care must hence be taken when constructing the test set. A supplementary approach to pure performance testing is to use XAI methods (cf. subsection 4.3), which have often been used to expose problems which had gone unnoticed in extensive testing (Lapuschkin et al., 2019).

2.5. Operation

A model that has successfully completed testing and evaluation may go into operation. Usually, the model is part of a more complex IT system, and mutual dependencies between the model and other components may exist. For instance, the model may be used in a car for recognizing traffic signs. In this case, it receives input from sensors within the same IT system, and its output may in turn be used for controlling actuators. The embedded model is tested once before practical deployment or continuously via a monitoring process. If necessary, one can adjust its embedding or even start a new training process using modified boundary conditions and iterate this process until achieving the desired performance.

Classical attacks can target the system at different levels and impact the input or output of the AI model without affecting its internal operation. Attacks may be mounted on the hardware (Clements and Lao, 2018) and operating system level or concern other software executed besides the model. Such attacks are not specific to AI models and are thus not in the focus of this publication. They need to be mitigated using classical countermeasures for achieving a sufficient degree of IT security. Due to the black-box property of AI systems, however, these attacks can be harder to detect than in a classical setting.

A qualitatively new type of attacks, called evasion attacks, focuses on AI systems (cf. **Figure 5**). Evasion attacks have been well-known in adversarial ML for years (Biggio and Roli, 2018). In the context of deep learning, these attacks are called adversarial attacks. Adversarial attacks target the inference phase of a trained model and perturb the input data in order to change the output of the model in a desired way (Szegedy et al., 2014; Goodfellow et al., 2015). Depending on the attacker's knowledge, adversarial attacks can be mounted in a white-box or gray-box setting:

1. In white-box attacks, the attacker has complete information about the system, including precise knowledge of defense mechanisms designed to thwart attacks. In most cases, the attacker computes the perturbation using the gradient of the targeted model. The Fast Gradient Sign Method of Goodfellow et al. (2015) is an early example, which was later enhanced by stronger attacks designed to create the perturbation in an iterative manner (Papernot et al., 2016c; Carlini and Wagner, 2017c; Chen et al., 2018, 2020; Madry et al., 2018).
2. In gray-box attacks, the attacker does not have access to the internals of the model and might not even know the exact training set, although some general intuition about the design of the system and the type of training data needs to be present, as pointed out by Biggio and Roli (2018). In this case, the attacker trains a so-called surrogate model using data whose distribution is similar to the original training data and, if applicable, queries to the model under attack (Papernot et al., 2016b). If the training was successful, the surrogate model approximates the victim model sufficiently well to proceed to the next step. The attacker then creates an attack based on the surrogate model, which is likely to still perform well when applied to the targeted model, even if the model classes differ. This property of adversarial examples, which is very beneficial for attackers, has been termed transferability (Papernot et al., 2016a).

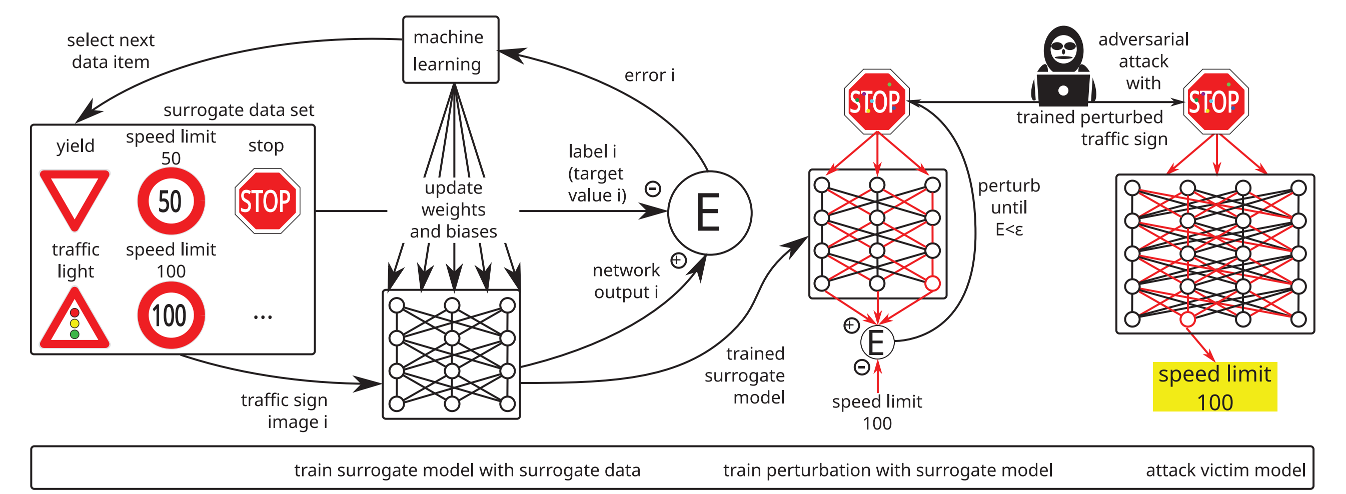


FIGURE 5 | Adversarial attacks may be conducted without white-box access to the victim model: First, a surrogate model is trained using a surrogate data set. Labels for this data set might optionally be obtained via queries to the victim model. Subsequently, the trained surrogate model is used to generate adversarial input examples. In many cases, these adversarial examples may then be used successfully for attacking the victim model.

Adversarial attacks usually choose the resulting data points to be close to the original ones in some metric, e.g., the Euclidean distance. This can make them indistinguishable from the original data points for human perception and thus impossible to detect by a human observer. However, some researchers have raised the question whether this restriction is really necessary and have argued that in many applications it may not be (Gilmer et al., 2018; Yakura et al., 2020). This applies in particular to applications where human inspection of data is highly unlikely and even blatant perturbations might well go unnoticed, as e.g., in the analysis of network traffic.

In most academic publications, creating and deploying adversarial attacks is a completely digital procedure. For situated systems acting in the sensory-motor loop, such as autonomous cars, this approach may serve as a starting point for investigating adversarial attacks but generally misses out on crucial aspects of physical instantiations of these attacks: First, it is impossible to foresee and correctly simulate all possible boundary conditions as e.g., viewing angles, sensor pollution and temperature. Second, sufficiently realistic simulations of the interaction effects between system modules and environment are hard to carry out. Third, this likewise applies to simulating individual characteristics of hardware components that influence the behavior of these components. This means the required effort for generating physical adversarial attacks that perform well is much larger as compared to their digital copies. For this reason, such attacks are less well-studied, but several publications have shown they can still work, in particular if attacks are optimized for high robustness to typically occurring transformations (e.g., rotation and translation in images) (Sharif et al., 2016; Brown et al., 2017; Evtimov et al., 2017; Eykholt et al., 2017; Athalye et al., 2018b; Song et al., 2018).

3. KEY FACTORS UNDERLYING AI-SPECIFIC VULNERABILITIES

As described in section 2, AI systems can be attacked on different levels. Whereas many of the vulnerabilities are just variants of more general problems in IT security, which affect not only AI systems, but also other IT solutions, two types of attacks are specific to AI, i.e., poisoning attacks and adversarial examples (also known as evasion attacks). This section aims to give a general intuition of the fundamental properties specific to AI which enable and facilitate these attacks, and to outline some general strategies for coping with them.

3.1. Huge Input and State Spaces and Approximate Decision Boundaries

Complex AI models contain many millions of parameters (weights and biases), which are updated during training in order to approximate a function for solving the problem at hand. As a result, the number of possible combinations of parameters is enormous and decision boundaries between input data where the models' outputs differ can only be approximate (Hornik et al., 1989; Blackmore et al., 2006; Montúfar et al., 2014) (cf. **Table 1**). Besides, due to the models' non-linearity small perturbations in input values may result in huge differences in the output (Pasemann, 2002; Goodfellow et al., 2015; Li, 2018).

TABLE 1 | The size of the input and state spaces of commonly used architectures in the field of object recognition (LeNet-5, VGG-16, ResNet-152) and natural language processing (BERT) is extremely large.

Model	Number of distinct possible inputs	Input size (in bit)	Output size (in bit)	Number of parameters	Number of layers
LeNet-5 (LeCun et al., 1998)	2^{6272}	$28 \cdot 28 \cdot 8 = 6272$	$10 \cdot 32$	≈ 60 K	7
VGG-16 (Simonyan and Zisserman, 2015)	$2^{1204224}$	$224 \cdot 224 \cdot 3 \cdot 8 = 1204224$	$1000 \cdot 32$	≈ 135 M	16
ResNet-152 (He et al., 2016)	$2^{1204224}$	$224 \cdot 224 \cdot 3 \cdot 8 = 1204224$	$1000 \cdot 32$	≈ 60 M	152
BERT (Devlin et al., 2019)	$\leq 2^{7680}$	$\leq 512 \cdot 15 = 7680$	$\leq 512 \cdot 1000 \cdot 32$	≈ 345 M	24

In general, AI models are trained on the natural distribution of the data considered in the specific problem (e.g., the distribution of traffic sign images). This distribution, however, lies on a very low-dimensional manifold as compared to the complete input space (e.g., all possible images of the same resolution) (Tanay and Griffin, 2016; Balda et al., 2020), which is sometimes referred to as the "curse of dimensionality." **Table 1** shows that the size of the input space for some common tasks is extremely large. Even rather simple and academic AI models as e.g., LeNet-5 for handwritten digit recognition have a huge input space. As a consequence, most possible inputs are never considered during training.

On the one hand, this creates a safety risk if the model is exposed to benign inputs which sufficiently differ from those seen during training, such that the model is unable to generalize to these new inputs (Novak et al., 2018; Jakubovitz et al., 2019). The probability of this happening depends on many factors, including the model, the algorithm used and especially the quality of the training data (Chung et al., 2018; Zahavy et al., 2018).

On the other hand, what is much more worrying, inputs which reliably cause malfunctioning for a model under attack, i.e., adversarial examples, can be computed efficiently and in a targeted way (Athalye et al., 2018b; Yousefzadeh and O'Leary, 2019; Chen et al., 2020). Although much work has been invested in designing defenses since adversarial examples first surfaced in deep learning, as of now, no general defense method is known which can reliably withstand adaptive attackers (Carlini and Wagner, 2017a; Athalye et al., 2018a). That is, defenses may work if information about their mode of operation is kept secret from an attacker (Song et al., 2019). As soon as an attacker gains this information, which should in most cases be considered possible following Kerckhoffs's principle, he is able to overcome them.

Besides the arms race in practical attacks and defenses, adversarial attacks have also sparked interest from a theoretical perspective (Goodfellow et al., 2015; Tanay and Griffin, 2016;

Biggio and Roli, 2018; Khoury and Hadfield-Menell, 2018; Madry et al., 2018; Ilyas et al., 2019; Balda et al., 2020). Several publications deal with their essential characteristics. As pointed out by Biggio and Roli (2018), adversarial examples commonly lie in areas of negligible probability, blind spots where the model is unsure about its predictions. Furthermore, they arise by adding highly non-random noise to legitimate samples, thus violating the implicit assumption of statistical noise that is made during training. Khoury and Hadfield-Menell (2018) relates adversarial examples to the high dimension of the input space and the curse of dimensionality, which allows constructing adversarial examples in many directions off the manifold of proper input data. In Ilyas et al. (2019), the existence of adversarial examples is ascribed to so-called non-robust features in the training data, which would also provide an explanation for their transferability property. By practical experiments (Madry et al., 2018) demonstrate defenses from the point of view of robust optimization that show comparatively high robustness against strong adversarial attacks. Additionally and in contrast to most other publications, theses defenses provide some theoretical guarantee against a whole range of both static and adaptive attacks.

Figure 6 illustrates the problem of adversarial examples and its root cause and presents an analogy from human psychophysics. Decision-making in humans (Loftus, 2005) as well as in AI systems (Jakubovitz et al., 2019) is error-prone since theoretically ideal boundaries for decision-making (task decision boundaries) are in practice instantiated by approximations (model decision boundaries). Models are trained using data (AI and humans) and evolutionary processes (humans). In the trained model, small changes in either sensory input or other boundary conditions (e.g., internal state) may lead to state changes whereby decision boundaries are crossed in state space, i.e., small changes in input (e.g., sensory noise) may lead to large output changes (here a different output class). Model and task decision, therefore, may not always match. Adversarial examples are found in those regions in input space where task and model decision boundaries differ, as depicted in **Figure 6**:

- Part A shows an example for human perception of ambiguous images, namely the so-called Necker cube: sensory input (image, viewpoint, lightening, ...), internal states (genetics, previous experience, alertness, mood, ...) and chance (e.g., sensory noise) determine in which of two possible ways the Necker cube is perceived: (top) either the square on the left/top side or the square on the right/bottom side is perceived as the front surface of the cube, and this perception may spontaneously switch from one to the other (bistability). Besides internal human states that influence which of the two perceptions is more likely to occur (Ward and Scholl, 2015), the input image may be slightly manipulated such that either the left/top square (left) or the right/bottom square (right) is perceived as the front surface of the cube.
- Part B shows how all these effects are also observed in AI systems. This figure illustrates adversarial examples for a simplified two-dimensional projection of an input space with three decision boundaries forming the model decision boundary of class A (yellow) modeling the task decision boundary (blue): small modifications can shift (red arrows) input data from one model decision class to another, with (example on boundary 2 on the left) and without (example on boundary 3 on the right) changing the task decision class. Most data are far enough from the model decision boundaries to exhibit a certain amount of robustness (example on boundary 1 on the bottom). It is important to note that this illustration, depicting a two-dimensional projection of input space, does not reflect realistic systems with high-dimensional input space. In those systems, adversarial examples may almost always be found within a small distance from the point of departure (Szegedy et al., 2014; Goodfellow et al., 2015; Khoury and Hadfield-Menell, 2018). These adversarial examples rarely occur by pure chance but attackers may efficiently search for them.

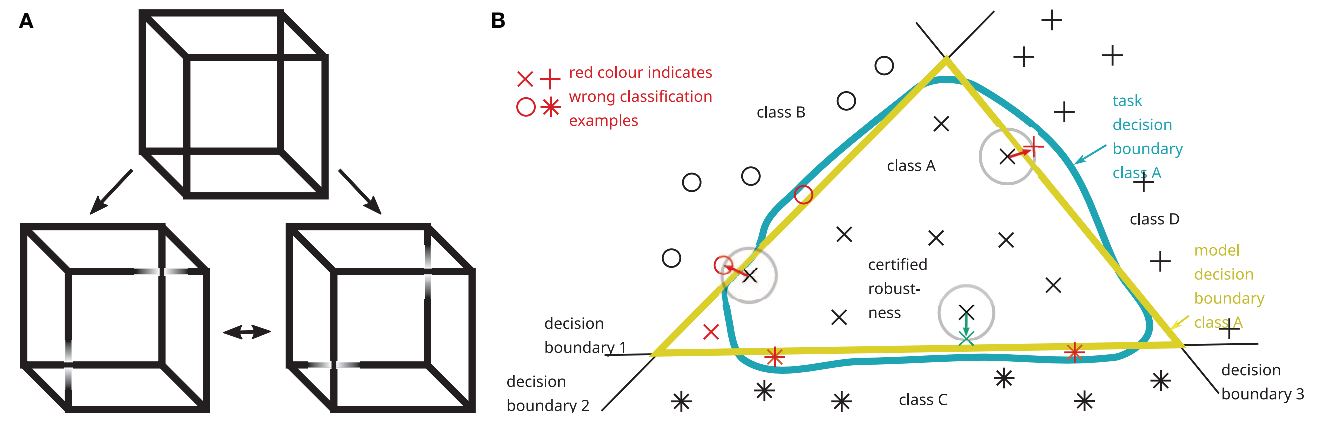


FIGURE 6 | Error-prone decision-making in humans **(A)** and AI systems **(B)** as exemplified by the Necker cube as an example of an ambiguous image **(A)** and a schematic depiction of adversarial examples in a 2D-projection of state space **(B)**. Task and model decision boundaries do not perfectly match and small changes in input may result in large changes in output. More details are given in the main text.

3.2. Black-Box Property and Lack of Interpretability

A major drawback of complex AI models like deep neural networks is their shortcoming in terms of interpretability and explainability (Rudin, 2019). Traditional computer programs solving a task are comprehensible and transparent at least to sufficiently knowledgeable programmers. Due to their huge parameter space as discussed in subsection 3.1, complex AI systems do not possess this property. In their case, a programmer can still understand the boundary conditions and the approach to the problem; however, it is infeasible for a human to directly convert the internal representation of a deep neural network to terms allowing him to understand how it operates. This is very dangerous from the perspective of IT security, since it means attacks can essentially only be detected from incorrect behavior of the model (which may in itself be hard to notice), but not by inspecting the model itself. In particular, after training is completed, the model's lack of transparency makes it very hard to detect poisoning and backdooring attacks on the training data. For this reason, such attacks should be addressed and mitigated by thorough documentation of the training and evaluation process and by protecting the integrity of intermediate results or alternatively by using training and test data that have been certified by a trustworthy party.

A straightforward solution to the black-box property of complex AI models would be to use a model which is inherently easier to interpret for a human, e.g., a decision tree or a rule list (Molnar, 2020). When considering applications based on tabular data, for instance in health care or finance, one finds that decision trees or rule lists even perform better than complex cAI models in most cases (Angelino et al., 2018; Rudin, 2019; Lundberg et al., 2020), besides exhibiting superior interpretability. However, in applications from computer vision, which are the focus of this paper, or speech recognition, sAI models cannot compete with complex models like deep neural networks, which are unfortunately very hard to interpret. For these applications, there is hence a trade-off between model interpretability and performance. A general rule of thumb for tackling the issue of interpretability would still consist in using the least complex model which is capable of solving a given problem sufficiently well. Another approach for gaining more insight into the operation of a black-box model is to use XAI methods that essentially aim to provide their users with a human-interpretable version of the model's internal representation. This is an active field of research, where many methods have been proposed in recent years (Gilpin et al., 2018; Samek et al., 2019; Molnar, 2020). Yet another approach is to use—where available—AI-systems which have been mathematically proven to be robust against attacks under the boundary conditions that apply for the specific use case (Huang et al., 2017; Katz et al., 2017; Gehr et al., 2018; Wong et al., 2018; Wong and Kolter, 2018; Singh et al., 2019). For more details, the reader is referred to subsection 4.3.

3.3. Dependence of Performance and Security on Training Data

The accuracy and robustness of an AI model is highly dependent on the quality and quantity of the training data (Zhu et al., 2016; Sun et al., 2017; Chung et al., 2018). In particular, the model can only achieve high overall performance if the training data are unbiased (Juba and Le, 2019; Kim et al., 2019). Despite their name, AI models currently used are not "intelligent," and hence they can only learn correlations from data but cannot by themselves differentiate spurious correlations from true causalities.

For economic reasons, it is quite common to outsource part of the supply chain of an AI model and obtain data and models for further training from sources which may not be trustworthy (cf. **Figure 7**). On the one hand, for lack of computational resources and professional expertise, developers of AI systems often use pre-trained networks provided by large international companies or even perform the whole training process in an environment not under their control. On the other hand, due to the efforts required in terms of funds and personnel for collecting training data from scratch as well as due to local data protection laws (e.g., the GDPR in the European Union), they often obtain whole data sets in other countries. This does not only apply to data sets containing real data, but also to data which are synthetically created (Gohorbani et al., 2019) in order to save costs. Besides synthetic data created from scratch, this especially concerns data obtained by augmenting an original data set, e.g., using transformations under which the model's output should remain invariant.

Both these facts are problematic in terms of IT security, since they carry the risk of dealing with biased or poor-quality data and of falling prey to poisoning attacks (cf. section 2), which are very hard to detect afterwards. The safest way to avoid these issues is not to rely on data or models furnished by other parties. If this is infeasible, at least a thorough documentation and cryptographic mechanisms for protecting the integrity and authenticity of such data and models should be applied throughout their whole supply chain (cf. subsection 4.2).

4. MITIGATION OF VULNERABILITIES OF AI SYSTEMS

4.1. Assessment of Attacks

A necessary condition for properly reasoning about attacks is to classify them using high-level criteria. The result of this classification will facilitate a discussion about defenses which are feasible and necessary. Such a classification is often referred to as a threat model or attacker model (Papernot et al., 2016d; Biggio and Roli, 2018).

An important criterion to consider is the **goal** of the attack. First, one needs to establish which security goal is affected. As already noted in section 1, attackers can target either integrity (by having the system make wrong predictions on specific input data), availability (by hindering legitimate users from properly using the system) or confidentiality (by extracting information without proper authorization). Besides, the scope of the attack may vary. An attacker may mount a targeted attack, which affects only certain data samples, or an indiscriminate one. In addition, the attacker may induce a specific or a general error. When considering AI classifiers, for instance, a specific error means that a sample is labeled as belonging to a target class of the

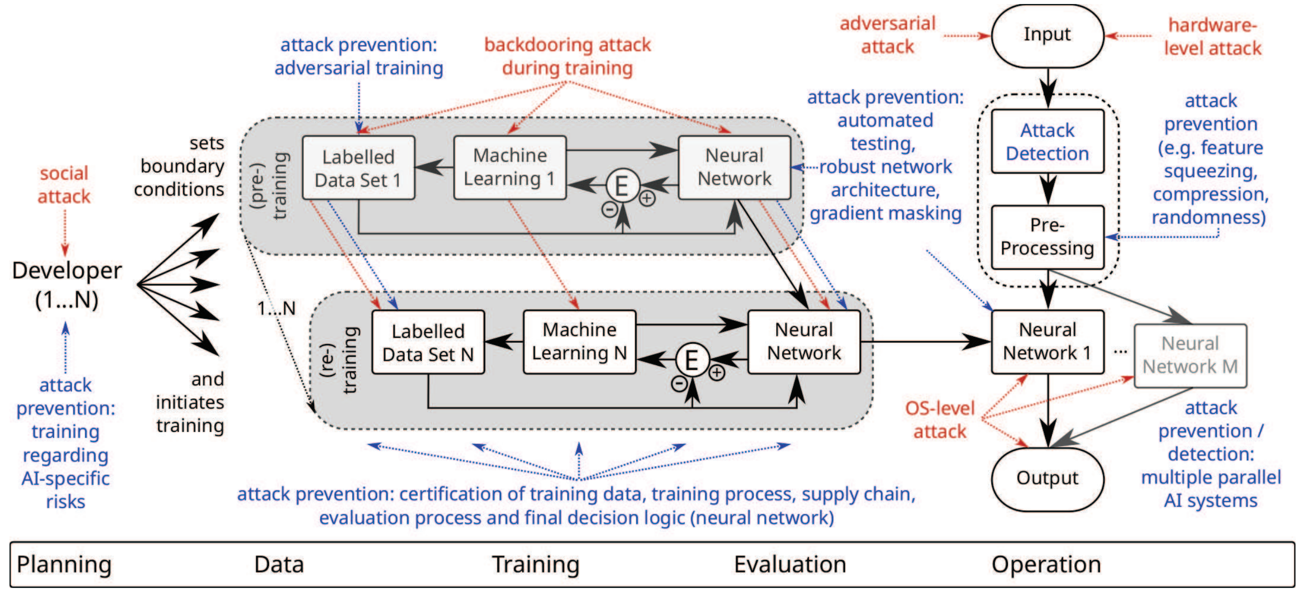

adversarial attack
Input
hardware-level attack
attack prevention: adversarial training
backdooring attack during training
attack prevention: automated testing, robust network architecture, gradient masking
Attack Detection
attack prevention (e.g. feature squeezing, compression, randomness)
social attack
sets boundary conditions
(pre-) training
Labelled Data Set 1
Machine Learning 1
Neural Network
E
Pre-Processing
Developer (1...N)
1...N
(re-) training
Labelled Data Set N
Machine Learning N
Neural Network
E
Neural Network 1
...
Neural Network M
attack prevention: training regarding AI-specific risks
and initiates training
OS-level attack
attack prevention / detection: multiple parallel AI systems
Output
attack prevention: certification of training data, training process, supply chain, evaluation process and final decision logic (neural network)
Planning
Data
Training
Evaluation
Operation

FIGURE 7 | Summary of possible attacks (red) on AI systems and defenses (blue) specific to AI systems depicted along the AI life cycle. Defenses not specific to AI systems, e.g., background checks of developers, hardware access control etc. are not shown here and should be adopted from classical IT security. Multiple AI training sessions with different data sets indicate the risk associated with pre-trained networks and externally acquired data.

attacker's choosing, whereas a general error only requires any incorrect label to be assigned to the sample. Furthermore, the ultimate objective of the attack must be considered. For example, this can be the unauthorized use of a passport (when attacking biometric authentication) or recognizing a wrong traffic sign (in autonomous driving applications). In order to properly assess the attack, it is necessary to measure its real-world impact. For lack of more precise metrics commonly agreed upon, as a first step one might resort to a general scale assessing the attack as having low, medium or high impact.

The **knowledge** needed to carry out an attack is another criterion to consider. As described in subsection 2.3, an attacker has full knowledge of the model and the data sets in the white-box case. In this scenario, the attacker is strongest, and an analysis assuming white-box access thus gives a worst-case estimate for security. As noted in Carlini et al. (2019), when performing such a white-box analysis, for the correct assessment of the vulnerabilities it is of paramount importance to use additional tests for checking whether the white-box attacks in question have been applied correctly, since mistakes in applying them have been observed many times and might yield wrong results.

In the case of a gray-box attack, conducting an analysis requires making precise assumptions on which information is assumed to be known to the attacker, and which is secret. Carlini et al. (2019) suggests that, in the same way as with cryptographic schemes, as little information as possible should be assumed to be secret when assessing the security of an AI system. For instance, the type of defense used in the system should be assumed to be known to the attacker.

The third criterion to be taken into account is the **efficiency** of the attack, which influences the capabilities and resources an attacker requires. We assume the cost of a successful attack to be the most important proxy metric from the attacker's point of view. This helps in judging whether an attack is realistic in a real-world setting. If an attacker is able to achieve his objective using a completely different attack which does not directly target the AI system and costs less, it seems highly probable a reasonable attacker will prefer this alternative (cf. the concise discussion in Gilmer et al., 2018). Possible alternatives may change over time though, and if effective defenses against them are put into place, the attacker will update his calculation and may likely turn to attack forms he originally disregarded, e.g., attacks on the AI system as discussed in this paper.

The cost of a successful attack is influenced by several factors. First, the general effort and scope of a successful attack have a direct influence. For instance, the fact whether manipulating only a few samples is sufficient for mounting a successful poisoning attack or whether many samples need to be affected can have a strong impact on the required cost, especially when taking into account additional measures for avoiding detection. Second, the degree of automation of the attack determines how much manual work and manpower is required. Third, the fact whether an attack requires physical presence or can be performed remotely is likewise important. For instance, an attack which allows only a low degree of automation and requires physical presence is much more costly to mount and especially to scale. Fourth, attacking in a real-world setting adds further complexity and might hence be more expensive than an attack in a laboratory setting, where all the side conditions are under control.

A fourth important criterion is the **availability of mitigations**, which may significantly increase the attacker's cost. However, mitigations must in turn be judged by the effort they require for the defender, their efficiency and effectiveness. In particular, non-adaptive defense mechanisms may provide a false sense of security, since an attacker who gains sufficient knowledge can bypass them by modifying his attack appropriately. This is a serious problem pointed out in many publications (cf. Athalye et al., 2018a; Gilmer et al., 2018). As a rule, defense mechanisms should therefore respect Kerckhoffs's principle and must not rely on security by obscurity.

4.2. General Measures

A lot of research has been done on how to mitigate attacks on AI systems (Bethge, 2019; Carlini et al., 2019; Madry et al., 2019). However, almost all the literature so far focuses on mitigations inside the AI systems, neglecting other possible defensive measures, and does not take into account the complete AI life cycle when assessing attacks. Furthermore, although certain defenses like some variants of adversarial training (Tramèr et al., 2018; Salman et al., 2019) can increase robustness against special threat models, there is, as of now, no general defense mechanism which is applicable against all types of attacks. A significant problem of most published defenses consists in their lack of resilience against adaptive attackers (Carlini and Wagner, 2017a,b; Athalye et al., 2018a). As already stated, the defense mechanisms used should be assumed to be public. The resistance of a defense against attackers who adapt to it is hence extremely important. In this section, we argue that a broader array of measures need to be combined for increasing security, especially if one intends to certify the safe and secure operation of an AI system, as seems necessary in high-risk applications like autonomous driving. An overview of defenses and attacks is presented in **Figure 7**.

There is no compelling reason to focus solely on defending the AI system itself without taking into account additional measures which can hamper attacks by changing side conditions. This observation does not by any means imply that defenses inside the AI system are unimportant or not necessary but instead emphasizes that they constitute a last line of defense, which should be reinforced by other mechanisms.

Legal measures are most general. They cannot by themselves prevent attacks, but may serve as a deterrent to a certain extent, if properly implemented and enforced. Legal measures may include the adoption of new laws and regulation or specifying how existing laws apply to AI applications.

Organizational measures can influence the side conditions, making them less advantageous for an attacker. For instance, in biometric authentication systems at border control, a human monitoring several systems at once and checking for unusual behavior or appearance may prevent attacks which can fool the AI system but are obvious to a human observer or can easily be detected by him if he is properly trained in advance. Restricting access to the development and training of AI systems for sensitive use cases to personnel which has undergone a background check is another example of an organizational measure. Yet another example is properly checking the identity of key holders when using a public key infrastructure (PKI) for protecting the authenticity of data.

Technical measures outside the AI system can be applied to increase IT security. The whole supply chain of collecting and preprocessing data, aggregating and transmitting data sets, pre-training models which are used as a basis for further training, and the training procedure itself can be documented and secured using classic cryptographic schemes like hash functions and digital signatures to ensure integrity and authenticity (this ultimately requires a PKI), preventing tampering in the process and allowing reproducing results and tracing back problems (Berghoff, 2020). Depending on the targeted level of security and traceability, the information covered may include all the training and test data, all AI models, all ML algorithms, a detailed logging of the development process (e.g., hyperparameters set by the developer, pseudo-random seeds, intermediate results) and comments of the developers concisely explaining and justifying each step in the development process. If the source of the data used is itself trusted, such documentation and cryptographic protection can later be validated to prove (with high probability) that no data poisoning attacks have been carried out, provided the validating party gets access to at least a sample of the original data and can check the correctness of intermediate results. As a further external technical measure, the AI system can be enhanced by using additional information from other sources. For example, in biometric authentication, biometric fakes can be detected using additional sensors (Marcel et al., 2019).

In a somewhat similar vein, the **redundant operation of multiple AI systems** running in parallel may serve to increase robustness to attacks, while at the same time increasing the robustness on benign data not seen during training. These systems can be deployed in conjunction with each other and compare and verify each other's results, thus increasing redundancy. The final result might be derived by a simple majority vote (cf. **Figure 7**). Other strategies are conceivable though. For instance, in safety-critical environments an alarm could be triggered in case the final decision is not unanimous and, if applicable, the system could be transferred to a safe fall-back state pending closer inspection. Increasing the redundancy of a technical system is a well-known approach for reducing the probability of undesired behavior, whether due to benign reasons or induced by an attacker. However, the transferability property of adversarial examples (cf. subsection 2.5, Papernot et al., 2016a) implies that attacks may continue to work even in the presence of redundancy, although their probability of success should at least slightly diminish. As a result, when using redundancy, one should aim to use conceptually different models and train them using different training sets that all stem from the data distribution representing the problem at hand, but have been sampled independently or at least exhibit only small intersections. While this does not in principle resolve the challenges posed by transferability, our intuition is that it should help to further decrease an attacker's probability of success.

4.3. AI-Specific Measures

On the AI level, several measures can likewise be combined and used in conjunction with the general countermeasures presented above. First and foremost, appropriate state-of-the-art defenses from the literature can be implemented according to their security benefits and the application scenario. One common approach for thwarting adversarial attacks is to make use of input compression (Dziugaite et al., 2016; Das et al., 2017), which removes high-frequency components from input data that are typical for adversarial examples. More prominent still is a technique called **adversarial training**, which consists in pre-computing adversarial examples using standard attack algorithms and incorporating them into the training process of the model, thus making it more robust and, in an ideal setting, immune to such attacks. State-of-the-art adversarial training methods may be identified using (Madry et al., 2018, 2019; Bethge, 2019). In general, when dealing with countermeasures against adversarial attacks, it is important to keep in mind that many proposed defenses have been broken in the past (Carlini and Wagner, 2017b; Athalye et al., 2018a), and that even the best defenses available and combinations thereof Carlini and Wagner (2017a) may not fully mitigate the problem of adversarial attacks.

In terms of **defenses against backdoor poisoning attacks** only a few promising proposals have been published in recent years (Tran et al., 2018; Chen et al., 2019; Wang et al., 2019). Their main idea lies in the creation of a method which proposes possibly malicious data samples of the training set for manual examination. Those methods use the fact that a neural network trained on such a compromised data set learns the false classification of backdoored samples as exceptions, which can be detected from the internal representation of the network. It needs to be kept in mind though that those defenses do not provide any formal guarantees and might be circumvented by an adaptive adversary.

As a first step, instead of preventing AI-specific attacks altogether, **reliably detecting** them might be a somewhat easier and hence more realistic task (Carlini and Wagner, 2017a). In case an attack is detected, the system might yield a special output corresponding to this situation, trigger an alarm and forward the apparently malicious input to another IT system or a human in the loop for further inspection. It depends on the application in question whether this approach is feasible. For instance, asking a human for feedback is incompatible by definition with fully autonomous driving at SAE level 5 (ORAD Committee, 2018).

A different approach lies in using methods from the area of **explainable AI (XAI)** to better understand the underlying reasons for the decisions which an AI system takes (cf. **Figure 8**). At the least, such methods may help to detect potential vulnerabilities and to develop more targeted defenses. One example is provided by Lapuschkin et al. (2019), which suggests a more diligent preprocessing of data for preventing the AI system from learning spurious correlations, which can easily be attacked. In principle, one can also hope that XAI methods will allow reasoning about the correctness of AI decisions under a certain range of circumstances. The field of XAI as focused on (deep) neural networks is quite young, and research has only started around 2015, although the general question of explaining decisions of AI systems dates back about 50 years (Samek et al.,

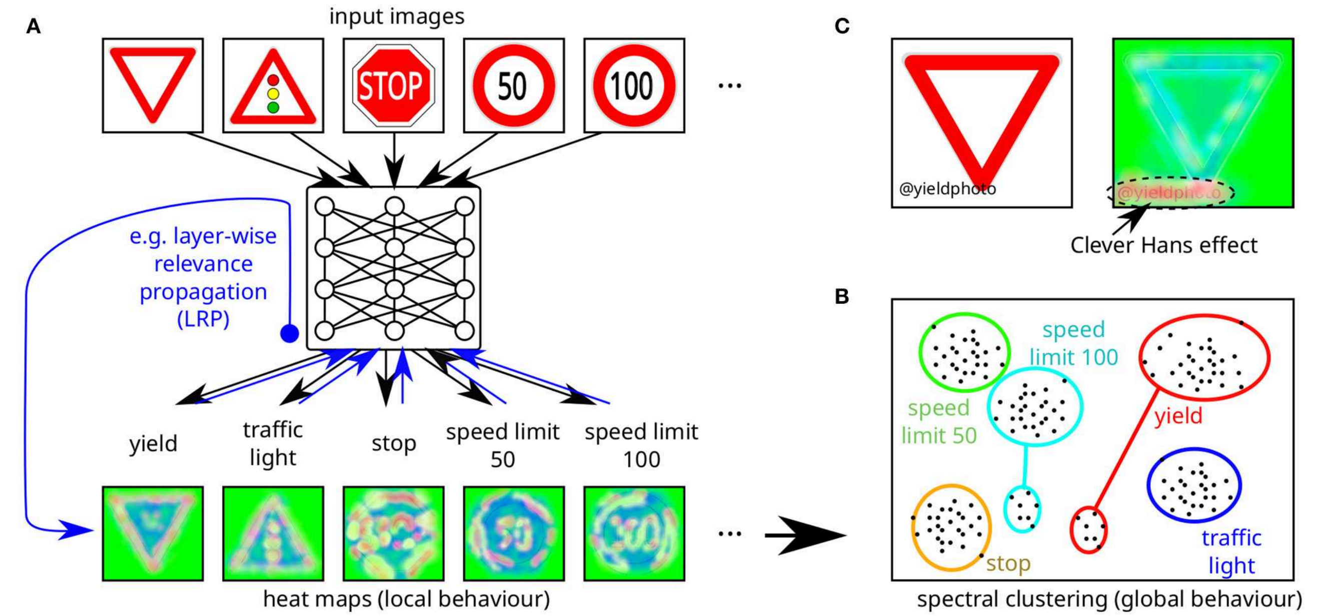


FIGURE 8 | Schematic illustration of the application of explainable AI (XAI) methods to deduce **(A)** local and **(B)** global model behavior of an AI system. **(A)** shows how heat maps are generated after labels were obtained for a specific input image, in this case using LRP (Samek et al., 2016), which assigns each input pixel a relative contribution to the output decision (green colors indicate lowest relevance, red colors highest relevance). **(B)** illustrates how many local model behavior explanations are combined to explain global model behavior, in this case using spectral analysis (cf. Lapuschkin et al., 2019). Here multiple topographically distinct clusters for individual labels shown in a 2D projection of input space indicate some kind of problem: the small cluster for *speed limit 100* represents the backdooring attack using modified stop signs (cf. **Figure 4**) and the small cluster for the yield sign represents the Clever Hans effect illustrated in detail in **(C)**, where specific image tags (here "@yieldphoto") correlate with specific input classes (here the yield sign) and the AI system focuses on these spurious correlations instead of causal correlations. Upon swapping the input data set (not containing any more spurious correlations of this kind), the AI model might show erroneous behavior.

2019, pp. 41–49). So far, it seems doubtful there will be a single method which will fit in every case. Rather, different conditions will require different approaches. On the one hand, the high-level use case has a strong impact on the applicable methods: When making predictions from structured data, probabilistic methods are considered promising (Molnar, 2020), whereas applications from computer vision rely on more advanced methods like layer-wise relevance propagation (LRP) (Bach et al., 2015; Samek et al., 2016; Montavon et al., 2017; Lapuschkin et al., 2019). On the other hand, some methods provide global explanations, while others explain individual (local) decisions. It should be noted that by using principles similar to adversarial examples, current XAI methods can themselves be efficiently attacked. Such attacks may either be performed as an enhancement to adversarial examples targeting the model (Zhang et al., 2018) or by completely altering the explanations provided while leaving model output unchanged (Dombrowski et al., 2019). Based on theoretical and practical observations, both Zhang et al. (2018) and Dombrowski et al. (2019) suggest countermeasures for thwarting the respective attacks.

A third line of research linked to both other approaches is concerned with **verifying and proving** the safety and security of AI systems. Owing to the much greater complexity of this problem, results in this area, especially practically usable ones, are scarce (Huang et al., 2017; Katz et al., 2017; Gehr et al., 2018; Wong et al., 2018; Wong and Kolter, 2018; Singh et al., 2019). A general idea for harnessing the potential of XAI and verification methods may be applied, provided one manages to make these methods work on moderately small models. In this case, it might be possible to **modularize** the AI system in question so that core functions are mapped to small AI models (Mascharka et al., 2018), which can then be checked and verified. From the perspective of data protection, this approach has the additional advantage that the use of specific data may be restricted to the training of specific modules. In contrast to monolithic models, this allows unlearning specific data by replacing the corresponding modules (Bourtoule et al., 2019).

5. CONCLUSION AND OUTLOOK

The life cycle of AI systems can give rise to malfunctions and is susceptible to targeted attacks at different levels. When facing naturally occurring circumstances and benign failures, i.e., in terms of safety, well-trained AI systems display robust performance in many cases. In practice, they may still show highly undesired behavior, as exemplified by several incidents involving Tesla cars (Wikipedia Contributors, 2020). The main problem in this respect is insufficient training data. The black-box property of the systems aggravates this issue, in particular when it comes to gaining user trust or establishing guarantees on correct behavior of the system under a range of circumstances.

The situation is much more problematic though when it comes to the robustness to attacks exhibited by the systems. Whereas a lot of attacks can be combated using traditional measures of IT security, the AI-specific vulnerabilities to poisoning and evasion attacks can have grave consequences and do not yet admit reliable mitigations. Considerable effort has been put into researching AI-specific vulnerabilities, yet more is needed, since defenses still need to become more resilient to attackers if they are to be used in safety-critical applications. In order to achieve this goal, it seems furthermore indispensable to combine defense measures at different levels and not only focus on the internals of the AI system.

Additional open questions concern the area of XAI, which is quite recent with respect to complex AI systems. The capabilities and limitations of existing methods need to be better understood, and reliable and sensible benchmarks need to be constructed to compare them (Osman et al., 2020). The topic of formal verification of the functionality of an AI system is an important enhancement that should further be studied. A general approach for obtaining better results from XAI and verification methods is to reduce complexity in the models to be analyzed. We argue that for safety-critical applications the size of AI systems used for certain tasks should be minimized subject to the desired performance. If possible, one might also envision using a modular system containing small modules, which lend themselves more easily to analysis. A thorough evaluation using suitable metrics should be considered a prerequisite for the deployment of any IT system and, therefore, of any AI system.

Thinking ahead, the issue of AI systems which are continuously being trained using fresh data (called continual learning, Parisi et al., 2019) also needs to be considered. This approach poses at least two difficulties as compared to the more static life cycle considered in this article. On the one hand, depending on how the training is done, an attacker might have a much better opportunity for poisoning training data. On the other hand, results on robustness, resilience to attacks or correctness guarantees will only be valid for a certain version of a model and may quickly become obsolete. This might be tackled by using regular checkpoints and repeating the countermeasures and evaluations, at potentially high costs.

Considering the current state of the art in the field of XAI and verification, it is unclear whether it will ever be possible to formally certify the correct operation of an arbitrary AI system and construct a system which is immune to the AI-specific attacks presented in this article. It is conceivable that both certification results and defenses will continue to only yield probabilistic guarantees on the overall robustness and correct operation of the system. If this assumption turns out true for the foreseeable future, its implications for safety-critical applications of AI systems need to be carefully considered and discussed without bias. For instance, it is important to discuss which level of residual risk, if any, one might be willing to accept in return for possible benefits of AI over traditional solutions, and in what way the conformance to a risk level might be tested and confirmed. For instance, humans are required to pass a driving test before obtaining their driver's license and being allowed to drive on their own. While a human having passed a driving test is not guaranteed to always respect the traffic rules, to behave correctly and to not cause any harm to other traffic participants, the test enforces a certain standard. In a similar vein, one might imagine a special test to be passed by an AI system for obtaining regulatory approval. In these cases the risks and benefits of using an AI system and the boundary conditions for which the risk assessment is valid should be made transparent to the user. However, the use of any IT system that cannot be guaranteed to achieve the acceptable risk level as outlined above could in extreme cases be banned for particularly safety-critical applications. Specifically, such a ban could apply to pure AI systems, if they fail to achieve such guarantees.

AUTHOR CONTRIBUTIONS

CB, MN, and AT conceived the article and surveyed relevant publications. CB wrote the original draft of the manuscript, with some help by MN. AT designed and created all the figures and tables, reviewed and edited the manuscript, with help by CB. All authors contributed to the article and approved the submitted version.

FUNDING

The article was written as part of the authors' employment at Federal Office for Information Security, Bonn, Germany. The authors did not receive any other funding.

ACKNOWLEDGMENTS

We would like to thank Ute Gebhardt, Rainer Plaga, Markus Ullmann, and Wojciech Samek for carefully proofreading earlier versions of this document and providing valuable suggestions for improvement. Further we would like to thank Frank Pasemann, Petar Tsankov, Vasilios Danos, and the VdTÜV-BSI AI work group for fruitful discussions. We would also like to thank the reviewers for their helpful comments. This manuscript has been released as a preprint at arXiv:2003.08837.

REFERENCES

Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., and Rudin, C. (2018). Learning certifiably optimal rule lists for categorical data. *J. Mach. Learn. Res.* 18, 1–78.

Athalye, A., Carlini, N., and Wagner, D. (2018a). "Obfuscated gradients give a false sense of security: circumventing Defenses to adversarial examples," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Volume 80 of Proceedings of Machine Learning Research*, eds J. G. Dy and A. Krause (Stockholm: PMLR), 274–283.

Athalye, A., Engstrom, L., Ilyas, A., and Kwok, K. (2018b). "Synthesizing robust and adversarial examples," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Volume 80 of Proceedings of Machine Learning Research*, eds J. G. Dy and A. Krause (Stockholm: PMLR), 284–293.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K. R., and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE* 10:e0130140. doi: 10.1371/journal.pone.0130140

Balda, E. R., Behboodi, A., and Mathar, R. (2020). *Adversarial Examples in Deep Neural Networks: An Overview, Volume 865 of Studies in Computational Intelligence* (Cham: Springer), 31–65.

Barreno, M., Nelson, B., Sears, R., Joseph, A. D., and Tygar, J. D. (2006). "Can machine learning be secure?" in *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2006*, eds F. C. Lin, D. T. Lee, B. S. Paul Lin, S. Shieh, and S. Jajodia (Taipei: ACM), 16–25.

Berghoff, C. (2020). Protecting the integrity of the training procedure of neural networks. *arXiv:2005.06928*.

Bethge, A. G. (2019). *Robust Vision Benchmark*. Available online at: https://robust.vision (accessed March 3, 2020).

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Srndic, N., Laskov, P., et al. (2013) "Evasion attacks against machine learning at test time," in *Machine Learning and Knowledge Discovery in Databases*, eds H. Blockeel, K. Kersting, S. Nijssen, and F. Železný (Berlin; Heidelberg: Springer), 387–402.

Biggio, B., Nelson, B., and Laskov, P. (2012). "Poisoning attacks against support vector machines," in *Proceedings of the 29th International Conference on Machine Learning (ICML)*, eds J. Langford and J. Pineau (Omnipress), 1807–1814.

Biggio, B., and Roli, F. (2018). Wild patterns: ten years after the rise of adversarial machine learning. *Pattern Recogn.* 84, 317–331. doi: 10.1016/j.patcog.2018.07.023

Blackmore, K. L., Williamson, R. C., and Mareels, I. M. Y. (2006). Decision region approximation by polynomials or neural networks. *IEEE Trans. Inform. Theory* 43, 903–907. doi: 10.1109/18.568700

Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C., Jia, H., Travers, A., Zhang, B., et al. (2019). Machine unlearning. *arXiv* abs/1912.03817.

Brown, T. B., Mané, D., Roy, A., Abadi, M., and Gilmer, J. (2017). Adversarial patch. *arXiv* abs/1712.09665.

Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., et al. (2019). On evaluating adversarial robustness. *arXiv* abs/1902.06705.

Carlini, N., and Wagner, D. (2017a). "Adversarial examples are not easily detected: bypassing ten detection methods," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security (AISec '17)*, eds B. M. Thuraisingham, B. Biggio, D. M. Freeman, B. Miller, and A. Sinha (New York, NY: Association for Computing Machinery), 3–14.

Carlini, N., and Wagner, D. (2017b). MagNet and "efficient defenses against adversarial attacks" are not robust to adversarial examples. *arXiv* abs/1711.08478.

Carlini, N., and Wagner, D. (2017c). "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy (SP)* (San Jose, CA), 39–57.

Chen, B., Carvalho, W., Baracaldo, N., Ludwig, H., Edwards, B., Lee, T., et al. (2019). "Detecting backdoor attacks on deep neural networks by activation clustering," in *Workshop on Artificial Intelligence Safety 2019 Co-located With the Thirty-Third AAAI Conference on Artificial Intelligence 2019 (AAAI-19), Volume 2301 of CEUR Workshop Proceedings*, eds H. Espinoza, S. hÉigeartaigh, X. Huang, J. Hernández-Orallo, and M. Castillo-Effen (Honolulu, HI: CEUR-WS.org).

Chen, J., Zhou, D., Yi, J., and Gu, Q. (2020). "A Frank-Wolfe framework for efficient and effective adversarial attacks," in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence 2020 (AAAI-20)* (New York, NY).

Chen, P. Y., Sharma, Y., Zhang, H., Yi, J., and Hsieh, C. J. (2018). "EAD: elastic-net attacks to deep neural networks via adversarial examples," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, eds S. A. McIlraith and K. Q. Weinberger (New Orleans, LA: AAAI Press), 10–17.

Chen, X., Liu, C., Li, B., Lu, K., and Song, D. (2017). Targeted backdoor Attacks on deep learning systems using data poisoning. *arXiv* abs/1712.05526.

Chung, Y., Haas, P. J., Upfal, E., and Kraska, T. (2018). Unknown examples & machine learning model generalization. *arXiv* abs/1808.08294.

Clements, J., and Lao, Y. (2018). Hardware trojan attacks on neural networks. *arXiv* abs/1806.05768.

Dalvi, N. N., Domingos, P. M., Mausam, Sanghai, S. K., and Verma, D. (2004). "Adversarial classification," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, eds W. Kim, R. Kohavi, J. Gehrke, and W. DuMouchel (Seattle, WA), 99–108.

Das, N., Shanbhogue, M., Chen, S. T., Hohman, F., Chen, L., Kounavis, M. E., et al. (2017). Keeping the bad guys out: protecting and vaccinating deep learning with JPEG compression. *arXiv* abs/1705.02900.

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, NAACL-HLT 2019, Volume 1 (Long and Short Papers), eds J. Burstein, C. Doran, and T. Solorio (Minneapolis, MN: Association for Computational Linguistics), 4171–4186.

Dombrowski, A. K., Alber, M., Anders, C. J., Ackermann, M., Müller, K. R., and Kessel, P. (2019). "Explanations can be manipulated and geometry is to blame," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, eds H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, (Vancouver, BC), 13567–13578.

Dziugaite, G. K., Ghahramani, Z., and Roy, D. M. (2016). A study of the effect of JPG compression on adversarial images. *arXiv* abs/1608.00853.

Eagleman, D. M. (2001). Visual illusions and neurobiology. *Nat. Rev. Neurosci.* 2, 920–926. doi: 10.1038/35104092

Evtimov, I., Eykholt, K., Fernandes, E., Kohno, T., Li, B., Prakash, A., et al. (2017). Robust physical-world attacks on machine learning models. *arXiv* abs/1707.08945.

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Song, D., Kohno, T., et al. (2017). Note on attacking object detectors with adversarial stickers. *arXiv* abs/1712.08062.

Facebook. *PyTorch*. Available online at: https://pytorch.org (accessed March 17, 2020).

Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., and Vechev, M. (2018). "AI2: safety and robustness certification of neural networks with abstract interpretation," in *IEEE Symposium on Security and Privacy (SP)* (San Francisco, CA), 3–18.

Gilmer, J., Adams, R. P., Goodfellow, I. J., Andersen, D., and Dahl, G. E. (2018). Motivating the rules of the game for adversarial example research. *arXiv* abs/1807.06732.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. (2018). "Explaining explanations: an overview of interpretability of machine learning," in *5th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2018*, eds F. Bonchi, F. J. Provost, T. Eliassi-Rad, W. Wang, C. Cattuto, and R. Ghani (Turin: IEEE), 80–89.

Gohorbani, A., Natarajan, V., Coz, D. D., and Liu, Y. (2019). "DermGAN: synthetic generation of clinical skin images with pathology," in *Proceedings of Machine Learning for Health (ML4H) at NeurIPS 2019* (Vancouver, BC).

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*. Available online at: http://arxiv.org/abs/1412.6572

Google Brain. *TensorFlow*. Available online at: https://www.tensorflow.org (accessed March 17, 2020).

Gu, T., Dolan-Gavitt, B., and Garg, S. (2017). BadNets: identifying vulnerabilities in the machine learning model supply chain. *arXiv* abs/1708.06733.

Haykin, S. (1999). *Neural Networks, 2nd Edn.* Upper Saddle River, NJ: Prentice Hall.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016* (Las Vegas, NV: IEEE Computer Society), 770–778.

Hornik, K., Stinchcombe, M. B., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Netw.* 2, 359–366.

Huang, X., Kwiatkowska, M., Wang, S., and Wu, M. (2017). "Safety verification of deep neural networks," in *Computer Aided Verification–29th International Conference, CAV 2017, Proceedings, Part I, Volume 10426 of Lecture Notes in Computer Science*, eds R. Majumdar and V. Kuncak (Heidelberg: Springer), 3–29.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. (2019). "Adversarial examples are not bugs, they are features," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, eds H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett (Vancouver, BC, Canada), 125–136.

INRIA. *Scikit-Learn*. Available online at: https://scikit-learn.org/stable/ (accessed March 17, 2020).

Jakubovitz, D., Giryes, R., and Rodrigues, M. R. D. (2019). "Generalization error in deep learning," in *Compressed Sensing and Its Applications. Applied and Numerical Harmonic Analysis*, eds H. Boche, G. Caire, R. Calderbank, G. Kutyniok, R. Mathar, and P. Petersen (Cham: Birkhäuser). doi: 10.1007/978-3-319-73074-5_5

Ji, Y., Liu, Z., Hu, X., Wang, P., and Zhang, Y. (2019). Programmable neural network trojan for pre-trained feature extractor. *arXiv* abs/1901.07766.

Juba, B., and Le, H. S. (2019). "Precision-recall versus accuracy and the role of large data sets," in *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*, Vol. 33 (Honolulu, HI).

Katz, G., Barrett, C. W., Dill, D. L., Julian, K., and Kochenderfer, M. J. (2017). "Reluplex: an efficient SMT solver for verifying deep neural networks," in *Computer Aided Verification–29th International Conference, CAV 2017, Proceedings, Part I, Volume 10426 of Lecture Notes in Computer Science*, eds R. Majumdar and V. Kuncak (Heidelberg: Springer), 97–117.

Khoury, M., and Hadfield-Menell, D. (2018). On the geometry of adversarial examples. *arXiv* abs/1811.00525.

Kim, B., Kim, H., Kim, K., Kim, S., and Kim, J. (2019). "Learning not to learn: training deep neural networks with biased data," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Long Beach, CA).

Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., and Müller, K. R. (2019). Unmasking Clever Hans predictors and assessing what machines really learn. *Nat. Commun.* 10, 1–8. doi: 10.1038/s41467-019-08987-4

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 2278–2324.

Lederberg, J. (1987). "How DENDRAL was conceived and born," in *Proceedings of the ACM Conference on History of Medical Informatics*, ed B. I. Blum (Bethesda, MD: ACM), 5–19.

Li, H. (2018). Analysis on the nonlinear dynamics of deep neural networks: topological entropy and chaos. *arXiv* abs/1804.03987.

Liu, Q., Li, P., Zhao, W., Cai, W., Yu, S., and Leung, V. C. M. (2018). A survey on security threats and defensive techniques of machine learning: a data driven view. *IEEE Access* 6, 12103–12117. doi: 10.1109/ACCESS.2018.2805680

Liu, Y., Ma, S., Aafer, Y., Lee, W. C., Zhai, J., Wang, W., et al. (2018). "Trojaning attack on neural networks," in *25th Annual Network and Distributed System Security Symposium, NDSS 2018* (San Diego, CA: The Internet Society).

Loftus, E. F. (2005). Planting misinformation in the human mind: a 30-year investigation of the malleability of memory. *Learn. Mem.* 12, 361–366. doi: 10.1101/lm.94705

Lowd, D., and Meek, C. (2005). "Adversarial learning," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, eds R. Grossman, R. J. Bayardo, and K. P. Bennett (Chicago, IL: ACM), 641–647.

Lundberg, S. M., Erion, G. G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., et al. (2020). Explainable AI for trees: from local explanations to global understanding. *Nat. Mach. Intell.* 2, 56–67. doi: 10.1038/s42256-019-0138-9

Madry, A., Athalye, A., Tsipras, D., and Engstrom, L. (2019). *RobustML*. Available online at: https://www.robust-ml.org/ (accessed March 17, 2020).

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2018). "Towards deep learning models resistant to adversarial attack," in *6th International Conference on Learning Representations* (Vancouver, BC). Available online at: http://arxiv.org/abs/1706.06083

Marcel, S., Nixon, M. S., and Fierrez, J. (Eds.). (2019). *Handbook of Biometric Anti-Spoofing: Presentation Attack Detection*. Advances in Computer Vision and Pattern Recognition (Basel: Springer International Publishing).

Mascharka, D., Tran, P., Soklaski, R., and Majumdar, A. (2018). "Transparency by design: closing the gap between performance and interpretability in visual reasoning," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018* (Salt Lake City, UT: IEEE Computer Society), 4942–4950.

McCulloch, W., and Pitts, W. (1943). A logical calculus of ideas immanent in nervous activity. *Bull. Math. Biophys.* 5, 115–133.

Mei, S., and Zhu, X. (2015). "Using machine teaching to identify optimal training-set attacks on machine learners," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, eds B. Bonet and S. Koenig (Austin, TX: AAAI Press), 2871–2877.

Molnar, C. (2020). *Interpretable Machine Learning–A Guide for Making Black Box Models Explainable*. Available online at: https://christophm.github.io/interpretable-ml-book/ (accessed March 17, 2020).

Montavon, G., Bach, S., Binder, A., Samek, W., and Müller, K. R. (2017). Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recogn.* 65, 211–222. doi: 10.1016/j.patcog.2016.11.008

Montúfar, G. F., Pascanu, R., Cho, K., and Bengio, Y. (2014). "On the number of linear regions of deep neural networks," in *NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems*, Vol. 2 (Montreal, QC), 2924–2932.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. (2018). "Sensitivity and generalization in neural networks: an empirical study," in *International Conference on Learning Representations* (Vancouver, BC).

On-Road Automated Driving (ORAD) Committee (2018). *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles J3016_201806*. Technical report, SAE International.

Osman, A., Arras, L., and Samek, W. (2020). Towards ground truth evaluation of visual explanations. *arXiv* abs/2003.07258.

Papernot, N., McDaniel, P. D., and Goodfellow, I. J. (2016a). Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv* abs/1605.07277.

Papernot, N., McDaniel, P. D., Goodfellow, I. J., Jha, S., Celik, Z. B., and Swami, A. (2016b). Practical black-box attacks against deep learning systems using adversarial examples. *arXiv* abs/1602.02697.

Papernot, N., McDaniel, P. D., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. (2016c). "The limitations of deep learning in adversarial settings," in *IEEE European Symposium on Security and Privacy, EuroS&P 2016* (Saarbrücken), 372–387.

Papernot, N., McDaniel, P. D., Sinha, A., and Wellman, M. P. (2016d). "SoK: security and privacy in machine learning," in *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018* (London: IEEE), 399–414.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: a review. *Neural Netw.* 113, 54–71. doi: 10.1016/j.neunet.2019.01.012

Pasemann, F. (2002). Complex dynamics and the structure of small neural networks. *Netw. Comput. Neural Syst.* 13, 195–216. doi: 10.1080/net.13.2.195.216

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* 1, 206–215. doi: 10.1038/s42256-019-0048-x

Saha, A., Subramanya, A., and Pirsiavash, H. (2020). "Hidden trigger backdoor attacks," in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence 2020 (AAAI-20)* (New York City, NY).

Salman, H., Li, J., Razenshteyn, I. P., Zhang, P., Zhang, H., Bubeck, S., et al. (2019). "Provably robust deep learning via adversarially trained smoothed classifiers," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, eds H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett (Vancouver, BC), 11289–11300.

Samek, W., Montavon, G., Binder, A., Lapuschkin, S., and Müller, K. R. (2016). Interpreting the predictions of complex ML models by layer-wise relevance propagation. *arXiv* abs/1611.08191.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L. K., and Müller, K. R. (eds.). (2019). *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Cham: Springer.

Sharif, M., Bhagavatula, S., Bauer, L., and Reiter, M. K. (2016). "Accessorize to a crime," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, eds E. R. Weippl, S. Katzenbeisser, C. Kruegel, A. C. Myers, and S. Halevi (Vienna: ACM), 1528–1540.

Simonyan, K., and Zisserman, A. (2015). "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations*, eds Y. Bengio and Y. LeCun (San Diego). Available online at: http://arxiv.org/abs/1409.1556

Singh, G., Gehr, T., Püschel, M., and Vechev, M. (2019). "An abstract domain for certifying neural networks," in *Proceedings of the ACM Symposium on Principles of Programming Languages 2019*, Vol. 3 (Cascais), 1–30.

Song, D., Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., et al. (2018). "Physical adversarial examples for object detectors," in *12th USENIX Workshop on Offensive Technologies, WOOT 2018*, eds C. Rossow and Y. Younan (Baltimore, MD: USENIX Association).

Song, Q., Yan, Z., and Tan, R. (2019). Moving target defense for deep visual sensing against adversarial examples. *arXiv* abs/1905.13148.

Stanford Vision Lab. (2016). *ImageNet*. available online at: http://image-net.org/index (accessed March 17, 2020).

Sun, C., Shrivastava, A., Singh, S., and Gupta, A. (2017). "Revisiting unreasonable effectiveness of data in deep learning era," in *IEEE International Conference on Computer Vision, ICCV 2017* (Venice: IEEE Computer Society), 843–852.

Sun, Q., Zhang, H., Zhang, J., and Zhang, X. (2018). Why can't we accurately predict others' decisions? Prediction discrepancy in risky decision-making. *Front. Psychol.* 9:2190. doi: 10.3389/fpsyg.2018.02190

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., et al. (2014). "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014, Conference Track Proceedings*, eds Y. Bengio and Y. LeCun (Banff, AB).

Tanay, T., and Griffin, L. D. (2016). A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv* abs/1608.07690.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. (2018). "Ensemble adversarial training: attacks and defenses," in *Proceedings of the 6th International Conference on Learning Representations* (Vancouver). Available online at: https://arxiv.org/abs/1705.07204

Tran, B., Li, J., and Madry, A. (2018). "Spectral signatures in backdoor attacks," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, eds S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Montréal, QC), 8011–8021.

Turner, A., Tsipras, D., and Madry, A. (2019). Label-consistent backdoor attacks. *arXiv* abs/1912.02771.

Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., and Belongie, S. J. (2017). "Learning from noisy large-scale datasets with minimal supervision," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* (Honolulu, HI: IEEE Computer Society), 6575–6583.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., et al. (2019). "Neural cleanse: identifying and mitigating backdoor attacks in neural networks," in *Proceedings of the IEEE Symposium on Security and Privacy (SP)* (San Francisco, CA), 707–723.

Wang, F., Chen, L., Li, C., Huang, S., Chen, Y., Qian, C., et al. (2018). "The devil of face recognition is in the noise," in *Computer Vision–ECCV 2018*, eds V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss (Cham: Springer International Publishing), 780–795.

Ward, E., and Scholl, B. (2015). Stochastic or systematic? Seemingly random perceptual switching in bistable events triggered by transient unconscious cues. *J. Exp. Psychol. Hum. Percept. Perform.* 41, 929–939. doi: 10.1037/a0038709

Werbos, P. (1982). "Applications of advances in nonlinear sensitivity analysis," in *System Modeling and Optimization. Lecture Notes in Control and Information Sciences*, Vol. 38, eds R. F. Drenick and F. Kozin (Berlin; Heidelberg; New York, NY: Springer), 762–770.

Wikipedia Contributors (2020). *Tesla Autopilot—Wikipedia, The Free Encyclopedia*. San Francisco, CA: Wikimedia Foundation, Inc.

Wong, E., and Kolter, J. Z. (2018). "Provable defenses against adversarial examples via the convex outer adversarial polytope," in *Proceedings of the 35th International Conference on Machine Learning, PMLR* (Stockholm), 5286–5295.

Wong, E., Schmidt, F. R., Metzen, J. H., and Kolter, J. Z. (2018). "Scaling provable adversarial defenses," in *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, QC), 8410–8419.

Wood, G., Vine, S., and Wilson, M. (2013). The impact of visual illusions on perception, action planning, and motor performance. *Atten. Percept. Psychophys.* 75, 830–834. doi: 10.3758/s13414-013-0489-y

Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., and Roli, F. (2014). Support vector machines under adversarial label contamination . *J. Neurocomput. Spec. Issue Adv. Learn. Label Noise* 160, 53–62. doi: 10.1016/j.neucom.2014.08.081

Xu, H., Ma, Y., Liu, H. C., Deb, D., Liu, H., Tang, J. L., et al. (2020). Adversarial attacks and defenses in images, graphs and text: a review. *Int. J. Autom. Comput.* 17, 151–178. doi: 10.1007/s11633-019-1211-x

Yakura, H., Akimoto, Y., and Sakuma, J. (2020). "Generate (non-software) bugs to fool classifiers," in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence 2020 (AAAI-20)* (New York, NY).

Yousefzadeh, R., and O'Leary, D. P. (2019). Investigating decision boundaries of trained neural networks. *arXiv* abs/1908.02802.

Zahavy, T., Kang, B., Sivak, A., Feng, J., Xu, H., and Mannor, S. (2018). "Ensemble robustness and generalization of stochastic deep learning algorithms," in *International Conference on Learning Representations Workshop (ICLRW'18)* (Vancouver, BC).

Zhang, X., Wang, N., Ji, S., Shen, H., and Wang, T. (2018). Interpretable deep learning under fire. *arXiv* abs/1812.00891.

Zhu, X., Vondrick, C., Fowlkes, C. C., and Ramanan, D. (2016). Do we need more training data? *Int. J. Comput. Vis.* 119, 76–92. doi: 10.1007/s11263-015-0812-2

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.

frontiers in Big Data

SYSTEMATIC REVIEW
published: 12 November 2020
doi: 10.3389/fdata.2020.587139

Securing Machine Learning in the Cloud: A Systematic Review of Cloud Machine Learning Security

Adnan Qayyum[1]*, *Aneeqa Ijaz*[2], *Muhammad Usama*[1], *Waleed Iqbal*[3], *Junaid Qadir*[1], *Yehia Elkhatib*[4] *and Ala Al-Fuqaha*[5]

[1]Information Technology University (ITU), Lahore, Pakistan, [2]AI4Networks Research Center, University of Oklahoma, Norman, OK, United States, [3]Social Data Science (SDS) Lab, Queen Mary University of London, London, United Kingdom, [4]School of Computing and Communications, Lancaster University, Lancaster, United Kingdom, [5]Hamad Bin Khalifa University (HBKU), Doha, Qatar

With the advances in machine learning (ML) and deep learning (DL) techniques, and the potency of cloud computing in offering services efficiently and cost-effectively, Machine Learning as a Service (MLaaS) cloud platforms have become popular. In addition, there is increasing adoption of third-party cloud services for outsourcing training of DL models, which requires substantial costly computational resources (e.g., high-performance graphics processing units (GPUs)). Such widespread usage of cloud-hosted ML/DL services opens a wide range of attack surfaces for adversaries to exploit the ML/DL system to achieve malicious goals. In this article, we conduct a systematic evaluation of literature of cloud-hosted ML/DL models along both the important dimensions—*attacks* and *defenses*—related to their security. Our systematic review identified a total of 31 related articles out of which 19 focused on attack, six focused on defense, and six focused on both attack and defense. Our evaluation reveals that there is an increasing interest from the research community on the perspective of attacking and defending different attacks on Machine Learning as a Service platforms. In addition, we identify the limitations and pitfalls of the analyzed articles and highlight open research issues that require further investigation.

Keywords: Machine Learning as a Service, cloud-hosted machine learning models, machine learning security, cloud machine learning security, systematic review, attacks, defenses

OPEN ACCESS

Edited by:
Bhavya Kailkhura,
United States Department of Energy (DOE), United States

Reviewed by:
Giovanni Apruzzese,
University of Liechtenstein, Liechtenstein
Cheng Chen,
The University of Utah, United States

****Correspondence:***
Adnan Qayyum
adnan.qayyum@itu.edu.pk

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Big Data

Received: *24 July 2020*
Accepted: *08 October 2020*
Published: *12 November 2020*

Citation:
Qayyum A, Ijaz A, Usama M, Iqbal W, Qadir J, Elkhatib Y, Al-Fuqaha A (2020) Securing Machine Learning in the Cloud: A Systematic Review of Cloud Machine Learning Security. Front. Big Data 3:587139. doi: 10.3389/fdata.2020.587139

1 INTRODUCTION

In recent years, machine learning (ML) techniques have been successfully applied to a wide range of applications, significantly outperforming previous state-of-the-art methods in various domains: for example, image classification, face recognition, and object detection. These ML techniques—in particular deep learning (DL)–based ML techniques—are resource intensive and require a large amount of training data to accomplish a specific task with good performance. Training DL models on large-scale datasets is usually performed using high-performance graphics processing units (GPUs) and tensor processing units. However, keeping in mind the cost of GPUs/Tensor Processing Units and the fact that small businesses and individuals cannot afford such computational resources, the training of deep models is typically outsourced to clouds, which is referred to in the literature as *"Machine Learning as a Service"* (MLaaS).

MLaaS refers to different ML services that are offered as a component of a cloud computing services, for example, predictive analytics, face recognition, natural language services, and data modeling APIs. MLaaS allows users to upload their data and model for training at the cloud. In addition to training, cloud-hosted ML services can also be used for inference purposes, that is, models can be deployed on the cloud environments; the system architecture of a typical MLaaS is shown in **Figure 1**.

MLaaS[1] can help reduce the entry barrier to the use of ML and DL through access to managed services of wide hardware heterogeneity and incredible horizontal scale. MLaaS is currently provided by several major organizations such as Google, Microsoft, and Amazon. For example, Google offers Cloud ML Engine[2] that allows developers and data scientists to upload training data and model which is trained on the cloud in the *Tensorflow*[3,] environment. Similarly, Microsoft offers Azure Batch AI[4,]—a cloud-based service for training DL models using different frameworks supported by both Linux and Windows operating systems and Amazon offers a cloud service named Deep Learning AMI (DLAMI)[5] that provides several pre-built DL frameworks (e.g., MXNet, Caffe, Theano, and Tensorflow) that are available in Amazon's EC2 cloud computing infrastructure. Such cloud services are popular among researchers as evidenced by the price lifting of Amazon's p2.16x large instance to the maximum possible—two days before the deadline of NeurIPS 2017 (the largest research venue on ML)—indicating that a large number of users request to reserve instances.

In addition to MLaaS services that allow users to upload their model and data for training on the cloud, *transfer learning* is another strategy to reduce computational cost in which a pretrained model is fine-tuned for a new task (using a new dataset). Transfer learning is widely applied for image recognition tasks using a convolutional neural network (CNN). A CNN model learns and encodes features like edges and other patterns. The learned weights and convolutional filters are useful for image recognition tasks in other domains and state-of-the-art results can be obtained with a minimal amount of training even on a single GPU. Moreover, various popular pretrained models such as AlexNet (Krizhevsky et al., 2012), VGG (Simonyan and Zisserman, 2015), and Inception (Szegedy et al., 2016) are available for download and fine-tuning online. Both of the aforementioned outsourcing strategies come with new security concerns. In addition, the literature suggests that different types of attacks can be realized on different components of the communication network as well (Usama et al., 2020a), for example, intrusion detection (Han et al., 2020; Usama et al., 2020b), network traffic classification (Usama et al., 2019), and malware detection systems (Chen et al., 2018). Moreover, adversarial ML attacks have also been devised for client-side ML classifiers, that is, Google's phishing pages filter (Liang et al., 2016).

Contributions of the article: In this article, we analyze the security of MLaaS and other cloud-hosted ML/DL models and provide a systematic review of associated security challenges and solutions. To the best of our knowledge, this article is the first effort on providing a systematic review of the security of cloud-hosted ML models and services. The following are the major contributions of this article:

(1) We conducted a systematic evaluation of 31 articles related to MLaaS attacks and defenses.
(2) We investigated five themes of approaches aiming to attack MLaaS and cloud-hosted ML services.
(3) We examined five themes of defense methods for securing MLaaS and cloud-hosted ML services.
(4) We identified the pitfalls and limitations of the examined articles. Finally, we have highlighted open research issues that require further investigation.

Organization of the article: The rest of the article is organized as follows. The methodology adopted for the systematic review is presented in **Section 2**. The results of the systematic review are presented in **Section 3**. **Section 4** presents various security challenges associated with cloud-hosted ML models and potential solutions for securing cloud-hosted ML models are presented in **Section 5**. The pitfalls and limitations of the reviewed approaches are discussed in **Section 6**. We briefly reflect on our methodology to identify any threats to the validity in **Section 8** and various open research issues that require further investigation are highlighted in **Section 7**. Finally, we conclude the article in **Section 9**.

2 REVIEW METHODOLOGY

In this section, we present the research objectives and the adopted methodology for the systematic review. The purpose of this article is to identify and systematically review the state-of-the art research related to the security of the cloud-based ML/DL techniques. The methodology followed for this study is depicted in **Figure 2**.

2.1 Research Objectives

The following are the key objectives of this article.

O1: To build upon the existing work around the security of cloud-based ML/DL methods and present a broad overview of the existing state-of-the-art literature related to MLaaS and cloud-hosted ML services.

O2: To identify and present a taxonomy of different attack and defense strategies for cloud-hosted ML/DL models.

O3: To identify the pitfalls and limitations of the existing approaches in terms of research challenges and opportunities.

2.2 Research Questions

To achieve our objectives, we consider answering two important questions that are described below and conducted a systematic analysis of 31 articles.

[1] We use MLaaS to cover both ML and DL as a Service cloud provisions.
[2] https://cloud.google.com/ml-engine/.
[3] A popular Python library for DL.
[4] https://azure.microsoft.com/en-us/services/machine-learning-service/.
[5] https://docs.aws.amazon.com/dlami/latest/devguide/AML2_0.html.

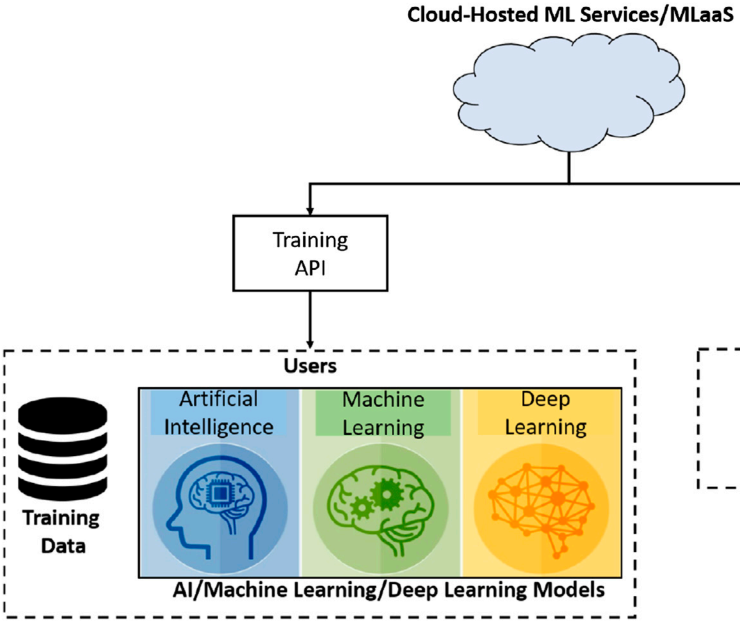


FIGURE 1 | Taxonomy of different defenses proposed for defending attacks on the third-party cloud-hosted machine learning (ML) or deep learning (DL) models.

Q1: What are the well-known attacks on cloud-hosted/third-party ML/DL models?

Q2: What are the countermeasures and defenses against such attacks?

2.3 Review Protocol

We developed a review protocol to conduct the systematic review; the details are described below.

2.3.1 Search Strategy and Searching Phase

To build a knowledge base and extract the relevant articles, eight major publishers and online repositories were queried that include ACM Digital Library, IEEE Xplore, ScienceDirect, international conference on machine learning, international conference on learning representations, journal of machine learning research, neural information processing systems, USENIX, and arXiv. As we added non-peer–reviewed articles from electric preprint archive (arXiv), we (AQ and AI) performed the critical appraisal using AACODS checklist; it is designed to enable evaluation and appraisal of gray literature (Tyndall, 2010), which is designed for the critical evaluation of gray literature.

In the initial phase, we queried main libraries using a set of different search terms that evolved using an iterative process to maximize the number of relevant articles. To achieve optimal sensitivity, we used a combination of words: attack, poisoning, Trojan attack, contamination, model inversion, evasion, backdoor, model stealing, black box, ML, neural networks, MLaaS, cloud computing, outsource, third party, secure, robust, and defense. The combinations of search keywords used are depicted in **Figure 3**. We then created search strategies with controlled or index terms given in **Figure 3**. Please note that no lower limit for the publication date was applied; the last search date was June 2020. The researchers (WI and AI) searched additional articles through citations and by snowballing on Google Scholar. Any disagreement was adjudicated by the third reviewer (AQ). Finally, articles focusing on the attack/defense for cloud-based ML models were retrieved.

2.3.2 Inclusion and Exclusion Criteria

The inclusion and exclusion criteria followed for this systematic review are defined below.

2.3.2.1 Inclusion Criteria

The following are the key points that we considered for screening retrieved articles as relevant for conducting a systematic review.

- We included all articles relevant to the research questions and published in the English language that discusses the attacks on cloud-based ML services, for example, offered by cloud computing service providers.
- We then assessed the eligibility of the relevant articles by identifying whether they discussed either attack or defense for cloud-based ML/DL models.
- Comparative studies that compare the attacks and robustness against different well-known attacks on cloud-hosted ML services (poisoning attacks, black box attacks, Trojan attacks, backdoor attacks, contamination attacks, inversion, stealing, and invasion attacks).
- Finally, we categorized the selected articles into three categories, that is, articles on attacks, articles on defenses, and articles on attacks and defenses.

2.3.2.2 Exclusion Criteria

The exclusion criteria are outlined below.

- Articles that are written in a language other than English.

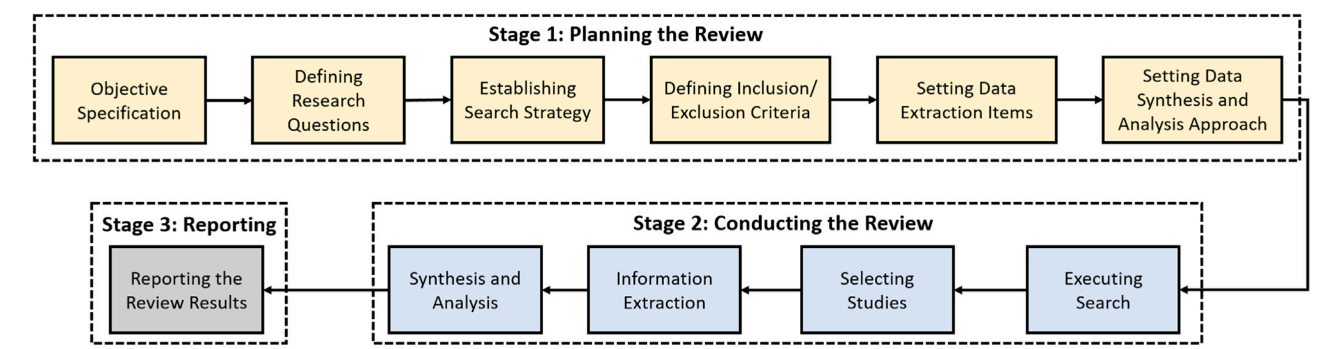

Stage 1: Planning the Review
Objective Specification
Defining Research Questions
Establishing Search Strategy
Defining Inclusion/ Exclusion Criteria
Setting Data Extraction Items
Setting Data Synthesis and Analysis Approach
Stage 2: Conducting the Review
Executing Search
Selecting Studies
Information Extraction
Synthesis and Analysis
Stage 3: Reporting
Reporting the Review Results

FIGURE 2 | An illustration of a typical cloud-based ML or machine learning as a service (MLaaS) architecture.

- Articles not available in full text.
- Secondary studies (e.g., systematic literature reviews, surveys, editorials, and abstracts or short papers) are not included.
- Articles that do not discuss attacks and defenses for cloud-based/third-party ML services, that is, we only consider those articles which have proposed an attack or defense for a cloud-hosted ML or MLaaS service.

2.3.3 Screening Phase

For the screening of articles, we employ two phases based on the content of the retrieved articles: 1) title and abstract screening and 2) full text of the publication. Please note that to avoid bias and to ensure that the judgment about the relevancy of articles is entirely based on the content of the publications, we intentionally do not consider authors, publication type (e.g., conference and journal), and publisher (e.g., IEEE and ACM). Titles and abstracts might not be true reflectors of the articles' contents; however, we concluded that our review protocol is sufficient to avoid provenance-based bias.

It is very common that the same work got published in multiple venues, for example, conference papers are usually extended to journals. In such cases, we only consider the original article. In the screening phase, every article was screened by at least two authors of this article that were tasked to annotate the articles as either relevant, not relevant, or need further investigation, which was finalized by the discussion between the authors until any such article is either marked relevant or not relevant. Only original technical articles are selected, while survey and review articles are ignored. Finally, all selected publications were thoroughly read by the authors for categorization and thematic analysis.

3 REVIEW RESULTS

3.1 Overview of the Search and Selection Process Outcome

The search using the aforementioned strategy identified a total of 4,384 articles. After removing duplicate articles, title, and abstract screening, the overall number of articles reduced to 384. A total of 230 articles did not meet the inclusion criteria and were therefore excluded. From the remaining 154 articles, 123 articles did not discuss attack/defense for third-party cloud-hosted ML models and were excluded as well. Of the remaining articles, a total of 31 articles are identified as relevant. Reasons for excluding articles were documented and reported in a PRISMA flow diagram, depicted in **Figure 4**. These articles were categorized into three classes, that is, articles that are specifically focused on attacks, articles that are specifically focused on defenses, and articles that considered both attacks and defenses containing 19, 6, and 6 articles each, respectively.

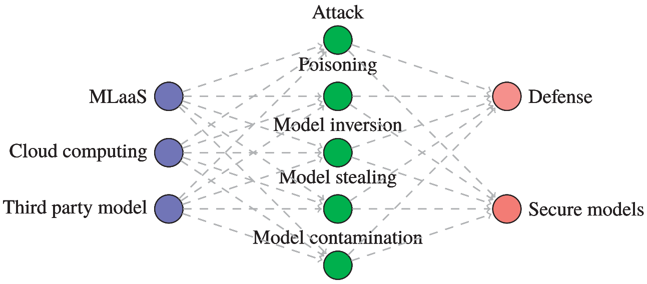


FIGURE 3 | The methodology for systematic review.

3.2 Overview of the Selected Studies

The systematic review eventually identified a set of 31 articles related to cloud-based ML/DL models and MLaaS, which we categorized into three classes as mentioned above and shown in **Figure 4**. As shown in **Figure 5**, a significant portion of the selected articles were published in conferences (41.94%); comparatively, a very smaller proportion of these articles were published in journals or transactions (19.35%). The percentage of gray literature (i.e., non-peer–reviewed articles) is 25.81%. Yet, a very small proportion of publications are published in symposia (6.45%), and this percentage is the same for workshop papers. The distribution of selected

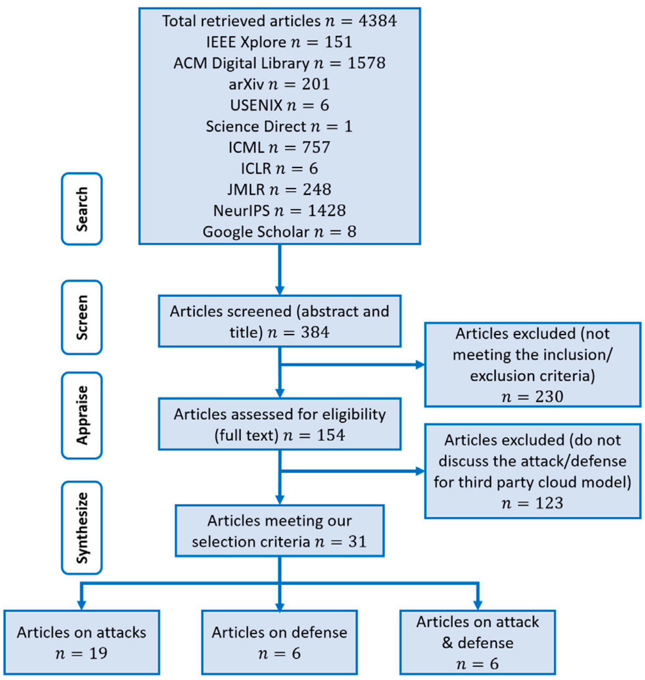

Search
Screen
Appraise
Synthesize
Total retrieved articles $n = 4384$
IEEE Xplore $n = 151$
ACM Digital Library $n = 1578$
arXiv $n = 201$
USENIX $n = 6$
Science Direct $n = 1$
ICML $n = 757$
ICLR $n = 6$
JMLR $n = 248$
NeurIPS $n = 1428$
Google Scholar $n = 8$
Articles screened (abstract and title) $n = 384$
Articles excluded (not meeting the inclusion/exclusion criteria) $n = 230$
Articles assessed for eligibility (full text) $n = 154$
Articles excluded (do not discuss the attack/defense for third party cloud model) $n = 123$
Articles meeting our selection criteria $n = 31$
Articles on attacks $n = 19$
Articles on defense $n = 6$
Articles on attack & defense $n = 6$

FIGURE 4 | Search queries used to identify publications to include in the systematic review.

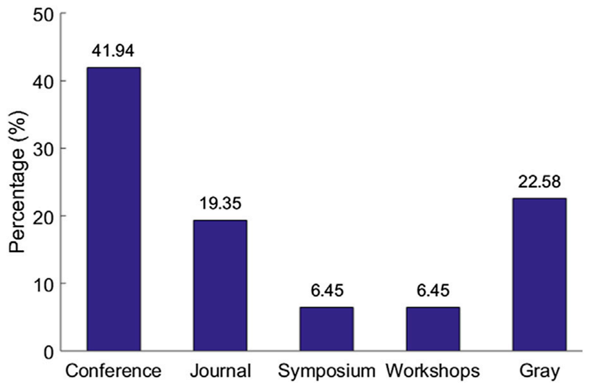


FIGURE 5 | Flowchart of systematic review and categorization.

publications by their types over the years is shown in **Figure 6**. The figure depicts that the interest in the security of cloud-hosted ML/DL models increased in the year 2017 and was at a peak in the year 2018 and was slightly lower in the year 2019 as compared to 2018. Also, the majority of the articles during these years were published in conferences. The distribution of selected publications by their publishers over the years is depicted in **Figure 7**, the figure shows that the majority of the publications have been published at IEEE, ACM, and arXiv. There is a similar trend in the number of articles in the year 2017, 2018, and 2019 as discussed previously.

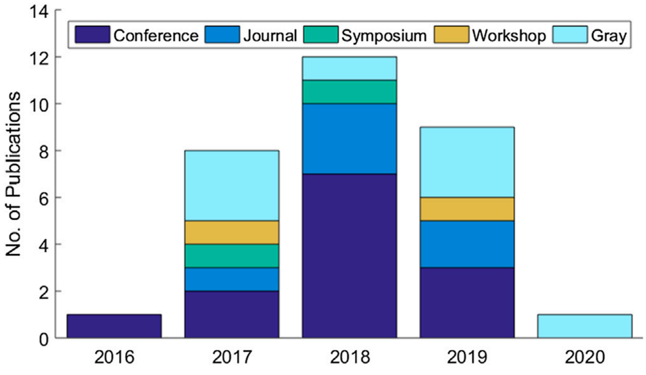


FIGURE 6 | Distribution of selected publications according to their types.

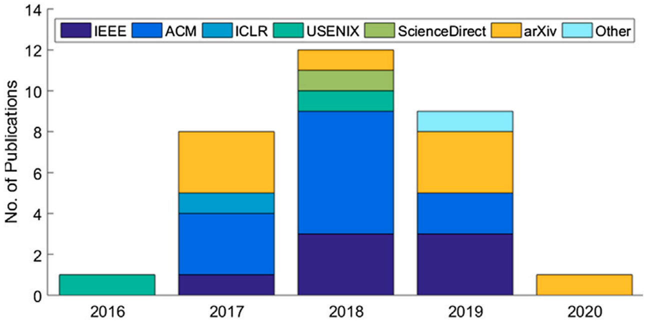


FIGURE 7 | Distribution of selected publications by types over years.

3.3 Some Partially Related Non-Selected Studies: A Discussion

We have described our inclusion and exclusion criteria that help us to identify relevant articles. We note, however, that some seemingly relevant articles failed to meet the inclusion criteria. Here, we briefly describe few such articles for giving a rationale why they were not included.

- Liang et al. (2016) investigated the security challenges for the client-side classifiers via a case study on the Google's phishing pages filter, a very widely used classifier for automatically detecting unknown phishing pages. They devised an attack that is not relevant to the cloud-based service.
- Demetrio et al. (2020) presented WAF-A-MoLE, a tool that models the presence of an adversary. This tool leverages a set of mutation operators that alter the syntax of a payload without affecting the original semantics. Using the results, the authors demonstrated that ML-based WAFs are exposed to a concrete risk of being bypassed. However, this attack is not associated with any cloud-based services.
- Authors in Apruzzese et al. (2019) discussed adversarial attacks where the machine learning model is compromised to induce an output favorable to the attacker. These attacks

are realized in a different setting as compared to the scope of this systematic review, as we only included the articles which discuss the attack or defense when the cloud is outsourcing its services as MLaaS.

- Han et al. (2020) conducted the first systematic study of the practical traffic space evasion attack on learning-based network intrusion detection systems; again it is out of the inclusion criteria of our work.
- Chen et al. (2018) designed and evaluated three types of attackers targeting the training phases to poison our detection. To address this threat, the authors proposed the detection system, KuafuDet, and showed it significantly reduces false negatives and boosts the detection accuracy.
- Song et al. (2020) presented a federated defense approach for mitigating the effect of adversarial perturbations in a federated learning environment. This article can be potentially relevant for our study as they address the problem of defending cloud-hosted ML models; however, instead of using a third-party service, the authors conducted the experiments on a single computer system in a simulated environment; therefore, this study is not included in the analysis of this article.
- In a similar study, Zhang et al. (2019) presented a defense mechanism for defending adversarial attacks on cloud-aided automatic speech recognition (ASR); however, it is not explicitly stated that the cloud is outsourcing ML services and also which ML/DL model or MLaaS was used in experiments.

4 ATTACKS ON CLOUD-HOSTED MACHINE LEARNING MODELS (Q1)

In this section, we present the findings from the systematically selected articles that aim at attacking cloud-hosted/third-party ML/DL models.

4.1 Attacks on Cloud-Hosted Machine Learning Models: Thematic Analysis

In ML practice, it is very common to outsource the training of ML/DL models to third-party services that provide high computational resources on the cloud. Such services enable ML practitioners to upload their models along with training data which is then trained on the cloud. Although such services have clear benefits for reducing the training and inference time; however, these services can easily be compromised and to this end, different types of attacks against these services have been proposed in the literature. In this section, we present the thematic analysis of 19 articles that are focused on attacking cloud-hosted ML/DL models. These articles are classified into five major themes: 1) attack type, 2) threat model, 3) attack method, 4) target model(s), and 5) dataset.

Attack type: A wide variety of attacks have been proposed in the literature. These are listed below with their descriptions provided in the next section.

- Adversarial attacks (Brendel et al., 2017);
- Backdoor attacks[6] (Chen et al., 2017; Gu et al., 2019);
- Cyber kill chain–based attack (Nguyen, 2017);
- Data manipulation attacks (Liao et al., 2018);
- Evasion attacks (Hitaj et al., 2019);
- Exploration attacks (Sethi and Kantardzic, 2018);
- Model extraction attacks (Correia-Silva et al., 2018; Kesarwani et al., 2018; Joshi and Tammana, 2019; Reith et al., 2019);
- Model inversion attacks (Yang et al., 2019);
- Model-reuse attacks (Ji et al., 2018);
- Trojan attacks (Liu et al., 2018).

Threat model: Cloud ML attacks are based on different threat models, with the salient types with examples are listed below.

- black box attacks (no knowledge) (Brendel et al., 2017; Chen et al., 2017; Hosseini et al., 2017; Correia-Silva et al., 2018; Sethi and Kantardzic, 2018; Hitaj et al., 2019);
- white box attacks (full knowledge) (Liao et al., 2018; Liu et al., 2018; Gu et al., 2019; Reith et al., 2019);
- gray box attacks (partial knowledge) (Ji et al., 2018; Kesarwani et al., 2018).

Attack method: In each article, a different type of method is proposed for attacking cloud-hosted ML/DL models; a brief description of these methods is presented in **Table 1** and is discussed in detail in the next section.

Target model(s): Considered studies have used different MLaaS services (e.g., Google Cloud ML Services (Hosseini et al., 2017; Salem et al., 2018; Sethi and Kantardzic, 2018), ML models of BigML Platform (Kesarwani et al., 2018), IBM's visual recognition (Nguyen, 2017), and Amazon Prediction APIs (Reith et al., 2019; Yang et al., 2019)).

Dataset: These attacks have been realized using different datasets ranging from small size datasets (e.g., MNIST (Gu et al., 2019) and Fashion-MNIST (Liu et al., 2018)) to large size datasets (e.g., YouTube Aligned Face Dataset (Chen et al., 2017), Project Wolf Eye (Nguyen, 2017), and Iris dataset (Joshi and Tammana, 2019)). Other datasets include California Housing, Boston House Prices, UJIIndoorLoc, and IPIN 2016 Tutorial (Reith et al., 2019), FaceScrub, CelebA, and CIFAR-10 (Yang et al., 2019). A summary of thematic analyses of these attacks is presented in **Table 1** and briefly described in the next section.

4.2 Taxonomy of Attacks on Cloud-Hosted Machine Learning Models

In this section, we present a taxonomy and description of different attacks described above in thematic analysis. A taxonomy of attacks on cloud-hosted ML/DL models is depicted in **Figure 8** and is described next.

[6] Backdoor attacks on cloud-hosted models can be further categorized into three categories (Chen et al., 2020): 1) complete model–based attacks, 2) partial model–based attacks, and 3) model-free attacks).

TABLE 1 | Summary of the state-of-the art attack types for cloud-based/third-party ML/DL models.

Author(s)	Attack type	Method	Target model (s)	Threat model	Data
(Brendel et al., 2017)	Adversarial attack	Presented a decision-based attack, i.e., the boundary attack	Two ML classifiers from Clarifai.com, i.e., brand and celebrity recognition	Black box	Two datasets: Natural images and celebrities
(Saadatpanah et al., 2019)	—	Crafted adversarial examples for copyright detection system	YouTube content ID and AudioTag copyright	White box and black box	N/A
(Hosseini et al., 2017)	—	Proposed two targeted attacks for video labeling and shot detection	Google cloud video intelligence API	Black box	—
(Kesarwani et al., 2018)	Extraction attack	Used information gain to measure model learning rate	Decision tree deployed on BigML platform	Gray box	Four BigML datasets, IRS tax pattern, GSS survey, email importance, steak survey
(Correia-Silva et al., 2018)	—	Knowledge extraction by querying the model with unlabeled data samples and then used responses to create fake dataset and model	Three local CNN models for visual recognition for facial expression, object, and crosswalk classification and Microsoft Azure Emotion API	Black box	Used three datasets for facial expression recognition, object, and satellite crosswalk classification
(Reith et al., 2019)	—	Performed model extraction attacks on the homomorphic encryption-based protocol for preserving SVR-based indoor localization	Support vector regressor (SVR) and SVM	White box	California housing, Boston house prices, UJIIndoorLoc, and IPIN 2016 tutorial
(Joshi and Tammana, 2019)	—	Proposed a variant of gradient driven adaptive learning rate (GDALR) for stealing MLaaS models	Used three different models	Black box	Iris, liver disease, and land satellite datasets
(Sethi and Kantardzic, 2018)	Exploration attack	Presented a seed-explore-exploit framework for generating adversarial samples	Google cloud prediction platform	Black box	10 real-world datasets
(Gu et al., 2019)	Backdoor attack	Realized attack by poisoning training samples and labels	MNIST and a U.S. street sign classifier, i.e., Faster-RCNN with outsourced training and transfer learning	White box	MNIST and U.S. traffic signs dataset
(Chen et al., 2017)	—	Used poisoning strategies to realized a targeted attack and proposed two types of backdoor poisoning attacks	Two face recognition models, i.e., DeepID and VGG-Face	Black box	YouTube aligned face dataset
(Liu et al., 2018)	Trojan attack	Proposed stealth infection on neural network-based Trojan attack	Cloud-based intelligent supply chain, i.e., MLaaS	White box	Fashion-MNIST
(Gong et al., 2019)	—	Proposed real-time adversarial example crafting procedure	Voice/speech enabled devices and Google Speech	Gray box	Voice-command dataset
(Ji et al., 2018)	Model reuse attack	Presented empirical evaluation of model-reuse attacks on primitive models and realizing attack by generating semantically similar neighbors and identifying salient features	Pretrained primitive models for speech recognition, autonomous steering, face verification, and skin cancer screening	Gray box	Speech commands, udacity self-driving car challenge, VGG Face2, and International Skin Imaging Collaboration (ISIC) datasets
(Liao et al., 2018)	Data manipulation attack	Studied data manipulation attacks for stealthily manipulating ML and DL models using transfer learning and gradient descent	Cloud-hosted ML and DL models	White box	Enron spam and MINIST
(Sehwag et al., 2019)	—	Crafted out-of-distribution exploratory adversarial examples to compromise ML/DL models of Clarifai's content moderation system in the cloud	Cloud-hosted ML and DL models	White box and black box	MINIST, CIFAR, and ImageNet

(Continued on following page)

TABLE 1 | (*Continued*) Summary of the state-of-the art attack types for cloud-based/third-party ML/DL models.

Author(s)	Attack type	Method	Target model (s)	Threat model	Data
(Nguyen, 2017)	Cyber kill chain attack	Proposed a high-level threat model for ML cyber kill chain and provided proof of concept	IBM visual recognition MLaaS (i.e., cognitive classifier for classification cats and female lions)	N/A	Project Wolf Eye
(Hilprecht et al., 2019)	Membership inference attack	Monte Carlo based attack and membership inference attack on GAN.	Amazon web services p2	Black box	MNIST, fashion-MNIST, and CIFAR
(Hitaj et al., 2019)	Evasion attacks	Realized evasion attacks using two ensemble neural networks	Watermarking detection models	Black box	MNIST
(Yang et al., 2019)	Iversion attacks	Constructed an auxiliary set for training the inversion model	CNN	Gray-box	FaceScrub, CelebA, and CIFAR-10
—	—	—	—	—	—

4.2.1 Adversarial Attacks

In recent years, DL models have been found vulnerable to carefully crafted imperceptible adversarial examples (Goodfellow et al., 2014). For instance, a decision-based adversarial attack namely *the boundary attack* against two black box ML models trained for brand and celebrity recognition hosted at Clarifai.com are proposed in (Brendel et al., 2017). The first model identifies brand names from natural images for 500 distinct brands and the second model recognizes over 10,000 celebrities. To date, a variety of adversarial examples generation methods have been proposed in the literature so far, the interesting readers are referred to recent surveys articles for detailed taxonomy of different types of adversarial attacks (i.e., Akhtar and Mian, 2018; Yuan et al., 2019; Qayyum et al., 2020b; Demetrio et al., 2020).

4.2.2 Exploratory Attacks

These attacks are inference time attacks in which adversary attempts to evade the underlying ML/DL model, for example, by forcing the classifier (i.e., ML/DL model) to misclassify a positive sample as a negative one. Exploratory attacks do not harm the training data and only affects the model at test time. A data-driven exploratory attack using the *Seed–Explore–Exploit* strategy for evading Google's cloud prediction API considering black box settings is presented in (Sethi and Kantardzic, 2018). The performance evaluation of the proposed framework was performed using 10 real-world datasets.

4.2.3 Model Extraction Attacks

In model extraction attacks, adversaries can query the deployed ML model and can use query–response pair for compromising future predictions and also, they can potentially realize privacy breaches of the training data and can steal the model by learning extraction queries. In Kesarwani et al. (2018), the authors presented a novel method for quantifying the extraction status of models for users with an increasing number of queries, which aims to measure model learning rate using information gain observed by query and response streams of users. The key objective of the authors was to design a cloud-based system for monitoring model extraction status and warnings. The performance evaluation of the proposed method was performed using a decision tree model deployed on the BigML MLaaS platform for different adversarial attack scenarios. Similarly, a model extraction/stealing strategy is presented by Correia-Silva et al. (2018). The authors queried the cloud-hosted DL model with random unlabeled samples and used their predictions for creating a fake dataset. Then they used the fake dataset for building a fake model by training an oracle (copycat) model in an attempt to achieve similar performance as of the target model.

4.2.4 Backdooring Attacks

In backdooring attacks, an adversary maliciously creates the trained model which performs as good as expected on the users' training and validation data, but it performs badly on attacker

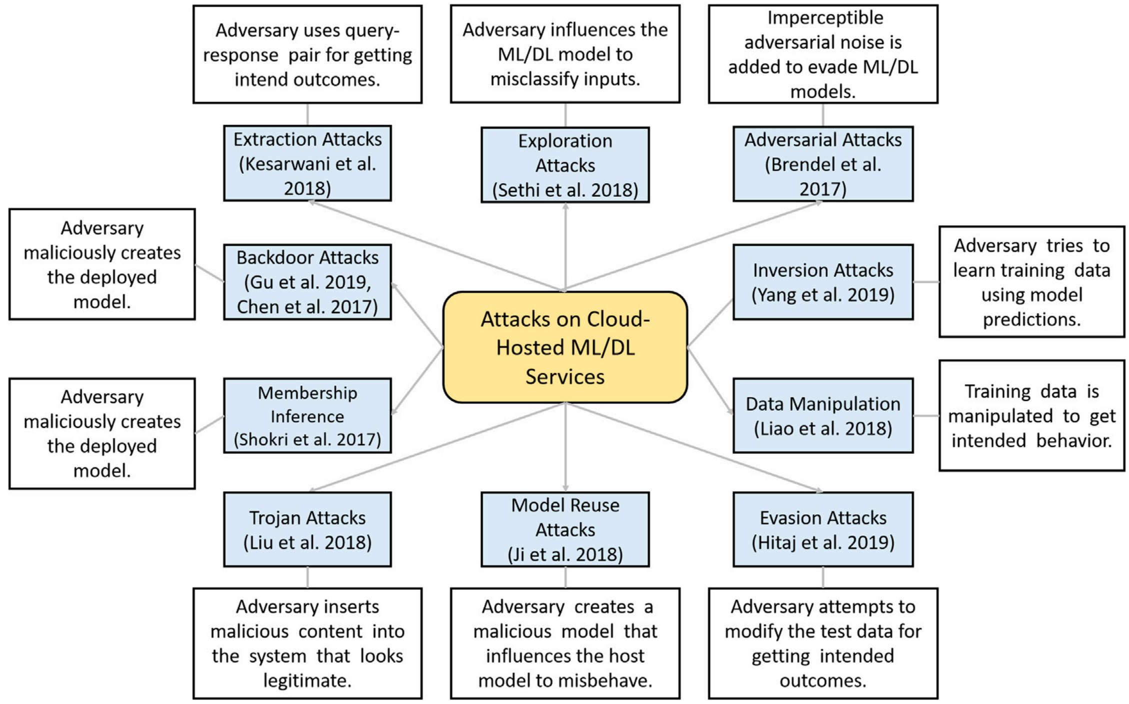

Attacks on Cloud-Hosted ML/DL Services
Extraction Attacks (Kesarwani et al. 2018)
Adversary uses query-response pair for getting intend outcomes.
Exploration Attacks (Sethi et al. 2018)
Adversary influences the ML/DL model to misclassify inputs.
Adversarial Attacks (Brendel et al. 2017)
Imperceptible adversarial noise is added to evade ML/DL models.
Backdoor Attacks (Gu et al. 2019, Chen et al. 2017)
Adversary maliciously creates the deployed model.
Inversion Attacks (Yang et al. 2019)
Adversary tries to learn training data using model predictions.
Membership Inference (Shokri et al. 2017)
Adversary maliciously creates the deployed model.
Data Manipulation (Liao et al. 2018)
Training data is manipulated to get intended behavior.
Trojan Attacks (Liu et al. 2018)
Adversary inserts malicious content into the system that looks legitimate.
Model Reuse Attacks (Ji et al. 2018)
Adversary creates a malicious model that influences the host model to misbehave.
Evasion Attacks (Hitaj et al. 2019)
Adversary attempts to modify the test data for getting intended outcomes.

FIGURE 8 | Distribution of selected publications by publishers over years.

input samples. The backdooring attacks on deep neural networks (DNNs) are explored and evaluated in (Gu et al., 2019). The authors first explored the properties of backdooring for a toy example and created a backdoor model for handwritten digit classifier and then demonstrated that backdoors are powerful for DNN by creating a backdoor model for a United States street sign classifier. Where, two scenarios were considered, that is, outsourced training of the model and transfer learning where an attacker can acquire a backdoor pretrained model online. In another similar study (Chen et al., 2017), a targeted backdoor attack for two state-of-the art face recognition models, that is, DeepID (Sun et al., 2014) and VGG-Face (Parkhi et al., 2015) is presented. The authors proposed two categories of backdooring poisoning attacks, that is, input–instance–key attacks and pattern–key attacks using two different data poising strategies, that is, input–instance–key strategies and pattern–key strategies, respectively.

4.2.5 Trojan Attacks

In Trojan attacks, the attacker inserts malicious content into the system that looks legitimate but can take over the control of the system. However, the purpose of Trojan insertion can be varied, for example, stealing, disruption, misbehaving, or getting intended behavior. In Liu et al. (2018), the authors proposed a stealth infection on neural networks, namely, SIN2 to realize a practical supply chain triggered neural Trojan attacks. Also, they proposed a variety of Trojan insertion strategies for agile and practical Trojan attacks. The proof of the concept is demonstrated by developing a prototype of the proposed neural Trojan attack (i.e., SIN2) in Linux sandbox and used Torch (Collobert et al., 2011) ML/DL framework for building visual recognition models using the Fashion-MNIST dataset.

4.2.6 Model-Reuse Attacks

In model-reuse attacks, an adversary creates a malicious model (i.e., adversarial model) that influences the host model to misbehave on targeted inputs (i.e., triggers) in extremely predictable fashion, that is, getting a sample classified into specific (intended class). For instance, experimental evaluation of model-reuse attacks for four pretrained primitive DL models (i.e., speech recognition, autonomous steering, face verification, and skin cancer screening) is evaluated by Ji et al. (2018).

4.2.7 Data Manipulation Attacks

Those attacks in which training data are manipulated to get intended behavior by the ML/DL model are known as data manipulation attacks. Data manipulation attacks for stealthily manipulating traditional supervised ML techniques and logistic regression (LR) and CNN models are studied by Liao et al. (2018). In the attack strategy, the authors added a new constraint on fully connected layers of the models and used gradient descent for retraining them, and other layers were frozen (i.e., were made non-trainable).

4.2.8 Cyber Kill Chain–Based Attacks

Kill chain is a term used to define steps for attacking a target usually used in the military. In cyber kill chain–based attacks, the cloud-hosted ML/DL models are attacked, for example, a high-level threat model targeting ML cyber kill chain is presented by Nguyen (2017). Also, the authors provided proof of concept by providing a case study using IBM visual recognition MLaaS (i.e., cognitive classifier for classification cats and female lions) and provided recommendations for ensuring secure and robust ML.

4.2.9 Membership Inference Attacks

In a typical membership inference attack, for given input data and black box access to the ML model, an attacker attempts to figure out if the given input sample was the part of the training set or not. To realize a membership inference attack against a target model, a classification model is trained for distinguishing between the predictions of the target model against the inputs on which it was trained and that those on which it was not trained (Shokri et al., 2017).

4.2.10 Evasion Attacks

Evasion attacks are inference time attacks in which an adversary attempts to modify the test data for getting the intended outcome from the ML/DL model. Two evasion attacks against watermarking techniques for DL models hosted as MLaaS have been presented by Hitaj et al. (2019). The authors used five publicly available models and trained them for distinguishing between watermarked and clean (non-watermarked) images, that is, binary image classification tasks.

4.2.11 Model Inversion Attacks

In model inversion attacks, an attacker tries to learn about training data using the model's outcomes. Two model inversion techniques have been proposed by Yang et al. (2019), that is, training an inversion model using auxiliary set composed by utilizing adversary's background knowledge and truncation-based method for aligning the inversion model. The authors evaluated their proposed methods on a commercial prediction MLaaS named Amazon Rekognition.

5 TOWARD SECURING CLOUD-HOSTED MACHINE LEARNING MODELS (Q2)

In this section, we present the insights from the systematically selected articles that provide tailored defense against specific attacks and report the articles that along with creating attacks propose countermeasure for the attacks for cloud-hosted/third-party ML/DL models.

5.1 Defenses for Attacks on Cloud-Hosted Machine Learning Models: Thematic Analysis

Leveraging cloud-based ML services for computational offloading and minimizing the communication overhead is accepted as a promising trend. While cloud-based prediction services have significant benefits, however, by sharing the model and the training data raises many privacy and security challenges. Several attacks that can compromise the model and data integrity, as described in the previous section. To avoid such issues, users can download the model and make inferences locally. However, this approach has certain drawbacks, including, confidentiality issues, service providers cannot update the models, adversaries can use the model to develop evading strategies, and privacy of the user data is compromised. To outline the countermeasures against these attacks, we present the thematic analysis of six articles that are focused on defense against the tailored attacks for cloud-hosted ML/DL models or data. In addition, we also provide the thematic analysis of those six articles that propose defense against specific attacks. These articles are classified into five major themes: 1) attack type, 2) defense, 3) target model(s), 4) dataset, and 5) measured outcomes. The thematic analysis of these systematically reviewed articles that are focused on developing defense strategies against attacks is given below.

Considered attacks for developing defenses: The defenses proposed in the reviewed articles are developed against the following specific attacks.

- Extraction attacks (Tramèr et al., 2016; Liu et al., 2017);
- Inversion attacks (Liu et al., 2017; Sharma and Chen, 2018);
- Adversarial attacks (Hosseini et al., 2017; Wang et al., 2018b; Rouhani et al., 2018);
- Evasion attacks (Lei et al., 2020);
- GAN attacks (Sharma and Chen, 2018);
- Privacy threat attacks (Hesamifard et al., 2017);
- ide channel and cache-timing attacks (Jiang et al., 2018);
- Membership inference attacks (Shokri et al., 2017; Salem et al., 2018).

Most of the aforementioned attacks are elaborated in previous sections. However, in the selected articles that are identified as either defense or attack and defense articles, some attacks are specifically created, for instance, GAN attacks, side channel, cache-timing attack, privacy threats, etc. Therefore, the attacks are worth mentioning in this section to explain the specific countermeasures proposed against them in the defense articles.

Defenses against different attacks: To provide resilience against these attacks, the authors of selected articles proposed different defense algorithms, which are listed below against each type of attack.

- Extraction attacks: MiniONN (Liu et al., 2017), rounding confidence, differential, and ensemble methods (Tramèr et al., 2016);
- Adversarial attacks: ReDCrypt (Rouhani et al., 2018) and Arden (Wang et al., 2018b);
- Inversion attacks: MiniONN (Liu et al., 2017) and image disguising techniques (Sharma and Chen, 2018);
- Privacy attacks: encryption-based defense (Hesamifard et al., 2017; Jiang et al., 2018);
- Side channel and cache-timing attacks: encryption-based defense (Hesamifard et al., 2017; Jiang et al., 2018);
- Membership inference attack: dropout and model stacking (Salem et al., 2018).

TABLE 2 | Summary of attack types and corresponding defenses for cloud-based/third-party ML/DL models.

Author	Attack	Defense	Target model	Data	Measured outcomes
(Liu et al., 2017)	Extraction attack and inversion attack	MiniONN: a defense against information leakage in DNN to transform into an oblivious NN	Cloud-hosted DL models, neural network for cloud-based prediction services	MNIST and CIFAR-10	Response latency and message sizes
(Rouhani et al., 2018)	Adversarial attacks	ReDCrypt: reconfigurable hardware-accelerated framework for the privacy-preserving	Cloud-hosted DL models	MNIST and MovieLens	Throughput
(Wang et al., 2018b)	—	Arden: To distribute DNN model computation among edge device and cloud data centers	Partial cloud-hosted DNN models	MNIST, SVHN, and CIFAR-10	Latency, accuracy, and privacy budget
(Hosseini et al., 2017)	—	Incorporating randomness to video analysis algorithms	Google cloud video intelligence API	Videos comprising of adversarial examples	Histogram peaks to detect shot change
(Sharma and Chen, 2018)	Inversion attack and GAN attack	Image disguising techniques to ensure the protection against model-based adversarial attacks	Cloud-hosted DL models	MNIST and CIFAR-10	Accuracy, average visual privacy, and Fano factor
(Hesamifard et al., 2017)	Privacy threats due to raw cloud data	Homomorphic encryption to preserve the privacy and integrity of data in DNN	Cloud-based DNN	Crab dataset, fertility dataset, climate dataset	Accuracy and training time
(Jiang et al., 2018)	Side channel and cache-timing attack	Secure logistic encryption along with hardware-based security enhancement by exploiting software guard extensions	Cloud-hosted LR models	Edinburgh MI, WI-Breast cancer, and MONK's prob	Area under the curve, complexity, and model training time
(Lei et al., 2020)	Evasion attack	Pelican: similarity-based analysis of unknown website with the known phishing Web site	BitDefender's partical processing hosted on cloud	PhishTank, PhishNet	Similarity index
(Tramèr et al., 2016)	Extraction attack	Rounding confidences to some precision, differential privacy to protect training data elements, ensemble methods	ML models hosted on BigML and amazon	102 categories flower dataset, face dataset, iris dataset, and traffic signs dataset	Success rate given the perturbation budget
(Shokri et al., 2017)	Membership inference attack	Top *k* class model predictions, increase entropy, regularization and reducing precision of prediction vector	MLaaS classification models of Google and Amazon APIs	CIFAR-10,purchases, locations, Texas hospital stays, MNIST, UCI adults	Accuracy and precision
(Salem et al., 2018)	—	Dropout and model stacking to prevent overfitting	Google cloud prediction API	Used eight different datasets	Precision and recall
(Wang et al., 2018a)	Misclassification attacks	Neuron distance model, ensemble method, dropout randomization	Google cloud ML, microsoft cognitive toolkit (CNTK), and the PyTorch	102-Class VGG flower, face dataset, iris dataset, and traffic signs dataset, Google's InceptionV3	Accuracy and success rate

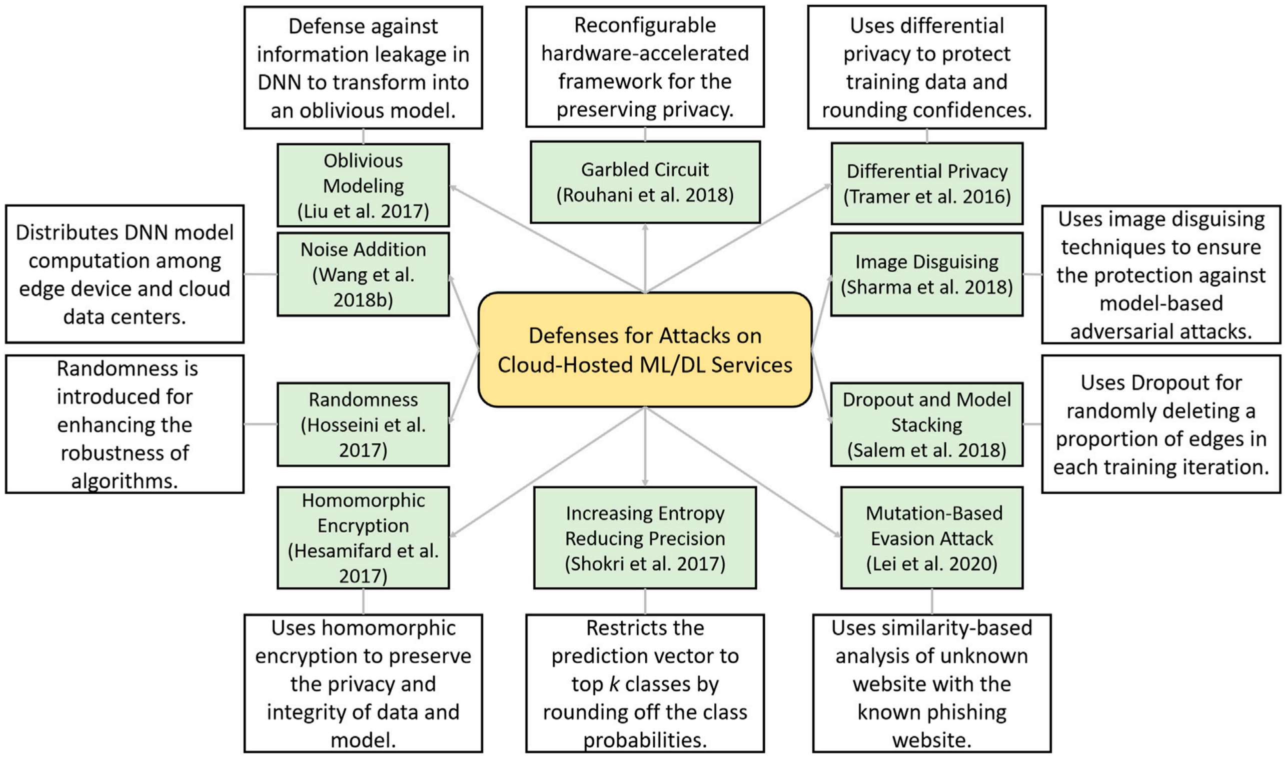

Defense against information leakage in DNN to transform into an oblivious model.
Reconfigurable hardware-accelerated framework for the preserving privacy.
Uses differential privacy to protect training data and rounding confidences.
Oblivious Modeling (Liu et al. 2017)
Garbled Circuit (Rouhani et al. 2018)
Differential Privacy (Tramer et al. 2016)
Distributes DNN model computation among edge device and cloud data centers.
Noise Addition (Wang et al. 2018b)
Image Disguising (Sharma et al. 2018)
Uses image disguising techniques to ensure the protection against model-based adversarial attacks.
Defenses for Attacks on Cloud-Hosted ML/DL Services
Randomness is introduced for enhancing the robustness of algorithms.
Randomness (Hosseini et al. 2017)
Dropout and Model Stacking (Salem et al. 2018)
Uses Dropout for randomly deleting a proportion of edges in each training iteration.
Homomorphic Encryption (Hesamifard et al. 2017)
Increasing Entropy Reducing Precision (Shokri et al. 2017)
Mutation-Based Evasion Attack (Lei et al. 2020)
Uses homomorphic encryption to preserve the privacy and integrity of data and model.
Restricts the prediction vector to top k classes by rounding off the class probabilities.
Uses similarity-based analysis of unknown website with the known phishing website.

FIGURE 9 | Taxonomy of different attacks realized on the third-party cloud-hosted machine learning (ML) or deep learning (DL) models.

Target model(s): Different cloud-hosted ML/DL models have been used for the evaluation of the proposed defenses, as shown in **Table 2**.

Dataset(s) used: The robustness of these defenses have been evaluated using various datasets ranging from small size datasets (e.g., MNIST (Liu et al., 2017; Wang et al., 2018b; Rouhani et al., 2018; Sharma and Chen, 2018)) and CIFAR-10 (Liu et al., 2017; Wang et al., 2018b; Sharma and Chen, 2018)), to large size datasets (e.g., Iris dataset (Tramèr et al., 2016), fertility and climate dataset (Hesamifard et al., 2017), and breast cancer (Jiang et al., 2018)). Other datasets include Crab dataset (Hesamifard et al., 2017), Face dataset, Traffic signs dataset, Traffic signs dataset (Tramèr et al., 2016), SVHN (Wang et al., 2018b), Edinburgh MI, Edinburgh MI, WI-Breast Cancerband MONKs Prob (Jiang et al., 2018), crab dataset, fertility dataset, and climate dataset (Hesamifard et al., 2017). Each of the defense techniques discussed above is mapped in **Table 2** to the specific attack for which it was developed.

Measured outcomes: The measured outcomes based on which the defenses are evaluated are response latency and message sizes (Liu et al., 2017; Wang et al., 2018b), throughput comparison (Rouhani et al., 2018), average on the cache miss rates per second (Sharma and Chen, 2018), AUC, space complexity to demonstrate approximated storage costs (Jiang et al., 2018), classification accuracy of the model as well as running time (Hesamifard et al., 2017; Sharma and Chen, 2018), similarity index (Lei et al., 2020), and training time (Hesamifard et al., 2017; Jiang et al., 2018).

5.2 Taxonomy of Defenses on Cloud-Hosted Machine Learning Model Attacks

In this section, we present a taxonomy and summary of different defensive strategies against attacks on cloud-hosted ML/DL models as described above in thematic analysis. A taxonomy of these defenses strategies is presented in **Figure 9** and is described next.

5.2.1 MiniONN

DNNs are vulnerable to model inversion and extraction attacks. Liu et al. (2017) proposed that without making any changes to the training phase of the model it is possible to change the model into an oblivious neural network. They make the nonlinear function such as *tanh* and *sigmoid* function more flexible, and by training the models on several datasets, the authors demonstrated significant results with minimal loss in the accuracy. In addition, they also implemented the offline precomputation phase to perform encryption incremental operations along with the SIMD batch processing technique.

5.2.2 ReDCrypt

A reconfigurable hardware-accelerated framework is proposed by Rouhani et al. (2018), for protecting the privacy of deep neural models in cloud networks. The authors perform an innovative and power-efficient implementation of Yao's Garbled Circuit (GC) protocol on FPGAs for preserving privacy. The proposed framework is evaluated for different

DL applications, and it has achieved up to 57-fold throughput gain per core.

5.2.3 Arden

To offload the large portion of DNNs from the mobile devices to the clouds and to make the framework secure, a privacy-preserving mechanism Arden is proposed by Wang et al. (2018b). While uploading the data to the mobile-cloud perturbation, noisy samples are included to make the data secure. To verify the robustness, the authors perform rigorous analysis based on three image datasets and demonstrated that this defense is capable to preserve the user privacy along with inference performance.

5.2.4 Image Disguising Techniques

While leveraging services from the cloud GPU server, the adversary can realize an attack by introducing malicious created training data, perform model inversion, and use the model for getting desirable incentives and outcomes. To protect from such attacks and to preserve the data as well as the model, Sharma and Chen (2018) proposed an image disguising mechanism. They developed a toolkit that can be leveraged to calibrate certain parameter settings. They claim that the disguised images with block-wise permutation and transformations are resilient to GAN-based attack and model inversion attacks.

5.2.5 Homomorphic Encryption

For making the cloud services of outsourced MLaaS secure, Hesamifard et al. (2017) proposed a privacy-preserving framework using homomorphic encryption. They trained the neural network using the encrypted data and then performed the encrypted predictions. The authors demonstrated that by carefully choosing the polynomials of the activation functions to adopt neural networks, it is possible to achieve the desired accuracy along with privacy-preserving training and classification.

In a similar study, to preserve the privacy of outsourced biomedical data and computation on public cloud servers, Jiang et al. (2018) built a homomorphically encrypted model that reinforces the hardware security through Software Guard Extensions. They combined homomorphic encryption and Software Guard Extensions to devise a hybrid model for the security of the most commonly used model for biomedical applications, that is, LR. The robustness of the Secure LR framework is evaluated on various datasets, and the authors also compared its performance with state-of-the-art secure LR solutions and demonstrated its superior efficiency.

5.2.6 Pelican

Lei et al. (2020) proposed three mutation-based evasion attacks and a sample-based collision attack in white-, gray-, and black box scenarios. They evaluated the attacks and demonstrated a 100% success rate of attack on Google’s phishing page filter classifier, while a success rate of up to 81% for the transferability on Bitdefender TrafficLight. To deal with such attacks and to increase the robustness of classifiers, they proposed a defense method known as Pelican.

5.2.7 Rounding Confidences and Differential Privacy

Tramèr et al. (2016) presented the model extraction attacks against the online services of BigML and Amazon ML. The attacks are capable of model evasion, monetization, and can compromise the privacy of training data. The authors also proposed and evaluated countermeasures such as rounding confidences against equation-solving and decision tree pathfinding attacks; however, this defense has no impact on the regression tree model attack. For the preservation of training data, differential privacy is proposed; this defense reduces the ability of an attacker to learn insights about the training dataset. The impact of both defenses is evaluated on the attacks for different models, while the authors also proposed ensemble models to mitigate the impact of attacks; however, their resilience is not evaluated.

5.2.8 Increasing Entropy and Reducing Precision

The training of attack using shadow training techniques against black box models in the cloud-based Google Prediction API and Amazon ML models are studied by Shokri et al. (2017). The attack does not require prior knowledge of training data distribution. The authors emphasize that in order to protect the privacy of medical-related datasets or other public-related data, countermeasures should be designed. For instance, restriction of prediction vector to top *k* classes, which will prevent the leakage of important information or rounding down or up the classification probabilities in the prediction. They show that regularization can be effective to cope with overfitting and increasing the randomness of the prediction vector.

5.2.9 Dropout and Model Stacking

In the study by Salem et al. (2018), the authors created three diverse attacks and tested the applicability of these attacks on eight datasets from which six are similar as used by Shokri et al. (2017), whereas in this work, news dataset and face dataset is included. In the threat model, the authors considered black box access to the target model which is a supervised ML classifier with binary classes that was trained for binary classification. To mitigate the privacy threats, the authors proposed a dropout-based method which reduces the impact of an attack by randomly deleting a proportion of edges in each training iteration in a fully connected neural network. The second defense strategy is model stacking, which hierarchically organizes multiple ML models to avoid overfitting. After extensive evaluation, these defense techniques showed the potential to mitigate the performance of the membership inference attack.

5.2.10 Randomness to Video Analysis Algorithms

Hosseini et al. designed two attacks specifically to analyze the robustness of video classification and shot detection (Hosseini et al., 2017). The attack can subtly manipulate the content of the video in such a way that it is undetected by humans, while the output from the automatic video analysis method is altered. Depending on the fact that the video and shot labels are generated by API by processing only the first video frame of every second, the attack can successfully deceive API. To deal with the shot removal and generation attacks, the authors proposed the inclusion of randomness for enhancing the robustness of algorithms. However, in this article, the authors thoroughly evaluated the applicability of these attacks in different video setting, but the purposed defense is not rigorously evaluated.

5.2.11 Neuron Distance Threshold and Obfuscation

Transfer learning is an effective technique for quickly building DL student models in which knowledge from a Teacher model is transferred to a Student model. However, Wang et al. (2018a) discussed that due to the centralization of model training, the vulnerability against misclassification attacks for image recognition on black box Student models increases. The authors proposed several defenses to mitigate the impact of such an attack, such as changing the internal representation of the Student model from the Teacher model. Other defense methods include increasing dropout randomization which alters the student model training process, modification in input data before classification, adding redundancy, and using orthogonal model against transfer learning attack. The authors analyzed the robustness of these attacks and demonstrated that the neuron distance threshold is the most effective in obfuscating the identity of the Teacher model.

6 PITFALLS AND LIMITATIONS

6.1 Lack of Attack Diversity

The attacks presented in the selected articles have limited scope and lack diversity, that is, they are limited to a specific setting, and the variability of attacks is limited as well. However, the diversity of attacks is an important consideration for developing robust attacks from the perspective of adversaries, and it ensures the detection and prevention of the attacks to be difficult. The diversity of attacks ultimately helps in the development of robust defense strategies. Moreover, the empirical evaluation of attack variabilities can identify the potential vulnerabilities of cybersecurity systems. Therefore, to make a more robust defense solution, it is important to test the model robustness under a diverse set of attacks.

6.2 Lack of Consideration for Adaptable Adversaries

Most of the defenses in the systematically reviewed articles are proposed for a specific attack and did not consider the adaptable adversaries. On the other hand, in practice, the adversarial attacks are an arms race between attackers and defenders. That is, the attackers continuously evolve and enhance their knowledge and attacking strategies to evade the underlying defensive system. Therefore, the consideration of adaptable adversaries is crucial for developing a robust and long-lasting defense mechanism. If we do not consider this, the adversary will adapt to our defensive system over time and will bypass it to get the intended behavior or outcomes.

6.3 Limited Progress in Developing Defenses

From the systematically selected articles that are collected from different databases, only 12 articles have presented defense methods for the proposed attack as compared to the articles that are focused on attacks, that is, 19. In these 12 articles, six have only discussed/presented a defense strategy and six have developed a defense against a particular attack. This indicates that there is limited activity from the research community in developing defense strategies for already proposed attacks in the literature. In addition, the proposed defenses only mitigate or detect those attacks for which they have been developed, and therefore, they are not generalizable. On the contrary, the increasing interest in developing different attacks and the popularity of cloud-hosted/third-party services demand a proportionate amount of interest in developing defense systems as well.

7 OPEN RESEARCH ISSUES

7.1 Adversarially Robust Machine Learning Models

In recent years, adversarial ML attacks have emerged as a major panacea for ML/DL models and the systematically selected articles have highlighted the threat of these attacks for cloud-hosted Ml/DL models as well. Moreover, the diversity of these attacks is drastically increasing as compared with the defensive strategies that can pose serious challenges and consequences for the security of cloud-hosted ML/DL models. Each defense method presented in the literature so far has been shown resilient to a particular attack which is realized in specific, settings and it fails to withstand for yet stronger and unseen attacks. Therefore, the development of adversarially robust ML/DL models remains an open research problem, while the literature suggests that worst-case robustness analysis should be performed while considering adversarial ML settings (Qayyum et al., 2020a; Qayyum et al., 2020b; Ilahi et al., 2020). In addition, it has been argued in the literature that most of ML developers and security incident responders are unequipped with the required tools for securing industry-grade ML systems against adversarial ML attacks Kumar et al. (2020). This indicates the increasing need for the development of defense strategies for securing ML/DL models against adversarial ML attacks.

7.2 Privacy-Preserving Machine Learning Models

In cloud-hosted ML services, preserving user privacy is fundamentally important and is a matter of high concern. Also, it is desirable that ML models built using users' data should not learn information that can compromise the privacy of the individuals. However, the literature on developing privacy-preserving ML/DL models or MLaaS is limited. On the other hand, one of the privacy-preserving techniques that have been used for privacy protection for building a defense system for cloud-hosted ML/DL models, that is, the homomorphic encryption-based protocol (Jiang et al., 2018), has been shown vulnerable to model extraction attack (Reith et al., 2019). Therefore, the development of privacy-preserving ML models for cloud computing platforms is another open research problem.

7.3 Proxy Metrics for Evaluating Security and Robustness

From systematically reviewed literature on the security of cloud-hosted ML/DL models, we orchestrate that the interest from the research community in the development of novel security-centric proxy metrics for the evaluation of security threats and model robustness of cloud-hosted models is very limited. However, with the increasing proliferation of cloud-hosted ML services (i.e., MLaaS) and with the development/advancements of different attacks (e.g., adversarial ML attacks), the development of effective and scalable metrics for evaluating the robustness ML/DL models toward different attacks and defense strategies is required.

8 THREATS TO VALIDITY

We now briefly reflect on our methodology in order to identify any threats to the validity of our findings. First, internal validity is maintained as the research questions we pose in **Section 2.2** capture the objectives of the study. Construct validity relies on a sound understanding of the literature and how it represents the state of the field. A detailed study of the reviewed articles along with deep discussions between the members of the research team helped ensure the quality of this understanding. Note that the research team is of diverse skills and expertise in ML, DL, cloud computing, ML/DL security, and analytics. Also, the inclusion and exclusion criteria (**Section 2.3**) help define the remit of our survey. Data extraction is prone to human error as is always the case. This was mitigated by having different members of the research team review each reviewed article. However, we did not attempt to evaluate the quality of the reviewed studies or validate their content due to time constraints. In order to minimize selection bias, we cast a wide net in order to capture articles from different communities publishing in the area of MLaaS via a comprehensive set of bibliographical databases without discriminating based on the venue/source.

9 CONCLUSION

In this article, we presented a systematic review of literature that is focused on the security of cloud-hosted ML/DL models, also named as MLaaS. The relevant articles were collected from eight major publishers that include ACM Digital Library, IEEE Xplore, ScienceDirect, international conference on machine learning, international conference on learning representations, journal of machine learning research, USENIX, neural information processing systems, and arXiv. For the selection of articles, we developed a review protocol that includes inclusion and exclusion formulas and analyzed the selected articles that fulfill these criteria across two dimensions (i.e., attacks and defenses) on MLaaS and provide a thematic analysis of these articles across five attack and five defense themes, respectively. We also identified the limitations and pitfalls from the reviewed literature, and finally, we have highlighted various open research issues that require further investigation.

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

AQ led the work in writing the manuscript and performed the annotation of the data and analysis as well. AI performed data acquisition, annotation, and analysis from four venues, and contributed to the paper write-up. MU contributed to writing a few sections, did annotations of papers, and helped in analysis. WI performed data scrapping, annotation, and analysis from four venues, and helped in developing graphics. All the first four authors validated the data, analysis, and contributed to the interpretation of the results. AQ and AI helped in developing and refining the methodology for this systematic review. JQ conceived the idea and supervises the overall work. JQ, YEK, and AF provided critical feedback and helped shape the research, analysis, and manuscript. All authors contributed to the final version of the manuscript.

REFERENCES

Akhtar, N., and Mian, A. (2018). Threat of adversarial attacks on deep learning in computer vision: a survey. *IEEE Access* 6, 14410–14430. doi:10.1109/access.2018.2807385

Apruzzese, G., Colajanni, M., Ferretti, L., and Marchetti, M. (2019). “Addressing adversarial attacks against security systems based on machine learning,” in 2019 11th International conference on cyber conflict (CyCon), Tallinn, Estonia, May 28–31, 2019 (IEEE), 900, 1–18

Brendel, W., Rauber, J., and Bethge, M. (2017). “Decision-based adversarial attacks: reliable attacks against black-box machine learning models,” in International Conference on Learning Representations (ICLR)

Chen, S., Xue, M., Fan, L., Hao, S., Xu, L., Zhu, H., et al. (2018). Automated poisoning attacks and defenses in malware detection systems: an adversarial machine learning approach. *Comput. Secur.* 73, 326–344. doi:10.1016/j.cose.2017.11.007

Chen, X., Liu, C., Li, B., Lu, K., and Song, D. (2017). Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv*

Chen, Y., Gong, X., Wang, Q., Di, X., and Huang, H. (2020). Backdoor attacks and defenses for deep neural networks in outsourced cloud environments. *IEEE Network* 34 (5), 141–147. doi:10.1109/MNET.011.1900577

Collobert, R., Kavukcuoglu, K., and Farabet, C. (2011). “Torch7: a Matlab-like environment for machine learning,” in BigLearn, NIPS workshop.

Correia-Silva, J. R., Berriel, R. F., Badue, C., de Souza, A. F., and Oliveira-Santos, T. (2018). “Copycat CNN: stealing knowledge by persuading confession with random non-labeled data,” in 2018 International joint conference on neural networks (IJCNN), Rio de Janeiro, Brazil, July 8–13, 2018 (IEEE), 1–8

Demetrio, L., Valenza, A., Costa, G., and Lagorio, G. (2020). “Waf-a-mole: evading web application firewalls through adversarial machine learning,” in Proceedings of the 35th annual ACM symposium on applied computing, Brno, Czech Republic, March 2020, 1745–1752

Gong, Y., Li, B., Poellabauer, C., and Shi, Y. (2019). "Real-time adversarial attacks," in Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), Macao, China, August 2019

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv*

Gu, T., Liu, K., Dolan-Gavitt, B., and Garg, S. (2019). BadNets: evaluating backdooring attacks on deep neural networks. *IEEE Access* 7, 47230–47244. doi:10.1109/access.2019.2909068

Han, D., Wang, Z., Zhong, Y., Chen, W., Yang, J., Lu, S., et al. (2020). Practical traffic-space adversarial attacks on learning-based nidss. *arXiv*

Hesamifard, E., Takabi, H., Ghasemi, M., and Jones, C. (2017). "Privacy-preserving machine learning in cloud," in Proceedings of the 2017 on cloud computing security workshop, 39–43

Hilprecht, B., Härterich, M., and Bernau, D. (2019). "Monte Carlo and reconstruction membership inference attacks against generative models," in Proceedings on Privacy Enhancing Technologies, Stockholm, Sweden, July 2019, 2019, 232–249

Hitaj, D., Hitaj, B., and Mancini, L. V. (2019). "Evasion attacks against watermarking techniques found in MLaaS systems," in 2019 sixth international conference on software defined systems (SDS), Rome, Italy, June 10–13, 2019 (IEEE)

Hosseini, H., Xiao, B., Clark, A., and Poovendran, R. (2017). "Attacking automatic video analysis algorithms: a case study of google cloud video intelligence API," in Proceedings of the 2017 conference on multimedia Privacy and security (ACM), 21–32

Ilahi, I., Usama, M., Qadir, J., Janjua, M. U., Al-Fuqaha, A., Hoang, D. T., et al. (2020). Challenges and countermeasures for adversarial attacks on deep reinforcement learning. *arXiv*

Ji, Y., Zhang, X., Ji, S., Luo, X., and Wang, T. (2018). "Model-reuse attacks on deep learning systems, "in Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (New York, NY: ACM), December 2018, 349–363

Jiang, Y., Hamer, J., Wang, C., Jiang, X., Kim, M., Song, Y., et al. (2018). Securelr: secure logistic regression model via a hybrid cryptographic protocol. *IEEE ACM Trans. Comput. Biol. Bioinf* 16, 113–123. doi:10.1109/TCBB.2018.2833463

Joshi, N., and Tammana, R. (2019). "GDALR: an efficient model duplication attack on black box machine learning models," in 2019 IEEE international Conference on system, computation, Automation and networking (ICSCAN),Pondicherry, India, March 29–30, 2019 (IEEE), 1–6

Kesarwani, M., Mukhoty, B., Arya, V., and Mehta, S. (2018). Model extraction warning in MLaaS paradigm. In Proceedings of the 34th Annual Computer Security Applications Conference (ACM), 371–380

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 1097–1105 Available at: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Kumar, R. S. S., Nyström, M., Lambert, J., Marshall, A., Goertzel, M., Comissoneru, A., et al. (2020). Adversarial machine learning–industry perspectives. *arXiv*. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3532474

Lei, Y., Chen, S., Fan, L., Song, F., and Liu, Y. (2020). Advanced evasion attacks and mitigations on practical ml-based phishing website classifiers. *arXiv*

Liang, B., Su, M., You, W., Shi, W., and Yang, G. (2016). "Cracking classifiers for evasion: a case study on the google's phishing pages filter," in Proceedings of the 25th international conference on world wide web Montréal, Québec, Canada, 345–356

Liao, C., Zhong, H., Zhu, S., and Squicciarini, A. (2018). "Server-based manipulation attacks against machine learning models," in Proceedings of the eighth ACM conference on data and application security and privacy (ACM), New York, NY, March 2018, 24–34

Liu, J., Juuti, M., Lu, Y., and Asokan, N. (2017). "Oblivious neural network predictions via minionn transformations," in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, October 2017, 619–631

Liu, T., Wen, W., and Jin, Y. (2018). "SIN 2: stealth infection on neural network—a low-cost agile neural Trojan attack methodology," in 2018 IEEE international symposium on hardware oriented security and trust (HOST), Washington, DC, April 30–4 May, 2018 (IEEE), 227–230

Nguyen, T. N. (2017). Attacking machine learning models as part of a cyber kill chain. *arXiv*

Parkhi, O. M., Vedaldi, A., Zisserman, A., et al. (2015). Deep face recognition. *Bmvc* 1, 6. doi:10.5244/C.29.41

Qayyum, A., Qadir, J., Bilal, M., and Al-Fuqaha, A. (2020a). Secure and robust machine learning for healthcare: a survey. *IEEE Rev. Biomed. Eng.*, 1. doi:10.1109/RBME.2020.3013489

Qayyum, A., Usama, M., Qadir, J., and Al-Fuqaha, A. (2020b). Securing connected & autonomous vehicles: challenges posed by adversarial machine learning and the way forward. *IEEE Commun. Surv. Tutorials* 22, 998–1026. doi:10.1109/comst.2020.2975048

Reith, R. N., Schneider, T., and Tkachenko, O. (2019). "Efficiently stealing your machine learning models," in Proceedings of the 18th ACM workshop on privacy in the electronic society, November 2019, 198–210

Rouhani, B. D., Hussain, S. U., Lauter, K., and Koushanfar, F. (2018). Redcrypt: real-time privacy-preserving deep learning inference in clouds using fpgas. *ACM Trans. Reconfigurable Technol. Syst.* 11, 1–21. doi:10.1145/3242899

Saadatpanah, P., Shafahi, A., and Goldstein, T. (2019). Adversarial attacks on copyright detection systems. *arXiv*.

Salem, A., Zhang, Y., Humbert, M., Berrang, P., Fritz, M., and Backes, M. (2018). ML-leaks: model and data independent membership inference attacks and defenses on machine learning models. *arXiv*.

Sehwag, V., Bhagoji, A. N., Song, L., Sitawarin, C., Cullina, D., Chiang, M., et al. (2019). Better the devil you know: an analysis of evasion attacks using out-of-distribution adversarial examples. *arXiv*.

Sethi, T. S., and Kantardzic, M. (2018). Data driven exploratory attacks on black box classifiers in adversarial domains. *Neurocomputing* 289, 129–143. doi:10.1016/j.neucom.2018.02.007

Sharma, S., and Chen, K. (2018). "Image disguising for privacy-preserving deep learning," in Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, (ACM, Toronto, Canada), 2291–2293

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017). "Membership inference attacks against machine learning models," in 2017 IEEE Symposium on Security and privacy (SP), San Jose, CA, May 22–26, 2017 (IEEE), 3–18

Simonyan, K., and Zisserman, A. (2015). "Very deep convolutional networks for large-scale image recognition,"in International Conference on Learning Representations (ICLR)

Song, Y., Liu, T., Wei, T., Wang, X., Tao, Z., and Chen, M. (2020). Fda3: federated defense against adversarial attacks for cloud-based iiot applications. *IEEE Trans. Industr. Inform.*, 1. doi:10.1109/TII.2020.3005969

Sun, Y., Wang, X., and Tang, X. (2014). "Deep learning face representation from predicting 10,000 classes," in Proceedings of the IEEE conference on computer vision and pattern recognition, Columbus, OH, June 23–28, 2014, (IEEE).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. "(2016). Rethinking the inception architecture for computer vision," in Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, June 27–30, 2016 (IEEE), 2818–2826

Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., and Ristenpart, T. (2016). "Stealing machine learning models via prediction APIs," in 25th USENIX security symposium (USENIX Security 16), 601–618

Tyndall, J. (2010). *AACODS checklist*. Adelaide, Australia: Adelaide Flinders University

Usama, M., Mitra, R. N., Ilahi, I., Qadir, J., and Marina, M. K. (2020a). Examining machine learning for 5g and beyond through an adversarial lens. *arXiv*. Available at: https://arxiv.org/abs/2009.02473.

Usama, M., Qadir, J., Al-Fuqaha, A., and Hamdi, M. (2020b). The adversarial machine learning conundrum: can the insecurity of ML become the achilles' heel of cognitive networks? *IEEE Network* 34, 196–203. doi:10.1109/mnet.001.1900197

Usama, M., Qayyum, A., Qadir, J., and Al-Fuqaha, A. (2019). "Black-box adversarial machine learning attack on network traffic classification, "in 2019 15th international wireless communications and mobile computing conference (IWCMC), Tangier, Morocco, June 24–28, 2019

Wang, B., Yao, Y., Viswanath, B., Zheng, H., and Zhao, B. Y. (2018a). "With great training comes great vulnerability: practical attacks against transfer learning," in 27th USENIX security symposium (USENIX Security 18), Baltimore, MD, August 2018, 1281–1297

Wang, J., Zhang, J., Bao, W., Zhu, X., Cao, B., and Yu, P. S. (2018b). "Not just privacy: improving performance of private deep learning in mobile cloud," in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining London, United Kingdom, January 2018, 2407–2416

Yang, Z., Zhang, J., Chang, E.-C., and Liang, Z. (2019). "Neural network inversion in adversarial setting via background knowledge alignment," in Proceedings of the 2019 ACM SIGSAC conference on computer and communications security, London, UK, November 2019, 225–240

Yuan, X., He, P., Zhu, Q., and Li, X. (2019). Adversarial examples: attacks and defenses for deep learning. *IEEE Trans. Neural. Netw. Learn. Syst.* 30 (9), 2805–2824. doi:10.1109/TNNLS.2018.2886017

Zhang, J., Zhang, B., and Zhang, B. (2019). "Defending adversarial attacks on cloud-aided automatic speech recognition systems, "in Proceedings of the seventh international workshop on security in cloud computing, New York, 23–31. Available at: https://dl.acm.org/doi/proceedings/10.1145/3327962

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.

frontiers
in Big Data

PERSPECTIVE
published: 23 November 2020
doi: 10.3389/fdata.2020.577974

Interdisciplinary Research in Artificial Intelligence: Challenges and Opportunities

Remy Kusters[1]*, Dusan Misevic[1]*, Hugues Berry[2], Antoine Cully[3], Yann Le Cunff[4], Loic Dandoy[1], Natalia Díaz-Rodríguez[5,6], Marion Ficher[1], Jonathan Grizou[1], Alice Othmani[7], Themis Palpanas[8], Matthieu Komorowski[3], Patrick Loiseau[9], Clément Moulin Frier[5], Santino Nanini[1], Daniele Quercia[10], Michele Sebag[11], Françoise Soulié Fogelman[12], Sofiane Taleb[1], Liubov Tupikina[1,13], Vaibhav Sahu[1], Jill-Jênn Vie[14] and Fatima Wehbi[1]

[1]INSERM U1284, Université de Paris, Center for Research and Interdisciplinarity (CRI), Paris, France, [2]Inria, Villeurbanne, France, [3]Imperial College London, London, United Kingdom, [4]University of Rennes, Rennes, France, [5]Inria Flowers, Paris and Bordeaux, France, [6]ENSTA Paris, Institut Polytechnique Paris, Paris, France, [7]Université Paris-Est, LISSI, Vitry sur Seine, France, [8]Université de Paris, France and French University Institute (IUF), Paris, France, [9]Université Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, Grenoble, France, [10]Nokia Bell Labs, Cambridge, United Kingdom, [11]TAU, LRI-CNRS–INRIA, Universite Paris-Saclay, France, [12]Hub France Intelligence Artificielle, Paris, France, [13]Nokia Bell Labs, Paris, France, [14]Inria, Lille, France

OPEN ACCESS

Edited by:
Bhavya Kailkhura,
United States Department of Energy (DOE), United States

Reviewed by:
Arndt Von Twickel,
Federal Office for Information Security, Germany
Jize Zhang,
Lawrence Livermore National Security, United States

****Correspondence:***
Remy Kusters
kusters.remy@gmail.com
Dusan Misevic
dusan.misevic@cri-paris.org

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Big Data

Received: *30 June 2020*
Accepted: *28 October 2020*
Published: *23 November 2020*

Citation:
Kusters R, Misevic D, Berry H, Cully A, Le Cunff Y, Dandoy L, Díaz-Rodríguez N, Ficher M, Grizou J, Othmani A, Palpanas T, Komorowski M, Loiseau P, Moulin Frier C, Nanini S, Quercia D, Sebag M, Soulié Fogelman F, Taleb S, Tupikina L, Sahu V, Vie J-J and Wehbi F (2020) Interdisciplinary Research in Artificial Intelligence: Challenges and Opportunities.
Front. Big Data 3:577974.
doi: 10.3389/fdata.2020.577974

The use of artificial intelligence (AI) in a variety of research fields is speeding up multiple digital revolutions, from shifting paradigms in healthcare, precision medicine and wearable sensing, to public services and education offered to the masses around the world, to future cities made optimally efficient by autonomous driving. When a revolution happens, the consequences are not obvious straight away, and to date, there is no uniformly adapted framework to guide AI research to ensure a sustainable societal transition. To answer this need, here we analyze three key challenges to interdisciplinary AI research, and deliver three broad conclusions: 1) future development of AI should not only impact other scientific domains but should also take inspiration and benefit from other fields of science, 2) AI research must be accompanied by decision explainability, dataset bias transparency as well as development of evaluation methodologies and creation of regulatory agencies to ensure responsibility, and 3) AI education should receive more attention, efforts and innovation from the educational and scientific communities. Our analysis is of interest not only to AI practitioners but also to other researchers and the general public as it offers ways to guide the emerging collaborations and interactions toward the most fruitful outcomes.

Keywords: artificial intelligence, interdisciplinary science, education, ethics, auditability, interpretability

INTRODUCTION

Artificial Intelligence (AI), which typically refers to the artificial creation of human-like intelligence that can learn, perceive and process information, is rapidly becoming a powerful tool for solving image recognition, document classification (Vapkin, 1995; LeCun et al., 2015) as well as for the advancement of interdisciplinary problems. It is often considered to be a powerful computational tool that can be applied to many complex problems which have not been successfully addressed so far. However, this is not a one way street, other fields such as neuroscience (Hassabis et al., 2017; Ullman, 2019), developmental psychology (Bennetot et al., 2020; Charisi et al., 2020), developmental robotics (Oudeyer, 2011; Moulin-Frier and Oudeyer, 2013; Doncieux et al., 2020) and evolutionary biology (Gobeyn et al., 2019) can inspire AI research itself, for example by suggesting novel ways to structure data (Timmis and Knight, 2002), or helping discover new algorithms, such as neural networks, which are inspired from the brain (Rosenblatt, 1958). Of course, combining AI with other fields is not without challenges. Like any time when fields synergize, barriers in communication arise, due to differences in terminologies, methods, cultures, and interests. How to bridge such gaps remains an open question, but having a solid education in both machine learning and the field of interest is clearly imperative. An example of cross-pollination interdisciplinary program showing the success of these approaches is not utopic is Frontier Development Lab, a cooperative agreement between NASA, the Seti Institute, and ESA set up to work on AI research for space science, exploration and all humankind (Frontier Development Lab). Besides multidisciplinarity, advocating for ethics and diversity (Agarwal et al., 2020) is a must to account for biased models (Hendricks et al., 2018; Denton et al., 2019) and avoid stereotypes being perpetuated by AI systems (Gebru, 2019). For instance, interdisciplinary approaches, e.g., including art and science, as well as ensuring minorities are well represented among both the users and the evaluators of the latest eXplainable AI techniques (Arrieta et al., 2020), can make AI more accessible and inclusive to otherwise unreachable communities.

While the AI revolution in research, healthcare and industry is presently happening at full speed, its long term impact on society will not reveal itself straight away. In research and healthcare, this might lead to blindly applying AI methods to problems for which, to date, the technology is not ready [e.g., IBM's Watson for oncology (Strickland, 2019)], and to ethically questionable applications [e.g., predicting sexual orientations from people's faces (Wang and Kosinski, 2017), using facial recognition in law enforcement or for commercial use (Clearview)]. AI can be used as a tool to improve data privacy (e.g., for deidentification, www.d-id.com) or for threat identification, but it is more often seen as itself being a threat to IT systems (Berghoff et al., 2020), e.g., in the cases of biometric security and privacy (Jiang et al., 2017). AI can be a target of attacks with vulnerabilities qualitatively new to AI systems [e.g. adversarial attacks and poisoning attacks (Qiu et al. , 2019)] as well as a powerful new tool used by the attackers (Dixon and Eagan, 2019). In industry, AI chatbots ended up being racist, reflecting the training data that was presented to the algorithm, recruitment software ended up being gender-biased; and risk assessment tools developed by a US contractor sent innocent people to jail (Dressel and Farid, 2018). A more careful consideration of the impact of AI is clearly needed by following global and local ethics guidelines for trustworthy (Smuha, 2019) and responsible AI (Arrieta et al., 2020).

While a large number of industries have seen a potential in this technology and invested colossal amounts of money to incorporate AI solutions in their businesses, predictions made by AI algorithms can be frightening and without a proper educational framework, lead to a societal distrust. In this opinion paper we put forward three research topics that we believe AI research should accentuate on,

(1) How can an interdisciplinary approach towards AI benefit from and contribute to the AI revolution? While AI is already used in various scientific fields, it should go beyond solely predicting outcomes towards conducting exploratory analysis and finding new patterns in complex systems. Additionally, in the future development of AI, the reverse direction should also be considered, namely investigating ways in which AI can take inspiration and can benefit from other fields of science.
(2) How could regulatory agencies help correct existing data biases and discriminations induced by AI? To ensure this, AI research must be accompanied by decision explainability and dataset and algorithm bias analysis as well as creation of regulatory agencies and development of evaluation methodologies and tools. In all cases, AI research should guarantee privacy as well as economical and ecological sustainability of the data and algorithms based on it.
(3) How can we manage the impact of this AI revolution once AI tools are deployed in the real world, particularly how to ensure trust of the scientific peers and the general public? This includes establishing public trust in AI through education, explainable solutions, and regulation.

By considering these three aspects, interdisciplinary research will go beyond the considerations of individual disciplines to take broader and more thoughtful views of the promised digital revolutions. Our recommendations are a result of in-person discussions within a diverse group of researchers, educators, and students, during a 3-day thematic workshop, which has been collectively written and edited during and after the meeting. While not comprehensive, we believe they capture a broad range of opinions from multiple stakeholders and synthesize a feasible way forward.

PART I: ARTIFICIAL INTELLIGENCE AND INTERDISCIPLINARY RESEARCH

The relationship between AI and interdisciplinary research must be considered as a two-way street. While one direction may be more well known (applying AI to other fields), here we consider both directions: 1) from AI to other fields and 2) from other fields to AI. Then we argue that applying knowledge from other fields to AI development is equally important in order to move forward and to achieve the full potential of the AI revolution.

From Artificial Intelligence to Other Fields

Using AI to make predictions or decisions in e.g. quantitative science, healthcare, biology, economy and finance has been extensively, and possibly excessively done over the past several years. While the application of AI to these domains remains an active area of research, we believe that the biggest challenge for the future of AI lies ahead. Rather than just predicting or making decisions, AI solutions should be developed to conduct exploratory analyses, i.e., to find new, interesting patterns in complex systems or facilitate scientific discovery (Raghu and Schmidt, 2020). Specific cases where this direction has already been explored include e.g., drug discovery (Vamathevan et al., 2019), the discovery of new material (Butler et al., 2018), symbolic math (Lample and Charton, 2019; Stanley et al., 2019) or the discovery of new physical laws (Both et al., 2019; Iten et al., 2020; Udrescu and Tegmark, 2020). *Will AI succeed in assisting humans in the discovery of new scientific knowledge? If so, in which domain will it happen first? How do we speed up the development of new AI methods that could reach such goals?* These are some questions that should inspire and drive the applications of AI in other fields.

Another possible approach consists of using AI models as experimental "guinea pigs" for hypothesis testing. In the domain of neuroscience, one standard methodology consists of analyzing which AI model is best at predicting behavioral data (from animals or humans) in order to support or inform hypotheses on the structure and on the function of biological cognitive systems (Gauthier and Levy, 2019). In that case, the process of training the AI-agent is an experiment in itself since the intrinsic interest does not lie in the performance of the underlying algorithm per se but instead in its ability to explain cognitive functions. *Can we create an AI algorithm that will replace all stages of scientific process, from coming up with questions, generating the data, to analysis and interpretation of results?* Such automated discovery is considered as the ultimate goal by some experts, but so far remains out of reach (Bohannon, 2017).

From Other Fields to Artificial Intelligence

Whereas AI approaches are readily impacting many scientific fields, those approaches also continue to benefit from insights from fields such as neuroscience (Hassabis et al., 2017; Samek et al., 2019; Ullman, 2019; Parde et al., 2020), for example the similarities between machine and human-like facial recognition (Grossman et al., 2019) and the use of the face space concept in deep convolutional neural networks (O'Toole et al., 2018; Parde et al., 2020). Other fields impacting AI research include evolutionary biology (Gobeyn et al., 2019) and even quantum mechanics (Biamonte et al., 2017). One of the biggest successes of integrating insights from other fields in modern day AI, the perceptron, became the prelude to the modern neural networks of today (Rosenblatt, 1958). Perceptrons and neural networks can be considered analogous to a highly reduced model of cortical neural circuitry. Other examples are algorithms such as reinforcement learning which drew inspiration from principles of developmental psychology from the 50s (Skinner, 2019) and have been influencing the field of developmental robotics (Cangelosi and Schlesinger, 2015) since the 2010s. Further illustration of this cross-fertilization can be seen in bio-inspired approaches, where principles from natural systems are used to design better AI, e.g., neuroevolution algorithms that evolve neural networks through evolutionary algorithms (Floreano et al., 2008). Finally, the rise of quantum computers and quantum-like algorithms could further expand the hardware and algorithmic toolbox for AI (Biamonte et al., 2017). Despite these important advances in the last decade, AI systems are still far from being comparable to human intelligence (and to some extent to animal intelligence), and several questions remain open. For instance, *how can an AI system learn and generalize while being exposed to only a small amount of data? How to bridge the gap between low-level neural mechanisms and higher-level symbolic reasoning?*

While AI algorithms are still mostly focused on the modeling of purely cognitive processes (e.g., learning, abstraction, planning...), a complementary approach could consider intelligence as an emergent property of cognitive systems through their coupling with environmental, morphological, sensorimotor, developmental, social, cultural and evolutionary processes. In this case, the highly complex dynamic of the ecological environment is driving the cognitive agents to continuously improve in an ever-changing world, in order to survive and to reproduce (Pfeifer and Bongard, 2006; Kaplan and Oudeyer, 2009). This approach draws inspiration from multiple scientific fields such as evolutionary biology, developmental science, anthropology or behavioral ecology. Recent advances in reinforcement learning have made a few steps in this direction. Agents capable of autonomously splitting a complex task into simpler ones (auto-curriculum) can evolve more complex behaviors through coadaptation in mixed cooperative-competitive environments (Lowe et al., 2017). In parallel, progress has also been made in curiosity-driven multi-goal reinforcement learning algorithms, enabling agents to autonomously discover and learn multiple tasks of increasing complexity (Doncieux et al., 2018). Finally, recent work has proposed to jointly generate increasingly complex and diverse learning environments and their solutions as a way to achieve open-ended learning (Doncieux et al., 2018). One related research direction are studies of systems that sequentially and continually learn (Lesort et al., 2020) in a lifelong setting, i.e., continual learning without experiencing the well known phenomenon of catastrophic forgetting (Traoré et al., 2019). When combined, this research puts forward the following questions: *How can we leverage recent advances that situate AI agents within realistic ecological systems? How does the dynamic of such systems drive the acquisition of increasingly complex skills?*

PART II: ARTIFICIAL INTELLIGENCE AND SOCIETY

The rise of AI in interdisciplinary science brings along significant challenges. From biased hiring algorithms, to deep fakes, the field has struggled to accommodate a rapid growth and an increasing complexity of algorithms (Chesney and Citron, 2019). Moreover, the lack of explainability (Arrieta et al., 2020) has slowed down its impact in areas such as quantitative research and prevents the community to further develop reproducible and deterministic protocols. Here we propose methodologies and rules to mitigate the inherent risks that arise from applying complex and non-deterministic AI methods. In particular we discuss how general scientific methodologies can be adapted for AI research and how auditability, interpretability and environmental neutrality of results can be ensured.

Adapting the Scientific Method to Artificial Intelligence-Driven Research

To ensure that AI solutions perform as we intended, it is important to clearly formulate the problem and to state the underlying hypothesis of the model. By matching formal problem expression/definitions to laws (intentions), functional and technical specifications, we ensure that the project has a well established scope and a path towards achieving this goal. These specifications have been set forward by the GDPR (General Data Protection Regulation) that published a self assessment template guiding scientists and practitioners to prepare their AI projects for society (Bieker et al., 2016). In short, products and services resulting from AI decision making must clearly define their applicability and limitations. Note that this differs from problem definition since it involves explicitly stating how the algorithm will address part or all of the original problem. The developers have to explicitly detail how they handle extreme cases and show that security of the user is ensured. It should be mandatory for the owner and user of the data to clearly and transparently state the known biases expressed by the dataset (similar to the way the secondary effects of medicines are clearly stated on the medication guide). While some of these are already addressed by the GDPR in the EU, similar regulation and standards are needed globally. An alternative, complementary approach would be to rely on the classical scientific method practices developed over the centuries. Relying on observation, hypothesis formulation, experimentation (Rawal et al., 2020) and evaluation allows us to understand causal relationships and promotes rigorous practices. AI would certainly benefit from explicitly integrating these practices into its research ecosystem (Forde and Paganini, 2019).

Biases and Ethical Standards in Artificial Intelligence

To control the functioning of AI algorithms and their potential inherent biases, clear, transparent and interpretable methodologies and best practices are required. Trustworthiness of AI-driven projects can be ensured by, for example, using open protocols of the algorithms functionality, introducing traceability (logs, model versioning, data used and transformations done on data) or the pre-definition of insurance datasets. In transversal domains such as software development, tools have been devised to prevent mistakes and model deterioration over time (such as automated unit tests). Establishing similar standards for AI would force data scientists to design ways to detect and eliminate biases, ultimately making sure that the algorithm is behaving as intended. If ethical standards can be encoded in the algorithm, then regulation can be imposed on the optimized objectives of AI models (Jobin et al., 2019).

Auditability and Interpretability

The goal of AI should be to improve human condition and not further aggravate either existing inequalities (Gebru, 2019) or environmental issues in our societies. The AI service and product developers are likely to be at the center of this challenge - they are the ones that can directly prevent errors and biases in input data or future applications. They present a priori knowledge that can lead to or prevent misuse (conscious or unconscious). It is tempting to extensively employ libraries and "ready-to-use" code samples, as these make the production process faster and easier. However, especially when used by non-experts, the key features of AI models, e.g., data recasting, could easily be implemented incorrectly. The secondary users of AI tools must be able to measure the biases of their input data and obtained results, which can be done only if they are both aware of potential problems and if they have the necessary tools readily available.

As with any software, failures and mistakes will inevitably arise and a system has to be in place to assess how AI tools and services behave not only during development but also "in production." The combination of decision logs and model versioning can allow us to verify and ensure the product outcomes are the ones intended. Here the question of independent authorities comes in order to regularly audit the AI products around us. Companies and AI product developers must be capable of "opening the black box" and clearly exposing the monitoring they perform over an algorithm. Opening the black box has already been set as an important goal in AI research (Castelvecchi, 2016), even if not all experts agree that this is necessary (Holm, 2019). It includes not only making the currently used model transparent, but more importantly being able to explain how it was designed, and examining its past states and decisions. For example, developers must track data drifting and deploy policies preventing an algorithm to produce unintended outcomes. So far, this has been left to good practices of individual developers, but we can envision construction of an authority in charge of auditing AI products regularly. One proposed approach has been to impose *Adversarial Fairness* during training or on the output (Adel et al., 2019). Independently of a particular way to ensure auditability and interpretability, the process should be co-designed not only by AI practitioners but all stakeholders, including the general public, following open science principles (Greshake Tzovaras et al., 2019). Auditability and interoperability considerations complement and extend the more obvious and direct requirements of robustness, security and data privacy in AI.

Finally, as for any technology, the usefulness of AI will have to be assessed against its environmental impact. In particular, life cycle assessment of AI solutions should be systematic. Here also, auditing by independent authorities could be a way to enforce environmental neutrality (Schwartz et al., 2019).

Education Through and About Artificial Intelligence Technologies

Besides impacting research and industry directly, AI is transforming the job market at a rapid pace. It is expected that approximately 80% of the population will be affected by these technological advancements in the near future (HolonIQ). Highly complex jobs (e.g., the medical, juridical or educational domains) will be redefined, some simpler, repetitive tasks will be replaced or significantly assisted by AI and new jobs will appear in the coming decades. For instance, budget readjustment and reeducation of people who lose their jobs, towards a clean energy shift, with only about 30% coming from governments (which amounts to less than 10% of the funds committed to coronavirus economic relief), could positively shift climate change (Florini, 2011). However, workers of these different fields received little to no formal education on AI, and more initiatives on sustainable AI (such as EarthDNA ambassadors or TeachSDG Ambassadors) are needed. Therefore, the AI transformation should come along with a transformation in education where educational and training programs will have to be adapted to these different existing professions.

The transformation in education can be implemented on four different levels: academic institutions, companies and governments. Academic institutions should not only prepare AI experts by providing in-depth training to move forward AI research but also focus on interdisciplinarity and attract diversity in AI. Three main axes for AI education should be: 1) high level AI experts who can train future generations 2) AI practitioners who can raise public awareness in their research and (3), broader public that can be informed directly, leading to decrease in a priori distrust.

The end users and beneficiaries of AI services and products, as the most numerous part of the population, must play a central role in their development. It is they who should have the final say on what global use of AI technologies should be pursued. However, to do so, they must have a chance to learn the fundamental principles of AI. This is not fundamentally different from educating the general public about any scientific topic with a global societal impact, may it be medical (e.g., antibiotic resistance, vaccination) or environmental (e.g., climate change). Providing the information and training at scale is not a trivial task, due to at least two major issues: 1) the motivation of the general public and 2) the existence of appropriate educational tools. Various online resources are available targeting the general public, such as Elements of AI in Finland or *Objectif'IA* in France. Interestingly, in the case of AI, the problem itself could also be a part of a possible solution - we can envisage AI playing a central role in creating adaptive learning paths, individual-based learning programs addressing the needs and interests of each person affected by AI technology. Educational tools designed with AI can motivate each individual by providing relevant, personalized examples and do it at the necessary scale. Interactions between AI and education is yet another example of interdisciplinarity in AI (Oudeyer et al., 2016), which can directly benefit not only the two fields, education and AI, but society and productivity as a whole.

CONCLUSION

AI is currently ever present in science and society, and if the trend continues, it will play a central role in the education and jobs of tomorrow. It inevitably interacts with other fields of science and in this paper we examined ways in which those interactions can lead to synergistic outcomes. We focused our recommendations on mutual benefits that can be harnessed from these interactions and emphasized the important role of interdisciplinarity in this process. AI systems have complex life cycles, including data acquisition, training, testing and deployment, ultimately demanding an interdisciplinary approach to audit and evaluate the quality and safety of these AI products or services. Furthermore in Part II we focused on how AI practitioners can prevent biases through transparency, explainability, inclusiveness and how robustness, security and data privacy can and should be ensured. Finally we emphasize the importance of education for and through AI to allow the whole society to benefit from this AI transition. We offer recommendations from the broad community gathered around the workshop resulting in this paper, with the goal of contributing, motivating and informing the conversion between AI practitioners, other scientists, and the general public. In this way, we hope this paper is another step towards harnessing the full potential of AI for good, in all its scientific and societal aspects.

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

RK and DM wrote the initial draft and supervised the project. All the other authors contributed equally to the conceptualization, data curation, and investigation presented in this paper.

ACKNOWLEDGMENTS

Thanks to the Bettencourt Schueller Foundation long term partnership, the workshop that gave rise to this paper was partially supported by the funding from CRI Research Collaboratory. The authors would like to thank Ariel Lindner for valuable input on the manuscript and the discussions.

REFERENCES

Adel, T., Valera, I., Ghahramani, Z., and Weller, A. (2019). "One-network adversarial fairness," in Proceedings of the AAAI conference on artificial intelligence, Honolulu, HI, January 27–February 1, 2019 (Palo Alto, CA: AAAI Press), 33, 2412–2420. doi:10.1609/aaai.v33i01.33012412

Agarwal, P., Betancourt, A., Panagiotou, V., and Díaz-Rodríguez, N. (2020). Egoshots, an ego-vision life-logging dataset and semantic fidelity metric to evaluate diversity in image captioning models. ArXiv ArXiv200311743 [Preprint]. doi:10.1287/c25d5e39-1f9b-4c44-b511-a0ca0a20131b

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., et al. (2020). Explainable Artificial Intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion.* 58, 82–115. doi:10.1016/j.inffus.2019.12.012

Bennetot, A., Charisi, V., and Díaz-Rodríguez, N. (2020). "Should artificial agents ask for help in human-robot collaborative problem-solving?," in IEEE international conference on robotics and automation (ICRA 2020), Paris, France, May 31, 2020.

Berghoff, C., Neu, M., and von Twickel, A. (2020). Vulnerabilities of connectionist AI applications: evaluation and defense. *Front. Big Data.* 3, 213005576. doi:10.3389/fdata.2020.00023

Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., and Lloyd, S. (2017). Quantum machine learning. *Nature* 549, 195–202. doi:10.1038/nature23474

Bieker, F., Friedewald, M., Hansen, M., Obersteller, H., and Rost, M. (2016). "A process for data protection impact assessment under the european general data protection regulation," in 4th annual privacy forum, APF 2016, Frankfurt/Main, Germany, September 7–8, 2016, 21–37.

Bohannon, J. (2017). A new breed of scientist, with brains of silicon. *Sci. AAAS.* Available at: https://www.sciencemag.org/news/2017/07/new-breed-scientist-brains-silicon (Accessed June 25, 2020).

Both, G.-J., Choudhury, S., Sens, P., and Kusters, R. (2019). DeepMoD: deep learning for model discovery in noisy data. ArXiv ArXiv190409406 [Preprint].

Butler, K. T., Davies, D. W., Cartwright, H., Isayev, O., and Walsh, A. (2018). Machine learning for molecular and materials science. *Nature* 559, 547–555. doi:10.1038/s41586-018-0337-2

Cangelosi, A., and Schlesinger, M. (2015). *Developmental robotics: from babies to robots*. London, UK: MIT press.

Castelvecchi, D. (2016). Can we open the black box of AI?. *Nature* 538, 20. doi:10.1038/538020a

Charisi, V., Gomez, E., Mier, G., Merino, L., and Gomez, R. (2020). Child-Robot collaborative problem-solving and the importance of child's voluntary interaction: a developmental perspective. *Front. Robot. AI.* 7, 15. doi:10.3389/frobt.2020.00015

Chesney, B., and Citron, D. (2019). Deep fakes: a looming challenge for privacy, democracy, and national security. *Calif. Law Rev.* 107, 1753. doi:10.2139/SSRN.3213954

Clearview. Available at: https://clearview.ai/ (Accessed November 6, 2020).

Denton, E., Hutchinson, B., Mitchell, M., and Gebru, T. (2019). Detecting bias with generative counterfactual face attribute augmentation. ArXiv ArXiv190606439 [Preprint].

Dixon, W., and Eagan, N. (2019). 3 ways AI will change the nature of cyber attacks. Davos, Switzerland: World Economic Forum. Available at: https://www.weforum.org/agenda/2019/06/ai-is-powering-a-new-generation-ofcyberattack-its-also-our-best-defence (Accessed January 22, 2019).

Doncieux, S., Bredeche, N., Goff, L. L., Girard, B., Coninx, A., Sigaud, O., et al. (2020). DREAM architecture: a developmental approach to open-ended learning in robotics. ArXiv ArXiv200506223 [Preprint].

Doncieux, S., Filliat, D., Díaz-Rodríguez, N., Hospedales, T., Duro, R., Coninx, A., et al. (2018). Open-ended learning: a conceptual framework based on representational redescription. *Front. Neuro. rob.* 12, 59. doi:10.3389/fnbot.2018.00059

Dressel, J., and Farid, H. (2018). The accuracy, fairness, and limits of predicting recidivism. *Sci. Adv.* 4, eaao5580. doi:10.1126/sciadv.aao5580

Floreano, D., Dürr, P., and Mattiussi, C. (2008). Neuroevolution: from architectures to learning. *Evol. Intel.* 1, 47–62. doi:10.1007/s12065-007-0002-4

Florini, A. (2011). The International Energy Agency in global energy governance. *Glob. Policy.* 2, 40–50. doi:10.1111/j.1758-5899.2011.00120.x

Forde, J. Z., and Paganini, M. (2019). The scientific method in the science of machine learning. ArXiv ArXiv190410922 [Preprint]. Available at: http://arxiv.org/abs/1904.10922 (Accessed June 22, 2020).

Frontier Development Lab. Available at: https://frontierdevelopmentlab.org/.

Gauthier, J., and Levy, R. (2019). Linking artificial and human neural representations of language. ArXiv ArXiv191001244 [Preprint] (Accessed November 6, 2020).

Gebru, T. (2019). Oxford handbook on AI ethics book chapter on race and gender. ArXiv ArXiv190806165 [Preprint].

Gobeyn, S., Mouton, A. M., Cord, A. F., Kaim, A., Volk, M., and Goethals, P. L. M. (2019). Evolutionary algorithms for species distribution modelling: a review in the context of machine learning. *Ecol. Model.* 392, 179–195. doi:10.1016/j.ecolmodel.2018.11.013

Greshake Tzovaras, B., Angrist, M., Arvai, K., Dulaney, M., Estrada-Galiñanes, V., Gunderson, B., et al. (2019). Open Humans: a platform for participant-centered research and personal data exploration. *GigaScience* 8, giz076. doi:10.1093/gigascience/giz076

Grossman, S., Gaziv, G., Yeagle, E. M., Harel, M., Mégevand, P., Groppe, D. M., et al. (2019). Convergent evolution of face spaces across human face-selective neuronal groups and deep convolutional networks. *Nat. Commun.* 10, 4934. doi:10.1038/s41467-019-12623-6

Hassabis, D., Kumaran, D., Summerfield, C., and Botvinick, M. (2017). Neuroscience-inspired artificial intelligence. *Neuron* 95, 245–258. doi:10.1016/j.neuron.2017.06.011

Hendricks, L. A., Burns, K., Saenko, K., Darrell, T., and Rohrbach, A. (2018). "Women also snowboard: overcoming bias in captioning models," in European conference on computer vision, Munich, Germany, October 7, 2018, 793–811.

Holm, E. A. (2019). In defense of the black box. *Science* 364, 26–27. doi:10.1126/science.aax0162

HolonIQ. Available at: https://www.holoniq.com/research/ (Accessed November 6, 2020).

Iten, R., Metger, T., Wilming, H., del Rio, L., and Renner, R. (2020). Discovering physical concepts with neural networks. *Phys. Rev. Lett.* 124, 010508. doi:10.1103/PhysRevLett.124.010508

R. Jiang, S. Al-maadeed, A. Bouridane, D. Crookes, and A. Beghdadi (Editors) (2017). *Biometric security and privacy: opportunities and challenges in the big data era*. Cham, Switzerland: Springer International Publishing.

Jobin, A., Ienca, M., and Vayena, E. (2019). The global landscape of AI ethics guidelines. *Nat. Mach. Intell.* 1, 389–399. doi:10.1038/s42256-019-0088-2

Kaplan, F., and Oudeyer, P.-Y. (2009). Stable kernels and fluid body envelopes. Available at: https://hal.inria.fr/inria-00348476 (Accessed October 11, 2012).

Lample, G., and Charton, F. (2019). Deep learning for symbolic mathematics. ArXiv ArXiv191201412 [Preprint].

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature* 521, 436–444. doi:10.1038/nature14539

Lesort, T., Lomonaco, V., Stoian, A., Maltoni, D., Filliat, D., and Díaz-Rodríguez, N. (2020). Continual learning for robotics: definition, framework, learning strategies, opportunities and challenges. *Inf. Fusion.* 58, 52–68. doi:10.1016/j.inffus.2019.12.004

Lowe, R., Wu, Y., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. (2017). "Multi-agent actor-critic for mixed cooperative-competitive environments," in Advances in neural information processing systems 30, Long Beach, CA, December 4–9, 2017, Editors I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, et al., 6379–6390. Available at: http://papers.nips.cc/paper/7217-multi-agent-actor-critic-for-mixed-cooperative-competitive-environments.pdf (Accessed June 25, 2020).

Moulin-Frier, C., and Oudeyer, P.-Y. (2013). "Exploration strategies in developmental robotics: a unified probabilistic framework," in 2013 IEEE third joint international conference on development and learning and epigenetic robotics (ICDL), Osaka, Japan, August 18–22, 2013, 1–6.

Oudeyer, P.-Y. (2011). *Developmental robotics*. New York, NY: Springer.

Oudeyer, P.-Y., Gottlieb, J., and Lopes, M. (2016). Intrinsic motivation, curiosity, and learning: theory and applications in educational technologies. *Prog. Brain Res.* 229, 257–284. doi:10.1016/bs.pbr.2016.05.005

O'Toole, A. J., Castillo, C. D., Parde, C. J., Hill, M. Q., and Chellappa, R. (2018). Face space representations in deep convolutional neural networks. *Trends Cognit. Sci.* 22, 794–809. doi:10.1016/j.tics.2018.06.006

Parde, C. J., Colón, Y. I., Hill, M. Q., Castillo, C. D., Dhar, P., and O'Toole, A. J. (2020). Single unit status in deep convolutional neural network codes for face identification: sparseness redefined. ArXiv ArXiv200206274 [Preprint]. Available at: http://arxiv.org/abs/2002.06274 (Accessed August 12, 2020).

Pfeifer, R., and Bongard, J. (2006). *How the body shapes the way we think: a new view of intelligence*. Cambridge, MA: MIT press.

Qiu, S., Liu, Q., Zhou, S., and Wu, C. (2019). Review of artificial intelligence adversarial attack and defense technologies. *Appl. Sci.* 9, 909. doi:10.3390/app9050909

Raghu, M., and Schmidt, E. (2020). A survey of deep learning for scientific discovery. ArXiv ArXiv200311755 [Preprint].

Rawal, A., Lehman, J., Such, F. P., Clune, J., and Stanley, K. O. (2020). Synthetic petri dish: a novel surrogate model for rapid architecture search. ArXiv ArXiv200513092 [Preprint].

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.* 65, 386–408. doi:10.1037/h0042519

Samek, W., Montavon, G., Vedaldi, A., Hansen, L. K., and Müller, K.-R. (2019). *Explainable AI: interpreting, explaining and visualizing deep learning*. Cham, Switzerland: Springer Nature.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. (2019). Green AI. ArXiv ArXiv190710597 [Preprint].

Skinner, B. F. (2019). *The behavior of organisms: an experimental analysis*. Cambridge, MA: BF Skinner Foundation.

Smuha, N. A. (2019). The eu approach to ethics guidelines for trustworthy artificial intelligence. *CRi-Comput. Law Rev. Int.* 20, 2194–4164. doi:10.9785/cri-2019-200402

Stanley, K. O., Clune, J., Lehman, J., and Miikkulainen, R. (2019). Designing neural networks through neuroevolution. *Nat. Mach. Intell.* 1, 24–35. doi:10.1038/s42256-018-0006-z

Strickland, E. (2019). IBM Watson, heal thyself: how IBM overpromised and underdelivered on AI health care. *IEEE Spectr.* 56, 24–31. doi:10.1109/mspec.2019.8678513

Timmis, J., and Knight, T. (2002). Artificial immune systems. *Heuristic Approach.*, 209–230. doi:10.4018/978-1-930708-25-9.ch011

Traoré, R., Caselles-Dupré, H., Lesort, T., Sun, T., Cai, G., Díaz-Rodríguez, N., et al. (2019). *DISCORL: continual reinforcement learning via policy distillation*. ArXiv ArXiv190705855 [Preprint].

Udrescu, S.-M., and Tegmark, M. (2020). AI Feynman: a physics-inspired method for symbolic regression. *Sci. Adv.* 6, eaay2631. doi:10.1126/sciadv.aay2631

Ullman, S. (2019). Using neuroscience to develop artificial intelligence. *Science* 363, 692–693. doi:10.1126/science.aau6595

Vamathevan, J., Clark, D., Czodrowski, P., Dunham, I., Ferran, E., Lee, G., et al. (2019). Applications of machine learning in drug discovery and development. *Nat. Rev. Drug Discov.* 18, 463–477. doi:10.1038/s41573-019-0024-5

Vapkin, V. N. (1995). The nature of statistical learning. Theory. Available at: https://ci.nii.ac.jp/naid/10020951890/ (Accessed June 22, 2020).

Wang, Y., and Kosinski, M. (2017). Deep neural networks are more accurate than humans at detecting sexual orientation from facial images. *J Pers. Soc Psychol.* 114, 246–257. doi:10.1109/ijcnn.2017.7965846

Conflict of Interest: Authors LT and DQ are employed by the company Nokia Bell Labs.

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.

frontiers in Big Data

ORIGINAL RESEARCH
published: 18 February 2021
doi: 10.3389/fdata.2020.590296

Subgroup Invariant Perturbation for Unbiased Pre-Trained Model Prediction

Puspita Majumdar[1], *Saheb Chhabra*[1], *Richa Singh*[2]* *and Mayank Vatsa*[2]

[1]*Department of Computer Science and Engineering, Indraprastha Institute of Information Technology, New Delhi, India,*
[2]*Department of Computer Science and Engineering, Indian Institute of Technology Jodhpur, Rajasthan, India*

Modern deep learning systems have achieved unparalleled success and several applications have significantly benefited due to these technological advancements. However, these systems have also shown vulnerabilities with strong implications on the fairness and trustability of such systems. Among these vulnerabilities, bias has been an *Achilles' heel problem*. Many applications such as face recognition and language translation have shown high levels of bias in the systems towards particular demographic sub-groups. Unbalanced representation of these sub-groups in the training data is one of the primary reasons of biased behavior. To address this important challenge, we propose a two-fold contribution: a bias estimation metric termed as *Precise Subgroup Equivalence* to jointly measure the bias in model prediction and the overall model performance. Secondly, we propose a novel bias mitigation algorithm which is inspired from adversarial perturbation and uses the PSE metric. The mitigation algorithm learns a single uniform perturbation termed as *Subgroup Invariant Perturbation* which is added to the input dataset to generate a transformed dataset. The transformed dataset, when given as input to the pre-trained model reduces the bias in model prediction. Multiple experiments performed on four publicly available face datasets showcase the effectiveness of the proposed algorithm for race and gender prediction.

Keywords: Fairness, trustability, bias estimation, bias mitigation, subgroup invariant perturbation, gender classification, race classification

OPEN ACCESS

Edited by:
Pin-Yu Chen,
IBM Research, United States

Reviewed by:
Chulin Xie,
University of Illinois at Urbana-Champaign, United States
Ren Wang,
University of Michigan, United States

****Correspondence:***
Richa Singh
richa@iitj.ac.in

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Big Data

Received: *31 July 2020*
Accepted: *20 November 2020*
Published: *18 February 2021*

Citation:
Majumdar P, Chhabra S, Singh R and Vatsa M (2021) Subgroup Invariant Perturbation for Unbiased Pre-Trained Model Prediction. Front. Big Data 3:590296. doi: 10.3389/fdata.2020.590296

1. INTRODUCTION

Increasing use of artificial intelligence (AI) and machine learning (ML) for automation coupled with instances of biased predictions has motivated and mandated researchers across the globe to pursue designing dependable AI systems. Out of the several attributes of dependability in AI systems such as interpretability, explainability, robustness, bias, and fairness (Mehrabi et al., 2019; Drozdowski et al., 2020; Ntoutsi et al., 2020), this research is focused towards bias and fairness.

Face analysis tasks such as face detection, face recognition, expression analysis, age and gender prediction are some of the AI applications in which several instances of biased or unfair predictions have been observed. For instance, Buolamwini and Gebru (2018) have shown that commercial gender classifiers perform better for lighter skin males while giving poor performance for darker skin females. Other instances include false identification of 28 members (specifically people of color) of the US Congress as criminals by Amazon's facial recognition tool (Paolini-Subramanya, 2018). Nagpal et al. (2019) analyzed several pre-trained face recognition models to determine where and how the bias manifests in the deep neural networks. In light of these incidents, while some corporate and government organizations have decided to minimize or ban the development or usage of automated face analysis systems (Conger et al., 2019), several others are continuing the deployment and usage. Therefore, it is of paramount importance that we design mechanisms to improve the trustability and dependability of these systems. To address the challenges related to biased predictions of AI systems, researchers are broadly pursuing three directions: understanding bias, mitigating bias, and accounting for bias (Ntoutsi et al., 2020). Understanding bias involves realizing the source of bias along with estimating it (Buolamwini and Gebru, 2018; Celis and Rao, 2019; Nagpal et al., 2019; Radford and Joseph, 2020) whereas mitigation strategies involve designing algorithms that address bias (Creager et al., 2019; Nagpal et al., 2020).

In the literature, it has been demonstrated that if the training data used for learning the models is not balanced in terms of demographic subgroups, for instance, *male* and *female* are two different subgroups of *gender*, then there can be significant differences in the classification performance of pre-trained models observed on subgroups (Barocas and Selbst, 2016). Recent instances of biased predictions can be safely attributed to this observation as the training data required for deep learning models is often collected from the Internet using convenience sampling, which inherently leads to disparate proportions of data across subgroups. Models trained on historically biased datasets lead to biased results. Therefore, researchers have proposed several algorithms to mitigate the effect of bias on model prediction (Alvi et al., 2018; Gong et al., 2019). However, there is generally a trade-off between fairness and model performance (Du et al., 2019; Li and Vasconcelos, 2019). Removal of bias may affect the overall model performance while a high performing model may affect the performance of the under-represented subgroup. Therefore, it is important to 1) measure the trade-off between the effect of bias and the model performance through a unified metric and 2) mitigate the effect of bias without affecting the model performance. A solution to the problem is to re-train the models with large datasets having equal distribution of samples across different subgroups. However, in a real-world scenario, collecting such diverse datasets is not a trivial task. Also, re-training the models require updating millions of parameters and is computationally expensive.

This research focuses on *estimating* the trade-off between the effect of bias and the model performance and *mitigating* the influence of demographic subgroup bias on pre-trained model prediction to improve the model performance. Existing metrics such as Disparate Impact, Average False Rate, and Degree of Bias provide information of only bias or error rates, but they do not provide the complete information. The first contribution of this research is a unified bias metric, termed as Precise Subgroup Equivalence (PSE) which provides a joint estimate of bias in model prediction and the overall model performance. The second contribution is to *mitigate* the influence of demographic subgroup bias on pre-trained model prediction to improve the model performance. We propose a novel algorithm based on adversarial perturbation for bias mitigation. In general, adversarial perturbation utilizes the vulnerability of deep models towards small changes in the input to reduce the confidence of model prediction. In this research, we have used this concept to reduce the effect of bias on model prediction. To the best of our knowledge, this is the first time that adversarial perturbation is used for bias mitigation. The proposed algorithm utilizes the model prediction to learn a single uniform Subgroup Invariant Perturbation (SIP) for a given dataset. SIP is added to the input dataset to generate a transformed dataset, which, when given as an input to the model, produces unbiased outcomes and improves the overall model performance. **Figure 1** shows a visual illustration of the proposed algorithm for bias mitigation using SIP. The proposed algorithm is used to mitigate the impact of demographic subgroup bias in race and gender model predictions.

The effectiveness of the algorithm is demonstrated under two scenarios: 1) *independent demographic subgroup analysis* and 2) *inter-sectional demographic subgroup analysis* on multiple datasets to showcase enhanced performance and reduced effect of bias on model prediction. The results show that PSE provides a unified score of both error and disparity in subgroups which is addressed using the proposed algorithm. Further, since the number of learned parameters is equal to the size of the input image, the proposed algorithm is observed to be computationally efficient as well.

2. RELATED WORK

Recent years have observed significant increase in the research on different aspects of bias and fairness in AI systems. Existing literature can be grouped into three broad categories: 1) Understanding and Estimating Bias, 2) Bias Mitigation Algorithms, and 3) Fairness Metrics.

Understanding and Estimating Bias: Researchers have focused on understanding the presence of bias in the prediction of commercial-off-the-shelf systems (COTS) and pre-trained deep models. Buolamwini and Gebru (2018) evaluated commercial gender classifiers from Microsoft, IBM, and Face ++ on four categories based on the skin type, namely, darker males, darker females, lighter males, and lighter females. It was found that the classifiers performed best for males with lighter skin tone and least for females with darker skin tone. Nagpal et al. (2019) provided an analysis of bias in deep face recognition models. They have shown that deep models encode race and age-specific features that lead to biased discrimination. According to various studies, the training data distribution has a huge impact on the model's performance (Torralba and Efros, 2011; Bolukbasi et al., 2016). Models trained on imbalanced datasets lead to biased outputs. Therefore, different data re-sampling techniques have been proposed by the researchers to balance the training data distribution. This is done either by over-sampling the minority class (Mullick et al., 2019) or under-sampling the majority class (Drummond et al., 2003). However, a recent study has shown that even models trained with balanced datasets amplify bias (Wang et al., 2019). It is shown that the learned models amplify

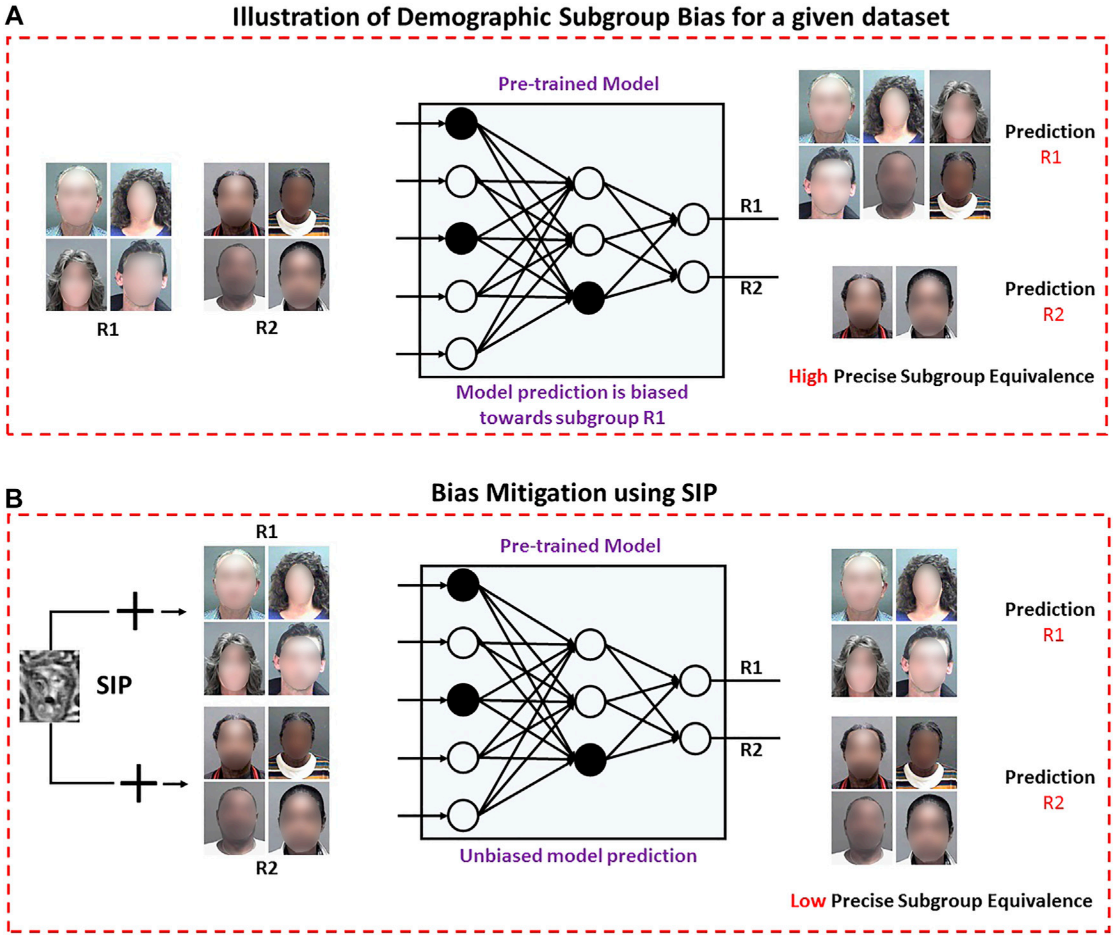

A
Illustration of Demographic Subgroup Bias for a given dataset
R1
R2
Pre-trained Model
R1
R2
Model prediction is biased towards subgroup R1
Prediction R1
Prediction R2
High Precise Subgroup Equivalence
B
Bias Mitigation using SIP
SIP
R1
R2
Pre-trained Model
R1
R2
Unbiased model prediction
Prediction R1
Prediction R2
Low Precise Subgroup Equivalence

FIGURE 1 | Effect of demographic subgroup bias on pre-trained model prediction. **(A)** Pre-trained model prediction is biased towards subgroup *R1*. **(B)** Bias mitigation using SIP to achieve equal performance across *R1* and *R2* (best viewed in color).

the association between labels and gender, which in turn leads to biased discrimination.

Bias Mitigation: Mitigation algorithms can either be applied as a pre-processing step or in-processing, or post-processing. Different algorithms have been proposed to mitigate the effect of bias. Ryu et al. (2017) addressed the problem of the performance gap in different subgroups of race and gender attributes. They hypothesized that faces look different across different genders and races, and proposed InclusiveNet which learns the demographic information prior to attribute detection. Dwork et al. (2018) proposed decoupled classifiers to increase fairness and accuracy in classification systems. The decoupled classifiers learn a separate classifier for sensitive attributes and can be used with any black-box network. Das et al. (2018) proposed a Multi-Task Convolution Neural Network (MTCNN) to classify gender, age, and ethnicity attributes and minimized the effect of bias by utilizing disjoint features of fully connected layers of a deep Convolution Neural Network (CNN). Alvi et al. (2018) proposed a joint learning and unlearning framework for mitigating bias in CNN models for gender, age, race, and pose classification. A disentangled representation learning technique is presented to obtain flexibly fair features by Creager et al. (2019). Kim et al. (2019) proposed a regularization algorithm to unlearn the bias information. Recently, Nagpal et al. (2020) proposed a filter drop technique for learning unbiased representations. Results are demonstrated for gender prediction across different ethnicity groups.

Apart from bias mitigation in attribute prediction, researchers have also focused on mitigating bias in face recognition. Gong et al. (2019) addressed the problem of bias in face recognition systems and proposed a debiasing adversarial network. The proposed network learns unbiased representation for both identity and demographic attributes. Huang et al. (2019) investigated the problem of deep imbalanced learning in the context of deep representation learning for attribute prediction and face recognition. They proposed Cluster-based Large Margin Local Embedding (CLMLE) method, which maintains inter-cluster margin among the same and different classes. Wang and Deng (2019) proposed a reinforcement learning-based race balance network (RL-RBN) to mitigate racial bias. Singh et al. (2020) provided a review of techniques related to bias in face recognition.

Fairness Metrics: To measure the fairness of deep models, different metrics have been proposed in the literature.

Statistical Parity (SP) (Calders and Verwer, 2010): It is one of the widely used fairness metrics. It suggests that a model gives unbiased output if the prediction is independent of the demographic group such as *race*, *gender*, and *religion*. Deviation from statistical parity is measured as the ratio of the probability of a positive classification for both subgroups of a demographic group. It is termed as Disparate Impact (DI) (Feldman et al., 2015) and computed as:

$$DI = \frac{P\left(\widehat{Y} = 1 \middle| D = 0\right)}{P\left(\widehat{Y} = 1 \middle| D = 1\right)} \tag{1}$$

where, D represents the demographic group, and $\widehat{Y}$ represents the predicted decision or class. A lower value of DI indicates higher bias in the model prediction.

Degree of Bias (DoB) (Gong et al., 2019): It is defined as the standard deviation of *Classification Accuracy* (*CAcc*) across different subgroups of a demographic group. Mathematically, it is represented as:

$$DoB = std\left(CAcc_{D_j}\right) \quad \forall j \tag{2}$$

where, D_j represents a subgroup of a demographic group D. High performance gap across different subgroups will result in high DoB, which in turn implies bias in the model prediction.

3. MATERIALS AND METHODS

The following subsections discuss the proposed metric, estimation of bias in model prediction, and bias mitigation using Subgroup Invariant Perturbation (SIP). There are two different scenarios for bias estimation and mitigation: 1) *independent demographic subgroup analysis* and 2) *intersectional demographic subgroup analysis*. In the first scenario, bias estimation/mitigation is performed across the subgroups of a demographic group. For example, bias estimation/mitigation is performed across the subgroups of *gender*. In the second scenario, bias estimation/mitigation is performed across the intersection of different demographic groups. For example, bias estimation/mitigation is performed across the intersectional subgroups of *race* and *gender*.

3.1. Proposed Metric: Precise Subgroup Equivalence

Existing fairness metrics evaluate the performance gap across different subgroups (Du et al., 2020). However, these do not reflect the overall model performance. For instance, if a model gives almost equal but low performance across different subgroups, then DI will be high, and DoB will be low. Therefore, the model prediction will be considered unbiased across different subgroups. However, an unbiased but low performing model is undesirable. Therefore, in this research, Precise Subgroup Equivalence (PSE) metric is introduced that jointly estimates the effect of demographic subgroup bias on model prediction and the overall model performance. Precise Subgroup Equivalence (PSE) is the average of Disparate Impact (DI), Average False Rate (AFR), and Degree of Bias (DoB).

$$PSE = \frac{(1 - DI) + AFR + DoB}{3} \tag{3}$$

Since a lower value of DI indicates higher bias in model prediction, therefore higher value of $(1 - DI)$ indicates higher bias in model prediction. Here, AFR is the mean of *False Positive Rate* (*FPR*) and *False Negative Rate* (*FNR*). It is robust to the subgroup imbalance problem and reflects the overall model performance. On the other hand, $(1 - DI)$ and DoB reflects the bias in the model prediction. Therefore, PSE provides a joint estimate of the overall model performance and the impact of bias. A model with low PSE indicates an unbiased high performing model.

3.2. Bias Estimation

For joint estimation of pre-trained model performance and the impact of demographic subgroup bias, PSE of the model prediction corresponding to a given dataset is computed. Let $\mathbf{X}$ be the training set with n number of images.

$$\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n\} \tag{4}$$

where, each image $\mathbf{X}_i$ is associated with m demographic groups. Let $\mathbf{D}$ and $\mathbf{E}$ are the two demographic groups and s and t be the number of subgroups in $\mathbf{D}$ and $\mathbf{E}$, respectively.

$$\mathbf{D} = \{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_s\} \quad \text{and} \quad \mathbf{E} = \{\mathbf{E}_1, \mathbf{E}_2, \ldots, \mathbf{E}_t\} \tag{5}$$

where, $\mathbf{D}_j$ and $\mathbf{E}_j$ represent a subgroup of the respective demographic group. Let $\phi_{\mathbf{D}}$ be a pre-trained model with weight $\mathbf{W}$ and bias b trained for predicting demographic group $\mathbf{D}$.

For the first scenario, the probability of predicting an input image $\mathbf{X}_i$ to subgroup $\mathbf{D}_j$ is represented as:

$$P\left(\mathbf{D}_j \middle| \mathbf{X}_i, \mathbf{D}\right) = \phi_{\mathbf{D}}\left(\mathbf{X}_i, \mathbf{W}, b\right) \tag{6}$$

For the second scenario, the probability of predicting an input image $\mathbf{X}_i$ to subgroup $\mathbf{D}_j$ across demographic group $\mathbf{E}$ is represented as:

$$P\left(\mathbf{D}_j \middle| \mathbf{X}_i, \mathbf{E}, \mathbf{D}\right) = \phi_{\mathbf{D}}\left(\mathbf{X}_i, \mathbf{W}, b\right) \tag{7}$$

The PSE of model $\phi_{\mathbf{D}}$ corresponding to dataset $\mathbf{X}$ is computed as:

$$PSE_{\phi_{\mathbf{D}}} = \frac{\left(1 - DI_{\phi_{\mathbf{D}}}\right) + AFR_{\phi_{\mathbf{D}}} + DoB_{\phi_{\mathbf{D}}}}{3} \tag{8}$$

where, $DI_{\phi_{\mathbf{D}}}$, $AFR_{\phi_{\mathbf{D}}}$, and $DoB_{\phi_{\mathbf{D}}}$ are the Disparate Impact, Average False Rate, and Degree of Bias of model $\phi_{\mathbf{D}}$ corresponding to dataset $\mathbf{X}$, respectively.

3.3. Bias Mitigation

After estimating the bias in the prediction of a pre-trained model $\phi_{\mathbf{D}}$ corresponding to dataset $\mathbf{X}$, the next task is to mitigate the effect of bias to improve the overall model performance. For this

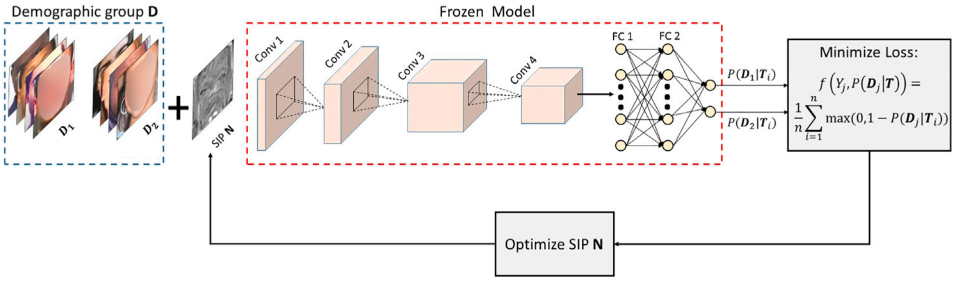

Demographic group **D**
D_1
D_2
+
SIP N
Frozen Model
Conv 1
Conv 2
Conv 3
Conv 4
FC 1
FC 2
$P(\mathbf{D}_1|\mathbf{T}_i)$
$P(\mathbf{D}_2|\mathbf{T}_i)$
Minimize Loss:
$f\left(Y_j, P(\mathbf{D}_j|\mathbf{T})\right) = \frac{1}{n}\sum_{i=1}^{n}\max(0,1 - P(\mathbf{D}_j|\mathbf{T}_i))$
Optimize SIP **N**

FIGURE 2 | Block diagram of the steps involved in learning Subgroup Invariant Perturbation (SIP). In the first step, SIP **N** is initialized with zeros and added to the images of the training set to generated the transformed set. In the next step, the transformed set is given as input to the pre-trained model and model prediction is obtained. Next, loss is computed and optimization is performed over **N** to minimize PSE. The updated **N** is added to the training set and the process is repeated until convergence (best viewed in color).

purpose, a single uniform Subgroup Invariant Perturbation (SIP) is learned by minimizing the *PSE* corresponding to the first scenario for the given dataset **X**. The aim is to generate a transformed dataset **T** by adding SIP to all the images of dataset **X**, such that when **T** is given as input to the pre-trained model $\phi_{\mathbf{D}}$ produces unbiased outcomes and improves the overall performance. We hypothesize that the learned SIP is effective for mitigating the bias corresponding to the second scenario as well. In order to validate this, multiple experiments are performed, and the results are discussed in **Section 5.2**. The optimization process for learning SIP **N** is discussed below.

Let **N** be the Subgroup Invariant Perturbation (SIP), initialized with zeros. Each image $\mathbf{X}_i$ of the dataset has pixel values in the range of $\{0, 1\}$. Let **T** be the transformed dataset obtained by adding **N** to the dataset **X**. To bring the pixel values of each image in the transformed dataset in the range of $\{0, 1\}$, *tanh* function is applied as follows:

$$\mathbf{T}_i = \frac{1}{2}\left(tanh\left(\mathbf{X}_i + \mathbf{N}\right) + 1\right) \tag{9}$$

where, $\mathbf{T}_i$ represents the transformed image corresponding to the input image $\mathbf{X}_i$. The probability of predicting a transformed image $\mathbf{T}_i$ to subgroup $\mathbf{D}_j$ is given by:

$$P\left(\mathbf{D}_j\middle|\mathbf{T}_i, \mathbf{D}\right) = \phi_{\mathbf{D}}\left(\mathbf{T}_i, \mathbf{W}, b\right) \tag{10}$$

For models that yield biased predictions, there is a performance gap across different subgroups, where the performance of some subgroups are better than others. Therefore, the objective is to reduce *PSE* by 1) enhancing the performance of the low performing subgroups and 2) maintaining/enhancing the performance of high performing subgroups. In order to achieve both the objectives, the following objective function is used.

$$f\left(Y_{i,j}, P\left(\mathbf{D}_j\middle|\mathbf{T}_i, \mathbf{D}\right)\right) \tag{11}$$

where, $Y_{i,j}$ represents the true label and $f(.,.)$ is the function to minimize the distance between the true label and the probability of predicting the true class. The above objective function is optimized corresponding to SIP **N**. For this purpose, the following function is minimized:

$$\min_{\mathbf{N}} \quad f\left(Y_{i,j}, P\left(\mathbf{D}_j\middle|\mathbf{T}_i, \mathbf{D}\right)\right) \qquad \forall j \tag{12}$$

$$f\left(Y_j, P\left(\mathbf{D}_j\middle|\mathbf{T}, \mathbf{D}\right)\right) = \frac{1}{q}\sum_{i=1}^{q} max\left(0, 1 - P\left(\mathbf{D}_j\middle|\mathbf{T}_i, \mathbf{D}\right)\right)$$

where, $j \in \{1, \ldots, s\}$ and q is the number of images belonging to subgroup j with $q < n$. $f(.,.)$ will increase the probability of predicting the true class, which in turn reduces the $PSE_{\phi_{\mathbf{D}}}$. Low $PSE_{\phi_{\mathbf{D}}}$ will simultaneously ensure reduced effect of bias on model prediction along with improved model performance. **Figure 2** shows the block diagram of the steps involved in learning the SIP **N**.

4. EXPERIMENTAL SETUP

The performance of the proposed algorithm is evaluated for race and gender classification on four different datasets. The results are reported using the proposed metric PSE, two existing bias evaluation metrics, and one existing performance evaluation metric. The details of the datasets with the corresponding protocols and the pre-trained models used for the experiments are discussed below.

4.1. Databases and Protocols

Experiments are performed for race and gender prediction, using data corresponding to race *R1 (light skin color)* and *R2 (dark skin color)*, and gender *G1 (Male)* and *G2 (Female)*. The distribution of the number of images in each dataset across different race and gender subgroups is shown in **Table 1**; **Figure 3** shows sample images from each dataset.

MORPH dataset (Album-2) (Rawls and Ricanek, 2009) contains more than 54,000 images of 13,180 subjects. The dataset is partitioned into 60% training set, 20% validation set, and 20% testing set. The partitioning is done with non-overlapping subjects in each set.

UTKFace dataset (Zhang et al., 2017) contains more than 20,000 face images and divided into three parts, having 9,779, 10,718, 3,206 images in Part I, Part II, and Part III, respectively.

TABLE 1 | Distribution of number of images in the MORPH, UTKFace, LFWA, and CelebA datasets across different race and gender subgroups.

Dataset	Race		Gender	
	*R*1	*R*2	*G*1	*G*2
MORPH	10,662	42,725	46,835	8,527
UTKFace	10,076	4,525	12,389	11,312
LFWA	9,830	560	10,181	2,962
CelebA	—	—	75,976	1,06,756

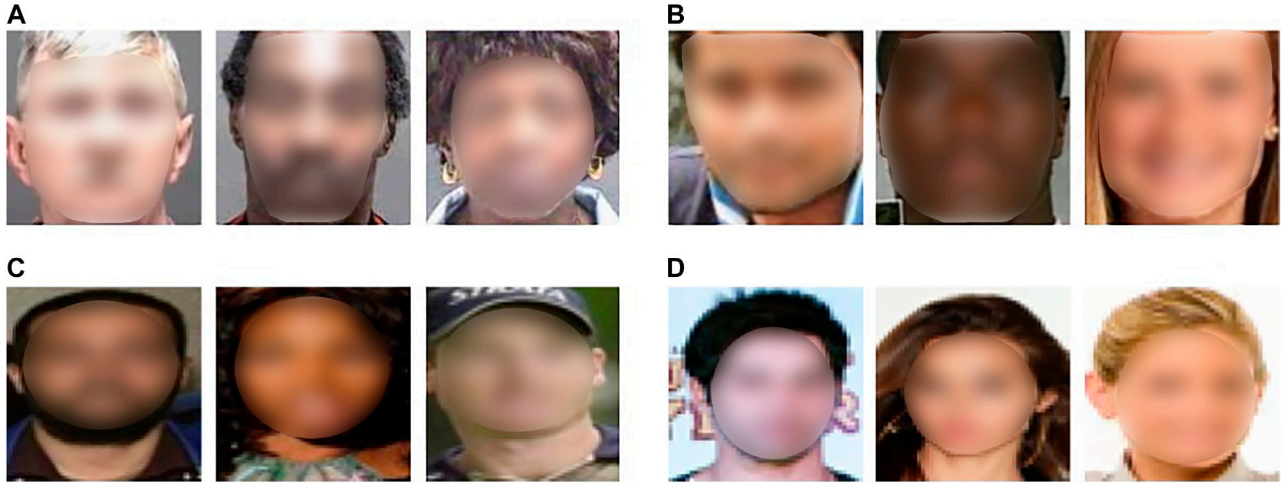


FIGURE 3 | Sample images of the **(A)** MORPH, **(B)** UTKFace, **(C)** LFWA, and **(D)** CelebA datasets.

Part I is used for training, Part II for testing, and Part III for validation (Das et al., 2018).

LFWA dataset (Huang et al., 2008) contains $13,233$ images of $5,749$ subjects with 73 attributes. Attributes corresponding to each image is annotated with intensity values. These are binarized by converting positive intensity values with label 1 and negative intensity values with label 0. For experiments, attributes corresponding to race *R1*, *R2*, and gender *G1* are taken. Images with label 0 for *G1* are considered as *G2*. Experiments are performed using the standard pre-defined protocol proposed by (Huang et al., 2008).

CelebA dataset (Liu et al., 2015) consists of a total of $2,02,599$ face images of more than $10,000$ celebrities with 40 annotated binary attributes. For experiments, the *G1* attribute is taken and images with label 0 for *G1* are considered as *G2*. The experiments are performed using the standard pre-defined protocol defined by (Liu et al., 2015).

Pre-trained models: Experiments are performed using pre-trained VGGFace (Parkhi et al., 2015) model, which is trained on the VGGFace dataset (Parkhi et al., 2015) for face recognition. VGGFace dataset is a large scale dataset of 2.6M facial images corresponding to 2.6K subjects. VGGFace model has shown high generalization abilities for face recognition. Therefore, we have used this model and fine-tuned it for race and gender prediction. In this research, three race prediction models and four gender prediction models are used for the experiments. The race prediction models are obtained by separately fine-tuning the pre-trained VGGFace model on the MORPH, UTKFace, and LFWA datasets. Similarly, the gender prediction models are obtained by fine-tuning on the MORPH, UTKFace, LFWA, and CelebA datasets. These models are treated as pre-trained race and gender prediction models in all the experiments.

4.2. Implementation Details

The implementation details of the network training and perturbation learning for mitigation are given below.

Network training: Each model is trained by adding two fully connected dense layers of 512 dimensions after the final convolutional layer of the VGGFace model. Models are trained for 20 epochs with Adam optimizer. The learning rate is set to 0.0001 for the first 10 epochs and reduced by 0.1 after every 5 epochs. Categorical cross-entropy loss is used to train the models.

Perturbation learning for mitigation: Perturbation is learned from the training set of a given dataset. In order to learn Subgroup Invariant Perturbation (SIP), a matrix is initialized with zeros of size $64 \times 64 \times 3$ (equal to the dimension of the input image), which results in 12,288 number of parameters. The parameters of this matrix are only trainable during SIP learning while keeping the parameters of the model frozen. In the first step, SIP is added to the images in the training set using **Equation 9** and given as input to the model to obtain the predictions. In the second step, model predictions are used to compute the loss using **Equation**

12. In the final step, the gradient of the loss is computed with respect to the given input, and this gradient is backpropagated to the input to update the parameters of the SIP matrix only. The process is repeated until convergence. For perturbation learning, Adam optimizer is used with a learning rate of 0.001. Depending upon the training set, the batch size is set between 500 and 1,000. Each batch is processed for 16 iterations.

5. RESULTS AND ANALYSIS

Models trained on datasets with over-representation of some demographic subgroups and under-representation of others often result in biased outputs. In a real-world scenario, it is difficult to have knowledge of the dataset used to train a model. However, depending on the training data distribution, the model could lead to biased prediction outputs. Therefore, it is important to first estimate the bias in model prediction, followed by mitigation. As discussed previously, the model's overall performance should also be considered during the estimation/mitigation of bias in model prediction to balance the trade-off between fairness and model performance. Therefore, in this research, we have jointly estimated bias in model prediction and the overall model performance using the proposed metric PSE. A series of experiments are performed where models pre-trained on some datasets are evaluated on others for bias estimation using the existing and proposed metrics. Next, we use the PSE of the model to mitigate the effect of bias in model prediction using the proposed algorithm.

We have segregated this section into: 1) Bias Estimation and 2) Bias Mitigation in **Sections 5.1** and **Sections 5.2**, respectively. Analysis of the experiments are performed under both the scenarios, *Independent demographic subgroup analysis* and *Intersectional demographic subgroup analysis*. In the first scenario of independent demographic subgroup analysis, bias estimation/mitigation algorithms are analyzed across the subgroups of a demographic group individually. Whereas, in the second scenario, analysis is performed across the intersection of different demographic groups. **Table 2** shows the details of the experiments performed in this research.

5.1. Bias Estimation

Bias estimation plays a key role in designing solutions for bias mitigation. Therefore, it is important to have a good metric to estimate bias in model prediction along with the overall model performance. There are various fairness and performance evaluation metrics, such as DI, DoB, and AFR. DI measures the deviation from statistical parity, and DoB represents the standard deviation of classification accuracy across different subgroups. On the other hand, AFR gives the average of the false positive rate and false negative rate. These metrics either evaluate the performance gap across different subgroups or the overall model performance. Therefore, we have introduced a new metric PSE that evaluates both fairness and model performance. To validate this fact, we have evaluated the performance of multiple pre-trained models (trained on different datasets) using existing and proposed metrics. The experimental setup of this experiment is discussed below:

Experimental Setup: In this experiment, the performance of pre-trained models is evaluated using five different evaluation metrics: subgroup-specific error rate, (1-DI), DoB, AFR, and PSE

TABLE 2 | Details of the experiments to estimate and mitigate the effect of demographic subgroup bias on pre-trained race and gender prediction models.

Task	Scenario	Model trained on	Bias estimation/mitigation
Race prediction	Independent/intersectional demographic subgroup analysis	MORPH UTKFace LFWA	UTKFace, LFWA MORPH, LFWA MORPH, UTKFace
Gender prediction	Independent demographic subgroup analysis	MORPH UTKFace LFWA CelebA	UTKFace, LFWA, CelebA MORPH, LFWA, CelebA MORPH, UTKFace, CelebA MORPH, UTKFace, LFWA
	Intersectional demographic subgroup analysis	MORPH UTKFace LFWA	UTKFace, LFWA MORPH, LFWA MORPH, UTKFace

TABLE 3 | Performance of pre-trained *race* prediction models (%) across different *race* subgroups for independent demographic subgroup analysis scenario.

Bias estimated on	Model trained on	Error		1 – DI	AFR	DoB	PSE
		R1	R2				
UTKFace	MORPH	27.72	22.47	6.77	25.09	2.62	11.49
	LFWA	0.04	97.54	97.53	48.78	48.75	65.02
MORPH	UTKFace	0.52	80.02	79.92	40.27	39.75	53.31
	LFWA	0.00	96.86	96.86	48.43	48.43	64.57
LFWA	MORPH	83.32	7.64	81.94	45.48	37.84	55.08
	UTKFace	17.82	60.37	51.78	39.09	21.27	37.38

for bias estimation. Evaluation of each pre-trained model is done on the training set of all the datasets except the one on which the model is trained. For instance, if the model is pre-trained on the MORPH dataset, then it is evaluated on the LFWA, CelebA, and UTKFace datasets. This setup is considered by keeping in mind the real-world scenario where the training set of the pre-trained model is unknown. Bias estimation is done on the training set because the PSE learned from the training set is used to mitigate the bias in model prediction for the corresponding dataset.

5.1.1. Independent Demographic Subgroup Analysis

In this scenario, the models are evaluated across different *race* and *gender* subgroups, respectively, of a given dataset. The error rate of each subgroup is computed to understand the variations in performance across subgroups. **Table 3** shows the performance of pre-trained race prediction models. It is observed that the error rate of the models varies significantly across different *race* subgroups. It is also observed that the distribution of training data plays a significant role in the performance of pre-trained models. For instance, the model trained on the MORPH dataset when evaluated on the UTKFace dataset results in 27.72% and 22.47% error rate corresponding to subgroup *R*1 and *R*2, respectively. On the other hand, when the LFWA model is evaluated on the UTKFace dataset, it gives 0.04% and 97.54% error rate corresponding to subgroup *R*1 and *R*2, respectively. The significant difference in the error rate of each subgroup obtained by different pre-trained models is due to the skewed training data distribution on which these models are trained as

TABLE 4 | Performance of pre-trained *gender* prediction models (%) across different *gender* subgroups for independent demographic subgroup analysis scenario.

Bias estimated on	Model trained on	Error		1 – DI	AFR	DoB	PSE
		G1	G2				
UTKFace	MORPH	29.42	42.75	18.90	36.08	6.66	20.54
	LFWA	24.94	49.12	32.22	37.02	12.09	27.11
	CelebA	53.05	41.70	19.47	47.37	5.67	24.17
MORPH	UTKFace	36.75	33.53	4.85	35.13	1.61	13.86
	LFWA	39.05	24.96	18.77	32.00	7.04	19.27
	CelebA	69.82	29.78	57.01	49.79	20.02	42.27
LFWA	UTKFace	30.27	36.69	9.21	33.48	3.21	15.30
	MORPH	19.27	57.66	47.56	38.46	19.19	35.07
	CelebA	16.79	45.74	34.80	31.26	14.47	26.84
CelebA	UTKFace	43.23	42.88	0.62	43.05	0.17	14.61
	MORPH	39.71	57.79	30.00	48.75	9.04	29.26
	LFWA	12.21	54.75	48.45	33.47	21.27	34.40

TABLE 5 | Performance of pre-trained *race* prediction models (%) across different *gender* subgroups and *gender* prediction models across *race* subgroups of a given dataset for intersectional demographic subgroup analysis scenario.

Bias estimated on	Model trained on	Error				1 – DI	AFR	DoB	PSE
		G1		G2					
		R1	R2	R1	R2				
		Race prediction across gender subgroups							
UTKFace	MORPH	35.70	18.41	20.76	26.47	14.21	25.33	5.74	15.09
	LFWA	1.15	97.52	1.30	100.00	98.75	49.98	48.76	65.83
MORPH	UTKFace	0.56	79.71	0.36	81.95	80.74	40.64	40.18	53.85
	LFWA	0.00	96.55	0.00	98.74	97.64	48.82	48.82	65.09
LFWA	UTKFace	20.49	58.96	9.52	67.40	56.17	39.09	24.08	39.78
	MORPH	83.24	6.99	83.58	10.87	81.78	46.17	37.23	55.06
		Gender prediction across race subgroups							
UTKFace	MORPH	32.07	38.95	16.42	54.91	28.09	35.58	11.34	25.00
	LFWA	28.97	43.68	9.46	62.75	39.79	36.21	16.99	30.99
MORPH	UTKFace	36.76	11.05	36.85	41.81	18.38	31.61	7.66	19.22
	LFWA	44.50	17.80	37.70	27.66	23.19	31.91	9.18	21.43
LFWA	UTKFace	30.70	35.28	31.01	50.00	17.08	36.75	5.89	19.91
	MORPH	19.65	57.28	17.47	58.70	48.39	38.27	19.71	35.46

TABLE 6 | Performance of *race* prediction models (%) after bias mitigation using the proposed and existing algorithms [Multi-task (Das et al., 2018) and Filter Drop (Nagpal et al., 2020)] for independent demographic subgroup analysis scenario.

Bias estimated on	Model trained on		Error		1 – DI	AFR	DoB	PSE
			R1	R2				
UTKFace	MORPH	Pre-trained	33.95	18.61	18.86	26.27	7.67	17.60
		Fine-tuned	4.85	46.37	43.63	25.61	20.76	30.00
		Multi-task	29.66	13.32	18.85	21.48	8.17	16.17
		Filter drop	27.95	16.16	14.06	22.04	5.90	14.00
		Proposed	14.55	19.74	6.08	17.17	2.59	**8.61**
	LFWA	Pre-trained	0.08	97.45	97.45	48.76	48.68	64.96
		Fine-tuned	3.55	53.44	51.72	28.49	24.94	35.05
		Multi-task	31.55	14.50	19.93	23.02	8.53	17.16
		Filter drop	26.78	14.84	14.01	20.80	5.97	13.59
		Proposed	15.42	25.16	11.52	20.29	4.87	**12.23**
MORPH	UTKFace	Pre-trained	0.31	86.45	86.41	43.37	43.07	57.62
		Fine-tuned	6.04	1.85	4.27	3.94	2.09	3.43
		Multi-task	3.32	5.99	2.75	4.65	1.34	2.91
		Filter drop	4.13	5.76	1.69	4.93	0.82	2.48
		Proposed	2.47	4.91	2.51	3.68	1.22	**2.47**
	LFWA	Pre-trained	0.00	98.11	98.11	49.05	49.05	65.40
		Fine-tuned	7.09	1.73	5.46	4.41	2.68	4.18
		Multi-task	2.07	7.96	6.01	5.00	2.95	4.65
		Filter drop	3.22	6.54	3.42	4.87	1.66	3.32
		Proposed	1.31	4.46	3.20	2.88	1.57	**2.55**
LFWA	UTKFace	Pre-trained	19.85	63.86	54.92	41.85	22.00	39.59
		Fine-tuned	0.71	81.76	81.63	41.23	40.52	54.46
		Multi-task	32.04	24.92	9.49	28.47	3.56	13.84
		Filter drop	35.62	28.78	9.61	32.19	3.42	15.07
		Proposed	8.56	22.11	14.83	15.33	6.77	**12.31**
	MORPH	Pre-trained	81.95	2.11	81.56	42.02	39.92	54.50
		Fine-tuned	1.47	78.25	77.93	39.86	38.39	52.06
		Multi-task	32.61	27.02	7.66	29.81	2.80	13.42
		Filter drop	32.69	26.32	8.64	29.50	3.19	13.78
		Proposed	8.75	23.16	15.80	15.95	7.20	**12.98**

The lowest PSE value is highlighted.

shown in **Table 1**. The MORPH dataset has under-representation of subgroup $R1$ and over-representation of subgroup $R2$. On the other hand, the LFWA dataset has a majority of subgroup $R1$. Therefore, the model trained on the MORPH dataset performs better for subgroup $R2$, while the LFWA model gives better performance for subgroup $R1$. A similar observation can be drawn when the evaluation is performed on the LFWA dataset using the models trained on the MORPH and UTKFace datasets.

On evaluating the performance of a pre-trained model using individual metrics for a given dataset, it is observed that PSE is a good indicator of fairness and model performance. For instance, the PSE of the LFWA model corresponding to the UTKFace dataset is 65.02%. The high value of PSE indicates a biased and low performing model. The values of metrics (1-DI) and DoB are 97.53% and 48.75%, indicating a biased model. However, these do not provide any insights about model performance. On the other hand, the AFR of the model is 48.78%, indicating that the model performance is low without providing any insight about the bias in model prediction. This shows that metric PSE provides a joint estimation of bias and model performance.

The performance of the *gender* prediction models is reported in **Table 4**. A similar observation is drawn regarding the effectiveness of metric PSE from **Table 4**. For instance, the performance of the model trained on the UTKFace dataset, when evaluated on the MORPH dataset, shows almost equal but high error rate across different *gender* subgroups. Therefore (1-DI) and DoB of this model are low, but AFR is high. Thus, none of the metrics is able to provide a unified estimate of fairness and model performance. On the other hand, the PSE of this model is 13.86% showing a joint estimate of both. A similar observation is obtained when this pre-trained model is evaluated on the LFWA and CelebA datasets. This showcases that PSE provides a unified score of both error and disparity in subgroups.

5.1.2. Intersectional Demographic Subgroup Analysis

Existing studies (Alvi et al., 2018; Das et al., 2018; Nagpal et al., 2020) have shown that the influence of one demographic group can affect the prediction of others. For instance, the performance of a gender prediction model may be affected due to the imbalance in ethnicity subgroups. In such a case, the model prediction will be biased towards the over-represented ethnicity subgroup. Therefore, it is important to estimate the bias of one

TABLE 7 | Performance of *gender* prediction models (%) with the proposed and existing bias mitigation algorithms for independent demographic subgroup analysis.

Bias estimated on	Model trained on		Error		1 – DI	AFR	DoB	PSE
			G1	*G2*				
UTKFace	MORPH	Pre-trained	22.64	41.37	24.56	32.13	9.36	22.02
		Fine-tuned	15.61	29.30	16.23	22.45	6.84	15.17
		Multi-task	23.65	37.87	18.62	30.75	7.11	18.83
		Filter drop	28.08	34.99	9.60	31.52	3.46	14.86
		Proposed	10.18	22.30	13.50	16.24	6.06	**11.93**
	LFWA	Pre-trained	18.25	51.74	40.97	34.99	16.74	30.90
		Fine-tuned	15.00	31.67	19.62	23.33	8.33	17.09
		Multi-task	24.01	38.52	19.09	31.26	7.25	19.20
		Filter drop	29.77	33.33	5.06	31.54	1.78	12.79
		Proposed	11.91	23.15	12.77	17.53	5.62	**11.97**
	CelebA	Pre-trained	42.96	47.32	7.65	45.13	2.18	18.32
		Fine-tuned	15.14	33.44	21.57	24.28	9.15	18.33
		Multi-task	30.86	38.08	10.43	34.46	3.61	16.17
		Filter drop	29.12	34.93	8.19	32.01	2.91	14.37
		Proposed	34.70	34.98	0.44	34.83	0.14	**11.80**
MORPH	UTKFace	Pre-trained	41.27	19.02	27.48	30.14	11.12	22.91
		Fine-tuned	5.84	28.52	24.09	17.17	11.34	17.53
		Multi-task	10.00	22.80	14.22	16.39	6.40	12.34
		Filter drop	8.88	21.67	14.03	15.27	6.40	11.90
		Proposed	14.11	3.59	10.92	8.84	5.26	**8.34**
	LFWA	Pre-trained	53.80	20.58	41.84	37.19	16.61	31.88
		Fine-tuned	8.69	18.75	11.02	13.72	5.03	9.92
		Multi-task	9.63	23.07	14.87	16.34	6.72	12.64
		Filter drop	9.88	20.94	12.26	15.4	5.53	11.06
		Proposed	15.92	4.75	11.72	10.33	5.58	**9.21**
	CelebA	Pre-trained	66.47	31.45	51.09	48.95	17.51	39.18
		Fine-tuned	7.71	22.01	15.50	14.85	7.15	12.50
		Multi-task	8.19	24.60	17.87	16.39	8.21	14.16
		Filter drop	9.80	26.47	18.48	18.12	8.33	14.98
		Proposed	19.49	3.26	16.78	11.37	8.11	**12.09**
LFWA	UTKFace	Pre-trained	29.88	39.39	13.57	34.63	4.75	17.65
		Fine-tuned	3.99	54.18	52.28	29.08	25.09	35.48
		Multi-task	27.46	36.70	12.73	32.07	4.62	16.47
		Filter drop	30.16	34.05	5.56	32.09	1.95	13.20
		Proposed	18.12	26.58	10.33	22.35	4.23	**12.30**
	MORPH	Pre-trained	19.12	58.50	48.69	38.30	19.69	35.56
		Fine-tuned	5.50	50.42	47.53	27.95	22.46	32.65
		Multi-task	39.67	28.74	15.34	34.20	5.47	18.34
		Filter drop	34.43	37.32	4.40	35.86	1.45	13.90
		Proposed	15.95	27.33	13.55	21.63	5.69	**13.62**
	CelebA	Pre-trained	16.31	45.21	34.53	30.76	14.45	26.58
		Fine-tuned	10.56	36.79	29.33	23.67	13.11	22.04
		Multi-task	25.41	37.41	16.07	31.40	6.00	17.82
		Filter drop	25.44	34.72	12.49	30.08	4.64	15.74
		Proposed	12.61	28.02	17.64	20.31	7.70	**15.22**
CelebA	UTKFace	Pre-trained	42.62	42.77	0.27	42.69	0.07	14.34
		Fine-tuned	28.26	11.97	18.51	20.11	8.14	15.59
		Multi-task	—	—	—	—	—	—
		Filter drop	—	—	—	—	—	—
		Proposed	33.90	33.92	0.04	33.91	0.00	**11.32**
	MORPH	Pre-trained	38.67	56.47	29.02	47.56	8.90	28.49
		Fine-tuned	23.85	14.26	11.19	19.05	4.79	11.68
		Multi-task	—	—	—	—	—	—
		Filter drop	—	—	—	—	—	—
		Proposed	14.99	21.73	7.28	18.08	3.37	**9.58**
	LFWA	Pre-trained	11.71	55.06	49.10	33.38	21.67	34.72
		Fine-tuned	26.03	12.94	15.04	19.48	6.54	13.69
		Multi-task	—	—	—	—	—	—
		Filter drop	—	—	—	—	—	—
		Proposed	11.57	23.33	13.31	17.45	5.88	**12.21**

The lowest PSE value is highlighted.

demographic group on the prediction of others. For this purpose, in this scenario, the pre-trained *race* prediction models are evaluated across different *gender* subgroups and vice versa. This scenario showcases the performance of the pre-trained models across the intersection of different demographic groups.

Table 5 shows the results of this experiment. On evaluating the performance across all the datasets using different pre-trained race prediction models, it is observed that the models trained on the UTKFace and LFWA datasets result in a high error rate for predicting race *R*2 across *G*2, i.e., subgroup (*R*2, *G*2). It is also observed that none of the samples in this intersectional subgroup are correctly classified by the model trained on the LFWA dataset when evaluated on the UTKFace dataset. This results in a high PSE value of 65.83%. For gender prediction across *race* subgroups, it is observed that all the pre-trained *gender* prediction models (except model trained on the LFWA dataset when evaluated on the MORPH dataset) perform worse for predicting gender *G*2 across *R*2, i.e., subgroup (*G*2, *R*2). The results from **Table 5** highlight that the majority of the pre-trained *race* and *gender* prediction models do not perform well for (*R*2, *G*2) and (*G*2, *R*2) subgroups, respectively.

5.2. Bias Mitigation

The experiments performed for bias estimation show that the pre-trained models do not give equal performance across different subgroups. Therefore, in this experiment, a single uniform Subgroup Invariant Perturbation is learned by minimizing the PSE of the pre-trained model prediction to achieve improved and almost equal performance across different subgroups. Multiple experiments are performed to evaluate the effectiveness of the proposed algorithm to mitigate the effect of bias in pre-trained model prediction. As mentioned in **Section 3.3**, SIP learned corresponding to the ‘independent subgroup analysis’ scenario is used to evaluate the performance of the proposed algorithm for the ‘intersectional subgroup analysis’ scenario as well. The performance of the proposed algorithm is compared with pre-trained and fine-tuned model predictions. Performance is evaluated using multiple existing metrics and the proposed metric PSE. Additionally, we have compared the number of trainable parameters of the proposed algorithm with model fine-tuning. Experimental setup of this experiment is discussed below.

Experimental Setup: In this experiment, SIP is learned corresponding to the training set of all the datasets individually other than the dataset on which the pre-trained model is trained. The learned SIP is added to the testing set of the corresponding dataset for evaluating the performance of the proposed algorithm. For instance, the model pre-trained on the MORPH dataset learns SIP using the training set of the UTKFace dataset and bias is estimated on the testing set of the UTKFace dataset. Similarly, during bias estimation of the MORPH model on the LFWA dataset, SIP learned on the training set of the LFWA dataset is used. For fine-tuning, the pre-trained model is updated using the training set of a given dataset and evaluated on the testing set of the corresponding dataset. The performance of the pre-trained model is evaluated on the testing set of the corresponding dataset.

5.2.1. Independent Demographic Subgroup Analysis

The results of the pre-trained model, fine-tuned model, and the proposed mitigation algorithm are summarized in **Table 6**. It is observed that the proposed algorithm reduces the bias in the model prediction and enhances the performance. For instance, the proposed algorithm reduces the PSE by 8.99% and 21.39% from the pre-trained and fine-tuned MORPH model predictions, respectively, for the UTKFace dataset. It is interesting to observe that fine-tuning increases the bias in the model prediction and decreases the overall performance. This is because the fine-tuned model decreases the error rate from 33.95 to 4.85% of subgroup *R*1 but increases the error rate of subgroup *R*2 from 18.61 to 46.37% compared to the pre-trained model. The UTKFace dataset has an under-representation of subgroup *R*2. Therefore, a model fine-tuned on this dataset decreases the error rate of subgroup *R*1 and penalizes subgroup *R*2. A similar observation can be drawn from the subgroup-specific error rates of fine-tuned MORPH and UTKFace models on the LFWA dataset, due to the minority of subgroup *R*2. On the other hand, the proposed algorithm overcomes the problem and reduces the performance gap across different subgroups.

The performance of *gender* prediction models is shown in **Table 7**. It is observed that the proposed algorithm reduces the PSE of each model corresponding to all the datasets. For instance, the PSE of the pre-trained and fine-tuned UTKFace model corresponding to the MORPH dataset is 22.91% and 17.53%, respectively. The proposed algorithm reduces the PSE to 8.34%. This showcase that the proposed algorithm is jointly able to reduce the bias in model prediction and improve the overall performance of the model. **Figure 4** shows the visualization of the learned Subgroup Invariant Perturbation

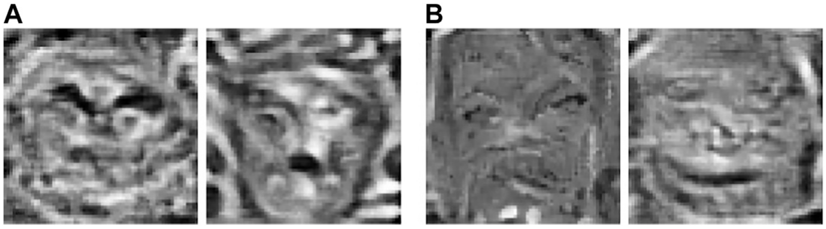


FIGURE 4 | Visualization of the learned Subgroup Invariant Perturbation (SIP) corresponding to the **(A)** race and **(B)** gender prediction models.

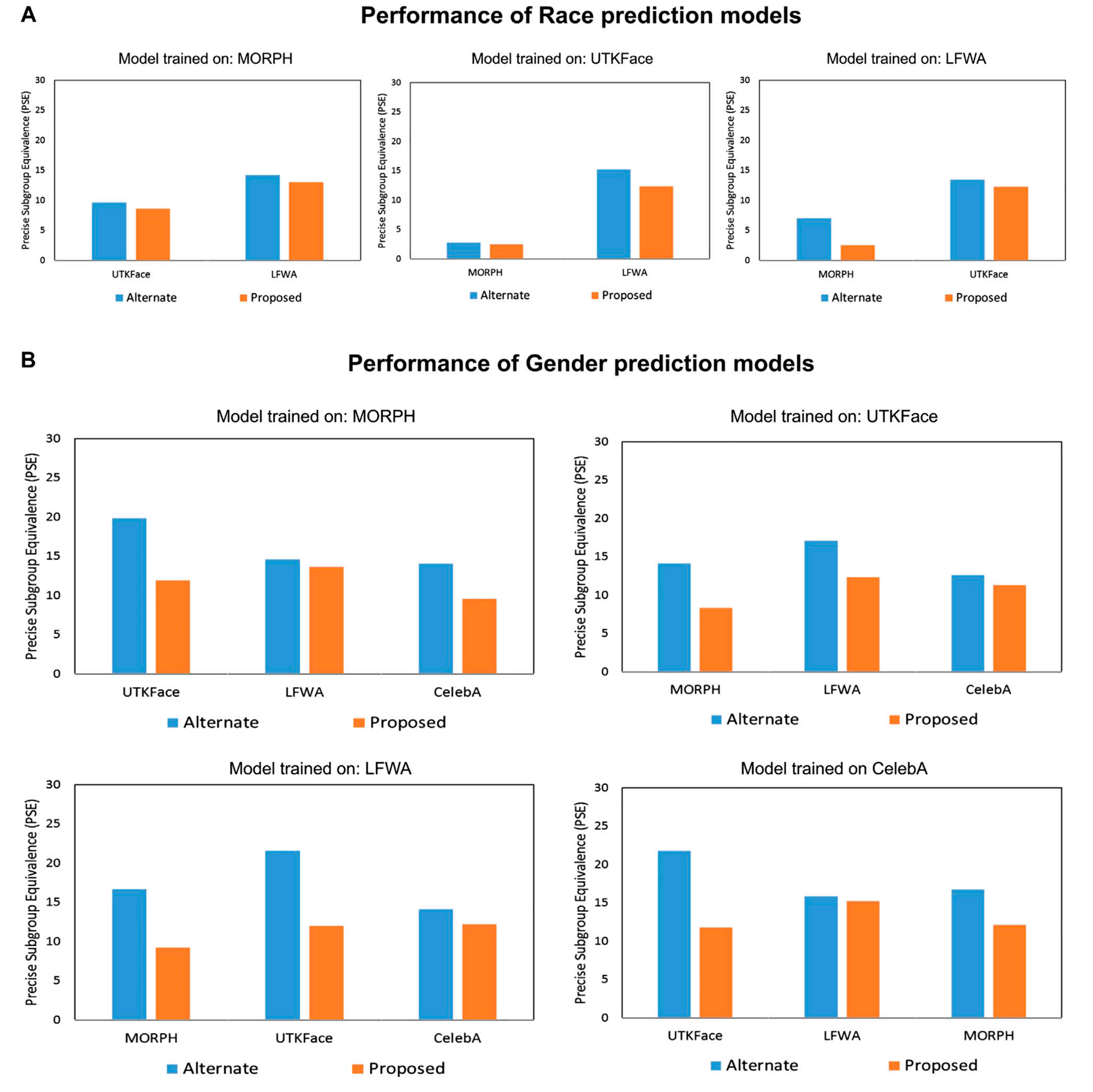


FIGURE 5 | Results shows the comparison of alternate and joint optimization using the proposed algorithm corresponding to the independent demographic subgroup analysis scenario. The proposed algorithm gives the lowest PSE.

(SIP). A face like structure can clearly be seen in all the perturbations.

The proposed algorithm is compared with two existing bias mitigation algorithms (Das et al., 2018; Nagpal et al., 2020). The comparison of the results for *race* and *gender* prediction are shown in **Tables 6** and **7**, respectively. It is observed that the proposed algorithm outperforms existing algorithms for both *race* and *gender* prediction. The proposed algorithm jointly optimizes bias and the overall model performance while the existing algorithms focus on bias optimization only. Therefore, the PSE of the proposed algorithm is minimum compared to others. For instance, in *gender* prediction (**Table 7**), the PSE of the CelebA model corresponding to the UTKFace dataset for Multi-task (Das et al., 2018), Filter Drop (Nagpal et al., 2020), and the

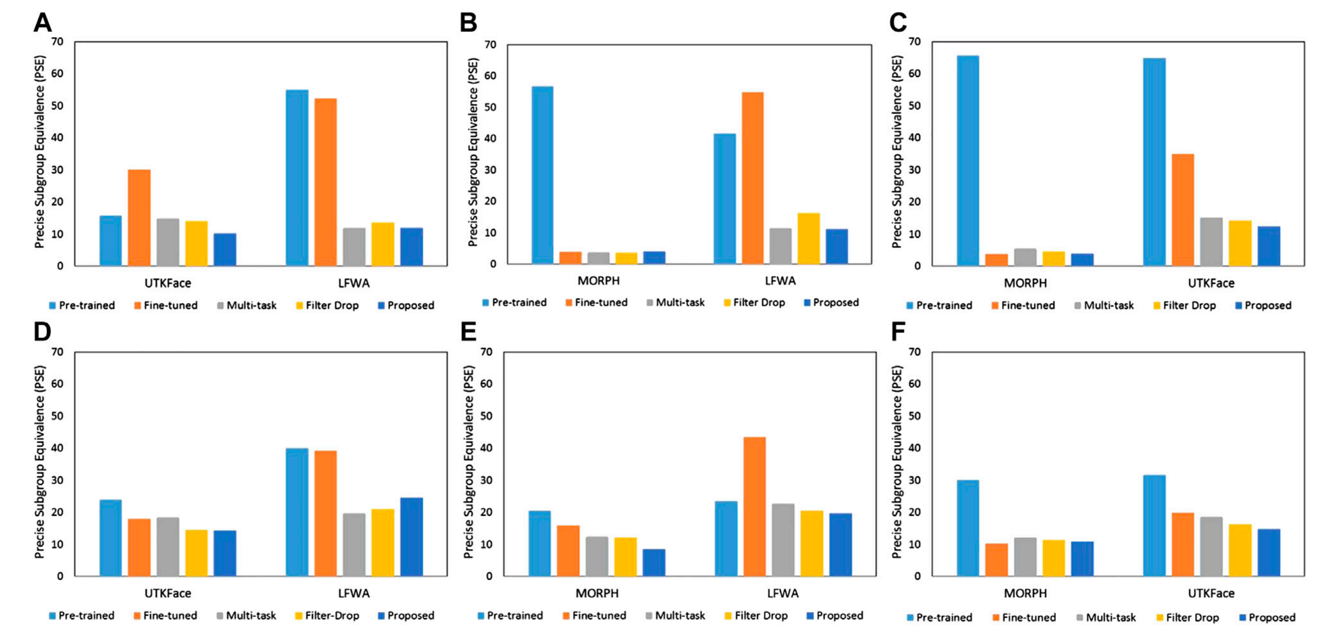


FIGURE 6 | Results of **(A–C)** race classification across *gender* subgroups corresponding to the MORPH, UTKFace, and LFWA models, respectively and **(D–F)** gender classification across *race* subgroups corresponding to the MORPH, UTKFace, and LFWA models, respectively for intersectional demographic subgroup analysis scenario. Comparison is shown with pre-trained and fine-tuned model predictions along with existing algorithms for bias mitigation.

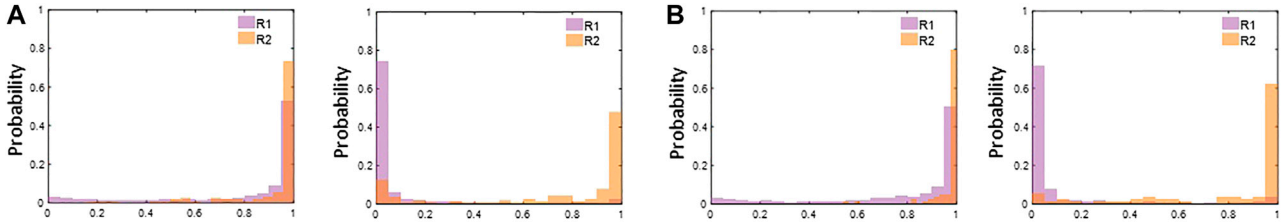


FIGURE 7 | Score distribution of the MORPH race prediction model across *gender* subgroups **(A)** *G*1 and **(B)** *G*2 on the LFWA dataset. The first graph of **(A)** and **(B)** shows the score distribution of the pre-trained model and the second graph shows the score distribution of the proposed algorithm.

proposed algorithm is 16.17%, 14.37%, and 11.80%, respectively. This shows the effectiveness of the proposed algorithm for independent demographic subgroup analysis scenario. In our experimental setup, the existing algorithms are not applicable for *gender* prediction on the CelebA dataset. Apart from this, we have also performed an experiment, where we reduce bias and improve the model performance alternatively using the proposed bias mitigation algorithm. The results of this experiment are compared with the proposed bias mitigation algorithm, where we jointly reduce the bias and improve the model performance. **Figure 5** shows the comparison of the results of alternate and joint optimization corresponding to the independent demographic subgroup analysis scenario. It is observed that joint optimization leads to better results as it provides combined supervision of bias and model performance for better learning of SIP that results in better performance.

5.2.2. Intersectional Demographic Subgroup Analysis

To further evaluate the effectiveness of the proposed algorithm across the intersection of different demographic groups, two different experiments are performed. In the first experiment, race classification is performed across *gender* subgroups. While in the second experiment, gender classification is performed across *race* subgroups. These experiments are performed to analyze the presence of *gender* bias on race prediction and *race* bias on gender prediction. Comparison is performed with pre-trained and fine-tuned model predictions. **Figure 6** shows the PSE corresponding to the first and second experiments. It is observed that in most of the cases, the proposed algorithm gives the lowest PSE. For instance, the PSE of pre-trained and fine-tuned UTKFace models corresponding to the MORPH dataset for gender prediction is 20.58% and 15.91%, respectively. The proposed algorithm reduces the PSE to 8.56%.

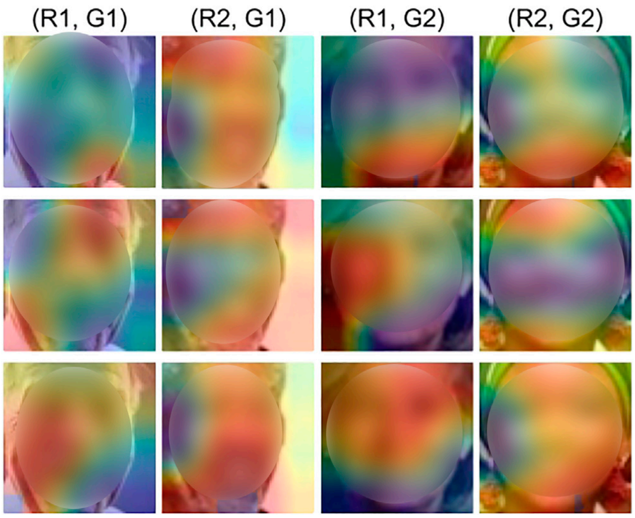


FIGURE 8 | Class Activation Map of race classification across *gender* subgroups on the UTKFace dataset using the MORPH race prediction model. Top row shows the visualization for the pre-trained model prediction, middle row for the fine-tuned model prediction, and the bottom row for the proposed algorithm. It is observed that the proposed algorithm focuses on the entire facial region instead of the subgroup-specific region for feature extraction.

This indicates that the proposed algorithm is able to reduce the effect of bias of one demographic group on the prediction of others. The reduction in PSE shows the effectiveness of the proposed algorithm.

Figure 7 compares the performance of the proposed algorithm and the pre-trained model using the score distribution of the model prediction. The results are shown for *race* prediction across different *gender* subgroups of the MORPH model on the LFWA dataset. It is observed that the proposed algorithm reduces the overlap among the subgroups and separates them from each other. Class Activation Map (CAM) of race classification across *gender* subgroups on the UTKFace dataset using the MORPH race prediction model is shown in **Figure 8**. It is observed that the pre-trained and fine-tuned models focus on different facial regions across the intersection of different demographic subgroups. On the other hand, the proposed algorithm tries to focus on the entire facial region irrespective of different subgroups. This showcases the effectiveness of the learned SIP to mitigate the effect of demographic subgroup bias by enforcing the model to extract features from the entire facial region for discrimination instead of subgroup-specific regions.

On comparing the number of trainable parameters of the proposed algorithm with model fine-tuning, it is observed that the proposed algorithm requires number of parameters equal to the size of the input image, i.e., 12K parameters. On the other hand, model fine-tuning requires updation of 0.52M parameters, which is approximately 43 times more than the proposed algorithm. This shows that the proposed algorithm is computationally efficient.

Figure 6 shows the comparison of the proposed algorithm with existing bias mitigation algorithms (Das et al., 2018; Nagpal et al., 2020). It is observed that in most of the cases, the proposed algorithm performs better than existing algorithms while giving comparable results for others. For instance, the PSE of the proposed and existing algorithms for *race* prediction across *gender* subgroups of the MORPH model corresponding to the UTKFace and LFWA datasets are 10.24%, 14.83%, 14.10% and 12.06%, 11.94%, 13.57%, respectively. It is important to note that the proposed algorithm does not require model training and therefore is computationally efficient.

6. DISCUSSION AND CONCLUSION

The effect of demographic subgroup bias on the performance of commercial and pre-trained models is studied in the past. A lot of progress is made towards estimating and mitigating the influence of bias on model prediction. However, studies have shown that there is a trade-off between fairness and model performance. Maintaining a balance between the two is an important factor. This motivated us to propose a unified metric to measure the trade-off and an algorithm to mitigate the effect of bias on pre-trained model prediction.

We used multiple pre-trained race and gender prediction models for bias estimation and mitigation. Since the existing metrics either evaluate the performance gap across different subgroups or the overall model performance, therefore we have introduced a unified metric, PSE, to jointly estimate the bias in model prediction and the overall model performance. Additionally, a novel algorithm is proposed to mitigate the effect of bias using adversarial perturbation by reducing the PSE of the model prediction. We showed that a single uniform Subgroup Invariant Perturbation (SIP), when added to the input images, is able to mitigate the effect of bias on model prediction.

During bias estimation, it is observed that PSE reflects both error and disparity in subgroups. On analyzing the existing metrics, it is observed that DI and DoB do not reflect the overall model performance, while AFR does not reflect the performance gap across different subgroups. On the other hand, we have experimentally validated in **Tables 3–5** that PSE considers the model performance along with fairness. Therefore, PSE is utilized by the proposed algorithm to learn SIP for bias mitigation. The performance of race and gender prediction models corresponding to the independent demographic subgroup analysis scenario are summarized in **Tables 6** and **7**, respectively. We have found that the proposed algorithm is able to reduce the PSE of all the pre-trained models corresponding to all the datasets. To test the proposed algorithm for mitigating the influence of bias corresponding to the intersectional subgroup analysis scenario, SIP learned corresponding to the independent subgroup analysis scenario is used. **Figure 6** shows that the proposed algorithm is effective in mitigating the intersectional subgroup bias. This is validated by the score distributions in **Figure 7** that shows that the proposed algorithm reduces the overlap between subgroups. We have also found that the proposed algorithm focuses on the entire face for feature extraction instead of subgroup-specific regions in **Figure 8**.

Existing research towards bias mitigation requires model training to suppress the element of bias for unbiased prediction. However, the proposed algorithm does not require model training for bias mitigation. It requires the number of trainable parameters equal to the size of the input image, which is significantly lower than the model fine-tuning approach. Therefore, the proposed algorithm is computationally efficient. This showcase the applicability of the proposed algorithm in real-world scenarios.

In the future, we plan to extend the proposed algorithm for mitigating the effect of bias due to the influence of multiple demographic subgroups via learning a single Subgroup Invariant Perturbation (SIP). Also, we will investigate the effect of bias on face recognition performance.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. This data can be found here:

- MORPH: https://ebill.uncw.edu/C20231_ustores/web/store_main.jsp?STOREID=4.
- UTKFace: https://susanqq.github.io/UTKFace/.
- LFWA: http://vis-www.cs.umass.edu/lfw/.
- CelebA: http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html.

ETHICS STATEMENT

Images used in this research are taken from publicly available datasets and the authors of the datasets have not taken explicit written consent from the individual(s) present in the dataset.

AUTHOR CONTRIBUTIONS

PM and SC developed the algorithm, conducted multiple experiments, and analyzed the results under the supervision of RS and MV. All the authors discussed the results, co-wrote, and reviewed the manuscript.

ACKNOWLEDGMENTS

PM is partly supported by DST INSPIRE PhD Fellowship and MV is partially supported through the SwarnaJayanti Fellowship. This research is also partially supported by Facebook Ethics in AI award.

REFERENCES

Alvi, M., Zisserman, A., and Nellåker, C. (2018). "Turning a blind eye: explicit removal of biases and variation from deep neural network embeddings", in European Conference on computer vision. doi:10.2307/2300364

Barocas, S., and Selbst, A. D. (2016). Big data's disparate impact. *Calif. Law Rev.* 104, 671. doi:10.2139/ssrn.2477899

Bolukbasi, T., Chang, K.-W., Zou, J. Y., Saligrama, V., and Kalai, A. T. (2016). "Man is to computer programmer as woman is to homemaker? debiasing word embeddings", in Advances in neural information processing systems, 4349–4357. doi:10.1007/978-3-030-52485-2_4

Buolamwini, J., and Gebru, T. (2018). "Gender shades: intersectional accuracy disparities in commercial gender classification", in Conference on fairness, accountability and transparency, 77–91. doi:10.1109/SPW.2016.114

Calders, T., and Verwer, S. (2010). Three naive bayes approaches for discrimination-free classification. *Data Min. Knowl. Discov.* 21, 277–292. doi:10.1007/s10618-010-0190-x

Celis, D., and Rao, M. (2019). "Learning facial recognition biases through vae latent representations," in International workshop on fairness, accountability, and transparency in multimedia, 26–32. doi:10.1007/s11042-020-08688-x

Conger, K., Fausset, R., and Kovaleski, S. F. (2019). *San francisco bans facial recognition technology* https://tinyurl.com/y4x6wbos. [Dataset].

Creager, E., Madras, D., Jacobsen, J.-H., Weis, M. A., Swersky, K., Pitassi, T., et al. (2019). "Flexibly fair representation learning by disentanglement", in International conference on machine learning, 1436–1445. doi:10.3389/frai.2020.00033

Das, A., Dantcheva, A., and Bremond, F. (2018). "Mitigating bias in gender, age and ethnicity classification: a multi-task convolution neural network approach," in European conference on computer vision. doi:10.1007/978-3-030-11009-3_35

Drozdowski, P., Rathgeb, C., Dantcheva, A., Damer, N., and Busch, C. (2020). Demographic bias in biometrics: a survey on an emerging challenge. *IEEE Trans. Technol. Soc.* 28, 1728. doi:10.1109/TTS.2020.2992344

Drummond, C., Holte, R. C., and Hu, X. (2003). "C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling," in Workshop on learning from imbalanced datasets II (Citeseer), 1–8.

Du, M., Yang, F., Zou, N., and Hu, X. (2019). *Fairness in deep learning: a computational perspective*. arXiv preprint arXiv:1908.08843.

Du, M., Yang, F., Zou, N., and Hu, X. (2020). Fairness in deep learning: a computational perspective. *IEEE Intell. Syst.* 17, 156. doi:10.1109/MIS.2020.3000681

Dwork, C., Immorlica, N., Kalai, A. T., and Leiserson, M. (2018). "Decoupled classifiers for group-fair and efficient machine learning", in Conference on fairness, accountability and transparency, 119–133. doi:10.1145/3357384.3357857

Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. (2015). "Certifying and removing disparate impact," in ACM SIGKDD international conference on knowledge discovery and data mining, 259–268. doi:10.1145/2783258.2783311

Gong, S., Liu, X., and Jain, A. K. (2019). Debface: de-biasing face recognition. arXiv preprint arXiv:1911.08080.

Huang, C., Li, Y., Chen, C. L., and Tang, X. (2019). Deep imbalanced learning for face recognition and attribute prediction. *IEEE Trans. Pattern Anal. Mach. Intell.* 1, 17. doi:10.1109/TPAMI.2019.2914680

Huang, G. B., Mattar, M., Berg, T., and Learned-Miller, E. (2008). "Labeled faces in the wild: a database for studying face recognition in unconstrained environments," in Workshop on faces in 'real-life' images: detection, alignment, and recognition. doi:10.1007/2F978-3-319-25958-1_8

Kim, B., Kim, H., Kim, K., Kim, S., and Kim, J. (2019). "Learning not to learn: training deep neural networks with biased data," in IEEE conference on computer vision and pattern recognition. 9012–9020. doi:10.1007/s10489-020-01658-8

Li, Y., and Vasconcelos, N. (2019). "Repair: removing representation bias by dataset resampling", in IEEE conference on computer vision and pattern recognition, 9572–9581. doi:10.1038/s41467-020-19784-9

Liu, Z., Luo, P., Wang, X., and Tang, X. (2015). "Deep learning face attributes in the wild", in IEEE international conference on computer vision, 3730–3738. doi:10.1109/ICCV.2015.425

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. (2019). *A survey on bias and fairness in machine learning.* arXiv preprint arXiv:1908.09635.

Mullick, S. S., Datta, S., and Das, S. (2019). "Generative adversarial minority oversampling", in IEEE international conference on computer vision, 1695–1704.

Nagpal, S., Singh, M., Singh, R., and Vatsa, M. (2020). Attribute aware filter-drop for bias invariant classification," in IEEE/CVF conference on computer vision and pattern recognition workshops, 32–33.

Nagpal, S., Singh, M., Singh, R., Vatsa, M., and Ratha, N. (2019). *Deep learning for face recognition: pride or prejudiced?*. arXiv preprint arXiv:1904.01219.

Ntoutsi, E., Fafalios, P., Gadiraju, U., Iosifidis, V., Nejdl, W., Vidal, M.-E., et al. (2020). Bias in data-driven artificial intelligence systems—an introductory survey. *Wiley Interdisciplinary Rev. Data Min. Knowl. Discov.* 10, e1356. doi:10.1002/widm.1356

Paolini-Subramanya, M. (2018). *Facial recognition, and bias* https://tinyurl.com/y7rat8vb. [Dataset].

Parkhi, O. M., Vedaldi, A., Zisserman, A., et al. (2015). “Deep face recognition,” in British machine vision conference, 41.1–41.12. doi:10.1155/2018/2861695

Radford, J., and Joseph, K. (2020). Theory in, theory out: the uses of social theory in machine learning for social science. *Front. Big Data* 3, 18. doi:10.3389/fdata.2020.00018

Rawls, A. W., and Ricanek, K. (2009). “*Morph: development and optimization of a longitudinal age progression database*,” in *European workshop on biometrics and identity management*. New York, NY: Springer, 17–24. doi:10.16451/j.cnki.issn

Ryu, H. J., Adam, H., and Mitchell, M. (2017). *Inclusivefacenet: Improving face attribute detection with race and gender diversity*. arXiv preprint arXiv: 1712.00193.

Singh, R., Agarwal, A., Singh, M., Nagpal, S., and Vatsa, M. (2020). “On the robustness of face recognition algorithms against attacks and bias,” in AAAI conference on artificial intelligence. doi:10.1609/aaai.v34i09.7085

Torralba, A., and Efros, A. A. (2011). “Unbiased look at dataset bias,” in IEEE conference on computer vision and pattern recognition, 1521–1528. doi:10.1007/978-3-642-33718-5_12

Wang, M., and Deng, W. (2019). *Mitigate bias in face recognition using skewness-aware reinforcement learning*. arXiv preprint arXiv:1911.10692.

Wang, T., Zhao, J., Yatskar, M., Chang, K.-W., and Ordonez, V. (2019). “Balanced datasets are not enough: estimating and mitigating gender bias in deep image representations,” in IEEE international conference on computer vision, 5310–5319. doi:10.1007/2Fs42413-020-00085-4

Zhang, Z., Song, Y., and Qi, H. (2017). “Age progression/regression by conditional adversarial autoencoder,” in IEEE conference on computer vision and pattern recognition, 5810–5818. doi:10.1007/s11277-020-07473-1

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.

frontiers
in Artificial Intelligence

ORIGINAL RESEARCH
published: 19 March 2021
doi: 10.3389/frai.2021.590215

Application of Seq2Seq Models on Code Correction

Shan Huang[1]*, *Xiao Zhou*[2] *and Sang Chin*[2,3,4]

[1]Department of Physics, Boston University, Boston, MA, United States, [2]Department of Computer Science, Boston University, Boston, MA, United States, [3]Department of Brain and Cognitive Science, Massachusetts Institute of Technology, Boston, MA, United States, [4]Center of Mathematical Sciences and Applications, Harvard University, Boston, MA, United States

We apply various seq2seq models on programming language correction tasks on Juliet Test Suite for C/C++ and Java of Software Assurance Reference Datasets and achieve 75% (for C/C++) and 56% (for Java) repair rates on these tasks. We introduce pyramid encoder in these seq2seq models, which significantly increases the computational efficiency and memory efficiency, while achieving similar repair rate to their nonpyramid counterparts. We successfully carry out error type classification task on ITC benchmark examples (with only 685 code instances) using transfer learning with models pretrained on Juliet Test Suite, pointing out a novel way of processing small programming language datasets.

Keywords: programming language correction, seq2seq architecture, pyramid encoder, attention mechanism, transfer learning

OPEN ACCESS

Edited by:
Bhavya Kailkhura,
United States Department of Energy (DOE), United States

Reviewed by:
Caiwen Ding,
University of Connecticut, United States
Aline Paes,
Fluminense Federal University, Brazil

****Correspondence:***
Shan Huang
sh2015@bu.edu

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *31 July 2020*
Accepted: *08 January 2021*
Published: *19 March 2021*

Citation:
Huang S, Zhou X and Chin S (2021) Application of Seq2Seq Models on Code Correction. Front. Artif. Intell. 4:590215. doi: 10.3389/frai.2021.590215

1 INTRODUCTION

Programming language correction (PLC), which can provide suggestions for people to debug code, identify potential flaws in a program, and help programmers to improve their coding skills, has been an important topic in the Natural Language Processing (NLP) area. Generally, code errors consist of two categories: one is explicit, syntax errors, and the other is implicit, logic errors that could cause failure during program execution, for example, memory allocation errors, redundant code, etc. The syntax error problem is relatively well studied; most compilers are able to catch syntax errors, and correcting syntax errors manually is not difficult even for beginner programmers. The latter problem, however, is much more challenging due to several reasons. First, the error space is vast. For example, Error-Prone, a rule-based Java code error detector developed by google, identifies 499 bug patterns. Second, recognizing and correcting these bugs requires a higher level of understanding of the code, including identifying the relationship between objects, making connections between blocks, and matching data types. These errors could be seen in even experienced programmers and can be time consuming to correct manually. Therefore, this study will focus on automatic correction of these logic errors in code body that pass compiling stage.

At present, most work in this field used rule-based methods [JetBrains (2016); Synopsys (2016); Google (2016a); Google (2016b); Singh et al., (2013)], using static analyzers, code transformations, or control flow to identify bug patterns and make corrections. These methods are quite mature, and some are even commercialized, like Resharper. Machine learning methods, however, have been a minority and are relatively new. There is also no canonical solution; people have used methods varying from reinforcement learning to recurrent neural network.

Given the good performance and wide usage of rule-based PLC methods, there is a major drawback: these methods are often case specific. The developer had to design specific correction strategy for each bug pattern. For example, the core code body of Error-Prone contains 499 java

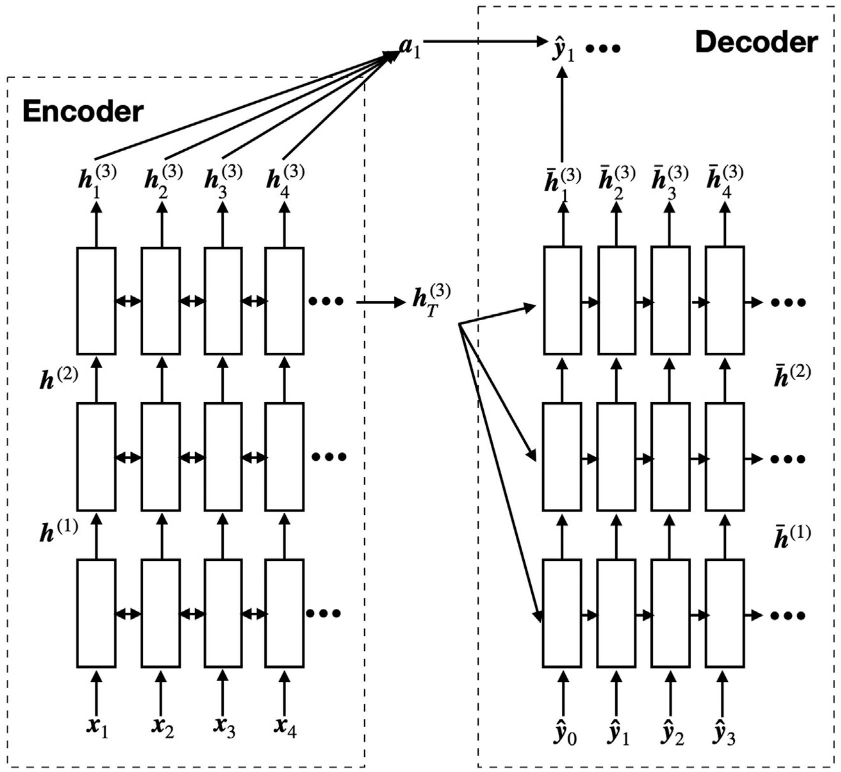

$\boldsymbol{a}_1$
$\hat{\boldsymbol{y}}_1$ •••
Decoder
Encoder
$\boldsymbol{h}_1^{(3)}$ $\boldsymbol{h}_2^{(3)}$ $\boldsymbol{h}_3^{(3)}$ $\boldsymbol{h}_4^{(3)}$
$\boldsymbol{h}_T^{(3)}$
$\bar{\boldsymbol{h}}_1^{(3)}$ $\bar{\boldsymbol{h}}_2^{(3)}$ $\bar{\boldsymbol{h}}_3^{(3)}$ $\bar{\boldsymbol{h}}_4^{(3)}$
$\boldsymbol{h}^{(2)}$
$\bar{\boldsymbol{h}}^{(2)}$
$\boldsymbol{h}^{(1)}$
$\bar{\boldsymbol{h}}^{(1)}$
$\boldsymbol{x}_1$ $\boldsymbol{x}_2$ $\boldsymbol{x}_3$ $\boldsymbol{x}_4$
$\hat{\boldsymbol{y}}_0$ $\hat{\boldsymbol{y}}_1$ $\hat{\boldsymbol{y}}_2$ $\hat{\boldsymbol{y}}_3$

FIGURE 1 | Model structure of a 3-layer seq2seq model with attention. The i^{th} layer takes the output of the previous layer ($\boldsymbol{h}^{(i-1)}$) as its input. $\boldsymbol{a}$ is the context vector, which can be calculated using different attention mechanisms.

script, each corresponds to a type of error. Therefore, rule-based PLC often requires large human labor to build. It also suffers from incompleteness and incapability of dealing with exceptions. In the long run, one could consider rule-based PLC vs. machine learning PLC as rule-based translation vs. statistical machine translation. Machine learning methods have the following advantages: first, they are self-sufficient; they teach themselves, requiring minimum amount of human development. Second, they can do self-improvement and self-prediction by grabbing data from users. Third, after sufficient training, one can expect them to perform better with coding style and fluency, like machine translations. One main obstacle that prevents machine code correction being as successful as machine translation is a general lack of data, which will be elaborated in a latter paragraph. This further leads to another drawback: insufficient training. However, machine code correction has an unlimited potential if more studies are carried out and more datasets are produced. This article aims to provide a successful example that might inspire further researches on machine code correction.

Despite good intentions of replacing hand-designed rule-based PLC method with machine-learning-based PLC method and its merits discussed above, some may express concerns about its environmental costs, as such concerns have been raised by ethical AI researchers (Hao, 2019). Although generally we do not agree that such concerns should overshadow the value of liberating human labor and pursuing potentially much better performances (as one did in machine translation), we leave such judgment to our readers. Since training a machine learning model takes mostly electricity and storage space, we provide an estimated power consumption and the detailed information of a number of parameters in our models (with chosen hyper parameters described in **Section 3.6**) in the Appendix: **Section 2**. Interested readers could refer to the information accordingly.

The machine learning models we choose are seq2seq models. Seq2seq (abbreviation of sequence to sequence) model is a group of neural-network-based models. It usually consists of an encoder and a decoder. The encoder takes a sequence as input and produces an encoded representation of the input sequence. The decoder takes this representation and produces an output sequence. It has been proved to be very successful in neural machine translation, natural language correction, text generation, etc. An example of a seq2seq model structure is shown in **Figure 1**. Our results show that seq2seq models successfully repair over 70% of the code instances if the beam search size is 1 and over 90% if the beam search size is 5.

Instead of just using regular seq2seq model, we introduce pyramid encoder structure to better suit the code correction task. The motivation is as follows: for NLC problems, the model works on a sentence level and the average length of a sentence lies around dozens of words. However, for PLC problems, the model works on the whole code instance. The average length of code

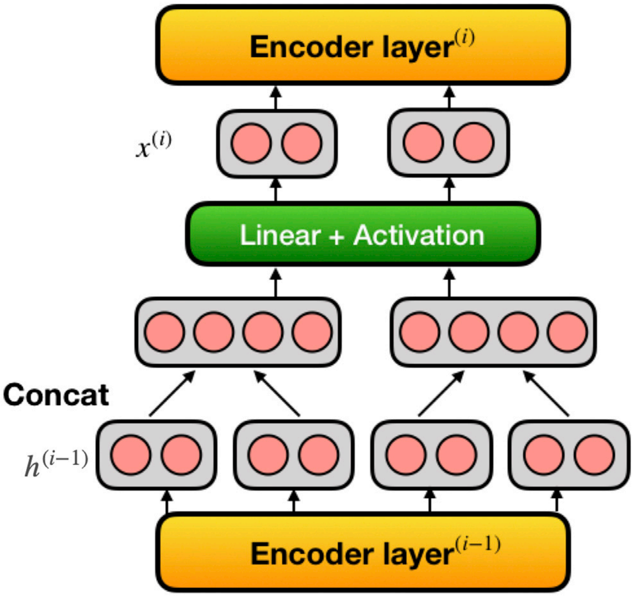


FIGURE 2 | Visualization of pyramid encoder in multilayer seq2seq models. Pyramid encoder reduces length of input sequence by half in every encoding layer. $h^{(i-1)}$ denotes output of $(i-1)^{\text{th}}$ encoder layer and $x^{(i)}$ denotes the input of i^{th} encoder layer.

instances in PLC is usually hundreds of syntax words, which results in enormous computational cost and memory requirement, especially combined with attention mechanisms. Pyramid structure aims to reduce these costs by contracting the data flow and discarding redundant information. **Figure 2** shows a visual representation of the pyramid encoder; it can be implemented to most of the multilayer seq2seq learning models. In our model comparison set, pyramid encoder increases networks' computational efficiency by 50%–100% and memory efficiency by up to 600%, while having similar ability of reparation.

On the other hand, due to the privacy policies, most of the publicly available datasets are not collected from realistic program errors and fixes but rather are generated by artificial tools. The ones that are collected realistically are usually very small. To handle this issue, we also applied transfer learning to inherit the knowledge learned from previous datasets to boost the network's performance on smaller and noisier datasets. Details of our project are available on GitHub[1].

2 RELATED WORK

Rule-based methods that work on PLC have a long history and are thus more mature. One of them is proposed by Singh et al., (2013), which is a rule-directed translation strategy synthesizing a correct program from a sketch. Their model is able to provide feedback for introductory programming problems and has achieved a correction rate of 64% on incorrect submissions. Some of these methods are quite mature. For instance, Google developed Error-Prone (Google, 2016a) and clang-tidy (Google, 2016b) as rule-based tools to help in identifying and correcting potential mistakes for programmers. Some of them are even commercialized, like Resharper (JetBrains, 2016), developed by Synopsys (2016). As a paid feature of Visual Studio, Resharper provides code analysis, refactoring, and code processing (including code generation and quick fixes for errors) as extra features to programmers.

In 2016, Pu et al.'s (2016) study became one of the first attempts to use machine learning method in PLC tasks. They used a Long Short Term Memory (LSTM) model on correcting MOOCs student assignment submissions. However, their dataset was not publicly available, putting difficulties on reproducing their work. Later in 2017, Gupta et al., (2017) proposed a seq2seq model for fixing student submissions (Deepfix), which is also a private dataset. In a later work, they (Gupta et al., 2018) used reinforcement learning based on the input code and the error messages returned by the compiler for the same task, on the same dataset. Our work, also based on seq2seq models, was carried out on a public dataset that contains more error categories.

The pyramid encoder played an important role in our research. It originated from Xie et al., (2016). We proposed its general form for all seq2seq models and thoroughly studied its performance in reduction of computational resources. We aimed to overcome difficulty brought by the extended length of code instances, compared to natural language sentences. These aspects of pyramid structure were not studied in Xie's work. We did the comparison of pyramid encoder and regular encoder under different attention mechanisms, showing that pyramid encoder could drastically reduce memory and computational cost in most setups that we considered.

3 MODEL

3.1 Overview

Given a code instance, we wish to identify and correct potential flaw in it, which might lead to a failure in execution after successful compilation. Each bad code instance contains exactly one flaw.

Formaly speaking, given an input code instance x, we wish to map it to an output code instance y and we seek to model $P(y|x)$. A code is "repaired" if the flaw that x contains is fixed in the output y. The "repair rate" is defined as the fraction between the number of code instances fixed and the total number of code instances that the model was applied on. We use repair rate as the evaluation metric in our experiments.

For this purpose, we applied two major families of seq2seq models: GRU and Transformer. We use learnable embedding layers, which allows the model to recognize the relationship between different words in the vocabulary. For the encoder, we applied pyramid encoder, where a pyramid module is added in between layers of regular multilayer encoders. For the purpose of testing generality of pyramid encoder, we combined it with different attention mechanisms.

[1] See https://github.com/b19e93n/PLC-Pyramid.

3.2 Word-Level Reasoning

In language correction, character-level reasoning is a more commonly applied method, Xie et al., (2016). However, in code correction, we apply word-level models. A "word" here is defined as a code syntax (e.g., "void", "{", space, " = ", "int", newline, etc.) or a custom variable name. The reason is that the basic building blocks of a code instance are related to the syntax. In the field of programming language processing, out-of-vocabulary (OOV) is less a problem than in natural language due to a fixed syntax pool.

In order to prevent the model suffering from vast variation of variable names, we performed a certain degree of variable renaming. We focused on renaming function names in our dataset while keeping other variables unchanged. This method reduced vocabulary size to ~1,000 and was proven to be effective in improving the performance.

We include our preprocessing method to a code instance in the Appendix.

3.3 Pyramid Encoder

Given a multilayer seq2seq encoder, its input at i^{th} layer at step t is $\boldsymbol{x}_t^{(i)}$ and the output is $\boldsymbol{h}_t^{(i)}$:

$$\boldsymbol{h}_t^{(i)} = \text{Layer}^{(i)}\left(\boldsymbol{x}_t^{(i)}\right) \tag{1}$$

In standard seq2seq models, the output of the i^{th} layer $\boldsymbol{h}^{(i)}$ is directly used as input of the $i + 1^{\text{th}}$ layer, $\boldsymbol{x}^{(i+1)}$:

$$\boldsymbol{x}_t^{(i+1)} = \boldsymbol{h}_t^{(i)} \tag{2}$$

and the time step $t = 1, 2, \ldots, T$, the layer number $i = 1, 2, \ldots, N$. Note that $\boldsymbol{x}_t^{(0)}$ is the embedded representation of the input instance.

For pyramid encoder, we introduce a pyramid module in between $\boldsymbol{h}^{(i)}$ and $\boldsymbol{x}^{(i+1)}$ as **Eq. 3 follows**:

$$\boldsymbol{x}_{t'}^{(i+1)} = \tanh\left(\boldsymbol{W}_{\text{pyr}}\left(\boldsymbol{h}_{2t}^{(i)}, \boldsymbol{h}_{2t+1}^{(i)}\right) + b_{pyr}\right) \tag{3}$$

This module reduced the length of the input $\boldsymbol{x}^{(i)}$ by half each time it is applied. The length of final output of the encoder is $T/2^{N-1}$. One could also take a bigger window such as 3, 4, 5... depending on their needs. The hope is that pyramid structure will extract the important information and reduce the redundant information of each of the neighboring hidden state, therefore reducing the training cost while keeping the accuracy of the correction. This is conceptually similar to a convolution, but without using filters.

For our GRU models, we used multilayer bidirectional GRU and we implemented pyramid encoder as described first in Xie et al., (2016):

$$\boldsymbol{f}_t^{(i)} = \text{GRU}\left(\boldsymbol{f}_{t-1}^{(i)}, \boldsymbol{x}_t^{(i)}\right) \tag{4}$$

$$\boldsymbol{b}_t^{(i)} = \text{GRU}\left(\boldsymbol{b}_{t+1}^{(i)}, \boldsymbol{x}_t^{(i)}\right) \tag{5}$$

$$\boldsymbol{h}_t^{(i)} = \boldsymbol{f}_t^{(i)} + \boldsymbol{b}_t^{(i)} \tag{6}$$

$$\boldsymbol{x}_{t'}^{(i+1)} = \tanh\left(\boldsymbol{W}_{\text{pyr}}\left(\boldsymbol{h}_{2t}^{(i)}, \boldsymbol{h}_{2t+1}^{(i)}\right) + b_{pyr}\right) \tag{7}$$

where $\boldsymbol{x}_{t'}^{(i+1)}$ denotes the input to next layer, $\boldsymbol{f}_t^{(i)}$ and $\boldsymbol{b}_t^{(i)}$ denote output from a forward and a backward GRU, respectively. GRU (Gated Recurrent Unit) is a RNN (Recurrent Neural Network) type model that includes a gating mechanism in the following equations (Cho et al., 2014):

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_{ir}\boldsymbol{x}_t + b_{ir} + \boldsymbol{W}_{hr}\tilde{\boldsymbol{h}}_{t-1} + b_{hr} \tag{8}$$

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_{iz}\boldsymbol{x}_t + b_{iz} + \boldsymbol{W}_{hz}\tilde{\boldsymbol{h}}_{t-1} + b_{hz} \tag{9}$$

$$\boldsymbol{n}_t = \tanh\left(\boldsymbol{W}_{in}\boldsymbol{x}_t + b_{in} + \boldsymbol{r}_t{}^{*}\boldsymbol{W}_{\text{hn}}\tilde{\boldsymbol{h}}_{\text{t}-1} + \text{b}_{\text{hn}}\right) \tag{10}$$

$$\tilde{\boldsymbol{h}}_t = (1 - \boldsymbol{z}_t) * \boldsymbol{n}_{\text{t}} + \boldsymbol{z}_{\text{t}} * \tilde{\boldsymbol{h}}_{\text{t}-1} \tag{11}$$

where $\tilde{\boldsymbol{h}}_t$ is the hidden state at step t, which is denoted by $\boldsymbol{f}_t$ in **Eq. 4** and $\boldsymbol{b}_t$ in **Eq. 5**. $\boldsymbol{r}_t$, $\boldsymbol{z}_t$, and $\boldsymbol{n}_t$ are the reset, update, and new gates, respectively. σ is the sigmoid function.

Transformer is a novel family of seq2seq model that works very differently than RNN type models. In the original Transformer (see **Figure 3**), a Feed Forward layer directly takes in the output from the Multihead attention layer $\boldsymbol{c}_{\text{att}}$, accompanied by a residual connection, shown in

$$\boldsymbol{c}_{\text{att}}^{(i)} = \text{MultiHeadAtt}\left(\boldsymbol{x}^{(i)}\right) + \boldsymbol{x}^{(i)} \tag{12}$$

$$\boldsymbol{x}^{(i+1)} = \boldsymbol{c}_{\text{att}}^{(i)} + \text{FeedForward}\left(\boldsymbol{c}_{\text{att}}^{(i)}\right) \tag{13}$$

In our model, we concatenated the neighboring elements in $\boldsymbol{c}_{\text{att}}$ before we feed it into the Feed Forward. As a result, the dimension of the first Linear layer in the Feed Forward layer has to change from $[d_{\text{model}} \times d_{\text{ff}}]$ to $[2d_{\text{model}} \times d_{\text{ff}}]$. Here we use the same notation as in Vaswani et al., (2017), where d_{model} is the size of input, output, and attention vectors and d_{ff} is the number of neurons in the Feed Forward layer. The residual connection also has to be changed accordingly; we tried two different approaches, simply averaging the neighboring element (**Eq. 14**) or concatenating the neighboring element and passing it through another affine transformation to recover its dimensions (**Eq. 15**). For simplicity, we denote the former method with subscript "ave" and the latter with subscript "aff".

$$\boldsymbol{x}_{t',\text{ave}}^{(i+1)} = \frac{\left(\boldsymbol{c}_{\text{att},2t}^{(i)} + \boldsymbol{c}_{\text{att},2t+1}^{(i)}\right)}{2} + \text{FeedForward}\left[\left(\boldsymbol{c}_{\text{att},2t}^{(i)}, \boldsymbol{c}_{\text{att},2t+1}^{(i)}\right)\right] \tag{14}$$

$$\begin{aligned}\boldsymbol{x}_{t',\text{aff}}^{(i+1)} &= \tanh\left[\boldsymbol{W}_{\text{aff}}\left(\boldsymbol{c}_{\text{att},2t}^{(i)}, \boldsymbol{c}_{\text{att},2t+1}^{(i)}\right) + b_{\text{aff}}\right] \\ &\quad + \text{FeedForward}\left[\left(\boldsymbol{c}_{\text{att},2t}^{(i)}, \boldsymbol{c}_{\text{att},2t+1}^{(i)}\right)\right]\end{aligned} \tag{15}$$

In our experiments, both methods show close performance. Therefore when showing the results, unless otherwise specified, we use the results of "ave" version.

3.4 Decoder and Attention Mechanisms

For our GRU models, we compared a regular multilayer unidirectional GRU:

$$\overline{\boldsymbol{h}}_t^{(i)} = \text{GRU}\left(\overline{\boldsymbol{h}}_{t-1}^{(i)}, \overline{\boldsymbol{h}}_t^{(i-1)}\right) \tag{16}$$

In our experiment, we did a comparison study on Bahdanau attention (**Eq. 17**) and different Luong attentions. Bahdanau attention is described in following set of equations.

$$u_{tk} = \left(\boldsymbol{W}_1\overline{\boldsymbol{h}}_t^{(M)} + b_1\right)^{\top}\left(\boldsymbol{W}_2\boldsymbol{h}_k^{(N)} + b_2\right) \tag{17}$$

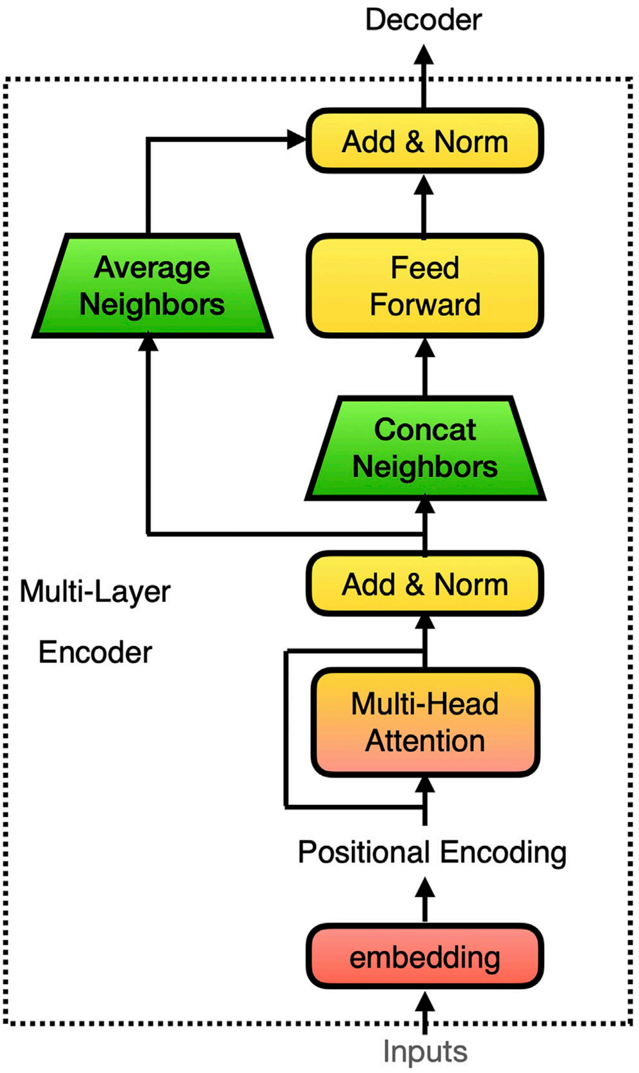


FIGURE 3 | The implementation of pyramid structure in Transformer's encoder.

$$\alpha_{tk} = \frac{u_{tk}}{\sum_j u_{tj}} \tag{18}$$

$$\boldsymbol{a}_t = \sum_j \alpha_{tj} \boldsymbol{h}_j^{(N)} \tag{19}$$

Here, u is the alignment score, h and $\bar{\boldsymbol{h}}$ denote the hidden state in encoder and decoder, respectively. M, N are the number of layers in decoder and encoder, respectively. a_t is the context vector, which will be concatenated with the decoder hidden state of last layer for predicting the next word $\hat{y}_t$.

Luong's global attentions are generalizations to Bahdanau attention, but using different alignment score calculation methods. For simplicity, we omit the superscript (M) and (N).

$$u_{tk} = \begin{cases} \bar{\boldsymbol{h}}_t^{\top} \boldsymbol{h}_k & \text{dot} \\ \bar{\boldsymbol{h}}_t^{\top} \boldsymbol{W}_a \boldsymbol{h}_k & \text{general} \\ \boldsymbol{v}_a^{\top} \tanh\left(\boldsymbol{W}_a \left[\bar{\boldsymbol{h}}_t, \boldsymbol{h}_k\right]\right) & \text{concat} \end{cases} \tag{20}$$

We also tried one example of Luong's local attention, which is done by imposing a Gaussian on **Eq. 19** at a desired attention center p_t:

$$\boldsymbol{a}_t = \sum_j \left(\alpha_{tj} \boldsymbol{h}_j\right) \exp\left(-\frac{(j - p_t)^2}{2\sigma^2}\right) \tag{21}$$

$$p_t = S \cdot \mathrm{sigmoid}\left(\boldsymbol{W}_p \bar{\boldsymbol{h}}_t\right) \tag{22}$$

where S denotes the total length of the hidden state from the last encoding layer and σ is a parameter chosen manually.

3.5 Beam Search

We use beam search in test and validation where text generation is involved. For each time step, we rank candidates based on their total negative logarithmic probability to current decoding time step t_{dec}:

$$\text{score} = -\sum_t^{t_{\mathrm{dec}}} \log\left(P(\hat{y})\right) \tag{23}$$

The search stops when there are five completed candidates.

3.6 Model Parameters

In all our experiments, we used a learnable embedding layer which embeds each "word" into a vector of length 400.

In our GRU models, we used a 3-layer bidirectional encoder; the size of the hidden states are 400 in all three layers. We used a 3-layer unidirectional decoder; the size of the hidden states are also 400.

In our Transformer models, following the original study, we used $d_{\text{model}} = 512$ and $d_{ff} = 2048$. We used 3-layer encoder and 3-layer decoder.

We did a coarse parameter space search to find these parameters chosen to be roughly optimal. But we did not fine-tune these parameters, because (1) we show that the overall performance of seq2seq model on PLC problem is satisfying and (2) we are more concerned about comparison between different attention mechanisms and between pyramid encoder and regular encoder.

4 DATASETS

We perform our experiments mainly on the Juliet Test Suite for C/C++ (v1.2) (created by NSA Center for Assured Software (2013)). This dataset contains 61,387 test cases, each test case contains one flawed code instance and one to several repaired code instance. These test cases contain more than 100 Common Weakness Enumerations (CWEs); each of them contains hundreds of example code instances. We note that the instances contain significant amount of dead code. To make the code more realistic, we remove the dead code. We also found that many of the code instances contain "if conditions", that, in the flawed code instance, executes one branch, while, in the repaired instance, executes the other. These instances are unrealistic; therefore, we removed them. We also performed function renaming. After the preprocessing, we obtained 31,082 pairs of good-bad code instances.

To test model's generality, for some of the models, we also tested their performance on Juliet Test Suite for Java (v1.3) (released by NSA Center for Assured Software (2018)). After similar preprocessing described above, we obtain 23,015 pairs of instances.

We did 4-fold cross-validation in all of our experiments to achieve statistically accurate results. An estimation of time and power consumption when running our experiments is provided in the **Supplementary Material** in a table, along with hardware requirements.

5 RESULTS

5.1 Repair Rate

We train our models on a GeForce GTX 1080 Ti graphic card. The metric we use for evaluation is the repair rate, which is the fraction of instances that are repaired after the model's edit. Since we performed beam search with beam width 5, each time a correction is being performed, we generate five correction candidates. Here we have two metrics in measuring the performance: one-candidate repair rate and five-candidate repair rate. The former corresponds to the scenario of code autocorrection, where there is no human judgment involved. The latter corresponds to correction suggesting, where the machine will identify an error and provide suggestions for the programmer for further judgment. The comparisons of the repair rates for considered models and their counterparts with pyramid encoder are listed in **Table 1** and **Table 2**. For comparison, we have attempted to test other machine-learning-based PLC tools that have been made. Gupta et al., (2018) take error messages while compiling as input, but our dataset focuses on logic flaws in programs that do not have syntax errors; therefore, this tool is not applicable. Pu et al., (2016) do not provide an open source repository, nor any documentations of their code. We have successfully trained Gupta et al., (2017) on our C/C++ dataset and included it in our work for comparison. Unfortunately, a tokenizer is required for preprocessing the data into a certain format, and they only provided that for C/C++, but not java.

From these results we see that pyramid encoder has close performance to regular encoder in most of the models we applied to, except for Luong's local attention. The reason is that the encoder output in pyramid encoder is very "coarse-grained"; each output position now represents information from $2^{(N-1)}$ words. This results in two drawbacks specifically to local attention: one, a much more "blurry" attention center and two, a much broader attention window. As a result, the attention is much less targeted, which damages the performance. Therefore, in the rest of the article, we will exclude this attention mechanism from our discussion.

5.2 Converging Speed

Since pyramid encoder reduces the sequence lengths in higher layers, one can expect a smaller training cost per batch in both GRU and Transformer models. To quantify this effect, for each of the regular encoder-pyramid encoder model pairs in **Table 1**, we set the same batch size and compare the average training speed in words per second, as shown in **Table 3**. Here the batch size is chosen so that it optimizes the training speed on the given GPU for each model. In the model, we also included number of epochs for the model to converge.

Apparently it takes similar number of epochs to converge for the same type of model with pyramid encoder and regular encoder. However, pyramid encoders largely increase the training speed, between 50 and 130%. Therefore it could easily shorten the training time by two to four folds while the same performance is achieved. As an example, **Figure 4** shows the learning curve for GRU model with general Luong's attention, comparing the regular encoder and pyramid encoder.

5.3 Memory Cost

The last thing we compared is the memory cost of the pyramid encoder and the regular encoder. This measure is crucial in some scenarios, where your input instances are very long; therefore, the memory of GPU is only capable of holding a very small batch. In code correction, this is often the case.

The metric we use for comparison is memory cost per instance, k, which is defined as

TABLE 1 | Repair rate of GRU and Transformer on Juliet Test Suite for C/C++, comparing the regular encoder and pyramid encoder. These results are averaged over a 4-fold cross-validation. We calculated the improvement of pyramid encoders compared to their nonpyramid pairs. Apparently pyramid encoder does not collaborate well with Luong's local attention; therefore, we exclude it from future discussions. It is also not included when calculating the average improvement.

Model	1-Candidate repair rate (%)		5-Candidate repair rate (%)	
	Regular encoder	Pyramid encoder	Regular encoder	Pyramid encoder
GRU + Bahdanau Att	76.92	76.09 (−0.83)	96.19	95.55 (−0.64)
GRU + Luong Att: Dot	74.38	73.04 (−1.34)	94.27	94.59 (+0.32)
GRU + Luong Att: General	75.79	74.85 (−0.94)	94.83	94.92 (+0.09)
GRU + Luong Att: Concat	50.34	47.26 (−3.08)	86.72	86.14 (−0.58)
GRU + Luong Att: Local	65.70	49.18 (−15.52)	92.46	86.24 (−6.22)
Transformer	75.48	72.39 (−3.09)	97.66	96.78 (−0.88)
Average improvement (%)	−1.95		−0.34	

TABLE 2 | Repair rate of GRU and Transformer on Juliet Test Suite for Java, comparing the regular encoder and pyramid encoder. We did not include result from DeepFix, because the provided data tokenizer only support C/C++.

Model	1-Candidate repair rate (%)		5-Candidate repair rate (%)	
	Regular encoder	Pyramid encoder	Regular encoder	Pyramid encoder
GRU + Bahdanau Att	54.65	56.21 (+1.56)	84.31	83.98 (−0.33)
GRU + Luong Att: Dot	54.30	55.66 (+1.36)	82.73	84.86 (+2.13)
GRU + Luong Att: General	53.15	52.54 (−0.61)	82.81	82.83 (+0.02)
GRU + Luong Att: Concat				
Transformer	56.68	57.35 (+0.67)	93.11	93.54 (+0.43)
Average improvement (%)	+0.74		+0.75	

TABLE 3 | Training speed of GRU and Transformer on Juliet Test Suite for C/C++.

Model	Batch size	Training speed (words/s)		Converge epoch	
		Regular	Pyramid	Regular	Pyramid
GRU + Bahdanau Att	8	754	1,185 (+57%)	18	18
GRU + Luong Att: General	16	441	853 (+108%)	23	27
GRU + Luong Att: Dot	128	4,646	10,408 (+124%)	36	34
GRU + Luong Att: Concat	6	1,418	2,344 (+65%)	23	29
Transformer	8	1,086	2,181 (+101%)	33	34

$$k = \frac{\Delta \text{Memory usage}}{\Delta \text{Batch size}} \tag{24}$$

Figure 5 shows the calculation process of k. Define $\mathcal{E} = 1/k$ as memory efficiency. We calculated the k and $\mathcal{E}$ value for each of the models we applied, shown in **Table 4**. We also included the number of parameters in each model, from which we see that each pair of models has roughly the same model size.

The pyramid encoder could increase the memory efficiency by 20%–600% depending on the attention mechanisms used, while only increase the memory occupied by the model itself by around 10%. One should note that the memory efficiency directly affects the maximum batch size one is able to use on a single GPU, and therefore affects the utility of the GPU. For example, for regular GRU with Bahdanau Attention, the memory of a GeForce GTX 1080 Ti graphic card can only support a batch size of 8, which does not fully utilize the GPU. With pyramid encoder, it can support up to 60 instances each batch. In practice, this will drastically reduce the training time by increasing the GPU utility, together with the smaller computational cost of pyramid encoder as addressed in previous section.

6 DISCUSSION

6.1 Length Analyses

Figure 6 shows the repair rate of the models with respect to the input length. We omitted the result of Transformer, Bahdanau's attention, and Luong's general attention, because they are qualitatively similar to the result of Luong's dot attention. Despite the different attention mechanisms, these seq2seq models (with pyramid encoder or regular encoder) are relatively robust to longer input lengths. The performance drops at around 250 words and above 500 words are likely resulting from the shortage of samples, which one can easily observe from **Figure 7**, the length histogram of source instances and target instances. The histogram also shows that the majority

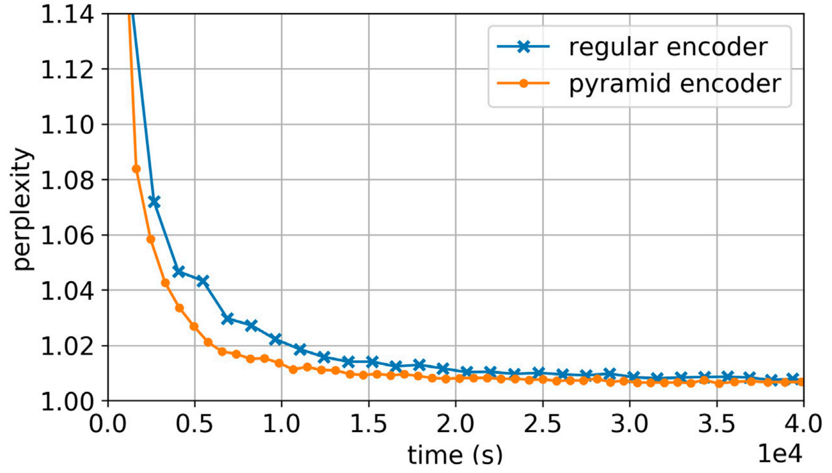


FIGURE 4 | Learning curve of GRU with Luong's general attention, comparing regular encoder to pyramid encoder. Pyramid encoder model shows fast converging speed.

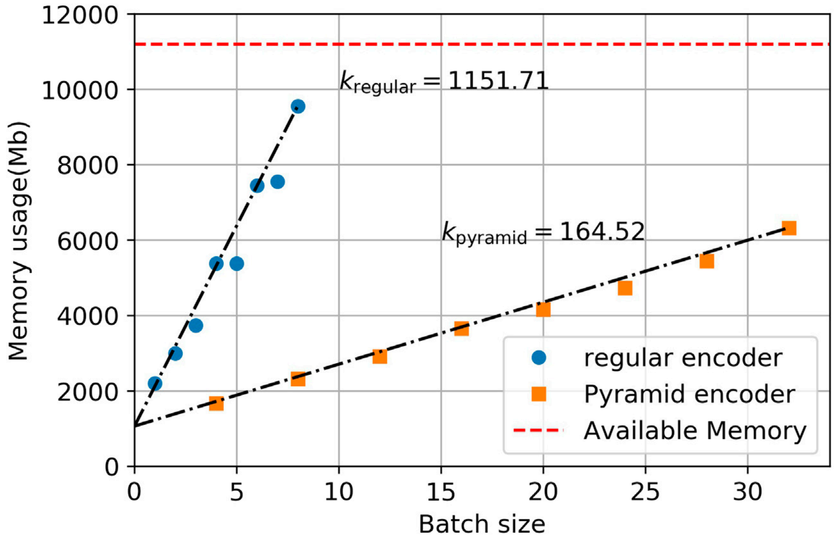


FIGURE 5 | Memory cost per instance for GRU models with Bahdanau attention, *k* is calculated by finding the slope of the linear fit (black dashed line). The red dashed line represents the maximum memory of a GeForce GTX 1080 Ti graphic card.

TABLE 4 | Memory cost for considered models, comparing regular encoder and pyramid encoder: pyramid encoder greatly increased the memory efficiency.

Model	**k (Mb/instance)**		$\mathcal{E}$ **(10^{-3})**		**Parameters (10^7)**	
	Regular	**Pyramid**	**Regular**	**Pyramid**	**Regular**	**Pyramid**
GRU + Bahdanau Att	1,151.71	164.52	0.86	6.08 (+600%)	1.24	1.11
GRU + Luong Att: General	830.71	165.03	1.20	6.05 (+403%)	1.22	1.10
GRU + Luong Att: Dot	65.91	52.42	15.17	19.08 (+26%)	1.20	1.08
GRU + Luong Att: Concat	1,381.6	431.87	0.72	2.31 (+220%)	1.24	1.11
Transformer	414.67	263.33	0.24	0.38 (+57%)	2.35	2.82

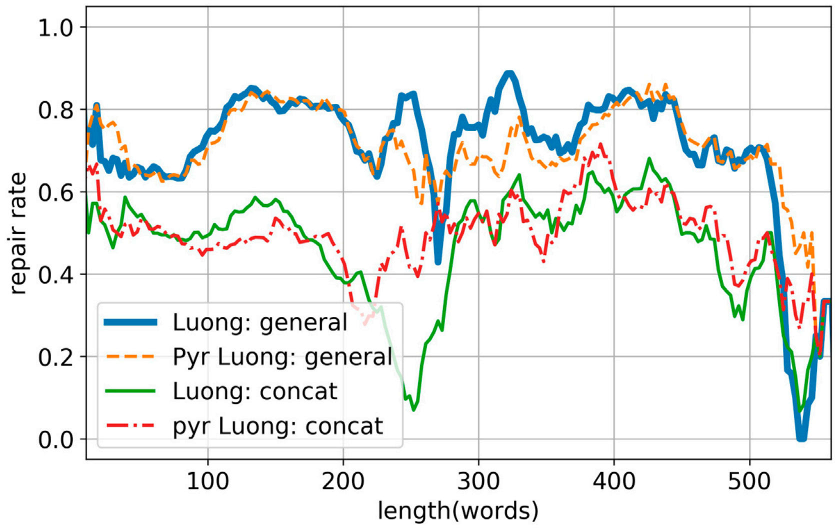


FIGURE 6 | Length analyses of Luong's general attention and Luong's concat attention. The results from the rest of the models are qualitatively similar to result of Luong's general attention and thus are omitted.

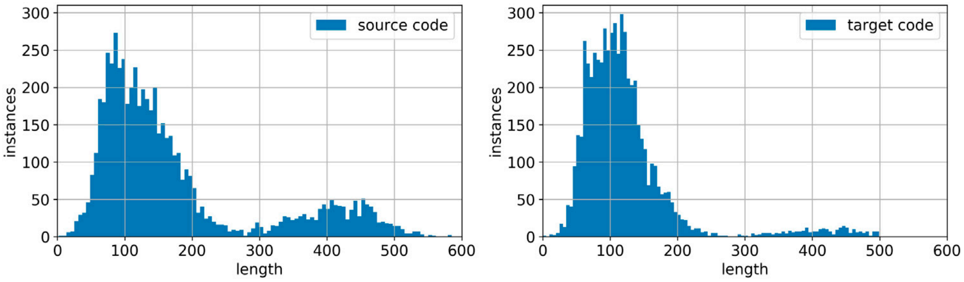


FIGURE 7 | Histogram of flawed code (left) and repaired code (right) instances.

of code instances contains several hundred words, while natural language sentences are typically not longer than 50 words. This feature of code instances calls for a much higher computational resource requirement for PLC problems than NLC problems, which makes pyramid structure especially useful.

6.2 Examples of Correction

In this section we give several examples of successful corrections from our Pyramid GRU model on Juliet C/C++ Test Suite for closer examination of model and datasets. The red striked out texts denote the original faulted instance, and blue buffed texts are the reparation done by the model.

Example 1: Memory allocation match

The flawed code creates a char variable whose size does not match its concatenating destination. The model is able to correct it so that their size matches each other.

```
   void main(){
       char * data;
       char data_buf[100];
       data=data_buf;
       ~~memset(data,’A’, 100-1);~~
       memset(data,’A’, 50-1);
       data[100-1]=’\0’;
   {
       char dest[50]="";
       strncat(dest,data,strlen(data));
       dest[50-1]=’\0’;
       printLine(data);
   }
}
```

Example 2: Redundant Code

This is an example that the model deletes repeated code where a variable is freed twice.

```
void main(){
    char * data;
    data=NULL;
    data=(char *)malloc(100*sizeof(char));
    free(data);
    ~~free(data);~~
}
```

Example 3: Possible Overflow

Here we show a slightly questionable example of correction provided by the dataset. In order to prevent potential string overflow emerging from environment variable, the repair suggestion given by the Juliet Test Suite is to abort the entire part of concatenating the environment string and replace the variable with an arbitrary string "*.*". This "correction" is easy for the model to learn; however, it has changed the original purpose of the program.

```
void main(){
    char * data;
    char data_buf[100]="";
    data=data_buf;
    ~~size_t data_len=strlen(data);~~
    ~~char * environment=GETENV(ENV_VARIABLE);~~
    ~~if(environment!=NULL){~~
        ~~strncat(data+data_len,~~
                ~~environment,100-data_len-1);~~
    ~~}~~
    strcat(data,"*.*");
    _spawnl(_P_WAIT,COMMAND_INT_PATH,
            COMMAND_INT_PATH,
            COMMAND_ARG1,COMMAND_ARG2,
            COMMAND_ARG3,NULL);
}
```

Example 4: Correction Across Functions

In this example, the models demonstrate the ability of making connections across the whole instance, between different functions. Here it prevents potential overflow in the sink function caused by a variable that was passed from the main function by adding an "if condition".

```
static void sink(unsigned char data)
{
    if (data < UCHAR_MAX){
        unsigned char result = data + 1;
    }
}
void main()
{
    unsigned char data;
    data = ’ ’;
    data = (unsigned char)rand();
    sink(data);
}
```

6.3 Generalizability to Syntax Error-Oriented Dataset

In the spirit of comparative study, we attempted to compare our method to Deepfix (Gupta et al., 2017), the only machine-learning-based PLC method that made their code and dataset open to the public, to the best of our knowledge. Unfortunately, the attempt of applying Deepfix onto Juliet Test Suite has failed, because Deepfix is aimed only to correct syntax errors and a compiler is used as the evaluator, marking any programs that could pass the compiling stage as "correct". This apparently contradicts the spirit of "identify logic errors from syntactically correct programs".

The difficulty that we are facing here comes from a more general problem in the field of Machine Learning PLC; the field is still disorganized and works in the field are uncorrelated. Each group might be using their own dataset and design their systems to match the specific purpose of that dataset. Comparative work is difficult to conduct not only because the datasets are hard to obtain due to private policies, but also because issues raised in PLC are versatile; each model is designed and optimized to best address the problem occurring in their particular dataset.

For the above reasons, we had to back off to a weaker comparative study, using seq2seq models on the dataset from Deepfix. Deepfix uses a generated dataset, originated from students' submission to an introductory C course in a web-based tutoring system (Das et al., 2016). For each student submission, they generate up to five syntax errors in the code instance, including replacing "}; " with "; }", deleting a semicolon, add an extra "}", replacing a semicolon with a period, and replacing a comma with a semicolon. If all of the syntax errors were fixed, then consider such a program as successfully repaired.

TABLE 5 | Comparison of our models with Deepfix, Gupta et al., (2017) on Deepfix dataset. All results are average of 5-fold cross validation.

Model	1-Candidate repair rate (%)		5-Candidate repair rate (%)	
	Regular encoder	Pyramid encoder	Regular encoder	Pyramid encoder
Transformer	51.96	43.78	67.16	59.32
GRU + Luong Att: General	51.86	34.80	66.33	48.44
GRU + Luong Att: Dot	58.63	41.09	72.31	54.47
GRU + Bahdanau Att	27.47	15.21	36.19	22.59
Deepfix	56			

Table 5 shows the comparison of repair rate of our seq2seq models compared to the method applied by Deepfix. We observe that pyramid encoder performs worse than regular encoder on this particular dataset. This is expected from how the dataset was generated. The generated syntax errors are extremely local in Deepfix's dataset. The fix usually only involves changing one token or two neighboring tokens, leaving the rest of the entire code piece unchanged. Therefore, while a pyramid encoder summarizes the information from neighboring tokens, it also blurs the local information.

We also observed that, in Luong's attention, dot has the best performance in this dataset and Bahdanau's attention performs the worst. After observing the dataset carefully, we came up with the following hypothesis: in this dataset, the network is only required to simply **copy** the original token most part of the instance and locally fix one or two tokens. This means that in the majority of times, for each decoder hidden state $\overline{\boldsymbol{h}}_t$, the normalized attention score score$(\overline{\boldsymbol{h}}_t, \boldsymbol{h}_{t'})$ needs to be close to one where $t = t$' and close to 0 everywhere else. In Luong's attention, a dot, which simply do an inner product of hidden states, could do the job easier, because latent vectors are mostly orthogonal to each other in the latent space due to high dimensionality. On the other hand, Bahdanau attention, which does an affine transformation to every hidden state $\boldsymbol{h}_t$, may overcomplicate the problem and fail to capture the correct attention.

6.4 Alternative Method for Small Datasets: Transfer Learning

One main difficulty that researchers often come across when attempting to apply machine learning methods to PLC problems is the availability of suitable datasets. Although there are many datasets and shared tasks available on Software Assurance Reference Dataset (2006), most of them include less than 1,000 examples. This makes neural-network-based methods nearly impossible. To tackle this problem, we take the idea of transfer learning from Pan and Yang (2009).

Our idea is to take the encoder part of the model that was trained on Juliet Test Suite and attach it to a untrained decoder, which was designed for the specific problem. We aim to take the advantage that codes written in the same coding language share the same syntax library and same construction rules.

Since many datasets available only provide the faulted code and their corresponding fault categories, here we give an example of fault classification using transfer learning, applying the model pretrained on Juliet Test Suite for C/C++ on ITC benchmark (Charles (2015)).

TABLE 6 | Comparison of the result of transfer learning on error type classification task. The models without transfer learning demonstrate no predicting power and no improvement during course of training.

Model	Accuracy (%)
Transfer learning: PyrGRU	60.5
Transfer learning: PyrTFM	69.1
Fresh pyramid GRU	16.7
Fresh pyramid transformer	7.1

6.4.1 Model Structure

Given a faulted code instance, our goal is to train a classification model that predicts the type of error of the faulted code from a given list of error categories.

We keep the encoder part of the pretrained model and use it directly as the encoder in the classification problem. The exception is the embedding layer, because the vocabulary in the new dataset will contain new variable names that did not occur in pretrained embedding, although the syntax will be the same. In practice, we manually expanded the embedding layer to accommodate the new "words" but keep the embeddings of the old "words" unchanged. In order to add variation from the original model, we also reinitialized the weights in the last encoding layer.

For the decoder, instead of generating a sequence, we take the output of the first time step of the reinitiated decoder and pass it to a linear layer that projects it to an n_{class} dimensional vector. n_{class} is the number of error classes. Model was trained to minimize cross-entropy loss with an ADAM optimizer.

6.4.2 Results

We extracted 566 C/C++ code instances from the ITC bench mark. These instances are organized into 44 error categories, with the largest category containing around 30 instance and the smallest only containing two instances. Then the instances are divided into a training set of 485 instances, a validation set of 42 instances, and test set of 39 instances. For comparison, we also tried Pyramid GRU and Pyramid Transformer with the same model structure but no prior knowledge from Juliet Test Suites. The result is shown in **Table 6**.

For the fresh GRU and Transformer models, we observed that the models have no predicting power as it produces constant prediction over all inputs. There is even no sufficient gradient on the loss landscape as the loss did not reduce during the training. Transfer learning, on the other hand, demonstrates a fair power of prediction, correctly classifying over 60% of instances, despite that ITC benchmark is written in very different style than Juliet Test Suites and that the dataset is 50 times smaller.

This result shows that one is able to use neural-network-based methods in code correction problems despite the shortage of data, which is a common problem in this field.

7 CONCLUSION

In our work, we show that seq2seq models, successful in natural language correction, are also applicable in programming language correction. Our results shows seq2seq models can be well applied in providing suggestions to potential errors and have a decent correct rate (above 70% in C/C++ dataset and above 50% in Java dataset) in code auto-correction. Although these results are only limited in Juliet Test Suites, we expect that, given sufficient training data, seq2seq models can also perform well when applied on other PLC problems.

Based on the commonly used encoder-decoder structure, we introduce a general pyramid encoder in seq2seq models. Our results demonstrates that this structure significantly reduces the memory cost and computational cost. This is helpful because PLC are generally more computationally expensive than NLC, due to its longer average instance length.

The publicly available datasets in PLC are mostly small and noisy. Most datasets we found contain close to or less than 1,000 code instances. This is far less than enough for training seq2seq and many other machine learning models. Our results on transfer learning pointed out a way of processing these small dataset using the pretrained model as an encoder, which boosts the performance by a large amount.

In future, we will further investigate the influence of different architectures in neural networks, for instance, parallel encoders/decoders, Tree2Tree models, etc. On the other hand, instead of code correction, we will modify and examine our model's performance on other tasks such as program generation and code optimizing. We will also examine the potential difference between artificial datasets and realistic datasets.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. This data can be found here: https://samate.nist.gov/SARD/around.php#juliet_documents https://samate.nist.gov/SARD/view.php?tsID=104.

AUTHOR CONTRIBUTIONS

SH came up with the generalized pyramid encoder, processed the dataset, programmed each of the seq2seq models, conducted experiments, respectively, and gathered results. He was responsible for writing parts 3, 4, 5, and 6 of the manuscript. XZ was responsible for literature reviews; he wrote part 2 of the manuscript independently and part 1 and 6 jointly with SH. He and SH also contributed together in finding supplementary datasets. SC was the advisor of the project; he gave advice on the general direction of the research, provided facilities to conduct experiments, and supervised the process of the research. He also provided the access to Juliet Test Suite, which was the main dataset used in the research. He helped proofread the manuscript. All three authors shared ideas, carried out discussions, and came up with solutions together over the course of research.

FUNDING

The study was funded by the National Science Foundation, DMS 1737897.

ACKNOWLEDGMENTS

This manuscript has been released as a pre-print at Arxiv (Huang et al., 2020).

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frai.2021.590215/full#supplementary-material.

REFERENCES

Charles, O. (2015). [Dataset] Itc-benchmarks. Available at: https://samate.nist.gov/SARD/testsuite.php (Accessed December 28, 2015).

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., et al. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. Preprint repository name [Preprint]. Available at: arXiv:1406.1078 (Accessed January 3, 2014).

Das, R., Ahmed, U. Z., Karkare, A., and Gulwani, S. (2016). Prutor: a system for tutoring cs1 and collecting student programs for analysis. Preprint repository name [Preprint]. Available at: arXiv:1608.03828 (Accessed August 12, 2016).

Google (2016a). Clang-tidy. Available at: http://clang.llvm.org/extra/clang-tidy/ (Accessed April 23, 2016).

Google (2016b). Error-prone. Available at: http://errorprone.info/ (Accessed January 25, 2016).

Gupta, R., Kanade, A., and Shevade, S. (2018). Deep reinforcement learning for programming language correction. Preprint repository name [Preprint]. Available at: arXiv:1801.10467 (Accessed January 31, 2018).

Gupta, R., Pal, S., Kanade, A., and Shevade, S. (2017). "Deepfix: fixing common c language errors by deep learning," in Proceedings of the thirty-first AAAI conference on artificial intelligence, San Francisco, California, February 4–9, 2017, (AAAI Press) 1345–1351.

Hao, K. (2019). Training a single ai model can emit as much carbon as five cars in their lifetimes. Available at: https://www.technologyreview.com/s/613630/

training-a-single-ai- model-can-emit-as-much-carbon-as-five-cars-in-their-lifetimes/ (Accessed September 28, 2019).

Huang, S., Zhou, X., and Chin, S. (2020). A study of pyramid structure for code correction. Preprint repository name [Preprint]. Available at: arXiv:2001.11367 (Accessed January 28, 2020).

JetBrains (2016). ReSharper. Available at: https://www.jetbrains.com/resharper/ (Accessed September 12, 2016).

NSA Center for Assured Software (2013). [Dataset] Juliet test suite c/c^{++}. Available at: https://samate.nist.gov/SARD/around.php#juliet_documents (Accessed May 15, 2013).

NSA Center for Assured Software (2018). [Dataset] Juliet test suite java. Available at: https://samate.nist.gov/SARD/around.php#juliet_documents (Accessed November 17, 2018).

Pan, S. J., and Yang, Q. (2009). A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* 22, 1345–1359. doi:10.1109/TKDE.2009.191

Pu, Y., Narasimhan, K., Solar-Lezama, A., and Barzilay, R. (2016). "sk_p: a neural program corrector for moocs," in Companion proceedings of the 2016 ACM SIGPLAN international conference on systems, programming, languages and applications: software for humanity (SPLASH Companion), 39–40.

Singh, R., Gulwani, S., and Solar-Lezama, A. (2013). "Automated feedback generation for introductory programming assignments," in Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation, Washington, DC, June 06, 2013 (ACM), 15–26.

Software Assurance Reference Dataset (2006). [Dataset] SARD datasets. Available at: https://samate.nist.gov/SARD/testsuite.php (Accessed January 6, 2006).

Synopsys (2016). Coverity. Available at: http://www.coverity.com// (Accessed July 11, 2016).

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*. 6000–6010.

Xie, Z., Avati, A., Arivazhagan, N., Jurafsky, D., and Ng, A. Y. (2016). Neural language correction with character-based attention. Preprint repository name [Preprint]. Available at: arXiv:1603.09727 (Accessed March 31, 2016).

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: 04 May 2021
doi: 10.3389/fdata.2021.589417

MARGIN: Uncovering Deep Neural Networks Using Graph Signal Analysis

Rushil Anirudh[1]*, *Jayaraman J. Thiagarajan*[1]*, *Rahul Sridhar*[2] *and Peer-Timo Bremer*[1]

[1] *Center for Applied Scientific Computing (CASC), Lawrence Livermore National Laboratory, Livermore, CA, United States,*
[2] *Walmart Labs, California, CA, United States*

Interpretability has emerged as a crucial aspect of building trust in machine learning systems, aimed at providing insights into the working of complex neural networks that are otherwise opaque to a user. There are a plethora of existing solutions addressing various aspects of interpretability ranging from identifying prototypical samples in a dataset to explaining image predictions or explaining mis-classifications. While all of these diverse techniques address seemingly different aspects of interpretability, we hypothesize that a large family of interepretability tasks are variants of the same central problem which is identifying *relative* change in a model's prediction. This paper introduces MARGIN, a simple yet general approach to address a large set of interpretability tasks MARGIN exploits ideas rooted in graph signal analysis to determine influential nodes in a graph, which are defined as those nodes that maximally describe a function defined on the graph. By carefully defining task-specific graphs and functions, we demonstrate that MARGIN outperforms existing approaches in a number of disparate interpretability challenges.

Keywords: graph signal processing, interpretability, influence sampling, adversarial attacks, machine learning

OPEN ACCESS

Edited by:
Xue Lin,
Northeastern University, United States

Reviewed by:
Bernhard C. Geiger,
Know Center, Austria
Akshay Mehra,
Tulane University, United States

****Correspondence:***
Rushil Anirudh
anirudh1@llnl.gov
Jayaraman J. Thiagarajan
jjayaram@llnl.gov

Specialty section:
This article was submitted to
Machine Learning and
Artificial Intelligence,
a section of the journal
Frontiers in Big Data

Received: *30 July 2020*
Accepted: *14 January 2021*
Published: *04 May 2021*

Citation:
Anirudh R, Thiagarajan JJ, Sridhar R and Bremer P-T (2021) MARGIN: Uncovering Deep Neural Networks Using Graph Signal Analysis. Front. Big Data 4:589417. doi: 10.3389/fdata.2021.589417

INTRODUCTION

With widespread adoption of deep learning solutions in science and engineering, obtaining post-hoc interpretations of the learned models has emerged as a crucial research direction. This is driven by a community-wide effort to develop a new set of meta-techniques able to provide insights into complex neural network systems, and explain their training or predictions. Despite being identified as a key research direction, there exists no well-accepted definition for interpretability. Instead, in different contexts, it may refer to a variety of tasks ranging from debugging models (Ribeiro et al., 2016), to determining anomalies in the training data (Koh and Liang, 2017). While some recent efforts (Lipton, 2016; Doshi-Velez and Kim, 2017) provide a more formal definition for interpretability as generating *interpretable rules*, these focus on instance-level explanations, i.e. understanding how a network arrived at a particular decision for a single instance. In practice, interpretability covers a wider range of challenges, such as characterizing data distributions and separating hyperplanes of classifiers, identifying noisy labels during training, detecting adversarial attacks, or generating saliency maps for image classification. As discussed below, solutions to all such problems have been proposed each using custom tailored, task-specific approaches. For example, a variety of tools aim to explain which parts of an image are the most responsible for a prediction. However, these cannot be easily re-purposed to identify which samples in a dataset were most helpful or harmful to train a classifier.

Instead, we argue that many existing interpretability techniques solve a variant of essentially the same task–understanding *relative* changes in the model's prediction, where the changes are either *global* in nature, i.e., across an entire distribution or *local*, i.e., within a single sample. In this paper, we

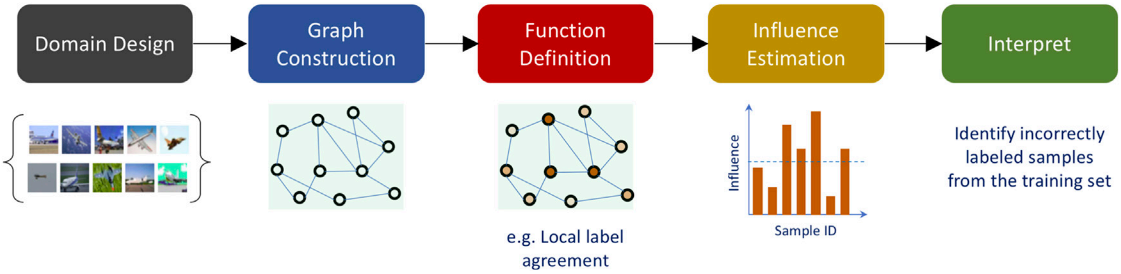


FIGURE 1 | MARGIN—An overview of the proposed protocol for post-hoc interpretability tasks. In this illustration, we consider the problem of identifying incorrectly labeled samples from a given dataset. MARGIN identifies the most important samples that need to be corrected so that fixing them will lead to improved predictive models.

propose the MARGIN (Model Analysis and Reasoning using Graph-based Interpretability) framework, which directly applies to a wide variety of interpretability tasks. MARGIN poses each task as an *hypothesis* test and derives a measure of *influence* that indicates which parts of the data/model maximally support (or contradict) the hypothesis. More specifically, for each task we construct a graph whose nodes represent entities of interest, and define a function on this graph that encodes a hypothesis. For example, if the task is to determine which samples need to be reviewed in a dataset containing noisy labels, the domain is the set of samples, while the function can be local label agreement that measures how misaligned are the neighborhoods of the input samples (or their features) and their corresponding labels. Using graph signal processing (Sandryhaila and Moura, 2013; Shuman et al., 2013) one can then identify which nodes are essential to reconstructing the chosen function (hypothesis), which most likely will correspond to those with flipped labels. In order words, through a careful selection of graph construction strategies and hypothesis functions, this general procedure can be used to solve a wide-range of post-hoc interepretability tasks.

This generic formulation, while extremely simple in its implementation, provides a powerful protocol to realize several meta-learning techniques, by allowing the user to incorporate rich semantic information, in a straightforward manner. In a nutshell, the proposed protocol is comprised of the following steps: 1) identifying the domain for interpretability (for e.g. intra-sample vs inter sample), 2) constructing a neighborhood graph to model the domain (for e.g. pixel space vs. latent space), 3) defining an *explanation function* at the nodes of the graph, 4) performing graph signal analysis to estimate the *influence* structure in the domain, and 5) creating interpretations based on the estimated influence structure. **Figure 1** illustrates the steps involved in MARGIN for *a posteriori* interpretability.

Overview

Using different choices for graph construction and the explanation function design, we present five case studies to demonstrate the broad applicability of MARGIN for *a posteriori* interpretability. First, in *Case Study I—Prototypes and Criticisms* we study a unsupervised problem of identifying samples which well characterize the underlying data distribution, referred to as prototypes and criticisms respectively (Kim et al., 2016). We show that the MARGIN is highly effective at characterizing data distributions and can shed light into the regimes where classifier performance can suffer. In *Case Study II—Explanations for Image Classification*, we obtain pixel-level explanations from an image classifier using MARGIN, without the need to access the model internals, i.e., black-box and show that the inferred feature importance estimates are meaningful. In *Case Study III—Detecting Incorrectly Labeled Samples*, we employ MARGIN to identify label corruptions in the training data and demonstrate significant improvements over popular approaches such as influence functions. In *Case Study IV—Interpreting Decision Boundaries*, we illustrate the application of MARGIN in analyzing pre-trained classifiers and identifying the most influential samples in describing the decision surfaces, akin to memorable examples in continual learning (Pan et al., 2020). Finally, in *Case Study V—Characterizing Statistics of Adversarial Examples* we extend two recently proposed statistical techniques to detect adversarial examples from harmless examples, and demonstrate that incorporating them inside MARGIN improves their discriminative power significantly.

RELATED WORK

We outline recent works that are closely related to the central framework, and themes around MARGIN. Papers pertinent to individual case studies are identified in their respective sections. Our goal in this paper is to design a core framework that is capable of being repurposed to interpretability tasks, ranging from explaining decisions of a predictive model, detecting outliers to identifying label corruptions in the training data. While post-hoc explanation methods are the *modus-operandi* in interpreting the decisions of a black box model, their scope has widened significantly in the recent years. For example, popular sensitivity analysis such as LIME (Ribeiro et al., 2016) and SHAP (Lundberg and Lee, 2017) or gradient-based methods such as Saliency Maps (Simonyan et al., 2013), Integrated Gradients (Sundararajan et al., 2017), Grad-CAM (Selvaraju et al., 2017), DeepLIFT (Shrikumar et al., 2017) and DeepSHAP (Lundberg and Lee, 2017) are routinely used to produce sample-wise, local explanations by measuring the sensitivity of the black-box to perturbations in the input features (Fong and Vedaldi, 2017). Despite their wide-spread use, they cannot be readily utilized to obtain dataset-level explanations, e.g., which are the most influential examples in a dataset for a given test sample, or to detect distribution shifts (Thiagarajan et al., 2020). On the other hand, in (Koh and Liang, 2017), the authors proposed a strategy to select influential samples by extending ideas from robust statistics, which was shown to be applicable to a variety of scenarios. However, such methods cannot be used for obtaining feature importance estimates. Another important challenge with most existing post-hoc explanation techniques is their computational complexity. In contrast, MARGIN leverages the generality of graph structures to scalably generate explanations, and through of use of appropriate hypothesis functions can support a large-class of interpretations.

In a nutshell, MARGIN reposes the problem of generating explanations as an influential node selection problem, wherein the node can correspond to a sample-level or feature-level explanations and the influence is measured based on a hypothesis function. Defining suitable objectives for detecting influential features in an image or influential samples in a dataset has been an important topic of research in explainable AI. For example, CXPlain (Schwab and Karlen, 2019) and Attentive Mixture of Experts (Schwab et al., 2019) utilize a Granger-causality based objective to quantify feature importances. In addition, prediction uncertainties Chakraborty et al. (2017) or even loss estimates Thiagarajan et al. (2020) have been widely adopted to characterize vulnerabilities of a trained model. Note that, MARGIN can directly use any of these objectives to choose the most relevant explanations. In this paper, we consider a variety of interpretability tasks and recommend suitable hypothesis functions for each of the tasks.

Since MARGIN relies on ideas from graph signal processing (GSP) to select the most relevant explanations, we briefly review existing work in this area. Broadly, there are two classes of approaches in GSP–one that builds on spectral graph theory using the graph Laplacian matrix (Shuman et al., 2013), and the other based on algebraic signal processing that builds upon the graph shift operator (Sandryhaila and Moura, 2013). While both are applicable to our framework, we adopt the latter formulation. Our approach relies on defining a measure of influence at each node, which is related to sampling of graph signals. This is an active research area, with several works generalizing ideas of sampling and interpolation to the domain of graphs, such as (Pesenson, 2008; Gadde et al., 2014; Chen et al., 2015).

A GENERIC PROTOCOL FOR INTERPRETABILITY

In this section, we provide an overview of the different steps of MARGIN and describe the proposed influence estimation technique in the next section.

Domain Design and Graph Construction

The domain definition step is crucial for the generalization of MARGIN across different scenarios. In order to enable instance-level interpretations (e.g. creating saliency maps), a single instance of data, possibly along with its perturbed variants, will form the domain; whereas a more holistic understanding of the model can be obtained (e.g. extracting prototypes/criticisms) by defining the entire dataset as the domain. Regardless of the choice of domain, we propose to model it using nearest neighbor graphs, as it enables a concise representation of the relationships between the domain elements.

More specifically, given the set of samples $\{x_i\}$, we construct a k-nearest neighbor domain graph that captures local geometry of the data samples. The metric for graph construction (that determines neighborhoods/edges) can arise from prior knowledge about the domain or designed based on latent representations from pre-trained models. For example, if we use the latent features from AlexNet (Krizhevsky et al., 2012), the resulting graph respects the distance metric inferred by AlexNet for image classification. Though the difficulty in choosing an appropriate k for designing robust graphs is well known, designing better graphs is beyond the scope of this paper. In our experiments, we find that our results are not very sensitive to the choice of k.

Formally, an undirected weighted graph is represented by the triplet $\mathcal{G} = (\text{v}, \text{E}, W)$, where v denotes the set of nodes, E denotes the set of edges and W is an adjacency matrix that specifies the weights on the edges, where $W_{n,m}$ corresponds to the edge weight between nodes v_n and v_m. Let $N_n = \{m \big| W^1_{n,m} 0\}$ define the neighborhood of node v_n, i.e. the set of nodes connected to it. The normalized graph Laplacian, L, is then constructed as $I - D^{-1/2} W D^{-1/2}$, where $D_{nn} = \sum_m W_{n,m}$ is the degree matrix and I denotes the identity matrix.

Explanation Function Definition

A key component of MARGIN is to construct an explanation function that measures how well each node in the graph supports the presented hypothesis. The function acts on each vertex of the graph as: $f(n) : v_n a R$ for all n vertices in the graph $\mathcal{G}$. This function is also referred to as the graph signal defined on the graph domain. We expect this function to capture properties of the explaination that are deemed important. Let us illustrate this process with an example–in order to create saliency maps for image classification, one can build a graph where each node corresponds to a potential explanation (i.e. a subset of pixels), while the edges can measure how likely can two explanations produce similar predictions. In such a scenario, one can hypothesize that an *ideal* explanation will be *sparse*, in terms of the number of pixels, since that is more interpretable. Consequently, the size of an explanation can be used as the function. *Case Studies* will present a more detailed discussion.

Influence Estimation

This is the central analysis step in MARGIN for obtaining influence estimates at the nodes of $\mathcal{G}$, that can reveal which nodes can maximally describe the variations in the chosen explanation function. Implicitly, this step can be viewed as a *soft*-sample selection strategy with respect to the structure induced by the domain graph. We propose to perform this estimation using tools from graph signal analysis. *Proposed*

Influence Estimation describes the proposed algorithm for influence estimation.

From Influence to Interpretation

Depending on the hypothesis chosen for *a posteriori* analysis, this step requires the design of an appropriate strategy for transferring the estimated influences into an interpretable explanation.

PROPOSED INFLUENCE ESTIMATION

Given a nearest neighbor graph $\mathcal{G}$ along with an explanation function f, we propose to employ graph signal analysis to estimate node influence scores. Before we describe the algorithm, we will present a brief overview of the preliminaries.

Definitions. We use the notation and terminology from (Sandryhaila and Moura, 2013) in defining an operator analogous to the *time-shift* or *delay* operator in classical signal processing. The function f assigns a scalar value to each vertex as defined earlier, as a result the entire function is written as $f : \nu \mapsto R^N$, where $|\nu| = N$, i.e., f is a collection of scalar values at each vertex, ordered according to the same order of vertices in the graph. When the graph does not have any special structure (i.e., it is Euclidean), f is nothing but a vector valued function. We consider the simplest scenario here where the function only takes a scalar value at each node, however more general cases maybe considered where the value at each node is multi-dimensional. During a graph shift operation, the function $f(n)$ at node v_n is replaced by a weighted linear combination of its neighbors: $\widehat{f} = Af$, where A is the graph shift operator, which is the simplest, non-trivial graph filter. Commonly used choices for A include the adjacency matrix W, transition matrix $D^{-1}W$ and the graph Laplacian L.

The set of eigenvectors of the graph shift operator is referred to as the graph Fourier basis, $A = U\Lambda U^T$, where $U \in \mathbb{R}^{N\times N}$, and the Fourier transform of a signal $f \in \mathbb{R}^N$ is defined as $U^T f$. The ordered eigenvalues corresponding to these eigenvectors represent frequencies of the signal, with λ_1 to λ_N representing the smallest to largest frequencies. The notion of frequency on the graph corresponds to the rate of change of the function across nodes in a neighborhood. A higher change corresponds to a high frequency, while a smooth variation corresponds to a low frequency. In this context, the graph filtering using a graph shift operator corresponds to a *low-pass* filter that dispenses high frequency components in the function. Similarly, a simple *high-pass* filter can be easily designed as $\widehat{f_h} = f - \widehat{f}$.

Algorithm 1 MARGIN's simple influence estimation.

```
procedure MARGIN-INFLUENCE(X,$G$, f)          ▷ Domain, Graph and explanation function
    Construct graph shift operator A from X
    $\hat{\mathbf{f}} = \mathbf{f} - \mathbf{Af}$                            ▷ High pass spectral filter
    for $i \in \mathcal{V}$ do                    ▷ Iterate over all the nodes of the graph $\mathcal{G}$
        Compute $I(i) = ||\hat{\mathbf{f}}(i)||_2^2 \ \forall i \in \mathcal{V}$
    return $I(i) \forall i \in \mathcal{V}$                        ▷ Influence score for each node
```

Algorithm: The overall procedure to obtain influence scores at the nodes of $\mathcal{G}$ can be found in **Algorithm 1**. Intuitively, we design a high-pass filter that eliminates the low frequency content and retains the signal energy only at those nodes that characterize the extreme variations of the function. Following the high-pass filtering step, the influence score at a node is estimated as the magnitude of the filtered function value at that node:

$$I(i) = \left\| \widehat{f_h}(i) \right\|_2^2 \forall i \in \nu, \tag{1}$$

where $\widehat{f_h}$ corresponds to the high-pass filtered version of f. Interestingly, we find that analyzing the high frequency components of the explanation function often leads to a sparse influence structure, indicating the presence of multiple local optima that corroborate the hypothesis. Conversely, the influence structure obtained from low frequency components is typically dense and hence requires additional processing to qualify regions of disagreement.

Sensitivity to Graph Construction

A critical step in MARGIN is the graph construction process for datasets that do not naturally have a graph structure. In this work, we rely on a simple nearest neighbor graph for construction which can vary depending on the size of the neighborhood. This is a hyper parameter that must be set with validating examples, and in all our case studies we found a neighborhood size of 20-40 to be quite good in terms of computational efficiency in constructing the graph. This directly influences the quality of low pass filtering of a graph signal similar to the case in Euclidean signal processing in choosing a size of the window. As the neighborhood size increases, the filtering at each node becomes more aggressive since it averages the across several neighboring nodes, while for a small neighborhood the smoothing may not have any effect at all. MARGIN is agnostic to the type of graph construction used, since it ultimately only relies on the graph filtering process, and as a result it is applicable to more other graph constructions such as Reeb graphs (Pascucci et al., 2007) or β–skeletons.

CASE STUDIES

Considering MARGIN is very generic in nature, it is easy applicable to a wide variety of interpretability tasks. In this section we illustrate this felxibility on several example tasks. **Table 1** shows the domain design, graph construction, and function definition choices made for different use cases. Note in each case study, we construct a k-nearest-neighbor graph followed by the application of MARGIN with the main difference is in how the nodes of the graph are defined, followed by the type of function that is defined at each node.

Case Study I—Prototypes and Criticisms

A commonly encountered problem in interpretability is to identify samples that are prototypical of a dataset, and those that are statistically different from the prototypes (called criticisms). Together, they can provide a holistic understanding about the underlying data distribution. Even in cases where we do not have access to the label information, we seek a hypothesis that can pick samples which are representatives of their local neighborhood, while emphasizing statistically

TABLE 1 | Using MARGIN to solve different commonly encountered interpretability tasks.

Task	Domain	Nodes in G	Function	Explanation Modality
Prototypes/ Criticisms	Complete dataset	Samples	MMD (Global,Local)	Sample sub-selection
Explain prediction	Single image	Explanations	Sparsity	Saliency maps
Detect noisy-labels	Complete dataset	Samples	Local label-agreement	Samples to fix
Detect adversarial-attacks	Attacks/Noisy samples	Perturbed samples	MMD (Global)	Attack statistics
Study discrimination	Complete dataset	Samples	Local label-agreement	Confusing samples

anomalous samples. One such function was recently utilized in (Kim et al., 2016) to define prototypes and criticisms, and it was based on Maximum Mean Discrepancy (MMD).

Formulation

Following the general protocol in **Figure 1**, the domain is defined as the complete dataset, along with labels if available. Since this analysis does not rely on pre-trained models, we construct the neighborhood graph based on the Euclidean distance using k = 25 nearest neighbors. Inspired by (Kim et al., 2016), we define the following explanation function: For each sample x_i, we remove the chosen sample and all its connected neighbors from the graph to construct the set $\overline{X}_i = \{x_j, j \notin (i \cup N_i)\}$, and estimate the function at the i^{th} node as $f(i) = MMD(\overline{X}_i, \overline{X}_i \cup x_i)$. MMD gives us a way to measure the difference between two distributions, and since we artficially construct the two distributions by removing a single sample, we are able to determine the importance of an individual sample (and its neighbors) within the dataset using MARGIN. Let $k : X \times X \rightarrow R$ be a kernel such as the radial basis function (RBF) kernel, and $X = \overline{X}_i \cup x_i$, then we can use the approximation for MMD given in (c.f. Eq. 5 in Kim et al. (2016)) as:

$$MMD(x_i) = \frac{1}{|\overline{\mathcal{X}}_i|} \sum_{x_m \in \overline{\mathcal{X}}_i} k(x_i, x_m) + \frac{1}{|\mathcal{X}|} \sum_{x_j \in \mathcal{X}} k\left(x_i, x_j\right). \tag{2}$$

In cases of labeled datasets, the kernel density estimates for the MMD computation are obtained using only samples belonging to the same class. We refer to these two cases as *global* (unlabeled case) and *local* (labeled case) respectively. The hypothesis is that the regions of criticisms will tend to produce highly varying MMD scores, thereby producing high frequency content, and hence will be associated with high MARGIN scores. Conversely, we find that the samples with low MARGIN scores correspond to prototypes since they lie in regions of strong agreement of MMD scores. More specifically, we consider all samples with low MARGIN scores (within a threshold) as prototypes, and rank them by their actual function values. In contrast to the greedy inference approach in (Kim et al., 2016) that estimates prototypes and criticisms separately, they are inferred jointly in our case.

Experiment Setup and Results

We evaluate the effectiveness of the chosen samples through predictive modeling experiments with the idea that the most helpful samples should result in a good classifier, whereas a the most unhelpful/confusing samples should result in a poor classifier. We use the USPS handwritten digits data for this experiment, which consists of 9,298 images belonging to 10 classes. We use a standard train/test split for this dataset, with 7,291 training samples and the rest for testing. For fair comparisons with (Kim et al., 2016), we use a simple 1-nearest neighbor classifier. As described earlier, we consider both unsupervised (*global*) and supervised (*local*) variants of our explanation function for sample selection.

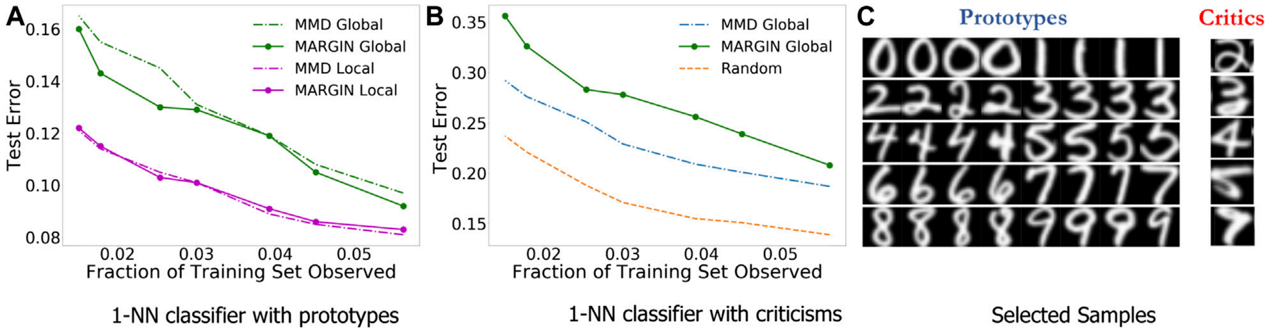
We expect the prototypical samples to be the most helpful in predictive modeling, i.e., good generalization. In **Figure 2A**, we observe that the prototypes from MARGIN perform competitively in comparison to the baseline technique. More importantly, MARGIN is particularly superior in the global case, with no access to label information. On the other hand, criticisms are expected to be the least helpful for generalization, since they often comprise boundary cases, outliers and under-sampled regions in space. Hence, we evaluate the test error using the criticisms as training data. Interestingly, as shown in **Figure 2B**, the criticisms from MARGIN achieve significantly higher test errors in comparison to samples identified using *MMD-critic* based optimization in (Kim et al., 2016). Furthermore, examples of the selected prototypes and criticisms from MARGIN are included in **Figure 2C**.

Case Study II—Explanations for Image Classification

Generating explanations for predictions is crucial to debugging black-box models and eventually building trust. Given a model, such as a deep neural network, that is designed to classify an image into one of r classes, a plausible *explanation* for a test prediction is to quantify the importance of different image regions to the overall prediction, i.e. produce a saliency map. We posit that perturbing the salient regions should result in maximal changes to the prediction. In addition, we expect *sparse* explanations to be more interpretable. In this section, we describe how MARGIN can be applied to achieve both these objectives.

Formulation

Since we are interested in producing explanations for instance-level predictions using MARGIN, the domain corresponds to a possible set of explanations for an image. Note that, the space of explanations can be combinatorially large, and hence we adopt the following greedy approach to construct the domain. We run the SLIC algorithm (Achanta et al., 2012) with varying number of superpixels, say $\{50, 100, 150, 200, 250, 300\}$, and define the



1-NN classifier with prototypes

1-NN classifier with criticisms

Selected Samples

FIGURE 2 | Using MARGIN to sample prototypes and criticisms. In this experiment, we study the generalization behavior of models trained solely using prototypes or criticisms.

domain as the union of superpixels from all the independent runs. In our setup, each of these superpixels is a plausible explanation and they become the nodes of $\mathcal{G}$. The edge between nodes m and n of this graph is defined based on the relative importance of each super-pixel, i.e., $e_{mn} = \left|\left|p_j(I) - p_j(I_m)\right| - \left|p_j(I) - p_j(I_n)\right|\right|$, where I is the original image, and I_m is the image with the m^{th} super-pixel masked out, and $p_j(\,)$ extracts the softmax scores for the j^{th} class in the image. This relative importance captures how two super-pixels are related in terms of the predictive model, which is related to a causal objective that is used in CXPlain Schwab and Karlen (2019).

For each of the explanations (super-pixels) m, we mask its pixels in the image and use the pre-trained model to obtain a measure of its saliency as before as $\left|p_j(I) - p_j(I_m)\right|$. Using these estimates, we obtain pixel-level saliency, S, as a weighted combination of their saliency from different superpixels (inversely weighted by the superpixel size). This dense saliency is similar to previous approaches such as (Zeiler and Fergus, 2014; Zhou et al., 2014).

Note that, this saliency estimation process did not impose the sparsity requirement. Hence, we use MARGIN to obtain influence scores based on their sparsity. The explanation function at each node is defined as the ratio of the size of the superpixel corresponding to that node and the size of the largest superpixel in the graph. Intuitively, MARGIN finds the sparsest explanation for different level sets of the saliency function. Subsequently, we compute pixel-level influence scores, I, as a weighted combination of their influences from different superpixels. The overall saliency map is obtained as $S_{\text{final}} = S \odot I$, where $\odot$ refers to the Hadamard product.

Experiment Setup and Results

Using images from the ImageNet database (Russakovsky et al., 2015), and the AlexNet (Krizhevsky et al., 2012) model, we demonstrate that MARGIN can effectively produce explanations for the classification. **Figure 3** illustrates the process of obtaining the final saliency map for an image from the *Tabby Cat* class. Interestingly, we see that the mouth and whiskers are highlighted as the most salient regions for its prediction. **Figure 4** shows the saliency maps from MARGIN for several other cases. For comparison, we show results from Grad-CAM (Selvaraju et al., 2017), which is a white-box approach that accesses the gradients in the network. We find that, using only a black-box approach, MARGIN produces explanations that strongly corroborate with Grad-CAM and in some cases produces more interpretable explanations. For example, in the case of an *Ice Cream* image, MARGIN identifies the ice cream, and the spoon, as salient regions, while Grad-CAM highlights only the ice cream and quite a few background regions as salient. Similarly, in the case of a *fountain* image, MARGIN highlights the fountain, and the sky, while Grad-CAM highlights the background (trees) slightly more than the fountain itself, which is not readily interpretable.

Case Study III—Detecting Incorrectly Labeled Samples

An increasingly important problem in real-world applications is concerned with the quality of labels in supervisory tasks. Since the presence of noisy labels can impact model learning, recent approaches attempt to compensate by perturbing the labels of samples that are determined to be high-risk of being corrupted, or when possible have annotators check the labels of those high-risk samples. In this section, we propose to employ MARGIN to recover incorrectly labeled samples. In particular, we consider a binary classification task, where we assume β% of the labels are randomly flipped in each class. In order to identify samples which were incorrectly labeled, we select samples with the highest MARGIN score, followed by simulating a human user correcting the labels for the top K samples. Ideally, we would like K, the number of samples checked by the user, to be as small as possible.

Formulation

Similar to Case Study I, the entire dataset is used to define the domain. Since we expect the flips to be random, we hypothesize that they will occur in regions where the labels of corrupted samples are different from their neighbors. Instead of directly using the label at each node as the explanation function, we believe a more smoothly varying function will allow us to extract regions of high frequency changes more robustly. As a result, we propose to measure the level of *distrust* at a given node, by measuring how many of its neighbors disagree with its label:

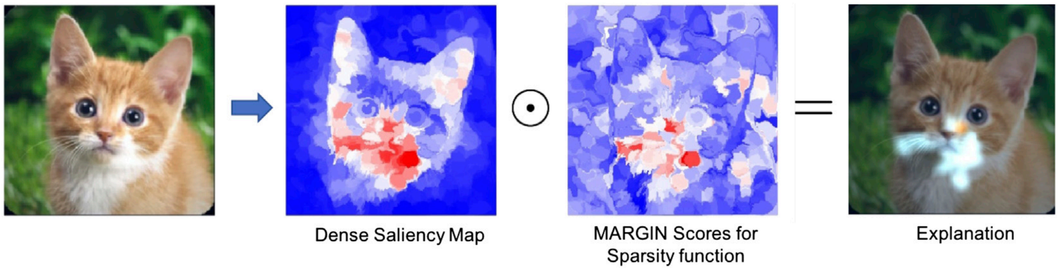


FIGURE 3 | We show the entire process of constructing the saliency map for one particular image (Tabby Cat) from ImageNet. From **left to right:** original image (dense) saliency map S, sparsity map I, and finally the explanation from MARGIN, S_{final}.

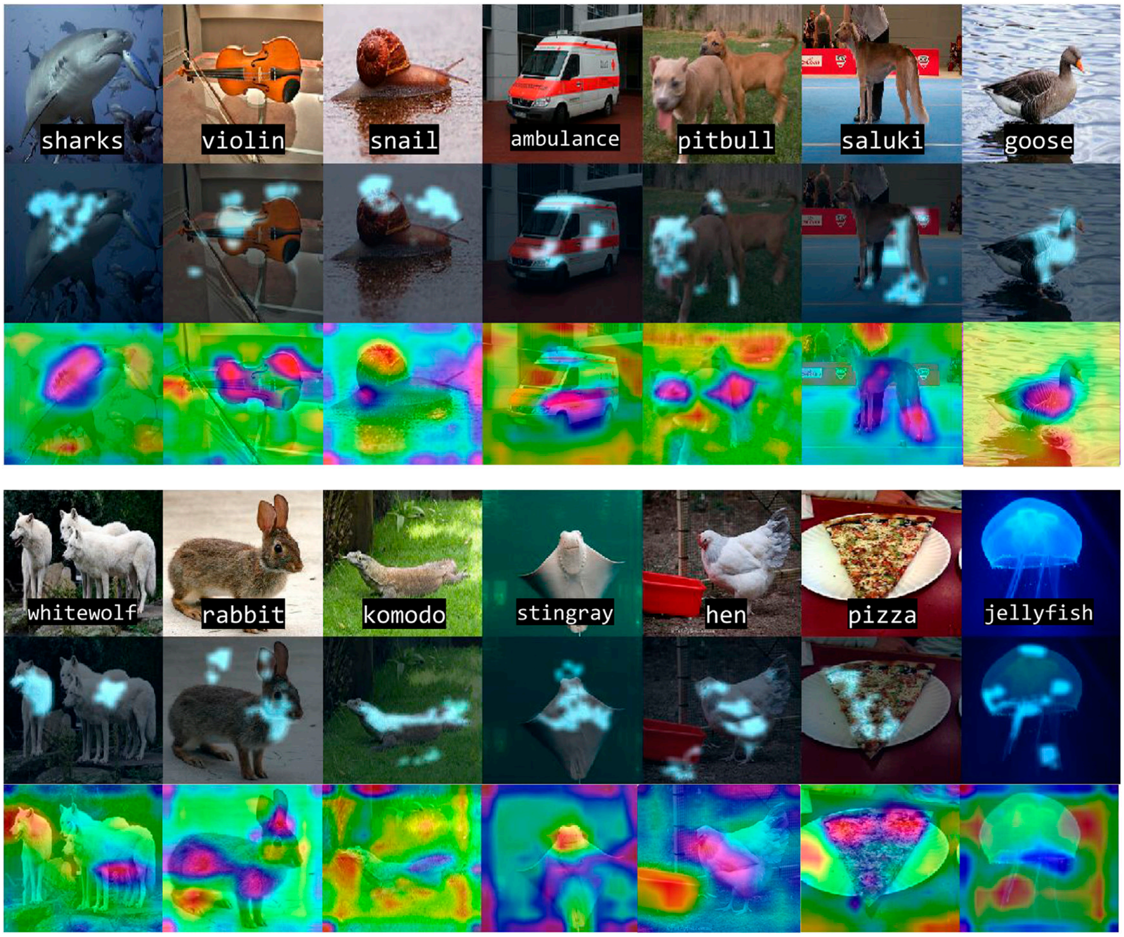


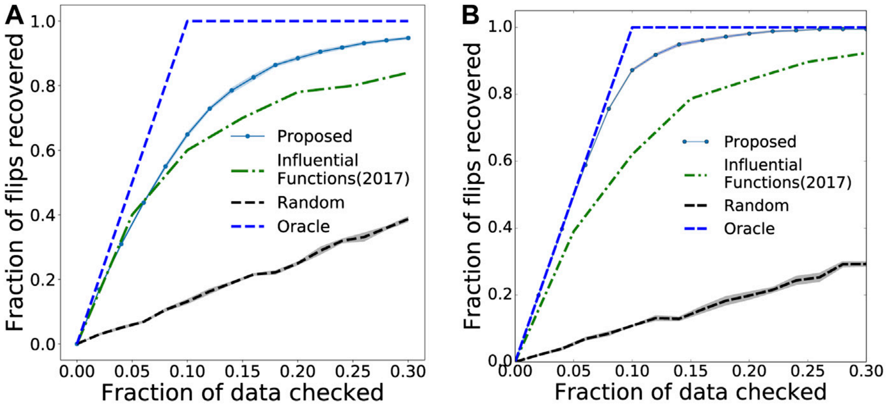
FIGURE 4 | Our approach identifies the most salient regions in different classes for image classification using AlexNet. From top to bottom: original image, MARGIN's explanation overlaid on the image, and Grad-CAM's (Selvaraju et al., 2017) explanation. Note our approach yields highly specific, and sparse explanations from different regions in the image for a given class.

$$f(i) = 1 - \frac{\sum_{j\epsilon N_i} L(j,i)}{|N_i|}, \tag{3}$$

where $L(j,i)$ is 1 only if nodes j and i share the same label; $|.|$ denotes the cardinality of a set.

Experiment Setup and Results

We perform our experiments on two datasets: 1) the Enron Spam Classification dataset (Metsis et al., 2006), containing 4138 training examples, with an imbalanced class split of around 70:30 (non-spam:spam), and 2) 3000 random images from



Detecting Label flips on the Enron Dataset

Detecting Label flips on the Kaggle Dataset

FIGURE 5 | MARGIN can be used to find samples with incorrect labels efficiently, much better than competing influence sampling based approaches. The "Oracle" here is the best case scenario, where the samples checked are exactly the ones that are corrupted.

Kaggle dog v cat classification dataset with almost equal number of images from each class[1]. Following standard practice, we randomly corrupt the labels of 10% of the samples. For the Enron Spam dataset, we extracted bag-of-words features of 500 dimensions corresponding to the most frequently occurring words. We observed these features to be noisy, so we use a simple PCA pre-processing step to reduce the dimensionality of the data down to 100. For Kaggle, we use penultimate features from AlexNet Krizhevsky et al. (2012) in order to construct a neighborhood graph. In both cases we use $k = 20$ as the number of neighbors for this purpose, we observed stable performance even when $k = 30$ or $k = 40$. The use of features instead of the data directly has become standard practice in several applications as it reduces the dimensionality of the data, while also providing a more semantically meaningful notion of neighborhood. We report average results from 10 repetitions of the experiment.

We compare our approach with three baselines: *1) Influence Functions:* We obtain the most influential samples using Influence Functions (Koh and Liang, 2017). *2) Random Sampling 3) Oracle:* The best case scenario, where the number of labels corrected is equal to the number of samples observed. Following (Koh and Liang, 2017), we vary the percentage of influential samples chosen, and compute the *recall* measure, which corresponds to the fraction of label flips recovered in the chosen subset of samples.

As seen in **Figure 5**, we see that our method is nearly 10 percentage points better than the state-of-the-art Influence Functions, achieving a recall of nearly 0.95 by observing just 30% of the samples. This difference is further improved when observing a balanced dataset like the Kaggle dogs v cats, as seen in **Figure 5B** where MARGIN outperforms Influence functions signficantly. On examining how MARGIN picks the samples, we see a clear trend which indicates a strong preference for samples that lie farther away from the classification boundary. In other words, this corresponds strongly to correcting the least number of samples which can lead to the most gain in validation performance when using a trained model.

Case Study IV—Interpreting Decision Boundaries

While studying black-box models, it is crucial to obtain a holistic understanding of their strengths, and more importantly, their weaknesses. Conventionally, this has been carried out by characterizing the decision surfaces of the resulting classifiers. In this experiment, we demonstrate how MARGIN can be utilized to identify samples that are the most confusing to a model, or more precisely those examples which are likely to be mis-classified by a pre-trained classifier. By definition these are images that are closest to the decision boundary inferred by the classifier.

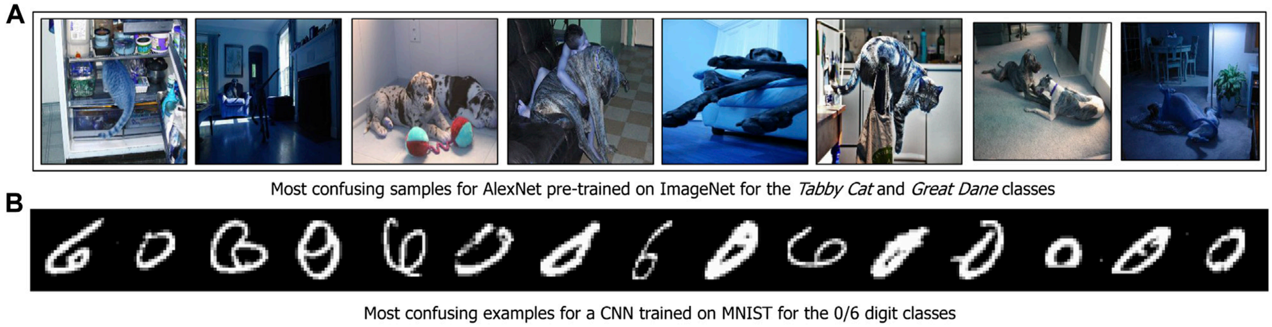
Formulation

In order to adopt MARGIN for analyzing a specific model, we construct a nearest neighbor graph $(k = 30)$ using latent representations inferred from the pre-trained classifier in consideration. This achieves two things–it gives us a semantic similarity measure as interpreted by the classifier, i.e., which similarities are considered important for the classification task. More importantly for this case study, this automatically distills the information regarding confusing samples into the graph that is constructed, since these samples are likely to be in regions of the neighborhood with high prediction uncertainty. Next, since the decision surface characterization is similar to case Study III, we use the local label agreement measure in **(3)** as the explanation function. This disagreement between the function and the neighborhood shows up as high frequency information which is exploited by MARGIN to identify the decision surface.

Experiment Setup

We perform an experiment on 2-class datasets extracted from ImageNet and MNIST. More specifically, in the case of ImageNet,

[1] https://www.kaggle.com/c/dogs-vs-cats/data



Most confusing samples for AlexNet pre-trained on ImageNet for the *Tabby Cat* and *Great Dane* classes

Most confusing examples for a CNN trained on MNIST for the 0/6 digit classes

FIGURE 6 | Using MARGIN to sample near decision boundaries.

we perform decision surface characterization on the classes *Tabby Cat* and *Great Dane*. We used the features from a pre-trained AlexNet's penultimate layer to construct the graph. For the MNIST dataset, we considered data samples from digits '0' and '6', and we used the latent space produced using a convolutional neural network for the analysis. A selected subset of samples characterizing the decision surfaces of both datasets are shown in **Figure 6**.

Results

From **Figure 6A**, we see that the model gets confused whenever the animal's face is not visible, or if it is in a position facing away from the camera. This is reasonable since we are only measuring the most confusing samples between the *Tabby Cat* and *Great Dane* classes which share a lot of semantic similarity. Similarly, in the MNIST dataset, the examples shown depict atypical ways in which the digits '0' and '6' can be written. These results suggest that MARGIN is effective in identifying these examples, with a combination of the appropriate neighborhoods (in the latent space of the model) and labels.

Case Study V—–Characterizing Statistics of Adversarial Examples

In this application, we examine the problem of quantifying the statistical properties of adversarial examples using MARGIN. Adversarial samples (Biggio et al., 2013; Szegedy et al., 2013) refer to examples that have been specially crafted, such that a particular trained model is 'tricked' into misclassifying them. This is done typically by perturbing a sample, sometimes in ways imperceptible to humans, while maximizing misclassification rates. In order to better understand the behavior of such adversarial examples, there have been studies in the past to show that adversarial examples are statistically different from normal test examples. For example, an MMD score between distributions is proposed in (Grosse et al., 2017), and a kernel density estimator (KDE) in (Feinman et al., 2017). However, these measures are global, and provide little insight into individual samples. We propose to use MARGINto develop these statistical measures at a sample level, and study how individual adversarial samples differ from regular samples.

Formulation

As in other case studies, MARGIN constructs a graph, where each node corresponds to an example that is either adversarial or harmless, and the edges are constructed using neighbors in the latent space of the model, against which the adversarial examples have been designed. We consider two kinds of functions in this experiment: 1)

MMD Global

Similar to *Case Study I—Prototypes and Criticisms*, we use the MMD score between the whole set, and the set without a particular sample and its neighbors. This provides a way to capture statistically rarer samples in the dataset; 2)

Kernel Density Estimator

We also use the KDE of each sample, as proposed in (Feinman et al., 2017), where we measure the discrepancy of each sample against the training samples from its predicted class. While these measures on their own may not be very illustrative, they are useful functions to determine influences within MARGIN.

Experiment Setup and Results

We perform experiments on 2000 randomly sampled test images from the MNIST dataset (LeCun, 1998), of which we adversarially perturb 1000 images. We measure MARGIN scores using both MMD Global, and KDE, against two popular attacks–the Fast Gradient Sign Method (FGSM) attack (Goodfellow et al., 2014), and the L2-attack (Carlini and Wagner, 2017b). We use the same setup as in (Carlini and Wagner, 2017a), including the network architecture for MNIST. The resulting MARGINscore determined using **Algorithm 1**, is more discriminative, as seen in **Figure 7**. As noted in (Carlini and Wagner, 2017a), the MMD and KDE measures were not very effective against stronger attacks such as the L2-attack. This is reflected to a much lower degree even in our approach, where there is a small overlap in the distributions. We also find that the overlapping regions correspond to samples from the training set that are extremely rare to begin with (like criticisms from *Case Study I—Prototypes and Criticisms*).

Case Study VI—Active Learning on Graphs

To demonstrate the applicability of MARGIN to work with graph structured data, we study the problem of active learning on

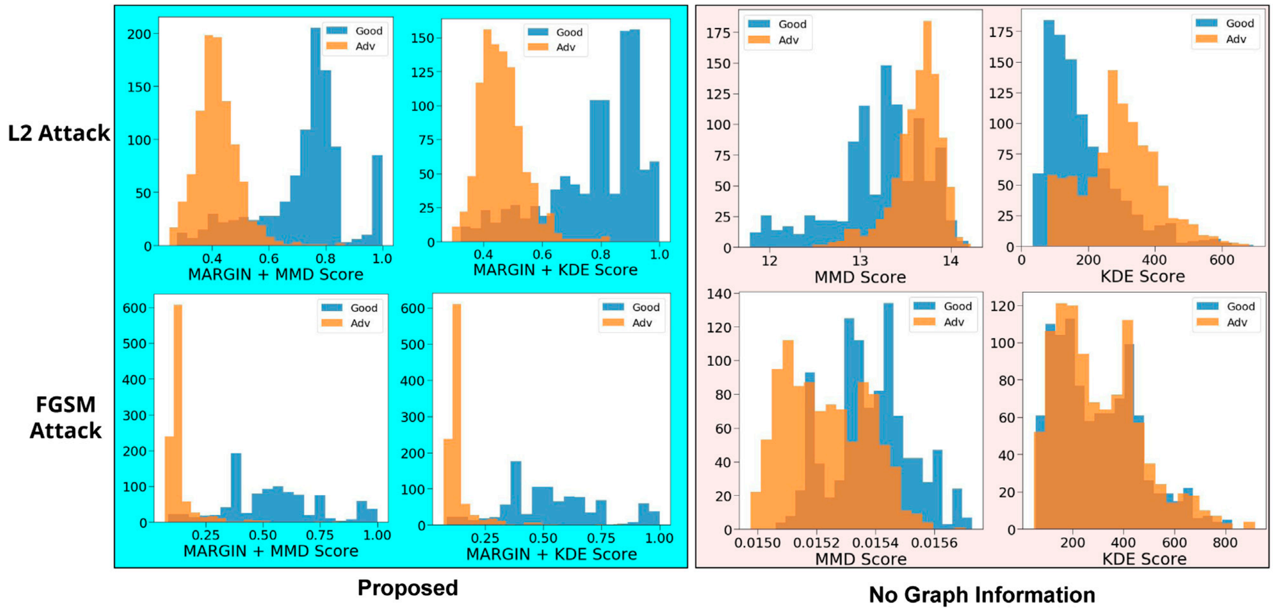


FIGURE 7 | We compare histograms of scores obtained from adversarial samples with and without incorporating graph structure. We see that including the structure results in a much better separation between adversarial and harmless examples. In addition, regions of overlap can easily be explained.

graphs, or in other words, generating highly influential samples for a label propagation task. Label propagation is a semi-supervised learning problem, where the task is to propagate labels from a small set of nodes to all the other nodes of the graph. In order to evaluate the samples chosen using our method, we study the test accuracies for varying sizes of the training set. In order to perform the semi-supervised learning, we use the Graph Convolutional Network (GCN) implementation by Kipf and Welling (2017), with 3 graph convolutional layers comprising 16 graph filters each, and a learning rate of 0.01. The rest of the hyper-parameters are those recommended in the GCN implementation[2].

Formulation

Since the attributes are independently defined on each node, they do not contain information about the neighborhoods in the graph and therefore do not directly provide us a notion of influence. Instead, we first embed the attributes using a graph convolutional autoencoder Kipf and Welling (2016), and compute the explanation function f as the as the norm of each latent feature at each node. Next, using MARGIN we compute the influences of the training samples alone, and sort them in decreasing order.

Datasets and Baselines

We consider two popularly used citation network datasets–Cora and Citeseer Sen et al. (2008). The Cora dataset consists of 2,708 nodes and 5,429 edges, while the Citeseer dataset consists of 3,327 nodes and 4,732 edges. The attributes at each node are comprised of a sparse bag-of-words feature vector with 3,703 dimensions for Citeseer, and 1,433 dimensions for Cora.

We compare with two baselines: *1) Probabilistic resampling on graphs:* The resampling strategy was proposed in Chen et al. (2017) as a way to efficiently resample dense point clouds. In this approach, the magnitude of the features at each node after a high pass filtering is directly used as a probability of influence at that node, $p(n)$. This is followed by a resampling of the nodes on the graph according to $p(n)$. While it is an effective strategy to resample dense point clouds, it tends to be less reliable for the label propagation experiment, as shown in **Figure 8**. Since we are sampling from a distribution, we sample 10 times, and report the mean and standard deviation. *2) Random sampling:* We also randomly sample from each class on the graph, and repeat this 10 times, while reporting the mean and standard deviation.

Results

In all cases, the accuracy of label propagation is measured on a test set of size 1,000 samples, by training on only 10-100s of samples. **Figure 8** shows the accuracy of label propagation for varying number of training set sizes. It is clear that our proposed sampling achieves state-of-theart performance on the graph. The performance is around 10–15% points higher in accuracy compared to the baseline techniques, especially in small training set regimes. While MARGIN's resampling method is deterministic, we repeat the other baselines 5 times and report average and standard deviation. As we observe in **Figure 8**, the influence computed by MARGIN is significantly better and more stable than the influence obtained by directly using the attributes as the function, as done in the case of probabilistic resampling. It is also interesting to note that this probabilistic method is highly unstable for a very low number of

[2] https://github.com/tkipf/gcn

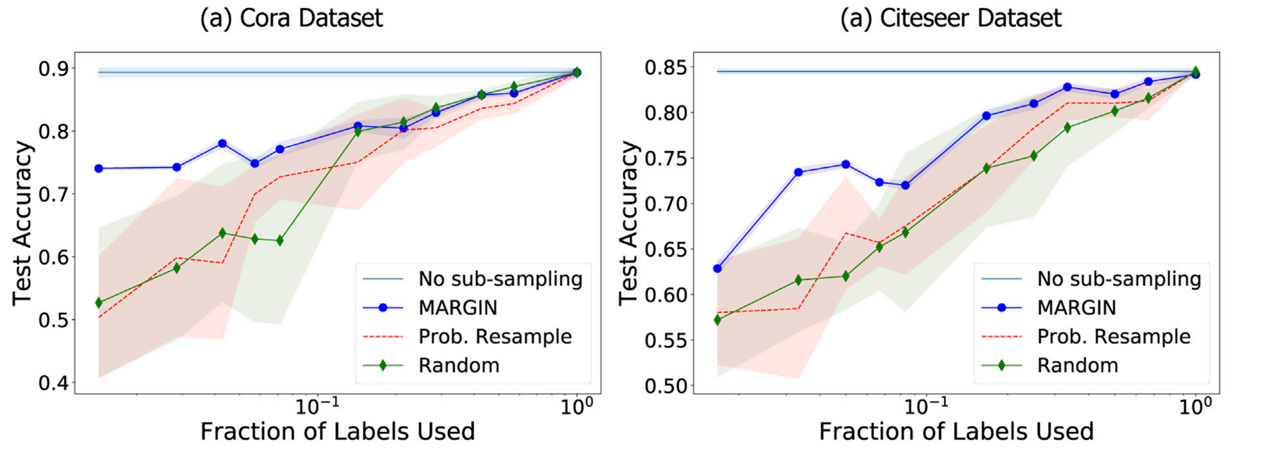


FIGURE 8 | MARGIN based sampling for graph signals shows significant improvement in label propagation performance, even for very small sets of samples.

samples, as it was originally proposed to resample dense point clouds. Finally, random sampling itself is a competitive baseline as the number of samples under consideration is very small.

CONCLUSION

We proposed a generic framework called MARGIN that is able to provide explanations to popular interpretability tasks in machine learning. These range from identifying prototypical samples in a dataset that might be most helpful for training, to explaining salient regions in an image for classification. In this regard, MARGIN exploits ideas rooted in graph signal processing to identify the most influential nodes in a graph, which are nodes that maximally affect the graph function. While the framework is extremely simple, it is highly general in that it allows a practitioner to include rich semantic information easily in three crucial ways–defining the domain (intra-sample vs inter-sample), edges (pre-defined/native/model latent space), and finally a function defined at each node. The graph based analysis easily scales to very sparse graphs with tens of thousands of nodes, and opens up several opportunities to study problems in interpretable machine learning.

PYTHON IMPLEMENTATION OF MARGIN

The graph analysis based influence estimation in MARGIN is extremely simple, in that it can be implemented using a few lines of python code.

```
import numpy as np
import networkx as nx
import scipy.sparse as sp
’’’
Inputs:
adj: adjacency matrix
f: function defined at each node
p: number of hops from each node
for filtering
Output:
I: Influence score per node
’’’
def MARGIN(adj,f,p=1):
  G = nx.Graph(adj) #graph object
  N = adj.shape[0] #number of nodes
  degree = G.degree()
  deg = [1./d[1] for d in degree.items()]
  tmp = np.zeros((N,N))
  Dinv = sp.csr_matrix(tmp)
  idx0,idx1 = np.diag_indices(N)
  Dinv[idx0,idx1] = deg
  A_n = np.sqrt(Dinv)*adj*np.sqrt(Dinv)
  P = np.power(A_n,p)
  M = np.sum(P>0,axis=1,dtype=np.float)
  f_0 = np.matrix(f)
  f_1 = (P*f_0)/M
  f_filter = f_1-(P*f_1)/M
  I = np.abs(f_filter)
  I = I/np.max(I)
  return I.A
```

DATA AVAILABILITY STATEMENT

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

AUTHOR CONTRIBUTIONS

RA and JT are responsible for conceptualization, writing, and design of the main ideas in the paper. RS helped with experiments in Case Study III—Detecting Incorrectly Labeled Samples and Case Study IV—Interpreting Decision Boundaries primarily, and PB helped in discussions, overall formulation and writing.

FUNDING

This work was performed under the auspices of the U.S. Dept. of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. This manuscript has been released as a pre-print at arxiv.org (Anirudh et al., 2017).

REFERENCES

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 2274–2282. doi:10.1109/tpami.2012.120

Anirudh, R., Thiagarajan, J. J., Sridhar, R., and Bremer, T. (2017). Margin: uncovering deep neural networks using graph signal analysis. Available at: http://arxiv.org/abs/1711.05407 (Accessed November 15, 2017).

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., et al. (2013). "Evasion attacks against machine learning at test time." in Joint European conference on machine learning and knowledge discovery in databases (Berlin, Germany: Springer), 387–402. doi:10.1007/978-3-642-40994-3_25

Carlini, N., and Wagner, D. (2017b). Towards evaluating the robustness of neural networks, in Security and Privacy (SP), 2017 IEEE Symposium on (Piscataway, NJ: IEEE), 39–57.

Chakraborty, S., Tomsett, R., Raghavendra, R., Harborne, D., Alzantot, M., Cerutti, F., et al. (2017). Interpretability of deep learning models: a survey of results. in Proceedings of the 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced and Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), San Francisco, CA, 4–8 August, 2017. Piscataway, NJ: IEEE, 1–6. doi:10.1109/UIC-ATC.2017.8397411

Chen, S., Tian, D., Feng, C., Vetro, A., and Kovačević, J. (2017). "Fast resampling of 3d point clouds via graphs." Available at: http://arxiv.org/abs/1702.06397 (Accessed Febrauary 11, 2017).doi:10.1109/icassp.2017.7952695

Chen, S., Varma, R., Sandryhaila, A., and Kovacevic, J. (2015). Discrete signal processing on graphs: sampling theory. *IEEE Trans. Signal. Process.* 63, 6510–6523. doi:10.1109/tsp.2015.2469645

Doshi-Velez, F., and Kim, B. (2017). "A roadmap for a rigorous science of interpretability." Available at: http://arxiv.org/abs/1702.08608.

Feinman, R., Curtin, R. R., Shintre, S., and Gardner, A. B. (2017). "Detecting adversarial samples from artifacts." Available at: http://arxiv.org/abs/1703.00410

Fong, R., and Vedaldi, A. (2017). "Interpretable explanations of black boxes by meaningful perturbation." Available at: http://arxiv.org/abs/1704.03296. doi:10.1109/iccv.2017.371

Gadde, A., Anis, A., and Ortega, A. (2014). Active semi-supervised learning using sampling theory for graph signals. in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining NY, United states: (ACM), 492–501.

Giacinto, N., and Wagner, D. (2017a). Adversarial examples are not easily detected: Bypassing ten detection methods, in Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security Newyork, United states: ACM, 3–14.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). "Explaining and Harnessing Adversarial Examples." Available at: http://arxiv.org/abs/1412.6572.

Grosse, K., Manoharan, P., Papernot, N., Backes, M., and McDaniel, P. (2017). "On the (Statistical) Detection of Adversarial Examples." Available at: http://arxiv.org/abs/1702.06280.

Huang, A., and Moura, J. M. F. (2013). Discrete Signal Processing on Graphs. *IEEE Trans. Signal. Process.* 61, 1644–1656. doi:10.1109/tsp.2013.2238935

Kim, B., Khanna, R., and Koyejo, O. O. (2016). "Examples Are Not Enough, Learn to Criticize! Criticism for Interpretability." in Advances in Neural Information Processing Systems (NIPS), 2280–2288.

Kipf, T. N., and Welling, M. (2017). Semi-supervised Classification with Graph Convolutional Networks, in International Conference on Learning Representations. (Ethiopia, United States: ICLR).

Kipf, T. N., and Welling, M. (2016). "Variational Graph Auto-Encoders." Available at: http://arxiv.org/abs/1611.07308.

Koh, P. W., and Liang, P. (2017). Understanding Black-Box Predictions via Influence Functions, in International Conference on Machine Learning, (Vienna, Austria. ICML).

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet Classification with Deep Convolutional Neural Networks, in Advances in neural information processing systems. 1097–1105.

LeCun, Y. (1998). The Mnist Database of Handwritten Digits. Available at: http://yann lecun.com/exdb/mnist/.

Lipton, Z. C. (2016). "The Mythos of Model Interpretability." Available at: http://arxiv.org/abs/1606.03490.

Lundberg, S. M., and Lee, S. I.. (2017). "A Unified Approach to Interpreting Model Predictions." in Advances in neural information processing systems, 4765–4774.

Metsis, V., Androutsopoulos, I., and Paliouras, G. (2006). Spam Filtering with Naive Bayes-Which Naive Bayes?. *CEAS* 17, 28–69.

Pan, P., Swaroop, S., Immer, A., Eschenhagen, R., Turner, R. E., and Khan, M. E. (2020). "Continual Deep Learning by Functional Regularisation of Memorable Past." Available at: http://arxiv.org/abs/2004.14070.

Pascucci, V., Scorzelli, G., Bremer, P.-T., and Mascarenhas, A. (2007). Robust On-Line Computation of Reeb Graphs: Simplicity and Speed papers. (NY, United states: ACM SIGGRAPH), 58. doi:10.1145/1275808.1276449

Pesenson, I. (2008). Sampling in Paley-Wiener Spaces on Combinatorial Graphs. *Trans. Amer. Math. Soc.* 360, 5603–5627. doi:10.1090/s0002-9947-08-04511-x

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why Should I Trust You?: Explaining the Predictions of Any Classifier, in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. NY, United states: ACM, 1135–1144.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* 115, 211–252. doi:10.07/s11263-015-0816-y

Schwab, P., and Karlen, W. (2019). Explain: Causal Explanations for Model Interpretation under Uncertainty. in Advances in Neural Information Processing Systems. 10220–10230.

Schwab, P., Miladinovic, D., and Karlen, W. (2019). Granger-causal Attentive Mixtures of Experts: Learning Important Features with Neural Networks, in Proceedings of the AAAI Conference on Artificial Intelligence, CA, United States. AAAI, 4846–4853. doi:10.1609/aaai.v33i01.33014846

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D.. (2017). "Grad-cam: Visual Explanations from Deep Networks via Gradient-Based Localization". in. Proceedings of the IEEE international conference on computer vision, 22–29 October, 2017. (Venice, Italy IEEE), 618–626.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. (2008). Collective Classification in Network Data. *AIMag* 29, 93. doi:10.1609/aimag.v29i3.2157

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning Important Features through Propagating Activation Differences. *Proc. Machine Learn. Res.* 70, 3145–3153.

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. (2013). The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *IEEE Signal. Process. Mag.* 30, 83–98. doi:10.1109/msp.2012.2235192

Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). "Deep inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps." Available at: http://arxiv.org/abs/1312.6034. doi:10.5244/c.27.8

Sundararajan, M., Taly, A., and Yan, Q. (2017). "Axiomatic Attribution for Deep Networks." Available at: http://arxiv.org/abs/1703.01365.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., et al. (2013). "Intriguing Properties of Neural Networks." Available at: http://arxiv.org/abs/1312.6199.

Thiagarajan, J. J., Narayanaswamy, V., Anirudh, R., Bremer, P.-T., and Spanias, A. (2020). "Accurate and Robust Feature Importance Estimation under Distribution Shifts." Available at: http://arxiv.org/abs/2009.14454 doi:10.2172/1670557

Zeiler, M. D., and Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. European conference on computer vision. Berlin, Germany: Springer, 818–833. doi:10.1007/978-3-319-10590-1_53

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2014). "Object Detectors Emerge in Deep Scene Cnns." Available at: http://arxiv.org/abs/1412.6856.

Disclaimer: This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Conflict of Interest: Author RS portion of the work was done during a summer internship at LLNL. He is currently employed by the company Walmart Labs. 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.

frontiers in Artificial Intelligence

METHODS
published: 18 May 2021
doi: 10.3389/frai.2021.589632

Preventing Failures by Dataset Shift Detection in Safety-Critical Graph Applications

Hoseung Song[1]*, *Jayaraman J. Thiagarajan*[2] *and Bhavya Kailkhura*[2]

[1]*Department of Statistics, University of California, Davis, CA, United States,* [2]*Lawrence Livermore National Laboratory, Livermore, CA, United States*

Dataset shift refers to the problem where the input data distribution may change over time (e.g., between training and test stages). Since this can be a critical bottleneck in several safety-critical applications such as healthcare, drug-discovery, etc., dataset shift detection has become an important research issue in machine learning. Though several existing efforts have focused on image/video data, applications with graph-structured data have not received sufficient attention. Therefore, in this paper, we investigate the problem of detecting shifts in graph structured data through the lens of statistical hypothesis testing. Specifically, we propose a practical two-sample test based approach for shift detection in large-scale graph structured data. Our approach is very flexible in that it is suitable for both undirected and directed graphs, and eliminates the need for equal sample sizes. Using empirical studies, we demonstrate the effectiveness of the proposed test in detecting dataset shifts. We also corroborate these findings using real-world datasets, characterized by directed graphs and a large number of nodes.

Keywords: graph learning, dataset shift, safety, two-sample testing, random graph models

OPEN ACCESS

Edited by:
Novi Quadrianto,
University of Sussex, United Kingdom

Reviewed by:
Bowei Chen,
University of Glasgow,
United Kingdom
Chetan Tonde,
Amazon, United States

****Correspondence:***
Hoseung Song
hosong@ucdavis.edu

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *31 July 2020*
Accepted: *26 April 2021*
Published: *18 May 2021*

Citation:
Song H, Thiagarajan JJ and Kailkhura B (2021) Preventing Failures by Dataset Shift Detection in Safety-Critical Graph Applications. Front. Artif. Intell. 4:589632. doi: 10.3389/frai.2021.589632

1 INTRODUCTION

Most machine learning (ML) applications, e.g., healthcare, drug-discovery, etc., encounter dataset shift when operating in the real-world. The reason for this comes from the bias in the testing conditions compared to the training environment introduced by experimental design. It is well known that ML systems are highly susceptible to such dataset shifts, which often leads to unintended and potentially harmful behavior. For example, in ML-based electronic health record systems, input data is often characterized by shifting demographics, where clinical and operational practices evolve over time and a wrong prediction can threaten human safety.

Although dataset shift is a frequent cause of failure of ML systems, very few ML systems inspect incoming data for a potential distribution shift (Bulusu et al., 2020). While some practical methods such as (Rabanser et al., 2019) have been proposed for detecting shifts in applications with Euclidean structured data (speech, images, or video), there are limited efforts in solving such issues for graph structured data that naturally arises in several scientific and engineering applications. In recent years there has been a surge of interest in applying ML techniques to structured data, e.g. graphs, trees, manifolds etc. In particular, graph structured data is becoming prevalent in several high-impact applications including bioinformatics, neuroscience, healthcare, molecular chemistry and computer graphics. In this paper, we investigate the problem of detecting distribution shifts in graph-structured datasets for responsible deployment of ML in safety-critical applications. Specifically, we propose to solve the problem of detecting shifts in graph-structured data through the lens of statistical two-sample testing. Broadly, the objective in two-sample testing for graphs is to test whether two populations of random graphs are different or not based on the samples generated from each of them.

Two-sample testing has been of significant research interest due to its broad applicability. An important class of testing methods relies on summary metrics that quantify the topological differences between networks. For example, in brain network analysis, commonly adopted topological summary metrics include the global efficiency (Ginestet et al., 2011) and network modularity (Ginestet et al., 2014). An inherent challenge with these approaches is that the topological characteristics depend directly on the number of edges in the graph, and can be insufficient in practice. An alternative class of methods is based on comparing the structure of subgraphs to produce a similarity score (Shervashidze et al., 2009; Macindoe and Richards, 2010). For example, Shervashidze et al. (2009) used the earth mover's distance between the distributions of feature summaries of their constituent subgraphs.

While these heuristic methods are reasonably effective for comparing real-world graphs, not until recently that a principled analysis of hypothesis testing with random graphs was carried out. In this spirit, Ginestet et al. (2017) developed a test statistic based on a precise geometric characterization of the space of graph Laplacian matrices. Most of these approaches for graph testing based on classical two-sample tests are only applicable to the restrictive low-dimensional setting, where the population size (number of graphs) is larger than the size of the graphs (number of vertices). To overcome this challenge, Tang et al. (2017a) proposed a semi-parametric two-sample test for a class of latent position random graphs, and studied the problem of testing whether two dot product random graphs are drawn from the same population or not. Other testing approaches that focused on hypothesis testing for specific scenarios, such as sparse networks (Ghoshdastidar et al., 2017a) and networks with a large number of nodes (Ghoshdastidar et al., 2017b), have been developed. More recently, Ghoshdastidar and von Luxburg (2018) developed a novel testing framework for random graphs, particularly for the cases with small sample sizes and the large number of nodes, and studied its optimality. More specifically, this test statistic was based on the asymptotic null distributions under certain model assumptions.

Unfortunately, all these approaches are limited to testing undirected graphs under the equal sample size (for two graph populations) setting. In real-world dataset shift detection problems, these assumptions are extremely restrictive, making existing approaches inapplicable to several applications. In order to circumvent these crucial shortcomings, we develop a novel approach based on hypothesis testing for detecting shifts in graph-structured data, which is more flexible (i.e., accommodates 1) both undirected and directed graphs and 2) unequal sample size cases). Moreover, it is highly effective even when the sample size grows. Notice that, similar to the setting in Ghoshdastidar and von Luxburg (2018), we also consider scenarios where all networks are defined from the same vertex set, which is common to several real-world applications. The main contributions of this paper are summarized below:

- We propose a new test statistic that can be applied to undirected graphs as well as directed graphs and/or unweighted graphs as well as weighted graphs, while eliminating the equal sample size requirement. The asymptotic distribution for the proposed statistic, based on the well-known U-statistic, is derived.
- A practical permutation approach based on a simplified form of the statistic is also proposed.
- We compare the new approach with existing methods for graph testing in diverse simulation settings, and show that the proposed statistic is more flexible and achieves significant performance improvements.
- In order to demonstrate the usefulness of the proposed method in challenging real-world problems, we consider several applications (including a healthcare application), and show the effectiveness of our approach.

2 PRELIMINARIES

We consider the following two-sample setting. Let two random graph populations with d vertices be denoted as $\mathcal{A}_1, \ldots, \mathcal{A}_m$ from $P \in [0,1]^{d\times d}$ and $\mathcal{B}_1, \ldots, \mathcal{B}_n$ from $Q \in [0,1]^{d\times d}$ with their adjacency matrices $A_1, \ldots, A_m$ and $B_1, \ldots, B_n$, respectively. We are concerned with testing hypotheses:

$$H_0 : P = Q \text{ vs } H_1 : P \neq Q. \tag{1}$$

Notice that we consider the cases where each population consists of independent and identically distributed samples, which encompasses a wide-range of network analysis problems, see, e.g., Holland et al. (1983), Newman and Girvan (2004), Newman (2006). In contrast to existing formulations, e.g., Ghoshdastidar and von Luxburg (2018), we consider a more flexible setup where 1) the sample sizes m and n are allowed to be different and 2) the graphs in p and Q can be weighted and/or directed.

While there have several efforts to two-sample testing of graphs (Bubeck et al., 2016; Gao and Lafferty, 2017; Maugis et al., 2017), recent works such as Tang et al. (2017a), Tang et al. (2017b); Ginestet et al. (2017) have focused on designing more general testing methods that are applicable to practical settings. For example, Ginestet et al. (2017) proposed a practical test statistic based on the correspondence between an undirected graph and its Laplacian under the inhomogeneous Erdős-Rényi (IER) assumption, which means all nodes are independently generated from a Bernoulli distribution (see details in **Section 3**). The test statistic, under the assumption of equal sample sizes m, can be described as follows:

$$T_{gin} = \sum_{i<j}^{d} \frac{\left[(\overline{A})_{ij} - (\overline{B})_{ij} \right]^2}{a}, \tag{2}$$

where

$$a = \frac{1}{m(m-1)} \sum_{k=1}^{m} \left[(A_k)_{ij} - (\overline{A})_{ij} \right]^2 + \frac{1}{m(m-1)} \sum_{k=1}^{m} \left[(B_k)_{ij} - (\overline{B})_{ij} \right]^2,$$

$$(\overline{A})_{ij} = \frac{1}{m} \sum_{k=1}^{m} (A_k)_{ij}, \ (\overline{B})_{ij} = \frac{1}{m} \sum_{k=1}^{m} (B_k)_{ij}.$$

The authors showed that T_{gin} converges to a chi-square distribution as $m \to \infty$ under H_0. However, this statistic can be interpreted as Hotelling's T^2 statistic for multivariate data, thus leading to no performance guarantees for "small m and large d" scenario. This is because the variance estimates used in **Eq. 2** are not stable for small m and large d, especially when graphs are sparse.

Recently, Ghoshdastidar and von Luxburg (2018) proposed a new class of test statistics, designed for different scenarios under the IER model assumption. More specifically, they focused on cases with small m and large d. For cases with $m > 1$, the following test statistic was used:

$$T_{spec} = \frac{\left\| \sum_{k=1}^{m} (A_k - B_k) \right\|_2}{\sqrt{\max_{1 \le i \le d} \sum_{j=1}^{d} \sum_{j=1}^{d} \sum_{k=1}^{m} \left[(A_k)_{ij} + (B_k)_{ij} \right]}}, \tag{3}$$

While it was suggested by the authors to perform this test using bootstraps from the aggregated data, this could be challenging for sparse graphs, since it is difficult to construct bootstrapped statistics from an operator norm. Hence, they considered an alternate test statistic based on the Frobenius-norm as follows:

$$T_{fro} = \frac{\sum_{i<j}^{d} \left(\sum_{k \le m/2} (\Delta_k)_{ij} \right) \left(\sum_{k > m/2} (\Delta_k)_{ij} \right)}{\sqrt{\sum_{i<j}^{d} \left(\sum_{k \le m/2} (S_k)_{ij} \right) \left(\sum_{k > m/2} (S_k)_{ij} \right)}}, \tag{4}$$

where $(\Delta_k)_{ij} = (A_k)_{ij} - (B_k)_{ij}$ and $(S_k)_{ij} = (A_k)_{ij} + (B_k)_{ij}$. It was shown that this test is provably effective and more reliable. Furthermore, they derived the asymptotic normality of T_{fro} as $d \to \infty$ to make the method instantly applicable without the bootstrap procedure. Despite the good properties of this method, this test can be used only when the two sample sizes are equal, and when graphs are undirected. In the rest of this paper, we develop a new test statistic which addresses these two crucial limitations.

3 PROPOSED TEST

To carry out two-sample testing, we want to measure the distance between two populations. Here, we utilize the Frobenius distance as the evidence for discrepancy between two populations:

$$T = \|P - Q\|_F^2. \tag{5}$$

Next, we provide finite sample estimates of this quantity. To accommodate more general settings for random graphs, the new test statistic is defined as follows:

$$T_{new} = \sum_{i=1}^{d} \sum_{j=1}^{d} T_{ij}, \tag{6}$$

where

$$T_{ij} = \frac{1}{m(m-1)} \sum_{k \ne l}^{m} (A_k)_{ij} (A_l)_{ij} + \frac{1}{n(n-1)} \sum_{k \ne l}^{n} (B_k)_{ij} (B_l)_{ij} - \frac{2}{mn} \sum_{k=1}^{m} \sum_{l=1}^{n} (A_k)_{ij} (B_l)_{ij}.$$

Note that the proposed test statistic accommodates scenarios where 1) the sample sizes m and n are different and 2) the graphs in p and Q are weighted and/or directed.

Next, we analyze the theoretical properties of the proposed test. For the ease of theoretical analysis, we focus on the case where graphs are unweighted and undirected. However, the proposed test and algorithmic tools are applicable to weighted and/or directed graph scenarios which is the main focus of the paper and is considered in our experimental evaluations. More specifically, in our theoretical analysis, we assume that graphs are drawn from the inhomogeneous Erdős-Rényi (IER) random graph process, which is considered as an extended version of the Erdős-Rényi (ER) model from Bollobás et al. (2007). In other words, we consider unweighted and undirected random graphs, where edges occur independently without any additional structural assumption on the population adjacency matrix. Note, the IER model encompasses other models studied in the literature including random dot product graphs (Tang et al., 2017b) and stochastic block models (Lei et al., 2016). A graph $\mathcal{G} \in [0,1]^{d \times d}$ from a population symmetric adjacency p with zero diagonal is considered to be an IER graph if $(\mathrm{G})_{\mathrm{ij}} \overset{\underline{i.i.d}}{\sim} \mathrm{Bernoulli}(\mathrm{P}_{\mathrm{ij}})$ for all $i, j \in \{1, \ldots, d\}$. Here, d denotes the cardinality of the vertex set. Next we analyze the theoretical properties of the proposed test under IER assumption.

LEMMA 3.1. T_{new} is an unbiased empirical estimate of T, that is,

$$E(T_{new}) = T. \tag{7}$$

PROOF. Under the IER assumptions, for all $i, j = 1, \ldots, d$, we have

$$(A_k)_{ij} (A_l)_{ij} \sim Bernoulli\left(P_{ij}^2\right),$$

$$\sum_{k \ne l}^{m} (A_k)_{ij} (A_l)_{ij} \sim Binomial\left(m(m-1), P_{ij}^2\right),$$

$$(B_k)_{ij}(B_l)_{ij} \sim Bernoulli\left(Q_{ij}^2\right),$$

$$\sum_{k \neq l}^{n} (B_k)_{ij}(B_l)_{ij} \sim Binomial\left(n(n-1), Q_{ij}^2\right),$$

since A_k and B_l are mutually independent $(k = 1, \ldots, m, \quad l = 1, \ldots, n)$. Then,

$$E(T) = \sum_{i=1}^{d} \sum_{j=1}^{d} \left[\frac{1}{m(m-1)} m(m-1) P_{ij}^2 + \frac{1}{n(n-1)} n(n-1) Q_{ij}^2 - \frac{2}{mn} mn P_{ij} Q_{ij} \right] = \sum_{i=1}^{d} \sum_{j=1}^{d} \left(P_{ij} - Q_{ij}\right)^2 = \| P - Q\|_F^2.$$

In the form of T_{ij}, the first term and the second term represent a similarity (closeness) within two samples, and the last term represents similarity between two samples. Hence, a relatively large value of T_{new} is the evidence against the null hypothesis. Note that the proposed statistic does not require equal sample sizes and undirected graphs assumptions.

When $m = n$, we have a simpler form of the estimate. Let $Z = (z_1, \ldots, z_m)$ be $i.i.d$ random variables $z_k = (A_k, B_k) \sim P \times Q \ (k = 1, \ldots, m)$. Then,

$$T_{new} = \sum_{i=1}^{d} \sum_{j=1}^{d} T_{ij}, \tag{8}$$

where

$$T_{ij} = \frac{1}{m(m-1)} \sum_{k \neq l}^{m} h(u_k, u_l)_{ij}, \tag{9}$$

and $h(z_k, z_l)_{ij} = (A_k)_{ij}(A_l)_{ij} + (B_k)_{ij}(B_l)_{ij} - (A_k)_{ij}(B_l)_{ij} - (A_l)_{ij}(B_k)_{ij}$. Since the proposed estimate has a form of U-statistics, which provides a minimum-variance unbiased estimator for T (Hoeffding, 1992; Serfling, 2009), the asymptotic distribution of T_{new} can be derived based on the asymptotic results of U-statistics.

Theorem 3.1 Assume $E(h^2) < \infty$. Under H_1, we have

$$\sqrt{m}(T_{new} - T) \xrightarrow{d} N\left(0, d^2\sigma^2\right), \tag{10}$$

where $\sigma^2 = \mathrm{var}_z\left(E_{z'} h(z, z')_{ij}\right)$. Under H_0, the U-statistic is degenerate and

$$mT_{new} \xrightarrow{d} \sum_{u=1}^{\infty} d^2 \lambda_u \left(\xi_u^2 - 1\right), \tag{11}$$

where $\xi_u \overset{i.i.d}{\sim} N(0,1)$ and λ_u are the solutions of

$$\lambda_u \phi_u(z) = \int_{z'} h(z, z')_{ij} \phi_u(z') dP(z'). \tag{12}$$

PROOF. These results can be obtained by applying the asymptotic properties of U-statistics as given in Serfling (2009) and the IER assumptions.

Having devised the test statistic, our next aim is to determine whether the new test statistic T_{new} is large enough to be outside the $1 - \alpha$ quantile of the limiting null distribution in **Eq. 11**, where a is the significance level of the test. One difficulty in implementing this test is that the asymptotic null distribution 11) and its a quantile do not have an analytic form unless $\lambda_u = 0$ or 1. Therefore, in order to estimate this quantile, we propose a permutation approach on the aggregated data. The main advantage of this method is that it yields a valid level a test in finite-sample scenarios (Lehmann and Romano, 2006). To this end, we first consider a simpler form of the test statistic (based on T_{new}) defined as follows:

$$T'_{new} = \sum_{i=1}^{d} \sum_{j=1}^{d} T'_{ij}, \tag{13}$$

where

$$T'_{ij} = \frac{1}{m(m-1)} \sum_{k \neq l}^{m} (A_k)_{ij}(A_l)_{ij} + \frac{1}{n(n-1)} \sum_{k \neq l}^{n} (B_k)_{ij}(B_l)_{ij}. \tag{14}$$

Although we do not use the last term of T_{ij} in the definition of T'_{ij}, the performance of the test statistic T'_{new} achieved by incorporating similarities in two samples is still maintained in the permutation framework. The permutation test is summarized in **Algorithm 1**; its computational cost is $O(R(m \vee n)^2)$, where $(m \vee n)$ indicates the maximum among m and n.

Algorithm 1Permutation test using T'_{new}.

```
Input: Graph samples $\mathcal{A}_1, \ldots, \mathcal{A}_m$ and $\mathcal{B}_1, \ldots, \mathcal{B}_n$; Significance level α; Number of permutation $R$.
Output: Reject the null hypothesis $H_0$ if $p$-value $\leq \alpha$.
1: Compute $T'_{new}$ by Eqs. 13, 14.
2: for $r = 1$ to $R$ do
3:     Randomly permute the pooled samples $\{\mathcal{A}_1, \ldots, \mathcal{A}_m, \mathcal{B}_1, \ldots, \mathcal{B}_n\}$ and divide into two groups with sample sizes m and n.
4:     Compute $T'_r$ which is $T'_{new}$ (as given in Eqs. 13, 14 calculated using permuted samples.
5: end for.
6: Calculate $p$-value $= \left|\{r : T'_r \geq T'_{new}\}/R\right|$
```

Unlike Ghoshdastidar and von Luxburg (2018) where the test is reliable even for a small number of samples, due to its asymptotic distribution, our test procedure needs a reasonable number of samples to implement the permutation test. Based on simulations, we see that as low as four samples are sufficient to obtain reliable results.

4 EXPERIMENTS

Here, we first examine the performance of the new test statistics under diverse settings through simulation studies. Later, we will apply the new test to real-world applications.

4.1 Simulated Data

To evaluate the performance of the new test, we examine sparse graphs from stochastic block models with two communities as studied in Tang et al. (2017a) an Ghoshdastidar and von Luxburg

(2018). Specifically, we consider sparse graphs with d nodes where the same $d/2$ size community is constructed with an edge probability p and $d/2$ size different community with an edge probability q. In other words, we define p and Q as follows:

$$P:\begin{pmatrix} p & q \\ q & p \end{pmatrix}_{d\times d} \quad vs \quad Q:\begin{pmatrix} p+\epsilon & q \\ q & p+\epsilon \end{pmatrix}_{d\times d}.$$

We generate m samples from p and n samples from Q. Under the null, $\epsilon = 0$, implying $P = Q$, whereas $\epsilon > 0$ under H_1, implying $P \neq Q$. Following Ghoshdastidar and von Luxburg (2018), we set $p = 0.1$, $q = 0.05$, and $\epsilon = 0$ for null, whereas $\epsilon = 0.04$ for the alternative hypothesis. We examine the performance of the new test for different choices of $d \in \{100, 200, 300, 400, 500\}$.

The performance of the test based on T'_{new} is studied and compared to existing methods. T_fro in Ghoshdastidar and von Luxburg (2018) is the bootstrap test based on T_{fro}, and T_asymp denotes the normal dominance test based on the asymptotic distribution of T_{fro} (also from Ghoshdastidar and von Luxburg (2018)). We denote the new test which is the permutation test based on T'_{new} as T'_new. The estimated power is calculated as the number of null rejections at $\alpha = 0.05$ level out of 100 independent trials for each of these methods. For T_fro and T_new, p-values are determined by 1,000 permutation runs to have a reliable comparison.

Figure 1 shows results for the undirected graph case under different settings. When two sample sizes are equal (upper panels), where existing methods can be applied, we see that the proposed test outperforms all other methods. Note that, when the sample size of two graph populations are different (i.e., $m \neq n$), the existing methods cannot be applied. We see that the proposed test still performs well under sample imbalance and the large d regime.

We also evaluate the performance of the new test for directed graphs under various configurations. (**Figure 2**). The existing methods are not applicable to directed graphs, but we transform T_{fro} so that it can be applied to directed graphs. The results show that the new test also has better power than the existing method in two-sample testing for directed graph and works well for large graphs.

Next, we examine the effect of the sparsity on the performance of the tests. To this end, we consider the same setting as above, but with different choices of $\epsilon \in \{0.02, 0.03, 0.04\}$ for each of methods. Small ϵ implies that there is small difference between p and Q, making the tests more difficult to detect discrepancy between two samples. **Table 1** shows results for undirected graphs with variations in the sparsity level ϵ. We see that, in general, the proposed method is consistently superior to existing methods. This indicates that our test statistic is more effective in detecting the inhomogeneity between two samples than the existing methods. The effect of a sparsity level ϵ on the performance of the proposed test for directed graphs can be found in **Table 2**. We see that the proposed test also performs better than the

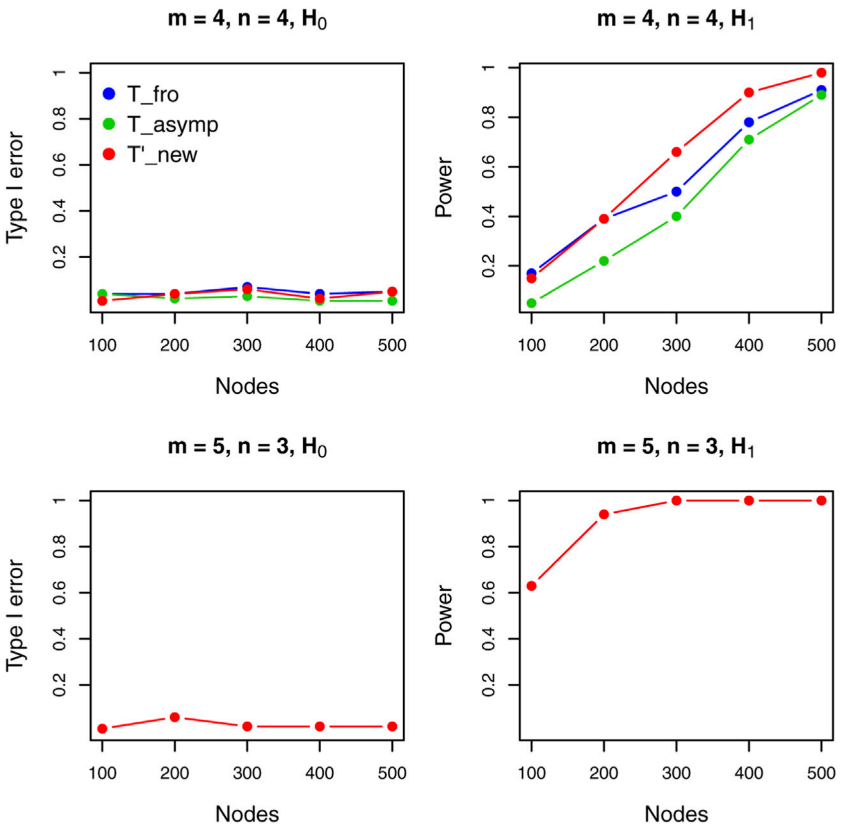


FIGURE 1 | Performance comparison of different tests for undirected graphs.

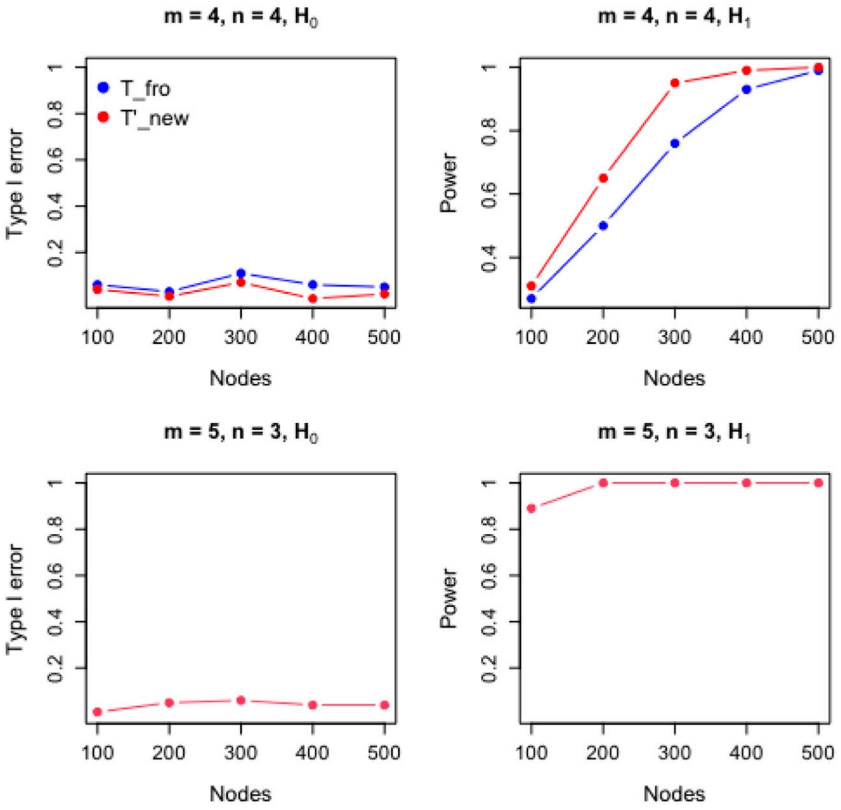


FIGURE 2 | Performance comparison of proposed test for directed graphs.

TABLE 1 | Power comparison of different tests for undirected graphs with varying sparsity levels.

$m = n = 4$	$\epsilon = 0.02$			$\epsilon = 0.03$			$\epsilon = 0.04$		
D	***T_fro***	***T_asymp***	***T'_new***	***T_fro***	***T_asymp***	***T'_new***	***T_fro***	***T_asymp***	***T'_new***
100	0.09	0.05	**0.10**	**0.10**	0.03	0.08	**0.17**	0.05	**0.17**
200	**0.09**	0.05	0.07	**0.18**	0.10	**0.18**	**0.39**	0.22	**0.39**
300	**0.17**	0.03	**0.17**	0.34	0.19	**0.37**	0.50	0.40	**0.66**
400	0.11	0.09	**0.15**	0.40	0.26	**0.53**	0.78	0.71	**0.90**
500	**0.22**	0.08	**0.22**	0.63	0.48	**0.75**	0.91	0.89	**0.98**
$m = n = 8$	$\epsilon = 0.02$			$\epsilon = 0.03$			$\epsilon = 0.04$		
d	***T_fro***	***T_asymp***	***T'_new***	***T_fro***	***T_asymp***	***T'_new***	***T_fro***	***T_asymp***	***T'_new***
100	**0.13**	0.05	0.08	0.17	0.08	**0.23**	0.39	0.21	**0.64**
200	0.19	0.09	**0.31**	0.40	0.20	**0.67**	0.80	0.66	**0.99**
300	0.36	0.22	**0.49**	0.73	0.58	**0.92**	0.98	0.94	**1.00**
400	0.37	0.19	**0.61**	0.92	0.86	**1.00**	**1.00**	0.99	**1.00**
500	0.51	0.31	**0.76**	0.98	0.96	**1.00**	**1.00**	**1.00**	**1.00**

Bold values indicate the largest power of the test under each condition.

existing method for directed graph settings, and as expected, the power increases as ϵ or the number of samples increases.

This observation becomes particularly evident when we have a large number of samples. To this end, we study how the performance of the tests is affected by the number of samples. For this study, we consider $m = n \in \{10, 20, 50\}$ with relatively small graphs $d \in \{50, 100, 150, 200\}$ and fix $\epsilon = 0.02$. This analysis is designed to reveal the potential impact of sample size in high-dimensional settings. **Tables 3**, **4** report numerical results for the performance of the tests with varying number of samples. We see that the proposed test in general outperforms the existing tests for both undirected and directed graphs. Hence, we can claim that the new test works well in high-dimensional settings.

TABLE 2 | Power of the proposed test for directed graphs with varying sparsity levels.

***m* = *n* = 4**	***ϵ* = 0.02**		***ϵ* = 0.03**		***ϵ* = 0.04**	
D	***T_fro***	***T′_new***	***T_fro***	***T′_new***	***T_fro***	***T′_new***
100	**0.13**	0.09	**0.11**	**0.11**	0.21	**0.26**
200	0.11	**0.12**	0.25	**0.27**	0.49	**0.66**
300	0.17	**0.22**	0.46	**0.61**	0.76	**0.94**
400	**0.20**	**0.20**	0.60	**0.72**	0.95	**1.00**
500	0.36	**0.37**	0.77	**0.93**	**1.00**	**1.00**

***m* = *n* = 8**	***ϵ* = 0.02**		***ϵ* = 0.03**		***ϵ* = 0.04**	
D	***T_fro***	***T′_new***	***T_fro***	***T′_new***	***T_fro***	***T′_new***
100	0.14	**0.18**	0.20	**0.42**	0.66	**0.93**
200	0.26	**0.38**	0.77	**0.94**	0.97	**1.00**
300	0.43	**0.68**	0.94	**1.00**	**1.00**	**1.00**
400	0.62	**0.89**	**1.00**	**1.00**	**1.00**	**1.00**
500	0.80	**0.96**	**1.00**	**1.00**	**1.00**	**1.00**

Bold values indicate the largest power of the test under each condition.

TABLE 4 | Power comparison of different tests for directed graphs with varying sample sizes.

Directed	***m* = *n* = 10**		***m* = *n* = 20**		***m* = *n* = 50**	
d	***T_fro***	***T′_new***	***T_fro***	***T′_new***	***T_fro***	***T′_new***
50	0.05	**0.09**	0.12	**0.28**	0.49	**0.77**
100	0.15	**0.24**	0.29	**0.43**	0.82	**0.99**
150	0.15	**0.21**	0.39	**0.52**	0.95	**1.00**
200	0.28	**0.42**	0.66	**0.86**	**1.00**	**1.00**

Bold values indicate the largest power of the test under each condition.

TABLE 5 | Test summary on the phone-call network.

Test statistic	***p*-value**
15.8131	< 0.001

4.2 Real-World Applications

4.2.1 Phone-Call Network

The MIT Media Laboratory conducted a study following 87 subjects who used mobile phones with a pre-installed device that can record call logs. The study lasted for 330°days from July 2004 to June 2005 (Eagle et al., 2009). Given the richness of this dataset, one question of interest to answer is that whether the phone call patterns among subjects are different between weekends and weekdays. These patterns can be viewed as a representation of the personal relationship and professional relationships of a subject. Removing days with no calls among subjects, there are $t = 299$ networks in total (corresponding to number of days) and 87 subjects (or nodes) with adjacency matrices N_t with value one for element (i, j) if subject i called j on day t and 0 otherwise. This in turn comprises of 85°days in weekends and 214°days in weekdays. This is an example of unweighted directed graphs with imbalanced sample sizes.

The test statistic and corresponding p-value are shown in **Table 5**. We see that the new test rejects the null hypothesis of equal distribution at 0.05 significance level. This outcome is intuitively plausible as phone call patterns in weekends (personal) can be different from the patterns in weekdays (work).

4.2.2 Safety-Critical Healthcare Application

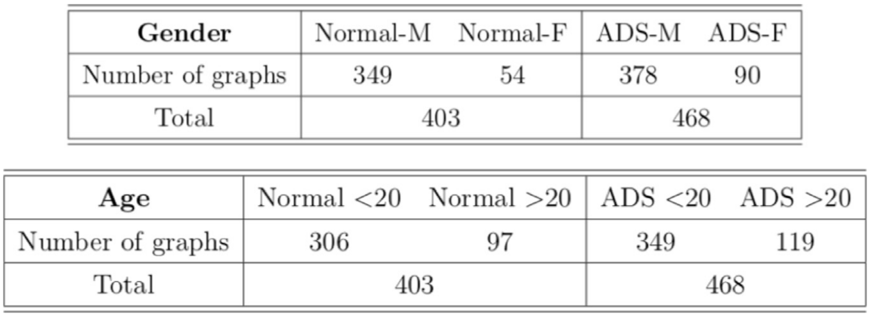
Modeling relationships between functional or structural regions in the brain is a significant step toward understanding, diagnosing, and eventually treating a gamut of neurological conditions including epilepsy, stroke, and autism. A variety of sensing mechanisms, such as functional-MRI, Electroencephalography (EEG), and Electrocorticography (ECoG), are commonly adopted to uncover patterns in both brain structure and function. In particular, the resting state fMRI (Kelly et al., 2008) has been proven effective in identifying diagnostic biomarkers for mental health conditions such as the Alzheimer disease (Chen et al., 2011) and autism (Plitt et al., 2015). At the core of these neuropathology studies is predictive models that map variations in brain functionality, obtained as time-series measurements in regions of interest, to clinical scores. For example, the Autism Brain Imaging Data Exchange (ABIDE) is a collaborative effort (Di Martino et al., 2014), which seeks to build a data-driven approach for autism diagnosis. Further, several published studies have reported that predictive models can reveal patterns in brain activity that act as effective biomarkers for classifying patients with mental illness (Plitt et al., 2015). Following current practice (Parisot et al., 2017), graphs are natural data structures to model the functional connectivity of human brain (e.g. fMRI), where nodes correspond to the different functional regions in the brain and edges represent the functional correlations between the regions. The problem of defining appropriate metrics to compare these graphs and thereby identify suitable biomarkers for autism severity has been of significant research interest. We show that the proposed two-sample test is highly effective at characterizing stratification based on demographics (e.g. age, gender) as well as autism severity states (normal vs abnormal) across a large population of brain networks.

TABLE 3 | Power comparison of different tests for undirected graphs with varying sample sizes.

	***m* = *n* = 10**			***m* = *n* = 20**			***m* = *n* = 50**		
d	***T_fro***	***T_asymp***	***T′_new***	***T_fro***	***T_asymp***	***T′_new***	***T_fro***	***T_asymp***	***T′_new***
50	0.08	0.08	**0.12**	0.11	0.04	**0.16**	0.28	0.15	**0.43**
100	0.16	0.08	**0.17**	0.18	0.05	**0.23**	0.61	0.42	**0.81**
150	**0.16**	0.03	0.15	0.21	0.14	**0.30**	0.70	0.52	**0.97**
200	0.14	0.06	**0.22**	0.37	0.21	**0.56**	0.94	0.89	**1.00**

Bold values indicate the largest power of the test under each condition.

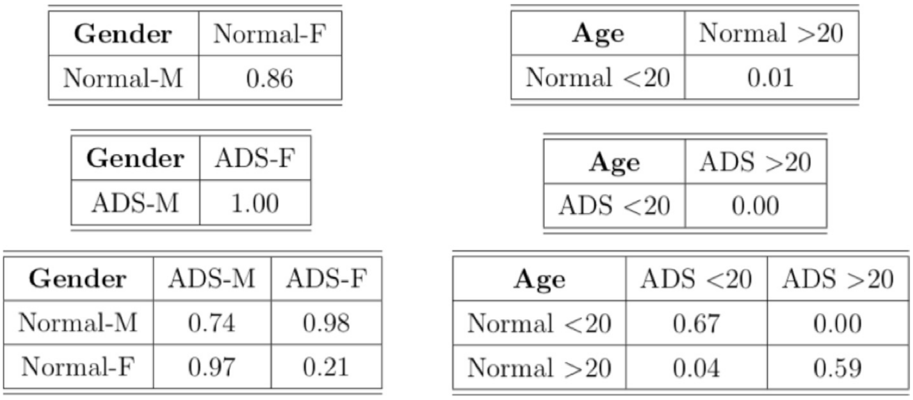
TABLE 6 | Distribution of graphs. "M" and "F" indicate male and female, respectively. '<20' and '>20' represent age less than 20 and over 20, respectively.



Gender	Normal-M	Normal-F	ADS-M	ADS-F
Number of graphs	349	54	378	90
Total	403		468	

Age	Normal <20	Normal >20	ADS <20	ADS >20
Number of graphs	306	97	349	119
Total	403		468	

TABLE 7 | *p*-values of the tests on the ABIDE dataset.



Gender	Normal-F
Normal-M	0.86

Age	Normal >20
Normal <20	0.01

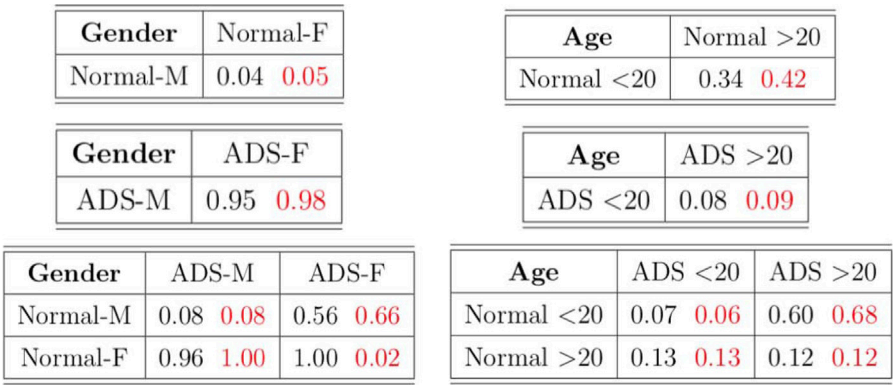
Gender	ADS-F
ADS-M	1.00

Age	ADS >20
ADS <20	0.00

Gender	ADS-M	ADS-F
Normal-M	0.74	0.98
Normal-F	0.97	0.21

Age	ADS <20	ADS >20
Normal <20	0.67	0.00
Normal >20	0.04	0.59

TABLE 8 | Estimated power of the tests with the significance level at 5%. Black numbers indicate the power of test based on *T_fro* and red numbers represent the power of test based on *T'_new*.



Gender	Normal-F
Normal-M	0.04 0.05

Age	Normal >20
Normal <20	0.34 0.42

Gender	ADS-F
ADS-M	0.95 0.98

Age	ADS >20
ADS <20	0.08 0.09

Gender	ADS-M	ADS-F
Normal-M	0.08 0.08	0.56 0.66
Normal-F	0.96 1.00	1.00 0.02

Age	ADS <20	ADS >20
Normal <20	0.07 0.06	0.60 0.68
Normal >20	0.13 0.13	0.12 0.12

In the dataset, there are total 871 graphs and each graph consists of 111 nodes (functional regions). Through this example, we study the effectiveness of our approach under the weighted and undirected graph setting. In particular, we focus on detecting variations across stratification arising from demographics (gender, age). Specifically, groups of normal control subjects as well as those diagnosed with Autism Spectrum Disorders (ADS) are further sub-divided according to their gender (Male or Female) and age (under 20 or over 20), and we compare these sub-groups using the proposed test. **Table 6** shows the distribution of graphs in the dataset and **Figure 3** shows an example of the network structure of normal-male and normal-female groups.

We conduct the two-sample test based on T'_new for each group with 10,000 permutations and the results are summarized in **Table 7**. We see that the new test rejects the null hypothesis of homogeneity in groups with respect to the treatment and age at 5% significance

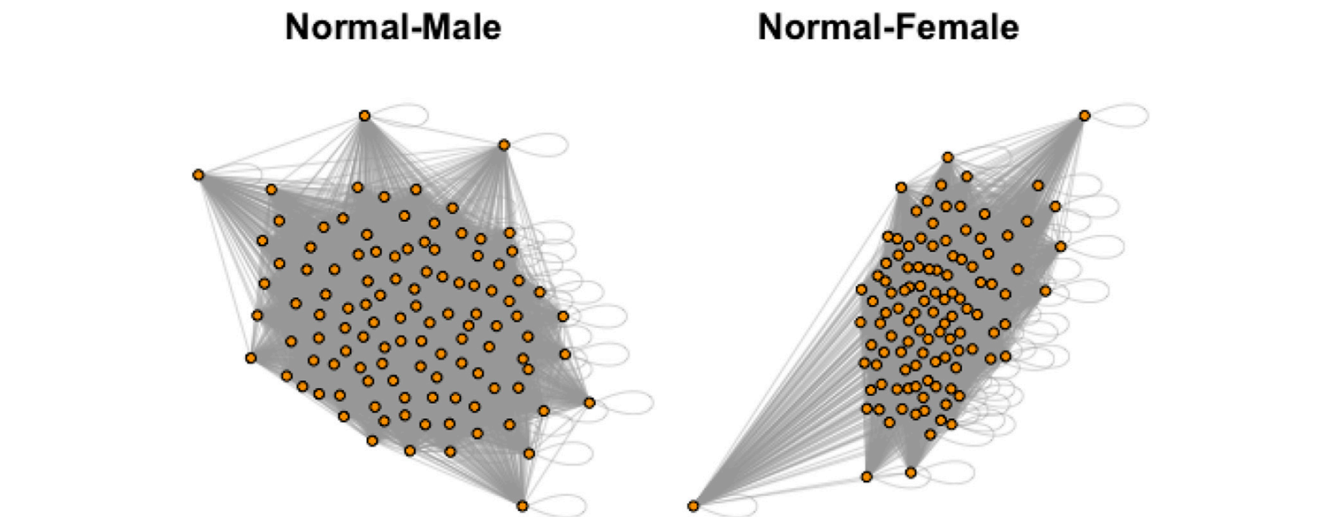


FIGURE 3 | Example networks from Normal-Male and Normal-Female groups.

level (Normal>20 vs ADS<20 and Normal<20 vs ADS>20). In addition, the new test rejects the null hypothesis of homogeneity in both normal and ADS groups with respect to the age difference (Normal<20 vs Normal>20 and ADS<20 vs ADS>20).

This conclusion indicates there is a dataset shift even within the same normal and ADS groups, depending on the age. Hence, the fact that normal and ADS groups are considered differently by age may affect the machine learning subjects classification and prediction task in population. Moreover, with the dataset in which the normal group and ADS group are determined differently by age and not by gender, the machine learning classification and prediction model may not be reliable. Hence, detecting dataset shift shed some light on the machine learning task for more reliable results.

We also compare the new test with the existing method $T_{\_}fro$ to this example. Note that the existing method $T_{\_}asymp$ may not be reliable due to the small number of nodes. Since $T_{\_}fro$ is only applicable to the balanced sample sizes, we randomly choose 54 graphs from each group as the smallest sample size among the groups is 54. We run the tests 100 times at the significance level 5%. The test powers are shown in **Table 8**. We see that the new test in general outperforms $T_{\_}fro$. Compared to the results in **Table 7**, some examples show inconsistent performance of the tests. This is because we only consider a subset of graphs due to the limitation of the existing approaches in that they cannot be applied to unbalanced sample size examples.

5 CONCLUSION

We propose the new two-sample test statistic for graph-structured data. Unlike the existing methods, the new test statistic is more versatile, which is applicable to directed graphs, imbalanced sample size cases, and even weighted graphs. The asymptotic distribution of the test statistic is presented and a practical testing procedure is proposed. The performance of the new method is studied under a number of settings. Experiments demonstrate that the new test in general outperforms state-of-the-art tests. The proposed test is also applied to two real datasets (including a safety-critical healthcare application), and we reveal that the new approach is effective to detecting the heterogeneity between disparate samples.

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

HS developed the main method and proposed the testing procedure based on the new test statistic. He conducted the simulation experiments and real data analysis. JJ and BK provided the intuition and the direction of the method and worked on simulation experiments with HS. JJ provided the real dataset, and JJ and BK discussed about the results with HS. HS, JJ, and BK generated the paper together.

FUNDING

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. This work was supported by the DOE Advanced Scientific Computing Research. Release number LLNL-JRNL-822138.

REFERENCES

Bollobás, B., Janson, S., and Riordan, O. (2007). The Phase Transition in Inhomogeneous Random Graphs. *Random Struct. Alg.* 31, 3–122. doi:10.1002/rsa.20168

Bubeck, S., Ding, J., Eldan, R., and Rácz, M. Z. (2016). Testing for High-Dimensional Geometry in Random Graphs. *Random Struct. Alg.* 49, 503–532. doi:10.1002/rsa.20633

Bulusu, S., Kailkhura, B., Li, B., Varshney, P. K., and Song, D. (2020). Anomalous Instance Detection in Deep Learning: A Survey. Available at: arXiv:2003.06979 (Accessed March 16, 2020).

Chen, G., Ward, B. D., Xie, C., Li, W., Wu, Z., Jones, J. L., et al. (2011). Classification of Alzheimer Disease, Mild Cognitive Impairment, and Normal Cognitive Status with Large-Scale Network Analysis Based on Resting-State Functional Mr Imaging. *Radiology* 259, 213–221. doi:10.1148/radiol.10100734

Di Martino, A., Yan, C.-G., Li, Q., Denio, E., Castellanos, F. X., Alaerts, K., et al. (2014). The Autism Brain Imaging Data Exchange: toward a Large-Scale Evaluation of the Intrinsic Brain Architecture in Autism. *Mol. Psychiatry* 19, 659–667. doi:10.1038/mp.2013.78

Eagle, N., Pentland, A., and Lazer, D. (2009). Inferring Friendship Network Structure by Using Mobile Phone Data. *Proc. Natl. Acad. Sci.* 106, 15274–15278. doi:10.1073/pnas.0900282106

Gao, C., and Lafferty, J. (2017). Testing Network Structure Using Relations between Small Subgraph Probabilities. Available at: arXiv:1704.06742 (Accessed April 22, 2017).

Ghoshdastidar, D., and von Luxburg, U. (2018). "Practical Methods for Graph Two-Sample Testing," in Advances in Neural Information Processing Systems, December, 2018, 3019–3028.

Ghoshdastidar, D., Gutzeit, M., Carpentier, A., and von Luxburg, U. (2017a). Two-sample Hypothesis Testing for Inhomogeneous Random Graphs. Available at: arXiv:1707.00833 (Accessed July 4, 2017).

Ghoshdastidar, D., Gutzeit, M., Carpentier, A., and von Luxburg, U. (2017b). Two-sample Tests for Large Random Graphs Using Network Statistics. Available at: arXiv:1705.06168v2 (Accessed May 26, 2017).

Ginestet, C. E., Fournel, A. P., and Simmons, A. (2014). Statistical Network Analysis for Functional Mri: Summary Networks and Group Comparisons. *Front. Comput. Neurosci.* 8, 51. doi:10.3389/fncom.2014.00051

Ginestet, C. E., Li, J., Balachandran, P., Rosenberg, S., and Kolaczyk, E. D. (2017). Hypothesis Testing for Network Data in Functional Neuroimaging. *Ann. Appl. Stat.* 11, 725–750. doi:10.1214/16-aoas1015

Ginestet, C. E., Nichols, T. E., Bullmore, E. T., and Simmons, A. (2011). Brain Network Analysis: Separating Cost from Topology Using Cost-Integration. *PloS one* 6, e21570. doi:10.1371/journal.pone.0021570

Hoeffding, W. (1992). "A Class of Statistics with Asymptotically Normal Distribution," in *Breakthroughs in Statistics (Springer)*, 308–334. doi:10.1007/978-1-4612-0919-5_20

Holland, P. W., Laskey, K. B., and Leinhardt, S. (1983). Stochastic Blockmodels: First Steps. *Social networks* 5, 109–137. doi:10.1016/0378-8733(83)90021-7

Kelly, A. M. C., Uddin, L. Q., Biswal, B. B., Castellanos, F. X., and Milham, M. P. (2008). Competition between Functional Brain Networks Mediates Behavioral Variability. *Neuroimage* 39, 527–537. doi:10.1016/j.neuroimage.2007.08.008

Lehmann, E. L., and Romano, J. P. (2006). *Testing Statistical Hypotheses*. Berlin, Germany: Springer Science & Business Media.

Lei, J., (2016). A Goodness-Of-Fit Test for Stochastic Block Models. *Ann. Stat.* 44, 401–424. doi:10.1214/15-aos1370

Macindoe, O., and Richards, W. (2010). Graph Comparison Using Fine Structure Analysis, IEEE Second International Conference on Social Computing. IEEE. doi:10.1109/socialcom.2010.35

Maugis, P., Priebe, C. E., Olhede, S. C., and Wolfe, P. J. (2017). Statistical Inference for Network Samples Using Subgraph Counts. Available at: arXiv:1701.00505 (Accessed January 2, 2017).

Newman, M. E., and Girvan, M. (2004). Finding and Evaluating Community Structure in Networks. *Phys. Rev. E* 69, 026113. doi:10.1103/physreve.69.026113

Newman, M. E. J. (2006). Modularity and Community Structure in Networks. *Proc. Natl. Acad. Sci.* 103, 8577–8582. doi:10.1073/pnas.0601602103

Parisot, S., Ktena, S. I., Ferrante, E., Lee, M., Moreno, R. G., Glocker, B., et al. (2017). "Spectral Graph Convolutions for Population-Based Disease Prediction," in International Conference On Medical Image Computing and Computer-Assisted Intervention, QC, Canada, September 10–14, 2017 (Springer), 177–185.

Plitt, M., Barnes, K. A., and Martin, A. (2015). Functional Connectivity Classification of Autism Identifies Highly Predictive Brain Features but Falls Short of Biomarker Standards. *NeuroImage: Clin.* 7, 359–366. doi:10.1016/j.nicl.2014.12.013

Rabanser, S., Günnemann, S., and Lipton, Z. (2019). "Failing Loudly: An Empirical Study of Methods for Detecting Dataset Shift," in 33rd Conference on Neural Information Processing Systems, Vancover, Canada, 1396–1408.

Serfling, R. J. (2009). Approximation Theorems of Mathematical Statistics. Hoboken, NJ: John Wiley & Sons.

Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., and Borgwardt, K. (2009). "Efficient Graphlet Kernels for Large Graph Comparison," in Artificial Intelligence and Statistics, 488–495.

Tang, M., Athreya, A., Sussman, D. L., Lyzinski, V., Park, Y., and Priebe, C. E. (2017a). A Semiparametric Two-Sample Hypothesis Testing Problem for Random Graphs. *J. Comput. Graphical Stat.* 26, 344–354. doi:10.1080/10618600.2016.1193505

Tang, M., Athreya, A., Sussman, D. L., Lyzinski, V., and Priebe, C. E. (2017b). A Nonparametric Two-Sample Hypothesis Testing Problem for Random Graphs. *Bernoulli* 23, 1599–1630. doi:10.3150/15-bej789

Disclaimer: The views and opinions of the authors do not necessarily reflect those of the U.S. government or Lawrence Livermore National Security, LLC neither of whom nor any of their employees make any endorsements, express or implied warranties or representations or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of the information contained herein.

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.

Advantages of publishing in Frontiers

OPEN ACCESS

Articles are free to read
for greatest visibility
and readership

FAST PUBLICATION

Around 90 days
from submission
to decision

HIGH QUALITY PEER-REVIEW

Rigorous, collaborative,
and constructive
peer-review

TRANSPARENT PEER-REVIEW

Editors and reviewers
acknowledged by name
on published articles

Frontiers
Avenue du Tribunal-Fédéral 34
1005 Lausanne | Switzerland

Visit us: www.frontiersin.org
Contact us: frontiersin.org/about/contact

REPRODUCIBILITY OF RESEARCH

Support open data
and methods to enhance
research reproducibility

DIGITAL PUBLISHING

Articles designed
for optimal readership
across devices

FOLLOW US

@frontiersin

IMPACT METRICS

Advanced article metrics
track visibility across
digital media

EXTENSIVE PROMOTION

Marketing
and promotion
of impactful research

LOOP RESEARCH NETWORK

Our network
increases your
article's readership