IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 10 June 2024, accepted 28 June 2024, date of publication 4 July 2024, date of current version 26 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3423323

SURVEY

A Systematic Review of Adversarial Machine Learning Attacks, Defensive Controls, and Technologies

JASMITA MALIK, (Student Member, IEEE), RAJA MUTHALAGU, AND PRANAV M. PAWAR, (Member, IEEE)
Department of Computer Science, Birla Institute of Technology and Science, Pilani, Dubai Campus, Dubai, United Arab Emirates
Corresponding authors: Jasmita Malik (p20210904@dubai.bits-pilani.ac.in) and Raja Muthalagu (raja.m@dubai.bits-pilani.ac.in)

This work was supported by the Birla Institute of Technology and Science, Pilani for paying the Article Processing Charges (APC) of this publication.

ABSTRACT Adversarial machine learning (AML) attacks have become a major concern for organizations in recent years, as AI has become the industry's focal point and GenAI applications have grown in popularity around the world. Organizations are eager to invest in GenAI applications and develop their own large language models, but they face numerous security and data privacy issues, particularly AML attacks. AML attacks have jeopardized numerous large-scale machine learning models. If carried out successfully, AML attacks can significantly reduce the efficiency and precision of machine learning models. They have far-reaching negative consequences in the context of critical healthcare and autonomous transportation systems. In this paper, AML attacks are identified, analyzed, and classified using adversarial tactics and techniques. This research also recommends open-source tools for testing AI and ML models against AML attacks. Furthermore, this research suggests specific mitigating measures against each attack. It aims to serve as a guidance for organizations to defend against AML attacks and gain assurance in the security of ML models.

INDEX TERMS Adversarial machine learning, AI assurance, cybersecurity, data privacy, secure software development lifecycle.

I. INTRODUCTION

Adversarial machine learning (AML) is a growing challenging threat in the AI industry. With AI and ML models being prominently used across various sectors, AML attacks have received increasing attention from developers and security researchers. Critical industrial systems such as autonomous transportation and healthcare systems utilize ML models. These systems if compromised can result into catastrophic health and safety accidents. For example, if an autonomous self-driving car that uses a traffic sign detection system is compromised by adversaries, it can result in the car falsely detecting a stop sign as a speed limit sign. This may lead to road accidents. Similarly, if a medical system is compromised by competing healthcare service providers, fraudulent insurance firms or threat actors, can misdiagnose a disease and provide wrong treatment plans to patients. New developments in natural language processing and computer vision [1], [2] have brought trained classifiers closer to the forefront of critical security systems as well. Virus detection, face recognition are some common examples. These advancements have made machine learning security crucial. Specifically, resistance against inputs chosen by adversaries has become a prominent design objective in most ML systems. Although trained ML models are often rather good at identifying benign inputs, it has been shown in past studies [3], [4] that adversaries can often alter the input such that the model produces an incorrect result. Therefore, it is crucial for security researchers to identify countermeasures to combat against AML attacks and assist organizations in secure deployment of ML models. So far, there has been no comprehensive study found within literature that provides a detailed overview of AML attacks, it's various types, potential impacts, security testing tools as well as

The associate editor coordinating the review of this manuscript and approving it for publication was Tiago Cruz.

countermeasures. Since, ML model architectures are usually quite complex and attacks on its auto-generated data is difficult to comprehend, the following question arises: ''How can one build secure ML models that are resistant to adversarial attacks?'' This paper aims to resolve this question.

A. PROBLEM STATEMENT

In cybersecurity, an attack against a secure system is an effort to make it do anything besides the tasks for which it was intended compromising it's confidentiality, integrity and availability. Similarly, an adversarial attack on an AI system is an effort to convince it to perform a function other than that for which it was intended, such as fooling it into generating false results.

By definition, AML attacks are executed by threat actors to cause a ML model to malfunction. These attacks can take place before, during the testing or training phases of a model or even after a model has been put into production. Creating an adversarial example such as an example that has been deliberately designed to be misclassified is usually necessary in order to execute an AML attack. AML attacks come in various forms, such as evasion, poisoning, privacy-based attacks. Attacks by adversaries might also be targeted or untargeted. Targeted attacks involve the calculation of an adversarial example that tricks the AI system into doing a specific action, such as installing spyware or forcing the system to shut down. An untargeted attack, on the other hand, tries to cause random harm to the system, such as tricking the system into misclassifying visuals or noises. For e.g., in image processing, we can often make two images with very small modifications appear to the human as identical to each other. However, in an ML model these resemble two entirely different images due to the different pixel values and classification boundaries. This is depicted in Figure 1.

B. PROBLEM DESCRIPTION

Here in Figure 1, it is noticed that the panda on the left is unmodified, and the convolutional network (ConvNet) trained on that dataset's image can identify it as such. However, after adding a minor adversarial perturbation to the image, the ConvNet fails to recognize it as a panda and instead classifies it as a gibbon.

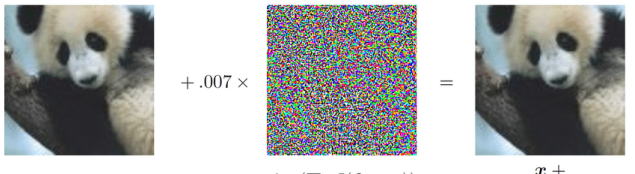

FIGURE 1. When an adversarial input is added to a normal image, the classifier incorrectly identifies the image of a panda as a gibbon [3].

The image in the middle gives the optimal direction to shift all pixels if one computes exactly how one might alter the image to cause the ConvNet to make a mistake. It appears to a human as noise, although it is not; it is carefully computed as a function of the network's parameters, and there is a lot of structure in there. However, it should be noted that the model does not have a great deal of confidence in that judgment. As a result, this image has a 58% likelihood of being a panda. A 32-bit floating point representation is used to send this image to the ConvNet; a small adjustment is made to the 32-bit floating point representation, leading the ConvNet to believe that this image of a panda is actually of a gibbon. When the image is multiplied by a small coefficient which is added to the original panda image, one gets an image that no human can distinguish from the original panda [3]. The ConvNet has a lot more confidence in its false prediction that the image on the right is a gibbon than it does in its incorrect prediction that the original is a panda. This is another intriguing aspect of it that affects more than just the class. Therefore, it is evident that a ML model can be evaded to provide an inaccurate result if minor perturbations are made to its inputs. There have been several real world AML attacks that have caused major impact to ML models by making such minor modifications to its input data.

C. MOTIVATION OF RESEARCH

AML attacks presents an unparalleled opportunity to delve into one of the most pressing challenges facing the field of AI today. In an era where ML algorithms underpin critical systems across various domains, the specter of adversarial attacks looms large, threatening the confidentiality, integrity and availability of these systems. This literature review aims to dissect the multifaceted landscape of adversarial machine learning, unraveling the intricate techniques employed by adversaries to manipulate and deceive algorithms, while also exploring the cutting-edge defenses and countermeasures devised by researchers to fortify these systems against such threats. By synthesizing and analyzing the latest advancements, theoretical frameworks, and empirical findings in this burgeoning field, this paper seeks to not only deepen our understanding of AML but also pave the way for robust and resilient AI systems that can withstand the relentless onslaught of adversarial manipulation.

D. CONTRIBUTIONS

This study offers a comprehensive examination of AML attacks, including attack tactics, techniques, and their real-world implications, along with open-source tools for testing against these attacks and supporting countermeasures for mitigation. The primary contributions of this research are as follows:

- The study conducts an extensive systematic review of AML attacks and presents a taxonomy of various types of AML attacks.
- The study sheds light on an adversary's tactics and techniques for executing various types of AML attacks and highlights the impact of AML attacks by presenting real-world case studies.

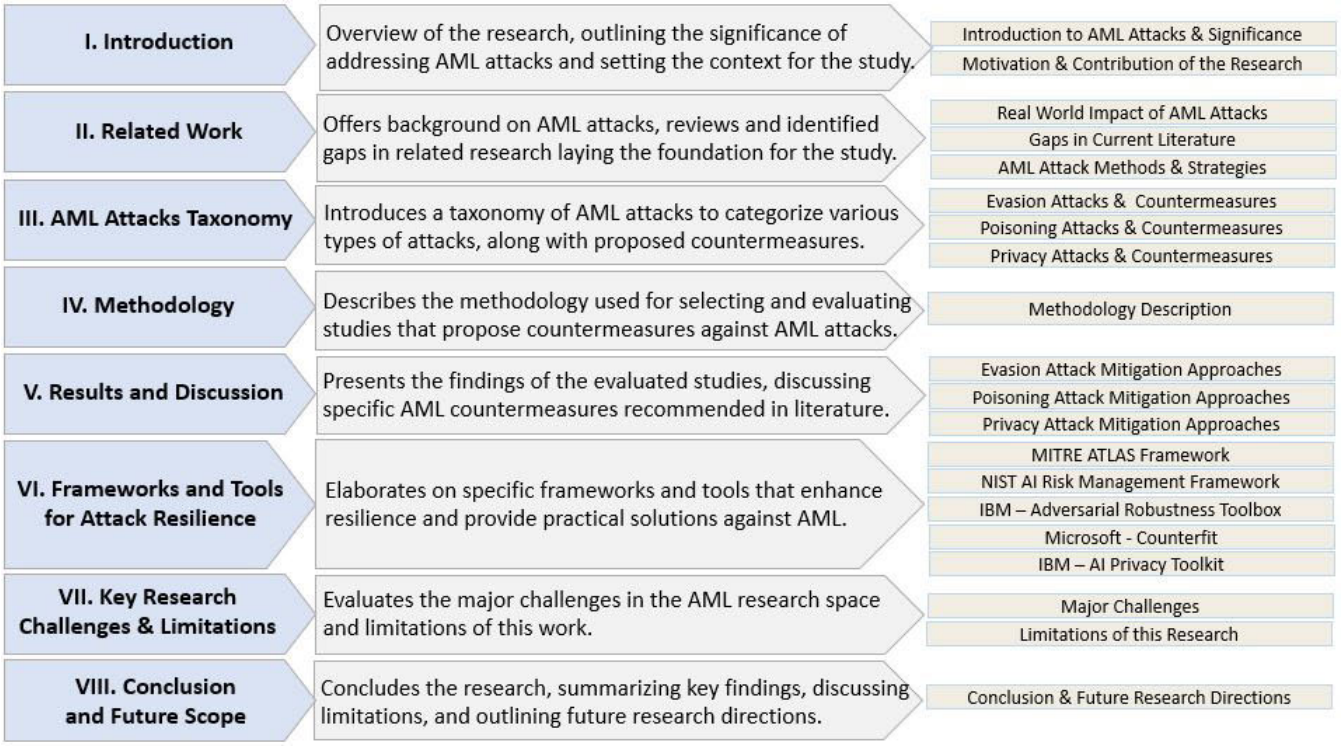

FIGURE 2. Structure of the review article.

- The study's findings shed light on previously unexplored countermeasures for each type of attack enriching the understanding of ML researchers and cybersecurity practitioners alike.
- Beyond theoretical advancements, this work offers practical solutions for defending against AML attacks by proposing open-source frameworks and tools available.

This research serves as a safeguard against AML attacks, enhancing the security posture of AI applications and underlying ML models. By detailing specific mitigating countermeasures for each type of attack, organizations can implement and test them to prevent AML attacks on their critical ML-based applications. This study targets AI and ML developers and security researchers, aiming to provide guidelines for creating secure AI and ML applications resilient to AML attacks, thereby contributing to the advancement of an AI-enabled intelligent world. While securing AI and ML models is a complex journey, this research represents an initial step in guiding organizations to establish a secure ML-model testing environment to enhance the security posture of their applications.

E. PAPER ORGANIZATION

The paper is organized in 8 sections. The overall structure of this paper and description of each section is highlighted below and illustrated in Figure 2

- *Section I - Introduction*: Provides an overview of the research focus outlining the significance of addressing AML attacks and setting the context for the study.
- *Section II - Related Work*: Offers background information on AML attacks, reviews related research, and identifies gaps in the existing literature, laying the foundation for the study.
- *Section III - AML Attacks Taxonomy*: Introduces AML attacks taxonomy to categorize and clarify the various types of attacks, along with proposed countermeasures.
- *Section IV - Methodology*: Describes the methodology used for selecting and evaluating academic studies that propose countermeasures against AML attacks.
- *Section V - Results and Discussion*: Presents the findings of the evaluated studies, discussing specific AML attack countermeasures recommended in literature.
- *Section VI - Recent Frameworks and Tools for Improving Cyber Resilience Against AML Attacks*: Elaborates on specific frameworks and tools that can enhance cyber resilience against AML attacks, providing practical solutions for mitigating risks.
- *Section VII - Key Research Challenges & Limitations*: Evaluates major challenges in the AML research space and limitations of this work.
- *Section VIII - Conclusion and Future Scope*: Concludes the research outlining future research directions to further advance the field.

II. RELATED WORKS

Attackers are now preying on AI and ML systems and causing adverse impacts to organizations by exploiting these systems through AML attacks. AML attacks can have a dramatic

impact on the security of AI applications. Therefore, building secure AI systems that are immune to external AML attacks is crucial. In this section, the researchers underscore the consequences of AML attacks in real-world scenarios, as well as elucidate the research gaps identified in existing literature.

A. MACHINE LEARNING MODELS AND ADVERSARIAL ATTACKS

ML models have shown great success when used to tackle a range of challenging data classification and analysis issues. ML models, a subset of AI, have revolutionized various fields by learning patterns in data and making predictions without explicit programming [5]. These models, particularly for classification algorithms, rely on strong mathematical models and logic to make decisions [6]. In medical diagnosis, ML algorithms have significantly improved accuracy by considering large amounts of patient data [7]. Furthermore, ML has the potential to enhance modeling and simulation practices, as demonstrated in healthcare and autonomous transportation use cases [8]. However, the existence of adverse inputs has raised serious concerns, especially in situations like autonomous driving where misclassification presents a serious risk [9].

FIGURE 3. A stop sign that underwent modification so that the ML model could identify it as a speed limit sign [10].

One can easily see the potential harm that self-driving cars can be impacted by in Figure 3. This shows an example of a stop sign that has been modified such that the ML model recognizes it as a speed restriction sign, but to a human this seems to be like any other stop sign on the street. In this section, the researchers shed light on the real-world impact of AML attacks, accentuating the pressing need for comprehensive solutions within both academic and practical domains. By delineating these gaps, the researchers lay the groundwork for a deeper comprehension of the AML attack methods, strategies used by adversaries thus fostering a more robust and informed approach to mitigating these threats. Additionally, this exploration serves as a catalyst for the introduction of the AML attack taxonomy, proposed countermeasures and open-source frameworks and tools in subsequent sections, poised to enrich both theoretical discourse and practical strategies for combating AML attacks.

B. REAL WORLD IMPACT OF AML ATTACKS AND IDENTIFIED GAPS

On July 21, 2019, vulnerabilities were discovered in the ML-algorithm based Cylance anti-virus program [11]. These vulnerabilities enabled a malicious party to create harmful files that the antivirus program would most likely mistake for benign ones. Attackers used this flaw to change any executable to which they had access. The attackers only required appending strings to the malware, not rewriting it. This allowed an attacker to easily and significantly improve their malware's attack strength against affected antivirus products. Undetected ransomware could be installed and encrypt a computer with Cylance anti-virus product installed. This is an example of an 'evasion attack'. On another occasion, Keen Labs, a recipient of Tesla's ''ethical hacking'' hall of fame award, stated in a research paper [12], [13] that it had found two ways to trick Tesla's Autopilot lane recognition system through changing the physical road surface. The first attempt was to confuse Autopilot using blurring patches on the left-lane line, which the Tesla engineering team said was too difficult for someone to actually deploy in the real world and easy for Tesla's computer to recognize. However, Keen Labs confirmed that by placing three little stickers on the sidewalk, it was able to cause Tesla's self-driving program which used ML models, to swerve off course. This was an example of a combined evasion and data poisoning attack. There was also a flaw reported in Proofpoint's Email Protection system that appeared in 2019 which allowed for the generation of email headers with an embedded spam score [14]. With these scores, an attacker could design a copycat spam detection engine and send spam emails that would not be detected. This is an example of another type of poisoning attack. There have also been multiple occurrences of privacy attacks on healthcare systems leaking patients sensitive data. In one of such occurrences, it was found that an adversary could identify if a person has HIV via a privacy-based, also referred to as membership inference attack on healthcare systems using ML models. The adversary could misuse patient's sensitive personal healthcare data. In general, there have been various such AML attacks recorded in the past that further mandate the need for implementation of better security and privacy controls for ML models. Below are some works in literature that describe impact of AML attacks and how ML models can be dissuaded. Furthermore, these highlights the gaps that need to be addressed in the understanding of ML security and AML attacks.

Lin et al., [15] described the adverse effects of real-world applications using ML techniques when compromised by threat actors. The author initiated the discussion by describing the classic example of how the learning process of spam filters can be manipulated by amending the content

of spam emails, for example, by adding positive words that isn't seen in spam emails but is usually found in authentic emails which can ultimately lead the anti-spam filter solution to misclassify possible spam emails that contain words of this nature and consider the emails to be legitimate. The ML model's filtering performance therefore significantly declined, leading the user to turn off the spam filtering service. This is an another example of a poisoning attack, and it makes the assumption that the attacker can sabotage learning of a model using unauthorized modification of training data. Poisoning attacks are among the most common types of AML attacks. This research also looked at other types of real-world attacks that have happened in the past such as one that impacted a chatbot Tay, a Twitter based chatbot designed to rapidly learn from online conversations. It started delivering offensive and hurtful tweets after it was poisoned by adversarial interactions by malicious Twitter users [16], [17]. Tay had to be shut down only after 16 hours of it's launch. This attack is also an example of a poisoning attack. Another type of AML attack was encrafted on Mozilla's DeepSpeech automated speech-to-text system that used ML audio classifiers. The attackers added nearly inaudible noises resulting in the system to recognize any sentence as the targeted sentence. They used evasion attacks to cause damage to the system. Later on, an advanced audio-based evasion attack was developed to target commercial speech recognition devices. The attackers targeted several speech recognition applications such as Google Cloud Speech-to-Text, Microsoft Bing Speech Service, IBM Speech-to-Text and Amazon Transcribe. Google Assistant, Google Home, Microsoft Cortana and Amazon Echo were in fact used to execute these attacks. The attackers were able to activate some services through inaudible commands without the user being aware. Thus, various ML and AI applications are vulnerable to different types of AML attacks and can have unexpected adverse outcomes. It is vitally important for security experts to secure ML systems against AML attacks.

Apruzzese et al., [18] described how the field of ML security is currently suffering a lack of research and knowledge among security practitioners and ML developers. The authors indicated that in recent years powerful algorithmic attacks against a wide variety of ML models have been encrafted by attackers and major targets have been the bigtech companies such as Meta, Google, Tesla and Amazon. While, these companies can withstand most attacks by developing proper mitigatory controls, mid-sized and smaller companies that adopt ML and AI based applications are at a much higher risk as they struggle to achieve the same level of defense due to lesser budgets and expertise. There are several real-world evidences that suggest that attackers use simple tactics and techniques to subvert ML model-driven systems and sometimes these are not even linked to the model but basic security hygiene controls such as improper access provisioning, absence of adequate training data monitoring, and so on. These can cause an AML attack to be successfully executed. The authors presented 4 case studies of AML attacks that were executed recently. Their study mentioned that in 2020 Gartner had predicted that "by 2022, 30 percent of cyber attacks will leverage poisoning of training-data, model theft, or adversarial examples to execute AML attacks [6]. However, the authors have claimed that the developer community has barely any idea of the importance of securing ML models during development. Below are a few references that indicate the same:

Kumar et al. [19] surveyed 28 businesses in 2020. Participants' opinions on AML were solicited. The majority of interviewees were mostly concerned about poisoning attacks and the participants were unsure on how to react to these. Boenisch et al. [20] issued a critical caution the next year after numerous ML developers acknowledged that "I never gave security for my ML models much thought." Sun et al.'s [21] study examined ML models used on mobile and surprisingly discovered that: "41 percent of ML-based mobile application development firms don't even attempt to safeguard their models. An additional year later, Bieringer et al. [22] spoke with several ML developers and found that, "most lack adequate understanding to secure ML systems in production, "A third said they are uneasy about AML." Grosse et al., [23] conducted interviews with hundreds of practitioners requesting their thoughts on defensive strategies for AML attacks. Generally speaking, they said, "Why do so?" These studies confirm the need for further research required in the AML attacks space. The unexpected reactions from interviewees in these surveys showcase how security is considered only as an afterthought and not a preemptive measure to be taken while designing and developing ML models.

Wiyatno et al., [24] conducted a similar study by focusing on ML models in the visual domain, as this is the area where the most amount of research has been conducted on producing and identifying adversarial attack methods so far. The authors described adversarial instances' transferability, defenses, and attacks in the real world. The authors also reviewed the advantages and disadvantages of several attack defense strategies. Their goal was to understand the mechanics of adversarial attack and defense mechanisms. The benefit of this study was that it covered over 29 types of various AML attacks. Unfortunately, it did not follow a proper taxonomy but it showcased the different methods attackers used in a variety of attacks which will be briefly discussed later. However, the limitation of this study was that it was only conducted over image recognition models and applicable to visual content. The study did not follow an easy to understand simplified taxonomy for the classification of AML attacks. It had too many technical jargons that had not been well explained in layman terms. Also, the study dated back to 2019 and several new types of AML attacks have originated since then. This research covers all types of common AML attacks based on a standard referential taxonomy that is easy to comprehend and can be mapped

effectively to specific countermeasures for each type of an attack.

Gilmer et al., [25] argued that several studies in literature provided impractical solutions to secure ML models from AML attacks. Most studies considered unrealistic ML-centric restrictions that reduced the quality and performance of the model. The authors focused on the idea of examining the relevance and realism of particular AML attack defenses. The study however did not end up providing effective security measures as underlined by recent ML/AI security frameworks. In this research, security measures from authoritarian sources are recommended which can be implemented effectively in real-world scenarios.

Biggio et al., [26] provided a different perspective on AML attacks. The authors mentioned that even though the research interest in AML attacks has grown over the years, the very first seminal work in the area of AML was conducted in 2004. At that time, the study was conducted on how AML attacks could compromise spam-filtering systems by tricking the ML model through making few carefully-crafted changes in the content of spam emails. The study provided insight on an attacker's goals, strategy, capability and knowledge. It also provided details on certain defensive measures that can be used to protect ML models from AML attacks. However, it failed to provide a clear road map to the cyber security and developer community on what are the exact mitigation countermeasures that should be implemented to secure ML models. It could not connect the specific defense countermeasures to the various types of AML attacks.

The majority of prior studies primarily emphasized the magnitude of the AML problem but lacked concrete suggestions for mitigation. In contrast, this research goes beyond merely highlighting the scale of the issue by presenting comprehensive solutions aimed at reducing the risk of AML attacks. Rather than stopping at the identification of the problem, this work delves into actionable strategies and methodologies designed to effectively combat AML threats. By addressing this gap in the existing literature, this research not only contributes to a deeper understanding of the complexities surrounding AML attacks but also offers tangible remedies for mitigating their impact. This research proposes practical measures that can be implemented by organizations to bolster their defenses against AML activities.

C. AML ATTACK METHODS

AML methods are used to craft adversarial examples. Adversarial examples are inputs to an ML model that are intentionally designed to cause the model to make a mistake. In this section, the researchers describe 3 commonly known AML attack methods.

1) FAST GRADIENT SIGN METHOD

Researchers Ian J. Goodfellow, Jonathan Shlens, and Christian Szegedy at Google suggested one of the first methods for building adversarial attacks. The attack strategy is known as the Fast Gradient Sign Method (FGSM) approach. It involved pushing a model to misclassify an image by adding a linear amount of undetected noise to it. The sign of the gradient associated with the image that want to be modified is multiplied by a tiny constant epsilon to produce this noise. The model is increasingly vulnerable to deception as epsilon rises, but it also gets simpler to identify the perturbations. The gradient is calculated in terms of the input image since the objective is to create an image that minimizes the loss for the original image [27]. Since the objective of traditional gradient descent for model training is to minimize the model's loss on an original dataset, the gradient is used to update the model's weights. Assuming that neural networks cannot withstand even linear changes to the input, the FGSM was presented as a quick method to create adversarial samples to alter the model. FGSM is a simple and computationally efficient method for generating adversarial examples. FGSM is effective but can be defended with techniques like adversarial training.

2) PROJECTED GRADIENT DESCENT METHOD

The Projected Gradient Descent (PGD) attack is a white-box attack in which the attacker get access to the model's gradients and a copy of the model's weights. Compared to black box attacks, this threat model gives the attacker far more power because the attacker can tailor their attack to mislead the model instead of relying solely on transfer attacks, which usually results in disturbances that are observable to humans. PGD eliminates all restrictions on the amount of time and effort an attacker may devote to creating the most effective attack, making it the most "full" white-box proponent. PGD is a more advanced method that iteratively applies FGSM with a small step size and then clips the perturbation to ensure that it stays within a specified epsilon range around the original input. This iterative process continues for a certain number of steps or until convergence. PGD is more powerful than FGSM and can find adversarial examples that are harder to detect [3], [27]. To comprehend a PGD attack, one must find the perturbation that maximizes a model's loss on a given input while keeping the perturbation's magnitude below a predetermined threshold. When applied correctly, it ensures that the adversarial example's content matches that of the unperturbed sample, or even makes it nearly impossible for humans to tell the adversarial example from the unperturbed sample [27].

3) CARLINI-WAGNER

In order to evaluate current adversarial attacks and defenses, Nicholas Carlini and David Wagner of the University of California, Berkeley proposed a quicker and more reliable method for generating adversarial samples [28]. The Carlini-Wagner (CW) attack is a state-of-the-art method for generating adversarial examples. It formulates the generation of adversarial examples as an optimization problem, where the goal is to find the smallest perturbation that causes the

model to misclassify the input. CW attacks are typically more effective than FGSM and PGD but are also more computationally expensive. CW's attack starts with an attempt to resolve a challenging non-linear optimization equation. The gradient descent, when used to solve this equation yields stronger adversarial examples than the FGSM approach. It also avoids defensive distillation, which was once believed to be helpful in dealing with adversarial samples. Among the most potent white-box attacks are the CW attacks.

To summarize, FGSM is simple but can be easily defended, while attacks that use PGD and CW methods are more powerful but require more computational resources by the adversary. Each method has its own approach and properties. Adversarial examples generated using these methods have raised concerns about the robustness and security of ML models in general. These types of methods are used in AML attacks to generate adversarial examples, which goes as an input to unsuspecting ML models and causes the model to make a mistake. There are several other attack methods, however these 3 types of attack methods are more prominently noticed.

D. ATTACK STRATEGIES

AML attacks target ML models of various kinds. Many of them work well with both conventional ML models like linear regression and support vector machines (SVMs) as well as deep learning systems. Some high-level examples of attack strategies are provided in the following subsections.

1) BLACK-BOX ATTACKS

Black box attacks assume that the adversary is unaware of the model's parameters or structure and can only derive outputs for predetermined inputs. In this case, the adversarial example is constructed with either a custom-made model or no model at all with the exception of the inability to query the original model. In any event, the goal of these attacks is to produce adversarial instances that can be used with the black box model in question [29].

2) WHITE-BOX ATTACKS

White box attacks presume that the adversary can obtain the input's labels and has access to the model's parameters [29]. These are a category of attacks aimed at compromising the performance or integrity of ML models by exploiting detailed knowledge about their internal workings, including architecture, parameters and training data. In contrast to black-box attacks, where attackers have limited knowledge about the model, white-box attacks assume full access to the model's structure and parameters. These attacks leverage the transparency of ML models, which is often essential for model interpretability and debugging but can also pose a security risk when exposed to adversaries. By exploiting this transparency, attackers can craft adversarial examples. Here, inputs that are intentionally perturbed to cause the model to make incorrect predictions or classifications.

3) TRAINING-TIME ATTACKS

ML involves two stages, a training stage, in which the model learns and a deployment stage in which the model gets deployed on new unlabeled data and is allowed to make predictions. Poisoning attacks are those that occur during the model's training phase as an adversary manipulates a portion of the training data [30] by adding or amending training samples. The adversary is in charge of the model and its parameters in a training-time attack [31]. While model poisoning attacks are the most common form of training-time attacks [32], where clients send local model updates to the aggregating server in a federated model set-up, data poisoning attacks can also be considered as training time attacks when applied during the training phase.

4) DEPLOYMENT-TIME ATTACKS

It is possible to launch two distinct attacks during testing and deployment phases. The first ones are evasion attacks which alter testing samples to produce adversarial examples [33], [34], which based on certain metrics, resemble the original sample but change the model's predictions to reflect the attacker's selections post deployment. The second ones are privacy attacks in which attackers have query access to a model and are able to execute different kinds of privacy-based inference attacks such as membership inference [35] and data reconstruction [36]. Therefore, deployment attacks can be further segregated into attacks against models i.e., evasion attacks and data privacy attacks.

5) TARGETED ATTACKS

Threat actors that seek and penetrate a target entity's infrastructure while remaining anonymous are known as targeted attackers. These attackers possess the knowledge and means required to carry out their plans over an extended length of time. They may adjust, improve, or change their attacks to counter the defenses put up by the organization [37]. Traditional online dangers including malware, malicious websites, emails, and vulnerabilities are frequently used in targeted attacks. Several characteristics set targeted attacks apart from other types of threats. Targeted attacks are commonly launched through campaigns. Advanced persistent threats which are a form of targeted attacks are typically conducted as part of campaigns and are a sequence of successful and unsuccessful attempts over time to gain access to the target's network [37]. They frequently target particular industries, such as corporations, governmental organizations, or political parties. Attackers frequently have long-term objectives in mind, including monetary gain, political benefit, and commercial data theft to name a few [37]. Attackers constantly modify, enhance, and adapt their methods in order to get beyond security controls depending on the characteristics of the industry they are targeting.

6) UNTARGETED ATTACKS

Attackers that don't care about the security of their targets try to compromise as many people, organizations, services, or devices as they can using untargeted attacks. Since there will be many vulnerable systems or services, they don't care who the victim is [37]. They achieve this by employing strategies that capitalize on the open nature of the Internet, such as:

- Phishing which is a malicious practice of sending bulk emails to individuals requesting sensitive information such as bank account data or pointing them in the direction of a malicious website [37].
- Water holing which is a malicious practice of fabricating a fake website or breaching an authentic one to exploit users [37].
- Ransomware, a type of extortion malware through which files are encrypted and held captive until a ransom is paid [3].
- Scanning networks to target random people, organizations, services or devices.

Through these strategies, attackers cause harm to people, organizations, services or devices, even though they do not have any specific target at the initial stages of the attack.

III. AML ATTACKS TAXONOMY

Different types of AML attacks can be executed, some over a portion of the training data, while some on the labels, the model parameters, or the code of ML algorithms itself. In case, the model has already been trained, the adversary may launch attacks during the deployment stage to compromise the integrity and alter the model's predictions or conduct attacks to deduce private information about the model or its training data. According to the NIST AML publication [38] which is considered to be an authoritarian source on information of AML attacks and has also been reviewed by industry experts, it states that there is a specific taxonomy on AML attacks as described below:

There are 3 main types of AML attacks - Evasion, Poisoning and Privacy-based attacks. Within Evasion Attacks, there are different types such as White-Box Evasion Attacks, Black-Box Evasion Attacks and Transferability of Attacks. Within Poisoning Attacks, there are several types such as Availability Poisoning Attacks, Targeted Poisoning Attacks, Backdoor Poisoning Attacks and Model Poisoning Attacks. Each of these attacks have been specifically described and countermeasures are available for all. These are quite different from each other in terms of their techniques. Within Privacy Attacks, there are several types such as Data Reconstruction, Memorization, Membership Inference, Model Extraction and Property Inference Attacks and these as well are different from each other. These attacks have been depicted in Figure 4 and classified in these 3 separate sections and extensively described based on attacker tactics and techniques with specific mitigating countermeasures for each. In further sections, the researchers provide details on each type of attack, adversary tactics and techniques and countermeasures proposed in literature to mitigate these.

A. EVASION ATTACKS

Over the past ten years, there has been a notable surge in interest in AML due to the revelation of evasion attacks against ML models. The objective of an adversary in an evasion attack is to produce adversarial examples, which are testing samples whose classification can be altered at deployment time to any attacker-selected class with the least amount of disruption [34]. The earliest known examples of evasion attacks were reported in 1988 by Kearns and Li [39], Szedegy et al. [34], then in 2004 by Dalvi et al. [40], Lowd and Meek [41], and others. Some of these studies demonstrated the visualization of adversarial samples and the ease of manipulation of deep neural networks utilized for image categorization using evasion attacks. The first application domains where evasion attacks were demonstrated were in anti-malware, spam filtering and image classification ML models. But as interest in adversarial machine learning grew, and ML technologies were used in a wide range of other application areas, such as speech recognition [42], natural language processing [43], and video classification [44], [45], evasion attacks on these areas came under closer examination. An evasion attack occurs when a deliberately perturbed input, known as a ''adversarial example,'' is fed to the network. This perturbed input appears and feels exactly like its unaltered copy to a human, but it completely confuses the ML classifier.

Evasion attack as shown in Figure 5 occur during the ML deployment stage in which the model has already been trained, and the adversary's aim is to breach the data's integrity and alter the ML model's predictions. Evasion attacks require altering testing samples to provide adversarial examples which are undetectable to humans and are incorrectly assigned by the model to a different class to alter the model's predictions. Evasion attacks are the most common type of AML attacks. For example, spammers and hackers frequently try to avoid detection by hiding the presence of malicious software and spam emails, in order to avoid detection and be accepted as genuine samples. Spoofing attacks on biometric verification systems is a form of evasion attacks. Image-based spam inserts its content into an attached image to avoid textual scrutiny by anti-spam filters, is another example of an evasion attack. Black box attacks and white box attacks are the two types of evasion attacks.

1) ATTACKER TACTICS AND TECHNIQUES IN EVASION ATTACKS

The adversary's objective in an evasion attack is to produce adversarial examples whose classification can be altered at deployment time to any attacker-selected class with the least amount of perturbation [34].

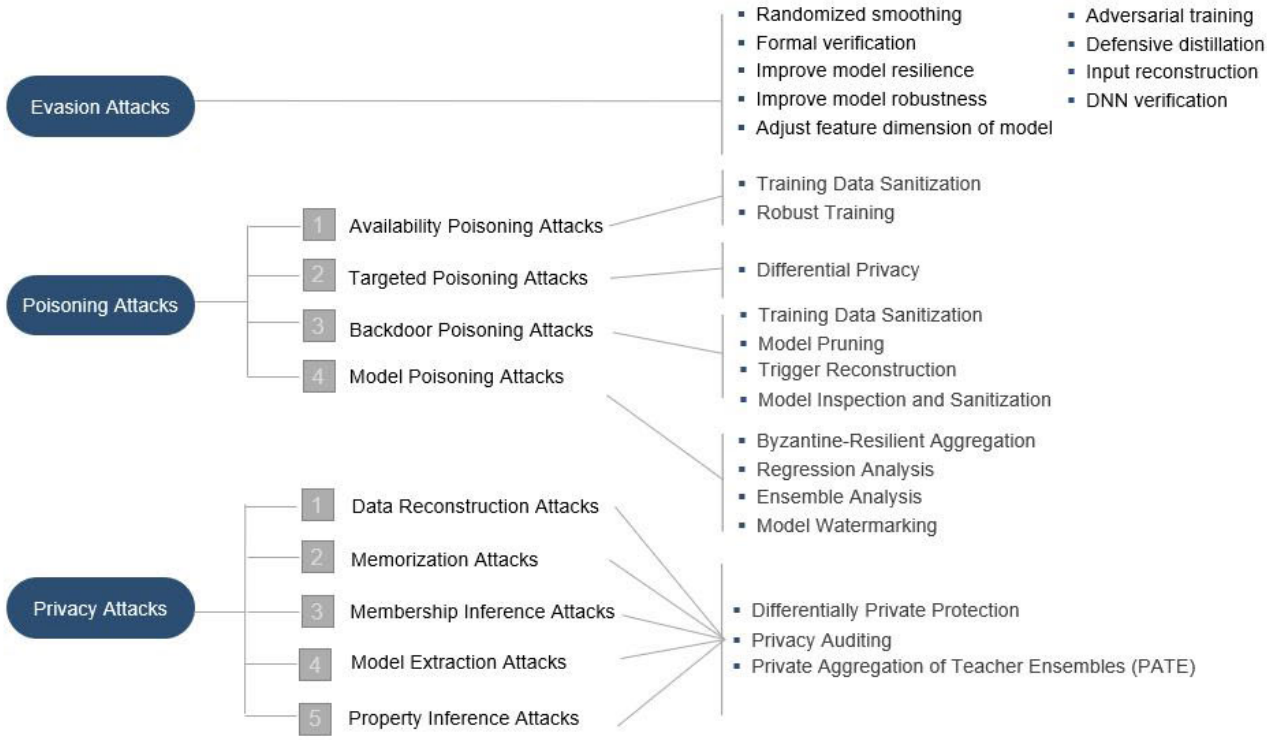

FIGURE 4. Types of AML attacks.

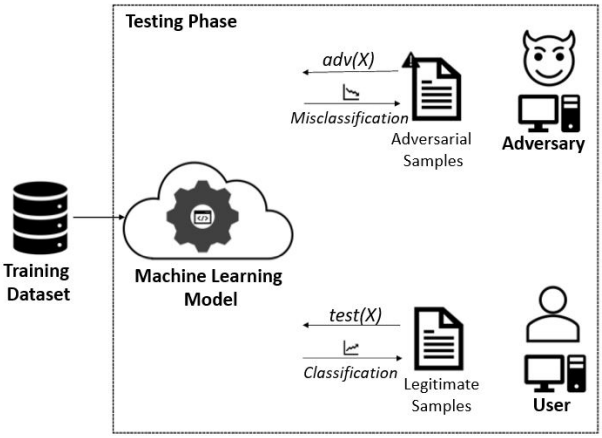

FIGURE 5. Evasion attack process flow.

In February 2020, the Microsoft Azure red team performed an evasion attack on a new Microsoft product designed for running AI workloads at the edge [46]. The goal of this attack was to continuously modify a target image using an automated system in order to generate misclassifications in the model. The attackers initiated the attack by searching for the victim's publicly available ML model research materials to gather information about the target model. Then, they acquired the public ML artifacts of the base model. They sent queries and analyzed the responses from the ML model. Finally, they created an automated system to continuously manipulate an original target image that tricked the model into producing incorrect inferences, however the perturbations in the image were unnoticeable to the human eye. This was then fed into the model and the red team was able to evade the model by causing misclassifications.

In another similar attack that was conducted by MITRE's AI ethical hacking team, the goal of was to execute a physical-domain evasion attack on a commercial face recognition service and deliberately cause a misclassification [47]. This required the team to search for the victim's publicly available research materials to gather information about the target ML model. They followed this by validating accounts to the said service that could help them gain access to the knowledge base documents of the face recognition API. The team then accessed the API of the target model and discovered the model's ontology by querying the model's API. Later, they acquired the public open source data that was used by the model and developed a proxy ML model that could optimize adversarial visual patterns as a physical domain patch-based attack using expectation over transformation method. The team caused issues in the face identification system by placing physical patches. The team successfully evaded the model using the physical patches and caused targeted misclassifications. Through these evasion attacks and several others found in various studies, the chronology of the tactics and techniques used by adversaries are similar and can be traced. In the next section, the researchers provide

several countermeasures that have been proposed in literature for detecting and preventing evasion attacks.

2) EVASION ATTACK COUNTERMEASURES

Adversarial inputs are common in a wide range of ML model architectures and application domains, making mitigation of evasion attacks challenging. The existence of adversarial inputs may be explained by the fact that ML models, especially in the field of computer vision rely on non-robust features that are beyond human perception [48]. Many of the countermeasures against adversarial inputs that have been suggested in recent years have proven ineffective and have been superseded by more powerful attack methods [49], [50], [51]. However, out of all the suggested countermeasures against adversarial evasion attacks, certain countermeasures have proven themselves to be effective and can mitigate evasion attacks. These are summarized in Table 1.

Randomized smoothing: Randomized smoothing is a technique used in ML to improve the robustness of models against adversarial attacks. It involves adding random noise to input data during both training and inference stages, which helps to regularize the model and make it less sensitive to small perturbations in the input. The key idea behind randomized smoothing is to utilize the principle of stochasticity to provide a form of defense against adversarial attacks. The model learns to generalize better and becomes less susceptible to adversarial examples by introducing random noise during training. Similarly, during inference, adding random noise to input samples can help to smooth out the decision boundaries of the model, making it more difficult for adversaries to craft effective adversarial perturbations. The effectiveness of randomized smoothing relies on the assumption that the added noise disrupts the gradient information used by adversaries to generate adversarial examples. This makes it harder for attackers to find perturbations that consistently fool the model. Randomized smoothing has shown to be effective in improving the robustness of various types of models, including deep neural networks, against evasion attacks. It is particularly useful in scenarios where model security and reliability are critical, such as in autonomous driving systems, medical diagnosis, and financial fraud detection. Randomized smoothing was first introduced by Lecuyer et al. [52] and refined by Cohen et al. [53] and is a technique that turns any classifier into a verifiable robust smooth classifier by generating the most likely predictions in the presence of gaussian noise perturbations. Provable resilience against evasion attacks is achieved by this method, even for classifiers trained on massively distributed datasets like ImageNet. A subset of testing samples are usually given certified predictions by randomized smoothing. Overall, randomized smoothing is a promising approach for enhancing the resilience of ML models against adversarial threats, contributing to the development of trustworthy AI systems.

Formal verification: Formal verification methods ensure the correctness and reliability of ML systems. It involves rigorous mathematical analysis and verification techniques to prove or disprove properties of ML models, algorithms, or systems. In traditional software engineering, formal verification is used to prove the correctness of software systems with respect to a specification. Similarly, in the context of ML, formal verification aims to provide guarantees about the behavior of ML models, such as their robustness, fairness, safety, and reliability. There are several approaches to formal verification in ML. The first one is mathematical proofs. This approach involves using mathematical techniques, such as theorem proving, to formally verify properties of ML models. For example, one might prove that a certain neural network architecture converge to the optimal solution under specific conditions. Another approach is model checking. Model checking involves exhaustively exploring the behavior of a model against a formal specification or a set of properties. In the context of ML, this could involve checking whether a model satisfies certain safety or fairness constraints under all possible inputs. Furthermore, there are constraint solving techniques that can be used to verify properties of ML models by formulating them as logical constraints and then solving them to check for validity. This approach is often used to verify properties related to the robustness of models against adversarial attacks. Finally, there are symbolic execution techniques that involve analyzing the behavior of a program or a model by representing its inputs symbolically. This technique can be used to systematically explore the behavior of ML models and verify properties such as correctness and safety. Providing formal specifications of desired properties or behaviors of ML systems can facilitate formal verification. These specifications can then be used as the basis for mathematical proofs or automated verification techniques. Formal verification systems check the robustness of small feed-forward networks using satisfiability modulo theories (SMT) solvers [54]. Several follow-up formal verification systems, including DeepPoly [55], ReluVal [56], and Fast Geometric Projections (FGP) [57] have expanded and scaled up these techniques to larger networks. While formal verification techniques hold great promise for verifying the robustness of neural networks, their primary drawbacks are their limited scalability and high computational cost. Formal verification in ML is still an active area of research, and it poses several challenges due to the complexity and non-linearity of many models. However, for ensuring the safety, reliability, and trustworthiness of ML systems, particularly in safety-critical applications such as autonomous vehicles, medical diagnosis, and financial systems, formal verification can be further delved upon.

Adversarial training: Adversarial training is a general technique that was first presented by Goodfellow et al. [33] and refined by Madry et al. [60]. It is a technique used in ML to improve the robustness and resilience of models against adversarial attacks. Many adversarial attacks involve making small, carefully crafted perturbations to input data with the intention of misleading the model into making incorrect predictions. Adversarial training aims

TABLE 1. Review of evasion attack studies.

Ref No.	Dataset	Type of ML Architecture / Method	Contribution	Proposed Countermeasure
[53]	ImageNet	Neural Network based classifier	Demonstrated how to create a new classifier that is verifiably robust to adversarial perturbations under the norm from any classifier that performs well under Gaussian noise using Randomized smoothing.	Randomized Smoothing
[52]	ImageNet	Autoencoder	Illustrated how differential privacy theory and adversarial training against adversarial examples are related and demonstrated how the connection can be used to create a certified defense against such attacks.	Adversarial Training; Differential Privacy
[54]	Not mentioned	Tranformers: Rectified Linear Unit (ReLU)	Presented a new, scalable, and effective method for formally verifying the properties of DNNs.	Formal Verification
[55]	MNIST; CIFAR10	Tranformers	Introduced a new system named DeepPoly for accurate certification of scalable DNNs and increase robustness of the network against evasion attacks.	Formal Verification
[56]	MNIST	Tranformers: Rectified Linear Unit (ReLU)	Proposed a novel approach to formally verify DNN security properties without the need for SMT solvers.	Formal Verification
[57]	MNIST, Fashion-MNIST, and CIFAR10	CNNs, Tranformers: Rectified Linear Unit (ReLU)	Provided a quick method for evaluating feed-forward neural networks with piecewise-linear activation functions for local robustness.	Adjusting Feature Dimension of Model
[58]	Penn Treebank (PTB) datase	Deep Neural Networks (DNNs)	Recommended a novel input reconstruction method for quantitatively evaluating the likelihood that uncommon or special training data.	Input Reconstruction
[59]	GPT-2 training datasets	LLMs: GPT-2	Showcased how an adversary in a LLM can conduct an evasion attack to retreive specific training examples. Derived specific adversarial training safeguards for LLMs.	Adversarial Training
[49]	MNIST; CIFAR10	Deep Neural Networks (DNNs): ResNet	Provided a few straightforward feature squeezing techniques.	Adjusting Feature Dimension of Model
[33]	MNIST	Deep Neural Networks (DNNs) & Transformers: LSTM & ReLU	Employed a novel method to offer instances for adversarial training and decreased the network's test set error.	Adversarial Training
[60]	MNIST; CIFAR10	Deep Neural Networks (DNNs)	Investigated the logical structure of neural networks to find dependable universal techniques for improving resistence to AML attacks using randomized smoothing.	Randomized Smoothing
[61]	MNIST; CIFAR10	Deep Neural Networks (DNNs)	Suggested an attack method for more neural networks that involves training a reconstructor network after receiving the weights of the targeted model as input.	Input Reconstruction

to mitigate the impact of such attacks by incorporating adversarial perturbed examples during the model training process. Under adversarial training, there are additional defense mechanisms like gradient masking, which prevents the adversary from obtaining important gradients to directly construct the adversarial examples. These defenses however don't work against black-box evasion attacks as the opponent is still able to acquire the gradient direction derived from the nearby replacement model using the adversarial examples' transferability [62].

It's interesting to note that models trained using adversarial training have greater semantic meaning than standard models [63], but this advantage is typically attained at the expense of a lower model accuracy on clean data. The iterative generation of adversarial examples during training is another reason why adversarial training is costly. Apruzzese et al., [18] addressed the problem of evasion attacks against flow-based botnet detectors by putting forth the first defensive strategy based on deep reinforcement learning to lessen adversarial perturbations against ML-based network intrusion detection systems. The authors proposed a framework that can independently produce evasive samples to evade a target botnet detector. The framework then used these samples to harden the detector by training on adversarial samples. The author's study paved the way for future research aiming to combat evasion attacks by developing robust detectors that maintain their performance in the face of adversarial perturbations. It helped in increasing the detection rate against known and novel evasion attacks. In order to create a new model that is resistant to attack perturbations, this technique generated adversarial samples during the model training stage using well-known attack methods. It then added these adversarial samples to the training set and retrained the model many more times. This technique increased the training set data by combining different adversarial samples, which improved the new model's robustness, accuracy, and standardization.

The key steps involved in adversarial training are as follows. First, adversarial examples are crafted by applying imperceptible perturbations to input data in a way that

maximizes the model's loss function. This is typically achieved by computing gradients of the loss function with respect to the input and then making small adjustments to the input in the direction that increases the loss. Then, the adversarial examples generated are added to the training dataset alongside the original clean examples. This augmentation ensures that the model learns to recognize and correctly classify adversarial inputs as well. The model is then trained on the augmented dataset, which now includes both clean and adversarial samples. By exposing the model to adversarial samples during training, it learns to become more robust and resilient to such attacks. After training, the model's performance is evaluated on a separate validation or test dataset to assess its robustness against AML attacks. This step helps to ensure that the adversarial training process has effectively improved the model's resilience without sacrificing performance on clean data. It provides a proactive defense mechanism by training models to recognize and appropriately respond to actual adversarial inputs, rather than simply reacting to attacks after deployment. However, adversarial training also has its limitations. It can be computationally expensive and may require additional labeled data for generating adversarial samples. Furthermore, adversaries may adapt their attack strategies to circumvent adversarially trained models, necessitating ongoing research and development of more sophisticated defense mechanisms.

Adjusting feature dimension of model: Adjusting the feature dimension of a model in ML refers to modifying the number of input features that the model processes. This adjustment can be necessary for various reasons, such as accommodating different input data shapes, reducing model complexity, or adapting to specific requirements of the task or architecture. Adjusting the feature dimension of a model is also sometimes referred to as Feature Squeezing. Below are a few scenarios where adjusting feature dimensions/feature squeezing might be relevant:

1) Dimensionality Reduction: In cases where the original feature space is high-dimensional or contains redundant information, dimensionality reduction techniques like Principal Component Analysis (PCA) or feature selection methods can be employed to reduce the number of features while retaining as much useful information as possible.
2) Reshaping Input Data: Some ML models, particularly neural networks, require input data to be in a specific shape or format. For example, convolutional neural networks (CNNs) commonly operate on image data with dimensions (height, width, channels). Adjusting feature dimensions in this context may involve reshaping input data to conform to the required input shape of the model.
3) Model Adaptation: When adapting pre-trained models to new tasks or datasets, it may be necessary to adjust the input feature dimensions to match the requirements of the pre-trained model. This adjustment could involve adding, removing, or transforming features to align with the model's architecture.
4) Handling Variable-Length Sequences: In natural language processing (NLP) tasks, input sequences such as sentences or documents may have variable lengths. Techniques like padding or truncation can be used to adjust the feature dimensions to ensure that all input sequences have uniform lengths, which is often required for processing with certain types of models like recurrent neural networks (RNNs) or transformers.
5) Feature Engineering: Feature engineering involves creating new features or transforming existing ones to improve model performance. Adjusting feature dimensions may involve adding new features, combining existing ones, or removing irrelevant or redundant features based on domain knowledge and experimentation.

Model feature dimensions are related to adversarial examples. To stop an evasion attack, one may include feature selection and reduction techniques [58], [59] to identify which dimensions are more prone to perturbation. Then, one may decrease the corresponding feature weights or eliminate the most vulnerable features. This type of technique can to some extent, restrict the generation of adversarial examples because it makes it harder for adversaries to add features with fewer available features and it would require more perturbations. Even though 5 random features can give adversarial samples a 50 percent success rate, the work in [49] demonstrated that limiting the number of features that the adversary can control does not totally prevent evasion attacks. Furthermore, for tasks requiring exceptionally high classification accuracy, reducing some vulnerable feature dimensions will result in an overall reduction in accuracy, which would be inappropriate. In summary, adjusting feature dimensions in ML involves modifying the input feature space to meet the requirements of the model, the task, or the data. It plays a crucial role in pre-processing, model adaptation, and feature engineering, all of which are essential steps in building effective ML systems.

Defensive distillation: Huang et al., [64] recommended using defensive distillation which is a method to prevent models from fitting too tightly and improves generalization. Defensive distillation enhance the robustness of models against adversarial attacks, particularly in the context of deep neural networks. The main idea behind defensive distillation is to train a secondary ''distilled'' model using the outputs of a primary model which has been trained on the original dataset. The distilled model is trained on the same dataset, but with higher temperature softmax output activations, making the model more confident and less sensitive to small perturbations in the input. The steps involved in defensive distillation are as follows:

1) Training the Primary Model: Initially, a primary model (often a deep neural network) is trained on the original dataset to perform the target task such as image classification. This model is typically trained

using standard techniques but it may be vulnerable to adversarial attacks.

2) Generating Soft Labels: Using the trained primary model, soft labels are generated for the training data. Soft labels provide a more nuanced representation of the model's predictions compared to hard labels (one-hot encoded vectors), allowing for richer information transfer during distillation.
3) Training the Distilled Model: A secondary distilled model is trained using the same dataset but with the soft labels generated by the primary model. Additionally, the temperature parameter in the softmax activation function is increased during training, resulting in smoother and more spread-out probability distributions over the output classes.
4) Evaluation: The distilled model is evaluated on a separate validation or test dataset to assess its performance and robustness against adversarial attacks.

Defensive distillation has shown to be effective in improving the robustness of models against various types of adversarial attacks, including those based on gradient information. By training the distilled model to be more confident in its predictions and smoothing out the decision boundaries, defensive distillation makes it more difficult for adversaries to generate effective adversarial perturbations. In order for the classification result produced by one DNN to be used for training the subsequent DNN, these technologies operate by concatenating multiple DNNs during the model training stage. Researchers discovered that knowledge transfer can increase an AI model's robustness and decrease its sensitivity to minute perturbations [61]. As a result, they suggested defending against evasion attacks by utilizing defensive distillation technologies. They discovered that distillation technologies can lower the success rate of particular attacks like Jacobian-based Saliency Map Attacks using tests on MNIST and CIFAR-10 data sets. However, it's importsnt to note that defensive distillation has its limitations, and it may not provide complete protection against all types of adversarial attacks. Furthermore, recent research has shown that defensive distillation may not be as effective as initially thought, and other defense mechanisms may be needed to enhance the security of ML models further.

Input reconstruction: Input reconstruction in ML refers to the process of generating a reconstructed version of the input data from a model's internal representations or latent space. This technique is commonly used in various domains, including unsupervised learning, generative modeling, and feature visualization. The goal of input reconstruction is to learn a mapping from the latent space back to the original input space, allowing the model to reconstruct or generate plausible samples resembling the input data. By learning this mapping, the model gains an understanding of the underlying structure and features of the data, which can be useful for tasks such as data denoising, anomaly detection, and generating new data samples. Input reconstruction is a versatile technique that facilitates understanding, manipulation, and generation of data samples by learning meaningful representations in the latent space. To protect against evasion attacks, this technique deforms input samples during the model inference stage. The normal classification function of the models remains unaffected by the deformed input. Adding noise, de-noising, or altering an input sample using an automatic encoder (autoencoder) are some methods for implementing input reconstruction. Input reconstruction is often employed in autoencoder architectures, which consists of an encoder network that maps input data to a lower-dimensional latent space and a decoder network that reconstructs the input data from the latent representation. During training, the model learns to minimize the reconstruction error, typically measured using a loss function such as mean squared error (MSE) or binary cross-entropy. Once trained, the decoder network can be used to generate reconstructions of input data by feeding sampled points from the latent space into the decoder. These reconstructions aim to capture the salient features of the original data while filtering out noise or irrelevant information. Input reconstruction has several applications across different domains:

- Data Denoising: Input reconstruction can be used to remove noise or artifacts from data samples by reconstructing clean versions of the input data from corrupted or noisy observations.
- Anomaly Detection: Deviations between the original input data and its reconstructed counterpart can be indicative of anomalies or outliers in the data. Input reconstruction can thus be leveraged for anomaly detection tasks.
- Generative Modeling: By learning the mapping from the latent space to the input space, input reconstruction enables the generation of new data samples that resemble the original input distribution. This is particularly common in generative adversarial networks (GANs) and variational autoencoders (VAEs).
- Feature Visualization: Input reconstruction provide insight into the features learned by the model by visualizing the reconstructed data samples. This can help interpret the model's behavior and understand which features are considered important for reconstruction.

There are intrinsic trade-offs between robustness and accuracy in all of these mitigations, and they all incur extra computational expenses during training. As a result, creating ML models that are accurate, cost optimal and resistant to evasion attacks is still a challenge. Several countermeasures as suggested by Ji et al., [65] such as input reconstruction are reactive countermeasures and are also applicable to other types of AML attacks such as poisoning attacks and privacy attacks. These will be discussed further in the results and discussion section.

B. POISONING ATTACKS

Poisoning is the process of intentionally corrupting training data with malicious information. It's a type of an attack where

an attacker manipulates the training data to compromise the performance or integrity of the trained model as shown in Figure 6. The attacker injects malicious samples into the training dataset influencing the behavior of the model during training time. Poisoning attacks occur in various ML settings, including supervised learning, where the model is trained on labeled data, and unsupervised learning, where the model learns patterns from unlabeled data. Data gathered during operations may be used to retrain ML systems. For example, intrusion detection systems (IDSs) frequently undergo retraining utilizing real-time operational data. By inserting malicious samples into the system while it is operating, an attacker can pollute this data and prevent retraining from functioning normally [66]. Poisoning attacks during the training stage of the ML algorithm, is another significant threat to ML systems. The first known poisoning attack was created for worm signature generation in 2006 [67], therefore poisoning attacks have a long history in cybersecurity. Later on, a number of application domains have seen a significant amount of research on poisoning attacks, including computer security (for spam detection [68], network intrusion detection [69], vulnerability prediction [70], malware classification [71], [72], computer vision [73], [74], [75], natural language processing [76], [77], [78] and tabular data in the healthcare and financial domains [77]. Poisoning attacks have also received increased attention in industrial settings lately. They are regarded as the most serious vulnerability in ML systems and are extremely potent, according to a Microsoft report [19]. Specifically, targeted and backdoor poisoning attacks are known to be more covert and may result in integrity violations on a limited number of target samples, whereas availability poisoning attacks cause the ML model to degrade indiscriminately on all samples [30]. There are several subcategories of poisoning attacks due to the extensive range of adversarial capabilities that are used in poisoning attacks, including data poisoning, model poisoning, label control, source code control, and test data control. The below sections describe the different types of poisoning attacks and their countermeasures. These are summarized in Table 2.

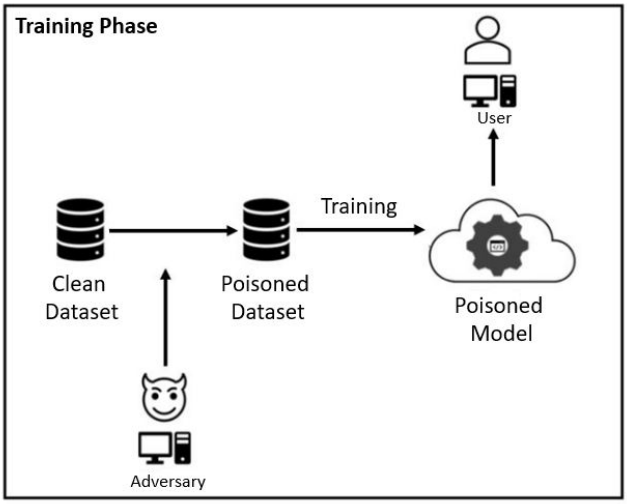

FIGURE 6. Poisoning attack process flow.

1) AVAILABILITY POISONING ATTACKS

Availability attacks against spam classifiers and worm signature generation were the first poisoning attacks found in cybersecurity applications. These attacks affect the entire model and essentially cause a denial-of-service attack on users of the AI system. Label flipping is a straightforward black-box poisoning attack technique that creates training examples with a victim label that the adversary chooses [104]. In order to mount an availability attack, this method needs a significant proportion of poisoning samples. It has been developed through the introduction of optimization-based poisoning attacks against Support Vector Machines (SVMs). Later, these optimization-based poisoning attacks were developed against neural networks [78] and linear regression [77] and they need white-box access to the model and training data. The most widely used technique for creating availability poisoning attacks in gray-box adversarial settings is transferabilities wherein poisoning samples are created for a replicated model and then transferred to the target model. Clean-label poisoning attacks in which adversaries can only control the training examples but not their labels represent a plausible threat model for supervised learning. This case study simulates situations where the labeling procedure is carried out independently of the training algorithm. For example, in malware classification, attackers may submit binary files to threat intelligence platforms, in which case the labeling process is carried out using external techniques such as anti-virus signatures. For neural network classifiers, clean-label availability attacks have been introduced by training a generative model and introducing noise to training samples in order to maximize the adversarial objective. Using gradient alignment and making only minor changes to the training set is an alternative strategy for clean-label poisoning [105]. An adversary may launch a model poisoning attack in federated learning to cause availability violations in the globally trained model.

2) TARGETED POISONING ATTACKS

Targeted poisoning attacks, as opposed to availability attacks alter the ML model's prediction on a limited set of selected samples. Label flipping is a successful targeted poisoning attack technique if the adversary has control over the training data's labeling function. By simply inserting multiple poisoned samples with the target label, the adversary can trick the model into learning the incorrect label because the attacker cannot access the labeling function in a clean-label setting, targeted poisoning attacks are primarily studied in this context. Numerous methods have been put forth to execute targeted attacks with clean labels.

TABLE 2. Review of poisoning attack studies.

Ref No.	Dataset	Attack Type	Contribution	Proposed Countermeasure
[79]	MNIST-1-7, Dogfish, IMDB	Availability	Created approximative upper bounds that execute empirical risk minimization after performing outlier removal for a wide range of poisoning attacks.	Adversarial training and Randomized Smoothing
[80]	Drug discovery, ENRON, Spambase	Availability	Presented a novel algorithm that can accept large ML models and strengthen the learner's resistance to outliers.	Training Data Sanitization
[77]	Multiple datasets	Availability	Conducted the first comprehensive analysis of linear regression models' countermeasures for poisoning attacks.	Regression Analysis
[81]	MNIST-1-7, Dogfish, IMDB	Availability	Offered a unified perspective on randomized smoothing over arbitrary functions.	Adversarial training and Randomized Smoothing
[82]	Vertebral Column dataset from UCI	Targeted	Proposed differential privacy as a preventative measure against poisoning attacks.	Differential Privacy
[83]	ImageNet	Targeted	Suggested reweighting losses according to class in order to produce robust and more equitable classifiers.	Model Pruning
[84]	MNIST; CIFAR10	Backdoor	Suggested model inspection and sanitization against a general poisoning threat model, defined as the insertion or deletion of samples to the training set.	Model Inspection and Sanitization
[85]	CIFAR10	Backdoor	Suggested a novel defense algorithm that amplifies the spectral signature of corrupted data by employing robust covariance estimation.	Training Data Sanitization
[86]	CIFAR10	Backdoor	Suggested a novel approach that can identify over 99% of tainted examples and eliminate them without affecting model performance.	Training Data Sanitization
[87]	CIFAR-10 and DPAcontest	Backdoor	Suggested that convolutional neural networks and multilayer perceptrons are the best machine learning architectures against most AML attacks.	Model Pruning
[88]	MNIST, GTSRB, YouTube Face, PubFig, VGG Face	Backdoor	Introduced the first reliable and broadly applicable DNN backdoor attack detection and mitigation system which reconstructs potential triggers and locates backdoors.	Model Inspection and Sanitization
[89]	MNIST; CIFAR10; TrojAI	Backdoor	Addressed the issue of Trojan detection specifically recognizing Trojaned models which were trained on tainted data.	Training Data Sanitization
[90]	CIFAR-10, GTSRB, ImageNet, VGG-Face, Age, USTS	Backdoor	Created a brand-new method for studying the behaviors of inner neurons by observing how their output actions alter in response to varying stimulation levels.	Model Pruning
[91]	MNIST, GTSRB	Backdoor	Presented NeuronInspect, a framework that uses output explanation techniques to find trojan backdoors in deep neural networks.	Model Inspection and Sanitization
[92]	MNIST, GTSRB, ResNet-18, Trojan Square, Trojan	Backdoor	To ensure safe model deployment, the authors presented DeepInspect, the first black-box Trojan detection system that requires little model knowledge beforehand.	Model Inspection and Sanitization
[93]	Multiple datasets	Backdoor	Presented a novel solution experimented on vision, speech, tabular data and natural language text datasets.	Training Data Sanitization
[94]	Not mentioned	Backdoor	Suggested a novel model repairing technique called Adversarial Neuron Pruning (ANP), which involved pruning certain sensitive neurons.	Model Pruning
[95]	CIFAR-10, GTSRB	Backdoor	Proposed a minimax formulation for eliminating backdoors from a given poisoned model.	Trigger Reconstruction
[96]	Multiple datasets	Model	Introduced FLTrust, a novel approach to federated learning where trust is bootstrapped by the service provider.	Model watermarking
[97]	MNIST; CIFAR10	Model	Introduced Garfield, a library to transparently create Byzantine-resilient machine learning (ML) applications.	Byzantine-resilient aggregation
[98]	MNIST; CIFAR10	Model	Investigated Byzantine collaborative learning, in which n nodes attempt to learn from each other's local data collectively.	Byzantine-resilient aggregation
[99]	MNIST-Fashion; CIFAR13	Model	Proposed a client-based defense that can mitigate model poisoning attacks that have already contaminated the global model.	Model watermarking
[100]	MNIST	Model	In order to achieve optimal statistical performance, the authors developed distributed learning algorithms that are provably robust against such failures.	Byzantine-resilient aggregation
[101]	CIFAR-10 and Reddit corpus	Model	Developed and evaluated a technique that incorporates defense into the attacker's loss function during training.	Regression analysis
[102]	Multiple datasets	Model	To counter local model poisoning attacks, the authors developed new defenses against local model poisoning attacks on federated learning.	Model watermarking
[103]	MNIST, CIFAR10, and Purchase	Model	A general framework for model poisoning attacks on federated learning was presented. The method defeated model poisoning attacks better than any Byzantine-robust algorithm currently in use.	Model watermarking

3) BACKDOOR POISONING ATTACKS

In 2017, Backdoor poisoning attacks, or BadNets, were first proposed by Gu et al. [30]. They demonstrated that adding a tiny patch trigger to a subset of images during training and altering their label to a target class can poison image classifiers. Any image, including the trigger or backdoor pattern, will be incorrectly classified to the target class during testing as the classifier learns to associate the trigger with that class. In parallel, backdoor attacks were presented by Chen et al., [106], in which the trigger is masked out of the training set. Subsequent research presented the idea of clean-label backdoor attacks, where the adversary is unable to alter the label of the tainted samples. For clean-label attacks to be successful, more poisoning samples are usually needed. Although the attack model of a clean-label attack is more realistic, it usually requires more poisoning samples to be effective. Backdoor attacks have grown more complex and covert over the past few years, making it more difficult to identify and counteract them. Even after the last few layers of the model were fine-tuned using clean data, latent backdoor attacks were intended to persist. In order for the model to learn the trigger in a location-invariant manner, the Backdoor Generating Network (BaN) [107] dynamically manipulates the trigger's location in the poisoned samples. There are embedded functional triggers in the image, or they can change based on input. Li et al [108], for example, concealed the trigger in the training set using steganography algorithms. A clean-label attack was first described by Liu et al. [109] and it makes use of an image's natural reflection as a backdoor trigger. Wenger et al.'s [110] use of tangible objects as triggers, like sunglasses and earrings compromised facial recognition systems. Although computer vision applications are the target of most backdoor poisoning attacks, this attack vector has also shown impact in other application domains like audio, natural language processing, and cybersecurity.

- Audio: Shi et al. [111] demonstrated how an adversary can insert a subtle audio trigger into real-time speech in the audio domains. During training, the adversary and the target model jointly optimize the trigger.
- NLP: Since text data in this context is discrete and maintaining a sentence's semantic meaning is required for the attack to remain undetectable, creating relevant poisoning samples is more difficult. However, recently it has been demonstrated that backdoor attacks in NLP domains are possible. For sentiment analysis and neural machine translation applications, for example, Chen et al. [74] introduced semantic-preserving backdoors at the character, word, and sentence levels. Li et al. [75] created hidden backdoors against transformer models.
- Cybersecurity: Severi et al. [71] demonstrated how clean-label poisoning attacks against malware classifiers can be generated using AI explainability techniques with small triggers. They used three malware datasets to attack multiple models, including neural networks, random forests, gradient boosting, and SVMs.

4) MODEL POISONING ATTACKS

Attacks known as "model poisoning" aim to introduce malicious features into the trained machine learning model by direct alteration. Liu et al., [31] used centralized learning to reverse engineer a trained neural network's trigger and then retrained the model by poisoning it with external data. In the context of federated learning, where clients submit local model updates to a server that aggregates them into a global model, the majority of model poisoning attacks have been developed. Malicious updates may be sent by compromised clients, contaminating the global model. In federated models, availability and integrity can both be compromised by model poisoning attacks. Model poisoning attacks can also occur in supply-chain scenarios when suppliers' models or parts of their models contain malicious code.

5) ATTACKER TACTICS AND TECHNIQUES FOR POISONING ATTACKS

The Twitter chatbot Tay briefly mentioned earlier was a type of a federated poisoning attack [17]. Tay, was a Twitter chatbot made by Microsoft with the goal of entertaining and involving users. Unlike other chatbots that responded to commands with pre-written scripts, Tay was able to learn from its interactions. Threat actors executed a poisoning attack which caused it to tweet abusive words and produce content that was offensive to its users. Within 24 hours of Tay's release, Microsoft shut it down and apologized to the public, sharing the lessons it had learned from the bot's failure. In this attack, the threat actors first interacted with Tay via Twitter messages and then coordinated with the intent of defacing it by exploiting its feedback loop. By repeated interactions with the bot using racist and offensive language, the threat actors were able to create bias in it's dataset. They used the "repeat after me" function, a command that forced Tay to repeat anything that was said to it. They were able to compromise Tay's conversation algorithms by generating offensive content. Tay began to repeat this when interacting with innocent users as a result of re-inforcement learning [17].

In another type of poisoning attack that was targeted against VirusTotal [112], a web application that enabled users to check for malware, viruses, worms, trojans, and other harmful content in files and URLs and uses ML, reported of a particular ransomware family that was increasingly observed by McAfee Advanced Threat Research analysts. A case investigation showed that, in a short period of time, numerous samples of that specific ransomware family were submitted via VirusTotal. Subsequent analysis showed that the samples were all equivalent in terms of string similarity and between 98 and 74 percent similar in terms of code similarity. It was interesting to note that every sample had the same compilation time. Further investigation led researchers

to the conclusion that the original file had been altered to produce mutant variants via the use of the metamorphic code manipulating tool ''metame.'' Metame, a basic metamorphic code engine for arbitrary executables, was acquired by the threat actor. To begin creating ''mutant'' variants, the actor started with a malware sample from a popular ransomware family. The performer posted ''mutant'' audio samples on the website. A number of vendors began categorizing the files and the majority of it were unusable as belonging to the ransomware family. The dataset used by the ML models to detect and categorize this family of ransomware was tainted by the ''mutant'' samples. In this attack, the threat actor compromised the ML supply chain model and poisoned the data. This was a form of a backdoor poisoning attack.

6) COUNTERMEASURES FOR AVAILABILITY POISONING ATTACKS

Since availability poisoning attacks result in a significant decline in the classifier metrics, they are typically identified by keeping an eye on the common performance metrics of ML models, including precision, recall, accuracy, F1 scores and area under the curve. However, it is less common to find these attacks during the ML testing or deployment phases. Therefore, existing mitigations try to proactively stop these attacks during the training phase in order to produce reliable ML models. Among the current mitigations, a few generally that work are as follows:

Training data sanitization: Training data sanitization in machine learning (ML) refers to the process of identifying and removing or mitigating the presence of erroneous, misleading, or malicious data in the training dataset before using it to train a model. The goal of training data sanitization is to ensure the quality, integrity, and reliability of the data is maintained, which in turn helps to improve the performance and robustness of the trained model. The following process is commonly followed in the training data sanitization process:

- Data Cleaning: Data cleaning involves identifying and correcting errors, inconsistencies, outliers, and missing values in the training dataset. This may include techniques such as imputation, outlier detection, and data normalization to ensure that the data is suitable for training the model.
- Anomaly Detection: Anomaly detection techniques are used to identify and remove data samples that deviate significantly from the norm or distribution of the dataset. These anomalies may be indicative of errors, noise, or maliciously injected samples and can negatively impact the performance of the trained model if left untreated.
- Noise Reduction: Noise in the training data can adversely affect the model's ability to learn meaningful patterns and generalization to unseen data. Training data sanitization involves filtering out noise data or reducing its influence on the model through techniques such as smoothing, filtering, or feature selection.
- Outlier Removal: Outliers are data points that lie far from the majority of the data and may skew the model's learning process or predictions. Training data sanitization may involve identifying and removing outliers or treating them separately to prevent them from unduly influencing the model.
- Adversarial Data Detection: In the context of adversarial attacks, training data sanitization aims to detect and mitigate the presence of maliciously crafted or poisoned data samples intended to manipulate the model's behavior. This may involve techniques such as adversarial example detection, robust data preprocessing, and data augmentation to improve the model's resilience to such attacks.
- Data Privacy and Security: Training data sanitization also addresses concerns related to data privacy and security by anonymizing sensitive information, removing personally identifiable information (PII), and ensuring compliance with data protection regulations such as GDPR.

This technique applies detection and purification to stop poisoning attacks from affecting models, and it focuses on controlling training data sets. This technique can be applied by comparing models to minimize sample data exploitable in poisoning attacks and filtering that data out, or by identifying potentially poisoned data points based on label characteristics and filtering those points out during retraining [62]. This technique takes advantage of the knowledge that normal training samples are not manipulated by adversaries and usually differ from poisoned samples. Therefore, before performing ML training, data sanitization must be conducted to cleanse the training set and eliminate the poisoned samples. A specific training data sanitization technique as proposed by Bahadoripour et al. [113], analyzed each sample and removed it from training if adding it caused the model's accuracy to drop. This early method was improved upon by sanitization techniques that were later proposed and had less computational complexity. A label cleaning technique created especially for label flipping attacks was presented by Paudice et al. [68]. Outlier detection techniques were suggested by Steinhardt et al. [79] as a means of detecting contaminated samples. Overall, training data sanitization plays an important role in ensuring the quality, reliability, and integrity of the training dataset used to train ML models. By removing errors, anomalies, noise, and malicious data, training data sanitization helps in improving the performance, robustness, and trustworthiness of the trained models, ultimately leading to more accurate and reliable predictions on unseen data.

Adversarial training and Randomized smoothing: Performing adversarial training rather than regular training on the ML training algorithm is an alternate strategy to mitigate availability poisoning attacks. By using model voting, the defender can train a group of several models and produce predictions. Robust optimization techniques, like using a trimmed loss function, are applied in several papers [77], [80]

for presenting a form of adversarial training. Randomized smoothing was also suggested as a technique by Rosenfeld et al. [81] as a way to add noise to training and get certified against label flipping attacks. These were previously discussed in the list of countermeasures for evasion attacks.

7) COUNTERMEASURES FOR TARGETED POISONING ATTACKS

Attacks using targeted poisoning are notoriously difficult to counter. Mitigation techniques for dataset origin and integrity attestation [114] should be applied sparingly to reduce some of the risks related to poisoning attacks as they may cause a degradation in the accuracy and performance of the model.

Differential privacy: Differential privacy is a framework for data privacy protection in which the privacy of individual data points is preserved while still allowing for useful analysis of the data as a whole. In the context of ML, differential privacy aims to enable the training and inference of models on sensitive data while minimizing the risk of revealing sensitive information about individual data points. The core idea behind differential privacy is to add noise to the data or the computations performed on the data in such a way that the output of the analysis remains statistically indistinguishable whether any individual data point is included or excluded from the dataset. This ensures that the presence or absence of any single data point does not significantly affect the outcome of the analysis, thereby protecting the privacy of individual data contributors. Ma et al. [82] proposed the use of Differential Privacy (DP) as a defense for targeted poisoning attacks. This technique used differential privacy during the model training phase to introduce noise into data or models. For example, in order to preserve the privacy of model data, some researchers suggest a technique [83] for creating gradients via differential privacy. However, it is widely known that ML models that are differentially private are less accurate than standard models. For every application, the trade-off between accuracy and robustness must be taken into account. Protection against targeted poisoning attacks is an added benefit if the application has strict data privacy requirements and uses differentially private training for privacy. However, differential privacy will not provide meaningful guarantees for large, poisoned sets, so the robustness provided by differential privacy starts to fade once the targeted attack uses multiple poisoning samples.

8) COUNTERMEASURES FOR BACKDOOR POISONING ATTACKS

Compared to other poisoning attacks, there is a plethora of literature on mitigating backdoor poisoning attacks. We go over a number of mitigation methods below, along with their drawbacks, such as data sanitization, trigger reconstruction and model inspection.

Training data sanitization: Training data sanitization as previously discussed can be used to detect backdoor poisoning attacks as well, much like availability poisoning attacks. For example, outlier detection in the latent feature space has proven to be successful when applied to convolutional neural networks in computer vision applications [85]. The purpose of activation clustering which is also a technique in training data sanitization [115] aims to isolate the backdoored samples in a distinct cluster by clustering training data in representation space. When through a backdoor poisoning attack controls a sizable portion of the training data, data sanitization performs better, but it is less effective against covert poisoning attacks. In general, this results in a trade-off between the detectability of malicious samples and the success of attacks. However, training data sanitization techniques have shown positive impact in defending against backdoor poisoning attacks.

Model pruning: Model pruning in ML refers to the process of reducing the size of a trained model by removing unnecessary or redundant parameters, connections, or layers. The goal of model pruning is to improve the efficiency, speed, and memory footprint of the model while maintaining or even improving its performance on the task at hand. Below are some key aspects and techniques involved in model pruning:

- Weight Pruning: Weight pruning involves identifying and removing individual weights or parameters in the model that contribute little to the overall performance. This is typically done by setting small weights to zero or removing connections with low magnitudes. Weight pruning can significantly reduce the size of the model and improve computational efficiency, particularly for dense neural network architectures.
- Unit Pruning: Unit pruning involves removing entire neurons or units from the model that are deemed unnecessary or redundant. This may include neurons with low activation values or those that have little impact on the model's output. Unit pruning can help to simplify the model architecture and reduce computational overhead.
- Filter Pruning: In convolutional neural networks (CNNs), filter pruning involves removing entire convolutional filters that contribute minimally to the model's performance. Filters with low activation values or those that capture redundant or irrelevant features may be pruned to reduce the model's size and computational cost.
- Layer Pruning: Layer pruning involves removing entire layers from the model that are deemed unnecessary or redundant. This may include fully connected layers, convolutional layers, or recurrent layers that do not significantly contribute to the model's performance. Layer pruning can help to simplify the model architecture and reduce computational complexity.
- Structured Pruning: Structured pruning techniques aim to remove entire clusters of weights, units, filters, or layers from the model while preserving its structural integrity. This allows for more efficient pruning without sacrificing the model's performance. Examples of

structured pruning techniques include magnitude-based pruning and sensitivity-based pruning.
- Iterative Pruning and Fine-Tuning: Pruning is often performed iteratively, with the model being pruned and fine-tuned multiple times to mitigate the performance degradation caused by pruning. Fine-tuning involves retraining the pruned model on the original or a subset of the training data to restore its performance and adapt to the changes introduced by pruning.

Model pruning is an effective technique for reducing the size and complexity of ML models, making them more efficient and suitable for deployment on resource-constrained devices or in real-time applications. By removing unnecessary parameters or structures, model pruning can help improve the scalability, speed, and memory efficiency of ML systems without sacrificing performance. Similar to training data sanitization techniques, with model pruning techniques, the likelihood that backdoor neurons will operate is decreased by pruning off neurons from the original model while maintaining normal functions. To stop backdoor attacks, fine-grained pruning can be used to eliminate the neurons that make up a backdoor [87].

Trigger reconstruction: One potential mitigation approach involves trigger reconstruction, where the goal is to identify and neutralize the triggers inserted by attackers. In the event that the backdoor trigger is present in the contaminated training samples at a fixed location, the goal of this class of mitigations is to reconstruct it. The first trigger reconstruction method was created by Wang et al. [88] with NeutralCleanse. They employed optimization to identify the most likely backdoor pattern that consistently misclassifies the test samples. The original method has been enhanced to support multiple triggers inserted into the model simultaneously and decrease performance time on multiple classes. Artificial Brain Simulation (ABS) by Liu et al. [90] is a representative system in this class; it stimulates multiple neurons and records the activations to reconstruct the trigger patterns. Overall, trigger reconstruction as a mitigation technique aims to identify and neutralize triggers inserted by attackers to protect against adversarial attacks. By removing or neutralizing triggers, the model can become more robust and resistant to manipulation, thereby enhancing its security and reliability in real-world applications.

Model inspection and sanitization: Similar to formal verification and training data sanitization mitigation approaches discussed earlier, model inspection and sanitization ML refers to the processes of examining, analyzing, and ensuring the reliability, fairness, and trustworthiness of ML models. These processes involve evaluating various aspects of the model, including its performance, behavior, and decision-making process, and taking steps to identify and mitigate potential issues or biases. Prior to deployment, model inspection examines the trained machine learning model to see if it was tainted. NeuronInspect [91] is an early work in this field that uses explainability techniques to identify features that differentiate between clean and backdoored models which are then used for outlier detection. DeepInspect [92] learns the probability distribution of trigger patterns using a conditional generative model, then applies model patching to eliminate the trigger. The Meta Neural Trojan Detection (MNTD) framework was introduced by Xu et al. [93]. It involved training a meta-classifier to determine if a particular ML model has been backdoored, or Trojaned, as the authors refer to it. This method is broad and works with various data modalities, including speech, vision, tabular data, and natural language processing. Model sanitization can be carried out by pruning [94], retraining [95], or fine-tuning [116] as well to restore the accuracy of the model after a backdoor is discovered. It is a mix of various techniques as highlighted earlier.

9) COUNTERMEASURES FOR MODEL POISONING ATTACKS

Gradient clipping and differential privacy were proposed in several papers as possible mitigation techniques for Model poisoning attacks [101] but these techniques typically reduce accuracy and don't offer full mitigation. Some of the mitigation techniques that actually work are:

Byzantine-resilient aggregation: Byzantine-resilient aggregation is a technique used in distributed ML to mitigate the impact of Byzantine faults, where nodes in a network behave arbitrarily or maliciously. In Byzantine-resilient aggregation, the goal is to accurately aggregate model updates from participating nodes, even in the presence of a certain number of faulty or adversarial nodes. Many Byzantine-resilient aggregation rules have been developed and assessed in order to protect federated learning environments against model poisoning attacks. When doing the server-side aggregation, majority of them try to detect and eliminate the malicious updates [96], [97], [117], [118]. Nevertheless, by introducing constraints in the attack generation optimization problem, determined adversaries can get around with this defense as well in certain scenarios [102], [103], [117]. Overall, byzantine-resilient aggregation techniques are essential for ensuring the security and integrity of distributed ML systems, particularly in settings where nodes may be untrusted or vulnerable to adversarial manipulation. These techniques help to maintain the accuracy and reliability of the global model despite the presence of Byzantine faults, thereby enabling effective collaborative training in distributed environments.

Regression analysis: Regression analysis is another technique to find noise and anomalous values in data sets using statistical techniques. They are different from training data sanitization techniques. Regression analysis in ML is a statistical method used to model and analyze the relationship between a dependent variable (often denoted as y) and one or more independent variables (often denoted as x). The goal of regression analysis is to predict the value of the dependent variable based on the values of the independent variables. To check for abnormal values, for instance, a model can be defined with various loss functions, or it can make

use of the data's distribution properties. While regression analysis itself may not be a direct defense against model poisoning attacks, it can provide valuable insights and techniques that contribute to the development of more robust and resilient models. By understanding the relationships between variables, detecting anomalies, improving model interpretability, and training data sanitization techniques, regression analysis can indirectly aid in mitigating the impact of adversarial attacks on ML systems [62].

Ensemble analysis: Ensemble analysis is a technique which combines multiple regression models to make predictions and can improve robustness against adversarial attacks. By aggregating predictions from multiple models trained on different subsets of the data or using different algorithms, ensemble methods can reduce the impact of individual models that may be vulnerable to adversarial manipulation. This mitigation technique places a strong emphasis on using several sub-models to strengthen an ML system's defense against model poisoning attacks. There is a lower chance of model poisoning attacks occurring in a system that has multiple independent models using distinct training data sets [62].

Model watermarking: Model watermarking is a technique used to embed unique identifiers or signatures, known as watermarks into ML models. These watermarks serve as a form of digital fingerprint that can be used to verify the ownership, authenticity, or integrity of the model. Model watermarking is primarily used for security and intellectual property protection purposes in scenarios where ML models are shared, distributed, or deployed in untrusted environments. The model watermarking technique works by:

- Embedding Watermarks: During the training or model development phase, a unique watermark is embedded into the model parameters or architecture. This watermark is typically designed to be imperceptible and robust to various transformations or modifications to the model.
- Verification: Once the model is trained and ready for deployment, the presence of the watermark can be verified using specialized techniques or algorithms. These verification methods check whether the model contains the expected watermark and validates its authenticity.
- Ownership and Authenticity Verification: Model watermarking allows for the verification of ownership and authenticity of the model. If the model is shared or distributed, the presence of the watermark can be used to trace the origin of the model back to its rightful owner or creator. This helps prevent unauthorized distribution, copying, or modification of the model.
- Integrity Checking: In addition to verifying ownership and authenticity, model watermarking can also be used to check the integrity of the model. If the model is modified or tampered with, the watermark may become invalid or distorted, indicating that the model has been compromised.
- Protection Against Unauthorized Use: By embedding watermarks into ML models, organizations can deter unauthorized use, distribution, or theft of their intellectual property. The presence of watermarks serves as a deterrent to potential attackers and provides a mechanism for detecting unauthorized use or infringement.

During the model training phase, model watermarking incorporates unique recognition neurons into the original model. These neurons make it possible to use a unique input sample to determine whether another model was created by stealing the original. It's important to note that model watermarking techniques should be designed to be robust, resistant to removal or tampering, and compatible with the specific requirements and constraints of ML models and applications. Additionally, while model watermarking can help protect against certain threats, it is not an ultimate solution and should be used in conjunction with other security measures and best practices for comprehensive defense in depth protection of ML models and data.

C. PRIVACY ATTACKS

Privacy attacks refer to various techniques or strategies employed by adversaries to compromise the privacy of individuals whose data is used to train or infer these models. Privacy attacks take advantage of over-generalization of training data, a typical issue in supervised ML models, to identify data utilized during model training. Attackers can carry out this action even in the absence of knowledge or access to a target model's parameters as shown in Figure 7. This poses a data privacy risk to models trained on confidential data such as medical records and or personally identifiable information [119]. Several companies are more driven to create models based on publicly available models as a result of the popularity of transfer learning and the accessibility of several state-of-the-art ML models. This makes it easier for attackers to obtain information about the kind and structure of the models being used [119]. Among privacy attacks, a specific type called as membership inference attacks is largely dependent on over-fitting as a consequence of poor ML procedures; as such, a model that successfully generalizes to the actual data distribution may be more resilient to membership inference attacks [119].

Although privacy concerns have existed for a while, Dinur and Nissim's [36] significant work on data reconstruction attacks marked the beginning of privacy attacks against aggregate statistical data gathered from user records. Data reconstruction attacks aim to obtain aggregate statistical data and use it to reverse engineer sensitive information about a specific user record or sensitive critical infrastructure data. In the context of large generative language models, like GPT-2, memorization attacks that regenerate or reconstruct the training data have been demonstrated more recently [59]. In membership inference attacks which allows an adversary to ascertain whether a specific record was included in the dataset used to calculate statistics or train a ML model may have other privacy implications such as trying to identify

is a certain person's information is available in the records. An example of membership inference attacks has been presented by Homer et al. [120] on genomic data. In most black-box scenarios, where adversaries have query access to a trained ML model, recent literature has focused on membership attacks against ML models. Model extraction attacks, which aim to extract confidential details about an ML model, like its architecture or parameters, are also a type of privacy attacks [121]. Finally, attacks known as property inference attacks seek to extract global data from a training dataset such as the percentage of training samples that possess a particular sensitive dataset attribute. The below sections covers various types of privacy attacks pertaining to membership inference, data reconstruction, model extraction, and property inference. It also covers mitigations for some of these attacks and unresolved issues with general mitigation strategy design.

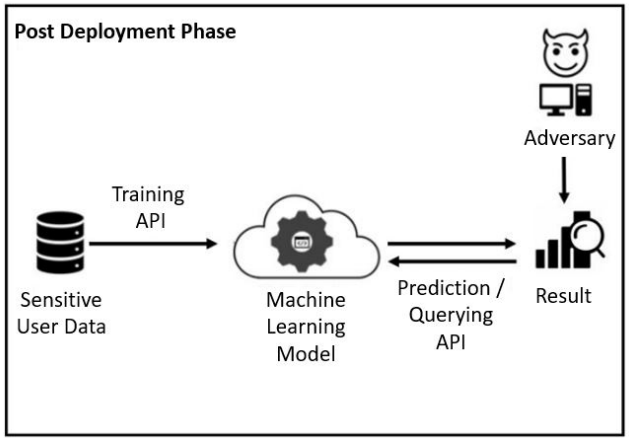

FIGURE 7. Privacy attack process flow.

1) DATA RECONSTRUCTION ATTACKS

Data reconstruction attacks in ML involve adversaries attempting to reconstruct sensitive or private training data from the model's predictions or outputs. These attacks exploit vulnerabilities in ML models that inadvertently leak information about the training data, potentially compromising the security and privacy of individuals whose data was used for training. Since data reconstruction attacks can retrieve personal data from publicly available aggregated statistical data, they are the most worrying privacy threats. Data reconstruction attacks were first introduced by Dinur and Nissim [36], who extracted user data from linear statistics. Although their initial attack necessitates an exponential amount of queries for reconstruction, later research had demonstrated that data reconstruction can be completed with a polynomial number of queries. Dwork et al. [122] presented a survey of privacy attacks, including data reconstruction attacks. Differential privacy was used in 2020 U.S. Census decennial release, as found in a large-scale study conducted by the U.S. Census Bureau on data reconstruction attacks on census data. Model inversion attacks were first described by Fredrickson et al. [123] in relation to ML classifiers. These attacks reconstruct class representatives using the training data of the model. Model inversion attacks i.e., a type of a data reconstruction attack cannot directly reconstruct the model's training data, but it can produce images that are semantically similar to those in the training set. In a recent study, Balle et al. [61] trained a reconstructor network that under the assumption of a strong adversary with knowledge of every other training sample can retrieve a data sample from a neural network model. By utilizing theoretical insights about implicit bias in neural networks, Haim et al. [124] demonstrated how the training data of a neural network can be reconstructed from access to the model parameters. Attribute inference is a pertinent data reconstruction attack wherein the perpetrator extract a sensitive attribute from the training set.

2) MEMORIZATION ATTACKS

A potent class of attacks known as memorization attacks enables an adversary to take training data out of generative ML models, like language models. Memorization attacks in ML involve exploiting vulnerabilities in models that have inadvertently memorized sensitive or private information from the training data. In these attacks, an adversary can extract or infer sensitive information about individuals from the model's predictions. The first practical demonstration of memorization attacks in language models was done by Carlini et al. [58]. They created a technique for extracting the attributes by introducing artificial attributes into the training set. They also introduced a metric called exposure to gauge memorization. Later research revealed that memorization is a risk factor in large language models like GPT-2 [59] and that models with higher capacities typically memorize more. Examining the relationship between generalization and memorization in ML models is an independent field of study. The topic of neural networks ability to memorize randomly chosen datasets was covered by Zhang et al. [125]. It was demonstrated that almost optimal generalization error in machine learning requires the memorization of training labels. Two learning tasks based on next-symbol prediction and cluster labeling were created by Brown et al. [126], where memorization is necessary for high-accuracy learning. Feldman and Zhang used an influence estimation method to empirically evaluate the usefulness of memorization for generalization [125].

3) MEMBERSHIP INFERENCE ATTACKS

Membership inference attacks in ML involve adversaries attempting to determine whether specific data points were used to train a ML model. These attacks exploit vulnerabilities in ML models that inadvertently leak information about the presence or absence of individual data samples in the training dataset. When releasing aggregate statistical data or ML models trained on user data, membership inference attacks pose a greater risk than reconstruction or

memorization attacks because they typically reveal personal information about a person. In certain circumstances, such as a medical study involving patients with a rare disease, determining that an individual is a part of the training set already has privacy implications. Furthermore, one can use membership inference as a foundation to launch extraction attacks [58], [59]. The attacker's objective in membership inference is to ascertain if a specific record or data sample was included in the training dataset in which the statistical or ML algorithm used. Known as ''tracing attacks,'' these attacks were first presented by Homer et al. [120] for use in statistical analyses on genomic data. When an adversary obtains noisy statistical data about the dataset, robust tracing attacks have been examined. The term membership inference has been used in the literature to describe attacks on ML models during the past five years. The majority of attacks documented in the literature target deep neural networks that are employed in classification.

4) MODEL EXTRACTION ATTACKS

Model extraction attacks in ML involve adversaries attempting to infer or recreate a target model's architecture, parameters, or functionality using limited access to the model's outputs or predictions. These attacks exploit vulnerabilities in ML models that inadvertently leak information about the model's internal workings. Cloud providers often use proprietary data to train large ML models in ML-as-a-service scenarios and they prefer to maintain the model architecture and parameters private. By posing queries to the ML model that has been trained by a ML-as-a-service provider, an attacker executing a model extraction attack seeks to obtain details about the model architecture and parameters. Tramer et al. [50] demonstrated the first model stealing attacks on a number of online ML services for various ML models, such as logistic regression, decision trees, and neural networks. But as Jagielski et al. [77] have demonstrated, it is impossible to extract ML models precisely. As an alternative, a functionally equivalent model that differs from the original model but performs comparably on the prediction task can be rebuilt. Even the more straightforward task of extracting functionally equivalent models is NP-hard, as demonstrated by Jagielski et al. [77].

5) PROPERTY INFERENCE ATTACKS

Property inference attacks in ML involve adversaries attempting to infer sensitive properties or characteristics of the training data used to train an ML model, based on the model's outputs or predictions. These attacks exploit vulnerabilities in ML models that inadvertently leak information about the properties of the underlying data. Through interaction with an ML model, the attacker attempts to gain global knowledge about the distribution of training data in property inference attacks. For example, an attacker could figure out what percentage of the training set has a particular sensitive attribute like demographic data that could reveal potentially private information about the training set that isn't supposed to be made public. Ateniese et al. [127] first described property inference attacks which were later formalized as a distinguishing game in which the attacker and the challenger train two models using varying percentages of sensitive data. Property inference attacks were developed for use in both black-box and white-box environments where the attacker used queries to learn the class probabilities or the predicted labels [128] while having access to the entire ML model.

6) ATTACKER TACTICS AND TECHNIQUES FOR PRIVACY ATTACKS

In May 2023, A vulnerability in ChatGPT known as ''indirect prompt injection'' was discovered. It allowed an attacker to take control of a chat session and steal the conversation's history by using ChatGPT plugins to feed malicious websites [129]. ChatGPT users would have been susceptible to personally identifiable information (PII) leakage from the extracted chat session as a result of this attack. The procedure carried out by the researchers who identified this vulnerability was as follows. Initially, when the researchers realized that ChatGPT uses open-source plugins, they created a malicious website based prompt injection that can be used instead. Using the open-source plugin, the researchers then designed a prompt injection attack that the large language model (LLM) consumes in order to alter its behavior when it is directed to access the malicious website during a chat session. In this use case, the researchers tool advantage of a ChatGPT plugin that was made to access a user-supplied URL that was intended to process the plain text on the website in order to retrieve confidential information. The indirect prompt injection attack instructed the LLM to compile the user's previous chat history and append it to the URL in order to exfiltrate more information at a later time when the plugin accesses this malicious website. Due to the hacker's access to the user's chat history, the user was now exposed to a number of risks, including the exposure of personally identifiable information. Similar tactics and techniques are used by real-life attackers to compromise personally identifiable information of application users and compromise the privacy controls of the application.

7) COUNTERMEASURES FOR ALL PRIVACY ATTACKS

Several countermeasures against various types of privacy attacks are documented in literature. These include restricting user queries to the model, identifying suspicious queries to the model and building stronger architectures to prevent side channel attacks. A number of studies have documented unfavorable outcomes from different mitigation techniques plotted against these attacks. A high accuracy ML model should logically yield some aggregate information about the training dataset. Although mitigating property inference attacks may not be simple, it remains unclear if users providing data for ML are truly at risk of privacy violations as a result of these attacks. These methods should only be used sparingly as determined and well-resourced attackers

TABLE 3. Review of privacy attack studies.

Ref No.	Dataset	Type of ML Architecture / Method	Attack Type	Contribution	Proposed Countermeasure
[130]	UCI Adult dataset, U.S. Census Income dataset, Bank Marketing dataset, CelebA dataset	Deep Neural Networks (DNNs)	Privacy Attack	Examined the scenario of property inference attacks, and an attacker's ability to contaminate a portion of the training dataset and ask questions about the trained target model. The authors created an effective property inference attack called SNAP that outperformed the state-of-the-art poisoning-based property inference attack in terms of attack success and poisoning requirements.	Differential Privacy
[131]	Census and Enron	Deep Neural Networks (DNNs)	Privacy Attack	Presented property inference poisoning attack, which enables the adversary to discover the prevalence of any selected property in the training data: it selects the target property, feeds in data based on a poisoned distribution, and then employs black box queries (label-only queries) on the trained model to ascertain the frequency of the selected property. The authors provided countermeasures for this specific attack in detail.	Differential Privacy
[132]	FMNIST, CIFAR10 and P100	Deep Neural Networks (DNNs): Logistic Regression and Feed-Forward Network	Privacy Attack	Examined whether Differentially Private Stochastic Gradient Descent provided more privacy in real-world scenarios than its cutting-edge analysis guarantees. Supplemented analytical work on differential privacy by employing a quantitative, empirical approach to understanding the privacy provided by particular implementations of differentially private algorithms.	Privacy Auditing
[133]	Not mentioned	Not mentioned	Privacy Attack	Used logistic regression and neural network models to conduct experiments and quantify the effect of decisions on privacy. Concluded that privacy cannot be obtained for free; looser definitions of differential privacy lead to higher measured leakage of privacy while lowering the amount of noise required to improve utility. For complex learning tasks, current mechanisms for differentially private machine learning rarely provide acceptable utility-privacy trade-offs: settings with limited accuracy loss offer little effective privacy, and settings with strong privacy lead to useless models.	Differential Privacy
[134]	Multiple datasets	Deep Neural Networks (DNNs): CNNs	Privacy Attack	Explored how to generate deep models that take into account the integration and combination of heterogeneous visual cues across sensory modalities, with the goal of improving the computer vision community's understanding of key concepts and algorithms of deep multimodal learning	Private Aggregation of Teacher Ensembles (PATE)

can exploit them. The few countermeasures that have been useful in reducing the risk of Privacy attacks are differential privacy, privacy auditing of the model and private aggregation of teacher ensembles as elaborated further.

Differential Privacy: Differential privacy refers to providing strong privacy guarantees by ensuring that the inclusion or exclusion of any individual data point does not significantly affect the output or results of the analysis. DP is often achieved through the use of randomized algorithms that introduce controlled amounts of noise or randomness into the data or computations. This randomization helps prevent adversaries from inferring sensitive information about individual data points. The rigorous definition of DP was motivated by the discovery of reconstruction attacks against aggregate statistical data. DP ensures a bound on the amount of information that an attacker possessing access to the algorithm's output can discover about each individual record in the dataset. A privacy parameter, or privacy budget, is present in the original pure definition of DP. It limits the likelihood that an attacker who has access to the algorithm's output will be able to ascertain whether a specific record was included in the dataset. DP offers defense against membership inference, training data memorization, and reconstruction attacks. However, DP does not offer defenses against attacks involving property inference or model extraction [128], [135]. Setting up the privacy parameters to achieve a trade-off between privacy and utility which is usually measured in terms of accuracy for ML models is one of the main challenges of using DP in practice.

Privacy auditing: Privacy auditing in ML refers to the process of evaluating and assessing the privacy risks associated with ML systems. The goal of privacy auditing is to identify potential privacy vulnerabilities, compliance violations, or data leakage risks and to ensure that appropriate measures are taken to protect the privacy of individuals whose data is being used or processed. Privacy auditing involves analyzing ML models, datasets, or systems to identify potential privacy risks and vulnerabilities. This may include assessing the sensitivity of the data being used, the types of ML algorithms being employed, the data processing and storage practices, and the potential impact of data breaches or unauthorized access. Privacy auditing also involves evaluating the compliance of ML systems with relevant privacy regulations, standards, and best practices. This may

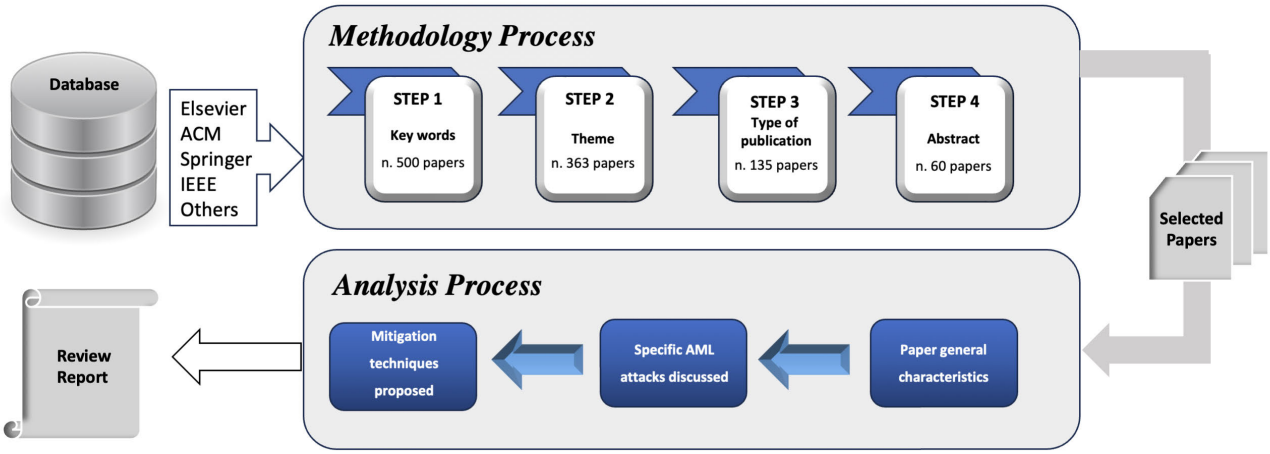

FIGURE 8. Methodology process flow.

include assessing compliance with laws such as the General Data Protection Regulation (GDPR), the Health Insurance Portability and Accountability Act (HIPAA), or industry-specific regulations governing data privacy and security. Based on the findings of the privacy audit, recommendations and strategies for mitigating privacy risks and vulnerabilities are developed. This may involve implementing technical measures such as data anonymization, differential privacy, encryption, or access controls, as well as organizational policies and procedures to ensure compliance and protect privacy. Recently, privacy auditing has emerged as a promising field of study, as described by Jagielski et al. [132]. Its objectives are to measure an algorithm's actual privacy guarantees empirically and determine privacy lower bounds by launching privacy attacks. Membership inference attacks can be used for auditing [133], but poisoning attacks are far more useful for empirical privacy auditing. Privacy auditing plays a crucial role in ensuring that ML systems are developed, deployed, and operated in a privacy-responsible manner, protecting the privacy rights and interests of individuals and maintaining compliance with relevant privacy laws and regulations.

Private Aggregation of Teacher Ensembles (PATE): PATE is a technique used in privacy-preserving ML to train models on sensitive data without exposing individual-level information. PATE involves training an ensemble of models, referred to as ''teachers,'' on the sensitive data. Each teacher model is trained on a distinct subset of the sensitive data, allowing them to learn different aspects of the underlying distribution. The ensemble of teacher models collectively makes predictions on a public dataset or test data, without directly accessing the sensitive data. These predictions are then aggregated using a voting scheme or averaging to obtain a final prediction. By aggregating predictions from multiple teacher models, PATE can help mitigate overfitting issues that may arise from training on limited datasets. The ensemble nature of PATE provides a more generalized view of the data, reducing the risk of over-reliance on specific patterns present in the training data. PATE provides a privacy guarantee by introducing noise into the label aggregation process. This noise ensures that the aggregated predictions do not reveal sensitive information about individual data points, even to the entity performing the aggregation. The aggregated predictions are used as labels to train a ''student'' model on the public dataset. The student model learns from the collective knowledge of the teacher ensemble while preserving the privacy of the sensitive data. PATE involves a trade-off between privacy and utility. By aggregating predictions from multiple teacher models and introducing noise, PATE protects the privacy of individual data points but may result in a loss of model accuracy or utility. In the training stage, this technique divides training data into multiple sets, each of which is used to train a separate model. Then, by voting, the independent models are used to jointly train a student model [134]. This technique protects the privacy of the training data by making sure the student model's inference does not reveal any information from a specific training data set. PATE is a powerful technique for training models on sensitive data. It enables organizations to leverage the collective knowledge of teacher ensembles without compromising the confidentiality of the underlying data, making it suitable for applications in healthcare, finance, and other domains with strict privacy requirements.

IV. METHODOLOGY

In this section, the researchers detail the methodology utilized to assess individual countermeasures and their practical implementation within ML environments. The study leveraged peer-reviewed academic journals such as IEEE, Elsevier, Springer, ACM, Taylor and Francis, along with conference proceedings, to survey the existing academic research on AML attacks. Initially, the researchers conducted searches for relevant studies in these journals and conference proceedings using specific keywords such as ''Evasion,''

"Poisoning," "Privacy," "Adversarial Machine Learning," and "Machine Learning Attacks." The identified academic papers were then filtered based on associated keywords, abstracts, and introductory sections, resulting in over 500 papers addressing ML/AI security and featuring countermeasures for AML attacks. Subsequently, the researchers narrowed down the selection by reviewing papers proposing specific countermeasures practical for detecting, preventing, or responding to AML attacks, resulting in 363 shortlisted papers. Finally, relevant papers were chosen based on predetermined search criteria, including keywords, abstracts, introductions, and the proposal of implementable countermeasures relevant to real-world environments combating evasion, poisoning, and privacy attacks. This process yielded 135 papers, which were categorized into separate groups and individually reviewed to identify pertinent countermeasures proposed in the research. Furthermore, the studies were scrutinized to assess the ease of implementation and the potential assistance they offer organizations in detecting, preventing, or responding to AML attacks. Countermeasures that lacked testing or implementation were excluded from consideration, resulting in a final set of 60 papers. The strengths and weaknesses of each proposed countermeasure from these papers are elaborated on in the subsequent section. The methodology process flow is illustrated in Figure 8. Additionally, aside from academic literature, the researchers also evaluated open-source AI/ML frameworks and tools recently released by leading standard developing organizations (SDOs) to construct secure ML models and enhance defense mechanisms against AML attacks.

Table 4 offers a comprehensive overview of all the reviewed papers, summarizing the proposed countermeasures for addressing AML attacks. These countermeasures are classified based on their effectiveness and feasibility of implementation within organizations. This categorization is determined qualitatively, taking into account the complexity associated with implementing the countermeasures as described in the reviewed studies. The results and discussion section provides additional insights into the strengths and limitations of each proposed countermeasure, offering clarity on their effectiveness in mitigating AML attacks.

V. RESULTS AND DISCUSSION

In this section, the conclusions drawn from the reviewed studies are consolidated and evaluated concerning the insights offered on countermeasures for each attack type. Additionally, the researchers offer a recapitulation of the surveyed frameworks and tools, along with highlighting the most notable resources accessible for organizations to employ in securing the code deployment of ML/AI models and conducting application testing. Several defense strategies have been suggested in the literature to safeguard against AML attacks. This section delineates the suggested countermeasures for evasion, poisoning, and privacy attacks, respectively.

A. EVASION ATTACKS MITIGATION APPROACHES

Evasion attacks typically involve two primary steps: first, manipulating an input, and then feeding the modified input into the targeted model. These attacks are often conducted subtly, resulting in adversarial input samples that are crafted in a manner imperceptible to the human eye. Adversarial input samples are meticulously constructed by malicious actors to deceive machine learning models. Therefore, the main objectives of proposed mitigatory countermeasures against evasion attacks are to identify these adversarial samples during both the model development and deployment stages. By detecting and mitigating the presence of adversarial input samples, these countermeasures aim to enhance the robustness and security of machine learning models against evasion attacks.

Table 5 lists out the strengths and weaknesses of each countermeasure proposed against evasion attacks. Based on the strengths and weaknesses of each countermeasure, it can be confirmed that Adversarial Training and Formal Verification methods are the most optimal mitigation strategies against Evasion attacks. Techniques such as adjusting feature dimension of a model and defensive distillation may not be as effective mitigation approaches for large networks as they are unscalable and decreases the accuracy of the model.

Adversarial training: In order to mitigate evasion attacks, one of the most popular methods being investigated is adversarial training. To ensure that the model won't be tricked by adversarial samples, adversarial training entails training the model on a training data set supplemented with appropriately labeled adversarial samples. Several techniques for mounting evasion attacks, including the fast gradient sign method (FGSM), can be used to compute the adversarial examples intended for training [25], [41], [49]. References [136] and [137] explore the use of adversarial training to improve the security of classifiers against evasion attacks. Reference [136] focuses on feature selection, proposing an adversary-aware model that enhances classifier security. On the other hand, [137] applied adversarial training to a modulation classifier radio, achieving varying levels of robustness against different attacks. Reference [138] introduced a blackbox morpher for evasion attacks, while [139] proposed a feedback learning method to improve the robustness of neural networks against various evasion attacks. These studies collectively highlight the potential of adversarial training in enhancing the security of classifiers against evasion attacks. Adversarial training is a better approach against evasion attacks because it enhances the robustness of ML models by teaching them to generalize better to adversarial inputs. By learning to handle adversarial examples during training, the model becomes more resilient to subtle variations or perturbations in the input data, reducing the effectiveness of evasion attacks. Adversarial training can be effective against both known and unknown evasion attacks. While traditional defenses may be susceptible to new or adaptive attack strategies,

TABLE 4. Mitigatory countermeasures for AML attacks.

Ref.No.	AML Attack Type	Suggested Mitigatory Countermeasures	Strength of Countermeasure (Strong / Weak)	Ease of Implementation (Easy / Hard)
[52], [53]	Evasion Attacks	Randomized Smoothing	Weak	Easy
[54], [55], [56], [57]	Evasion Attacks	Formal Verification	Strong	Hard
[58], [59], [49]	Evasion Attacks	Adjusting Feature Dimension of Model	Weak	Hard
[33], [60], [63] [136] [137] [138] [139]	Evasion Attacks	Adversarial Training	Strong	Hard
[61] [64]	Evasion Attacks	Defensive Distillation	Strong	Hard
[113]	Evasion Attacks	Input Reconstruction	Weak	Hard
[62] [68], [113], [79]	Availability Poisoning Attacks	Training Data Sanitization	Strong	Easy
[84], [140], [80], [77], [81], [114]	Availability Poisoning Attacks	Adversarial Training and Randomized Smoothing	Strong	Hard
[82], [83]	Targeted Poisoning Attacks	Differential Privacy	Strong	Easy
[85], [86], [141], [115]	Backdoor Poisoning Attacks	Training Data Sanitization	Strong	Easy
[87]	Backdoor Poisoning Attacks	Model Pruning	Weak	Easy
[88], [89], [142], [90]	Backdoor Poisoning Attacks	Trigger Reconstruction	Weak	Hard
[91], [92], [93], [94], [95], [116]	Backdoor Poisoning Attacks	Model Inspection and Sanitization	Strong	Hard
[117], [118], [96], [97], [143]	Model Poisoning Attacks	Byzantine-resilient Aggregation	Strong	Hard
[62]	Model Poisoning Attacks	Regression Analysis	Strong	Hard
[62]	Model Poisoning Attacks	Ensemble Analysis	Strong	Hard
[62]	Model Poisoning Attacks	Model Watermarking	Weak	Hard
[130], [128]	Privacy Attacks	Differential Privacy	Strong	Easy
[132] [133]	Privacy Attacks	Privacy Auditing	Strong	Easy
[134]	Privacy Attacks	Private Aggregation of Teacher Ensembles (PATE)	Strong	Hard

adversarial training provides a proactive defense mechanism that can adapt to novel attack techniques. By exposing the model to adversarial perturbations during training, the model learns to recognize and adapt to these perturbations, making it more robust against future attacks. Adversarial training can be applied to various types of ML models and architectures, including deep neural networks, convolutional neural networks, as well as recurrent neural networks. It is a versatile approach that can be scaled to large datasets and complex models, making it suitable for a wide range of applications. It is supported by theoretical frameworks and empirical evidence demonstrating its effectiveness in improving model robustness against evasion attacks. Research has shown that adversarial training can significantly reduce the vulnerability of ML models to adversarial perturbations. Finally, it can be combined with other defense mechanisms, such as randomized smoothing, adjusting feature dimension of the model and input reconstruction to enhance the overall defense against evasion attacks. By integrating multiple defense strategies, organizations can build more resilient ML systems that are better equipped to withstand adversarial threats. It is considered as a better mitigation approach against evasion attacks because it directly addresses the underlying vulnerability of ML models to adversarial perturbations and improves their robustness in the face of such attacks

Formal verification: While formal verification is a computationally expensive method and has its own limitations and challenges, the formal verification research studies prove that it can provide strong guarantees about the correctness and robustness of ML models through mathematical proofs. By formally specifying security properties and verifying them against the model's behavior, organizations can ensure that the model behaves as intended and is resilient to evasion attacks. It can help identify vulnerabilities and weaknesses in ML models that may otherwise get unnoticed by other mitigation techniques and still leave the model vulnerable to evasion attacks. By systematically analyzing the model's architecture, algorithms, and decision-making processes, organizations, it can uncover potential evasion attack vectors and strengthen the model's defenses accordingly. Formal verification techniques for certifiable training aim to train neural networks to increase their lower bound on robustness [34], [35], [36], [37]. One approach is currently in use in several ML models [33]. A range of studies have proposed formal verification techniques to enhance the security of machine learning models and networked

TABLE 5. Evasion attack countermeasures - strengths and weaknesses.

Countermeasures	Strengths	Weaknesses
Randomized Smoothing [48] – [49]	Provides a probabilistic defense mechanism against evasion attacks by adding noise to input data and offers strong theoretical guarantees against certain types of adversarial perturbations.	May introduce computational overhead and increase model complexity. The effectiveness of randomized smoothing may vary depending on the choice of smoothing parameters and the distribution of adversarial perturbations.
Formal Verification [[33], [50] – [53], [144] –[145]]	Offers rigorous mathematical guarantees by formally verifying the correctness and robustness of ML models. Provides a systematic approach to detecting vulnerabilities and ensuring compliance with security requirements.	Requires significant computational resources and expertise to perform formal verification, making it impractical for large-scale models or complex systems. Limited scalability and applicability to certain types of ML models and architectures.
Adversarial training [12], [21], [30], [37], [43], [54], [56], [135] – [134]	Effectively enhances the robustness of ML models against adversarial attacks by incorporating adversarial examples into the training process. Can be combined with other defense mechanisms to improve overall resilience.	May result in increased computational overhead and longer training times. Vulnerable to adaptive adversaries that can generate more sophisticated adversarial examples.
Adjusting feature dimension of model [43], [57] – [60]	Can help mitigate the impact of adversarial perturbations by reducing the model's sensitivity to irrelevant or noisy features. Improves the interpretability and generalization of the model by focusing on the most informative features.	Requires careful feature selection and dimensionality reduction techniques, which may not always be straightforward or effective. May lead to information loss or degradation in model performance if important features are omitted.
Defensive distillation [62] – [63]	Provides a defense mechanism against adversarial attacks by training a "distilled" model that smooths out the decision boundaries learned by the original model. Offers resistance against simple gradient-based attacks by making the model more robust to perturbations.	Vulnerable to adaptive adversaries that can craft adversarial examples specifically targeted at the distilled model. May not provide robust protection against more sophisticated attacks or adversarial strategies.
Input reconstruction [58] –[59]	Helps mitigate the impact of adversarial perturbations by reconstructing clean input data from perturbed inputs. Can be combined with other defense mechanisms to enhance the robustness of ML models.	Requires access to clean training data or a reconstruction model, which may not always be available in practical settings. May introduce additional computational overhead and complexity, particularly for real-time applications or large-scale models.

control systems against evasion attacks. Reference [146] introduces a method that transforms machine learning models into imperative programs for analysis, while [144] focuses on revealing stealth attacks on networked control systems. Reference [147] presents an automated verification framework for hardware circuits, specifically designed to protect cryptographic implementations against combined physical attacks. These studies collectively contribute to the development of robust security measures against evasion attacks. It can also be applied to different domains and types of ML models, including deep neural networks, decision trees, and support vector machines. This flexibility makes formal verification suitable for a wide range of applications and use cases. The subsequent studies for each countermeasure as highlighted in Table 1 have made valuable contributions by outlining multiple innovative methods and approaches to lessen the impact of evasion attacks. Other techniques such as randomized smoothing do not provide complete protection and should be used in conjunction with other defense mechanisms such as adversarial training and formal verification for optimal security.

B. POISONING ATTACKS MITIGATION APPROACHES

Attackers employ poisoning attacks to manipulate the learning process by introducing adversarial data samples or altering existing ones within the training dataset. This manipulation compromises the integrity of the trained model, leading to inaccurate outputs during inference. Poisoning attacks encompass various subtypes, including Availability Poisoning Attacks, Targeted Poisoning Attacks, Backdoor Poisoning Attacks, and Model Poisoning Attacks. Backdoor attacks, considered one of the most prevalent forms of poisoning attacks, involve the insertion of a backdoor, also known as a Trojan, into the targeted model during the training phase. This backdoor remains dormant for clean inputs during inference, exhibiting normal behavior. However, when presented with inputs containing a specific, predefined pattern or trigger, the model misbehaves or yields desired outcomes for the attacker. While backdoor poisoning attacks focus on embedding backdoors for future exploitation, other forms of poisoning attacks aim to compromise the overall functionality of the model. These attacks undermine the trustworthiness and reliability of machine learning models, posing significant threats to their security and effectiveness in real-world applications. It's essential to note that no single countermeasure can provide complete protection against all types of poisoning attacks. Organizations should carefully evaluate their specific requirements, risks, and constraints to determine the most appropriate combination of countermeasures for their machine learning systems. Table 6 lists out the strengths and weaknesses of each countermeasure proposed against poisoning attacks. As per studies evaluated, training data sanitization is the most effective defense currently available against poisoning attacks.

Training data sanitization: Training data sanitization is often considered the most fundamental and effective

TABLE 6. Poisoning attack countermeasures - strengths and weaknesses.

Countermeasures	Strengths	Weaknesses
Training Data Sanitization [55], [58], [113], [108]	Helps mitigate the risk of poisoning attacks and data contamination by identifying and removing malicious or suspicious data points. Enhances the reliability and trustworthiness of ML models by ensuring the integrity of the training data.	Requires careful data preprocessing and cleaning techniques, which may be resource-intensive and time-consuming. May inadvertently remove legitimate data points or introduce bias if not implemented carefully.
Adversarial Training & Randomized Smoothing [73], [109] – [111], [136] – [138],	Provides robust defense mechanisms against evasion attacks by incorporating adversarial examples into the training process and adding noise to input data. Offers strong theoretical guarantees against certain types of adversarial perturbations.	May result in increased computational overhead and longer training times, particularly when combined with randomized smoothing. Vulnerable to adaptive adversaries that can generate more sophisticated adversarial examples.
Model Pruning [82]	Helps improve the efficiency and performance of ML models by reducing redundancy and complexity. Can enhance the interpretability and generalization of the model by focusing on the most informative features or parameters.	Pruning techniques may be sensitive to the choice of pruning criteria and hyperparameters, which can affect model performance. May lead to information loss or degradation in model accuracy if important features or parameters are pruned excessively.
Trigger Reconstruction [83] –[85], [84] – [140]	Provides a defense mechanism against trigger-based attacks by detecting and reconstructing trigger patterns or backdoor signals in input data. Enhances the security and integrity of ML models by mitigating the risk of trigger-based manipulation.	Requires access to clean training data or a reconstruction model, which may not always be available or practical. May introduce additional computational overhead and complexity, particularly for real-time applications or large-scale models.
Model Inspection and Sanitization [115] – [92]	Offers a systematic approach to identifying and mitigating vulnerabilities in ML models by analyzing their structure, behavior, and decision-making process. Helps improve the transparency, trustworthiness, and reliability of ML systems by ensuring adherence to safety and security requirements.	Limited scalability and applicability to complex or black-box models, as model inspection techniques may struggle to provide insights into their internal workings. Requires domain expertise and specialized tools for effective model inspection and sanitization, which may not be readily available or accessible.
Byzantine-Resilient Aggregation [94] – [118], [86]	Provides a defense mechanism against Byzantine faults in distributed ML systems by aggregating model updates from multiple sources and mitigating the impact of malicious or faulty nodes. Enhances the robustness and reliability of collaborative learning approaches by ensuring the integrity of aggregated updates.	Byzantine-resilient aggregation techniques may introduce additional communication overhead and complexity, particularly in large-scale distributed systems. Limited effectiveness against sophisticated Byzantine adversaries that can coordinate attacks across multiple nodes or manipulate the aggregation process strategically.
Regression Analysis [55]	Offers a powerful tool for analyzing relationships between variables and identifying patterns in data, which can be valuable for detecting anomalies, identifying vulnerabilities, and informing decision-making in ML systems. Provides insights into the impact of model features on model performance and helps identify potential sources of bias or error.	Limited effectiveness in detecting subtle or complex patterns in high-dimensional data or non-linear relationships. Requires domain expertise and careful interpretation of results to draw meaningful conclusions, which can be challenging in complex or interdisciplinary applications.
Ensemble Analysis [55]	Enhances the robustness and reliability of ML models by combining predictions from multiple models or algorithms, reducing the risk of overfitting and improving generalization performance. Provides a defense mechanism against adversarial attacks by leveraging diversity among ensemble members to mitigate the impact of individual vulnerabilities.	Ensemble methods may introduce additional computational overhead and complexity, particularly when integrating diverse models or algorithms. Requires careful selection and tuning of ensemble components to optimize performance and achieve synergy among individual models.
Model Watermarking [55]	Asserts ownership and intellectual property rights over models.	May affect model performance and accuracy. Effectiveness depends on the robustness of the watermarking technique and sophisticated attackers may remove or alter watermarks.

countermeasure against poisoning attacks due to several reasons. It addresses poisoning attacks directly at the source by ensuring the cleanliness and integrity of the training dataset. By rigorously validating and preprocessing the training data, organizations can detect and remove malicious or anomalous data points before they are used to train ML models. This is a proactive approach that prevents poisoning attacks from infiltrating the model during the training phase. It is applicable across various types of machine learning models and architectures. Whether using deep neural networks, decision trees, support vector machines, or other models, ensuring the quality and trustworthiness of the training data is a critical step in building robust and reliable machine learning systems. [45] This generalizability makes training data sanitization a versatile and widely applicable countermeasure against poisoning attacks. By removing potentially malicious or adversarial data points from the training dataset, training data sanitization reduces the attack surface and vulnerability of machine learning models to poisoning attacks. By only training on clean and trustworthy data, organizations can minimize the risk of models being manipulated or compromised by malicious actors. It can be implemented at scale

TABLE 7. Privacy attack countermeasures - strengths and weaknesses.

Countermeasures	Strengths	Weaknesses
Differential Privacy [11], [88], [124], [144]	Provides strong privacy guarantees by ensuring that the inclusion or exclusion of any individual data point does not significantly affect the output or results of the analysis. Offers a rigorous and principled approach to privacy protection with well-defined mathematical frameworks and formal guarantees. Enables the sharing and analysis of sensitive data while preserving the privacy of individual data contributors.	Introduces noise or randomness into the data, which can reduce the accuracy or utility of the analysis. Requires careful tuning of privacy parameters and trade-offs between privacy and utility, which may be challenging to optimize in practice. May not be suitable for all types of ML models or applications, particularly those with stringent accuracy or performance requirements.
Privacy Auditing [127] – [128], [132]	Provides a systematic approach to evaluating and assessing the privacy risks associated with ML models, datasets, or systems. Helps identify potential privacy vulnerabilities, compliance violations, or data leakage risks, enabling organizations to take appropriate measures to protect privacy. Facilitates ongoing monitoring and review of ML systems to ensure compliance with privacy regulations, standards, and best practices.	Relies on the availability of comprehensive privacy auditing tools, expertise, and resources, which may be limited or unavailable to some organizations. May require access to sensitive data or proprietary information, raising concerns about data confidentiality and security. Limited scalability and applicability to certain types of ML models or systems, particularly those with complex architectures or distributed components.
Private Aggregation of Teacher Ensembles (PATE) [133]	Enables the training of ML models on sensitive data without exposing individual-level information, preserving the privacy of data contributors. Provides a practical and scalable approach to privacy-preserving machine learning, particularly in scenarios where access to sensitive data is restricted. Offers strong privacy guarantees by aggregating predictions from multiple teacher models and introducing noise into the label aggregation process.	May introduce computational overhead and complexity, particularly when training large ensembles of teacher models or aggregating predictions from diverse sources. Requires careful selection and tuning of hyperparameters to balance privacy and utility, which may be challenging in practice. Vulnerable to attacks that exploit weaknesses in the aggregation process or leverage auxiliary information to infer sensitive data.

and integrated into existing data processing pipelines and workflows. Automated data validation and preprocessing techniques can help organizations efficiently manage and sanitize large volumes of training data, reducing the manual effort and cost associated with securing machine learning models against poisoning attacks [69]. Also protecting the data supply chain from manipulations is one way to mitigate attacks that alter or lower the caliber of training data. If data is taken from a sanitized setting, there is significantly less chance of inaccurate or manipulated information [50] While training data sanitization offers significant advantages in mitigating the risks associated with poisoning attacks, it is important to note that it is not an ultimate solution and should be complemented with other defense mechanisms such as model validation, input verification, and ongoing monitoring to provide comprehensive protection against evolving poisoning attacks.

Adversarial training has also proven effective against Poisoning attacks. Other techniques such as Byzantine-resilient aggregation, Model Pruning, Trigger Reconstruction, Robust Training, Ensemble Analysis, Model Watermarking have several limitations. They are not typically considered as standalone methods for mitigating poisoning attacks in ML models. While these methods can help identify weaknesses in ML models and inform the developers of issues within the models, they cannot typically be employed as direct mitigation techniques against poisoning attacks. They are either expensive, difficult to scale or compromise the accuracy and quality of the model. However, they can still play a valuable role in the broader context of ML security by providing insights into the vulnerabilities and limitations of the models and helping the developers create more robust and resilient defense mechanisms against poisoning attacks.

C. PRIVACY ATTACKS MITIGATION APPROACHES

Data reconstruction, memorization, membership inference, model extraction, and property inference are the five types of privacy attacks. All privacy attacks attempt to identify a possible model input given a model output and permissions to query the model. Attacks known as membership inference attacks try to determine if a given data sample has access to the training dataset or not through model querying rights. Although the attack strategies for each of the five types may slightly differ, the attacker's objective is always the same. The attacker is interested in retrieving confidential data from a set of pairs representing the input and output of the model, along with the corresponding output confidence levels.

Table 7 lists out the strengths and weaknesses of each countermeasure proposed against privacy attacks in the studies reviewed. Based on that, differential privacy, privacy audit and private aggregation of teacher ensembles (PATE) can all be considered as suitable strategies for mitigating privacy attacks. No one specific countermeasure can defend against all types of privacy attacks.

Differential Privacy: Differential privacy allows one to quantify the privacy guarantees that an algorithm offers. Adding randomness to an algorithm's behavior is the fundamental idea. Differential privacy in learning reduces the possibility of revealing sensitive information about training data and offers verifiable privacy guarantees. The behaviors of a trained model that has been trained with differential privacy are less influenced by any one training set. Due to this,

it is challenging to determine which data record by looking at the model's behavior belongs to the training dataset [16], [93], [127], [128]. It is also referred to as data augmentation or synthetic data generation. It is proven to be effective against membership inference attacks. By introducing additional noise and variability into the dataset, organizations can make it more challenging for adversaries to infer membership status based on model predictions. Techniques such as differential privacy-based data synthesis can be used to generate synthetic data that closely resembles the original dataset while preserving privacy and reducing the risk of membership inference attacks.

Privacy Auditing: A few studies proposed the scheme for auditing privacy of ML/AI systems. By conducting privacy audits, organizations can assess whether their models adhere to privacy principles and legal requirements, such as data minimization, purpose limitation, and user consent. These are helpful but the challenge is that they are usually manual effort driven. Tools such as IBM Privacy Toolkit [145] can be useful in this regard [101], [116], [148]. Organizations can leverage privacy audit tools to proactively manage privacy risks and reduce the likelihood of privacy attacks.

Private Aggregation of Teacher Ensembles (PATE): PATE integrates several models trained on different datasets. Regardless of the learning algorithm, PATE offers differential privacy for training data. Several studies have proposed PATE to protect both teacher's and students' data on ML models. Reference [149] introduced a new framework that combined secret sharing with Intel Software Guard Extensions to protect both teachers' and students' data. Reference [150] extended PATE to quantum ML, while [151] proposed PATE to improve the accuracy of the student model by integrating advanced noisy label training mechanisms. Reference [152] addressed privacy threats in deep reinforcement learning by developing a differentially private mechanism to protect the teacher's training dataset. These studies collectively contribute to the development of more robust and secure privacy-preserving machine learning techniques. Although PATE can withstand membership inference attacks with provable robustness, achieving it is difficult and results in minimal utility loss [117].

VI. RECENT FRAMEWORKS AND TOOLS FOR IMPROVING CYBER RESILIENCE AGAINST AML ATTACKS

This section elaborates on specific frameworks such as the MITRE ATLAS Framework [153], NIST AI Risk Management Framework [154], ETSI Securing AI Mitigation Strategy Framework [155] designed to address the broader aspects of ML security and ensure the secure deployment of ML models. These frameworks offer comprehensive guidelines and best practices to organizations, covering all stages of AI/ML model and application development. By following these frameworks, organizations can implement robust security measures throughout the development lifecycle, from initial design to deployment and maintenance. This proactive approach helps mitigate potential risks and vulnerabilities, ensuring that ML models and AI systems are resilient against AML threats. Furthermore, tools such as the Adversarial Robustness Toolbox [156], Microsoft Counterfit [157], IBM Privacy Toolkit [145] are recommended for integration into organizations' code pipelines, ensuring that secure code development practices are ingrained into the development process of AI/ML applications. By incorporating these frameworks and tools early in the development lifecycle, organizations can adopt a ''shift-left'' approach to security, emphasizing the importance of addressing security concerns at the earliest stages of development. This proactive stance enables teams to identify and mitigate potential security risks before they escalate, ultimately enhancing the overall security posture of AI/ML applications.

A. MITRE ATLAS FRAMEWORK

The MITRE ATLAS (Adversarial Tactics, Techniques, and Common Knowledge for Machine Learning) framework [153] is a comprehensive resource developed by MITRE Corporation to address the security challenges associated with ML systems. Just like the ATT&CK framework, the new ATLAS framework provides a structured approach for understanding, evaluating, and mitigating threats and vulnerabilities specific to ML models and systems. The framework catalogs adversarial tactics and techniques used by attackers to exploit vulnerabilities in ML systems. These tactics and techniques are organized into a taxonomy similar to the original MITRE ATT&CK framework, with categories such as evasion attacks, poisoning attacks and privacy attacks. By categorizing adversarial techniques, the framework helps organizations understand the tactics employed by attackers and develop appropriate defense strategies for each stage in an AI/ML Application lifecycle as presented in Figure 9. The framework also includes use case scenarios that illustrate real-world examples of adversarial attacks and security challenges in ML systems. These use cases provide context and insight into the types of threats that organizations may face and help stakeholders understand the impact of security vulnerabilities on ML applications. It provides best practices and mitigation strategies for securing ML systems against adversarial attacks and security threats. These include recommendations for secure model development, robust training practices, input validation and sanitization, model monitoring, and incident response. By implementing these best practices across all stages of building a ML application, organizations can reduce the likelihood of successful attacks and mitigate the impact of AML attacks. By providing a structured approach to understanding and mitigating security risks, the framework helps organizations build more resilient and secure ML applications that can withstand adversarial threats and protect sensitive data. The framework focuses on understanding the various stages of a cyber attack in the context of ML systems. These stages are: Business and Data Understanding, Data Preparation, ML Model Engineering, ML Model Evaluation, Deployment, Monitoring and

Maintenance. While the framework is primarily tailored to ML security, the stages of a cyber attack described within it are generally applicable to cybersecurity in a broader context. Here are the stages of a cyber attack as outlined by the MITRE ATLAS framework:

1) Reconnaissance (Recon): In this initial stage, adversaries gather information about the target ML system, including its architecture, datasets, algorithms, and potential vulnerabilities. Reconnaissance activities may involve scanning public resources, analyzing documentation, probing APIs, or monitoring network traffic to gain insight into the target environment.
2) Resource Development: Once adversaries have collected sufficient information about the target ML system, they develop or acquire tools and techniques to exploit identified vulnerabilities or weaknesses. It involves crafting malicious payloads, code snippets, or data inputs that can be used to compromise the target system and achieve the attacker's objectives.
3) Initial Access: This stage involves the adversary gaining an initial foothold or access to the ML system, which could be through vulnerabilities in software, misconfigurations, or social engineering techniques.
4) ML Model Access: After gaining initial access to the target environment, adversaries install backdoors, rootkits, or other persistent malware to maintain access and establish a foothold within the system. Installation activities may include creating user accounts, modifying system configurations, or deploying additional malicious payloads to achieve persistence and evade detection.
5) Execution: Once access is established, the adversary executes malicious code or commands to carry out their objectives, such as compromising ML models or manipulating training data.
6) Persistence: In this stage, the adversary ensures that their access to the ML system persists over time, often by establishing backdoors or other means of maintaining access even after detection or removal attempts.
7) Privilege Escalation: If the adversary's initial access does not provide sufficient privileges, they may attempt to escalate their privileges within the ML system to gain access to sensitive resources or capabilities.
8) Defense Evasion: Adversaries employ various techniques to evade detection by security measures or monitoring systems, such as obfuscating malicious code, bypassing anomaly detection mechanisms, or exploiting weaknesses in security controls.
9) Credential Access: Adversaries may attempt to obtain credentials, such as usernames and passwords, to gain access to ML systems or data repositories.
10) Discovery: In this stage, the adversary gathers information about the ML system, including its architecture, components, datasets, and vulnerabilities, to plan and execute their attack more effectively.
11) Collection: Adversaries collect data or information from the ML system, such as training data, model parameters, or sensitive information, for their malicious purposes.
12) ML Attack Staging: Once inside the ML system, the adversary may move laterally across the network or infrastructure to explore and exploit other systems or resources.
13) Exfiltration: Adversaries exfiltrate stolen data or information from the ML system to external servers or endpoints under their control for further exploitation or monetization.
14) Impact: Adversaries carry out their primary objectives, which may include disrupting ML operations, manipulating model outputs, or causing financial or reputational damage to organizations.

FIGURE 9. AML attack chain.

These stages of a cyber attack as illustrated in Figure 9 provide a structured process flow for understanding the lifecycle of an adversary's activities and guiding defensive strategies to detect, prevent, and respond to cyber threats targeting ML systems and other critical assets. The framework includes tools and techniques for analyzing the attack surface of ML systems, including identifying potential entry points and attack vectors that adversaries may exploit. By conducting attack surface analysis, organizations can gain insights into the security posture of their ML systems and prioritize security controls to mitigate potential risks. While addressing complex security challenges, the MITRE ATLAS framework emphasizes pragmatic solutions that are practical to implement and align with real-world constraints and requirements. It offers guidance that is accessible to a wide range of stakeholders, including researchers, developers, data scientists, security analysts, and policymakers, enabling organizations to adopt effective security practices

regardless of their level of expertise or resources. It is continuously updated to reflect emerging threats, evolving attack techniques, and new developments in ML security research. By staying current with the latest trends and advancements in the field, the framework remains relevant and provides organizations with up-to-date guidance on mitigating security risks in their ML systems. It provides a platform for researchers, practitioners, and organizations to contribute insights, best practices, and threat intelligence, fostering a collaborative ecosystem for addressing ML security challenges collectively. To summarize, the MITRE ATLAS ML Security framework serves as a valuable resource for organizations seeking to enhance the security of their ML systems. By providing a structured approach to understanding and mitigating security risks, the framework helps organizations build more resilient and secure ML applications that can withstand adversarial threats and protect sensitive data.

B. NIST AI RISK MANAGEMENT FRAMEWORK

The NIST AI Risk Management framework [154], has also been developed by the National Institute of Standards and Technology (NIST), provides guidance and best practices for managing the risks associated with AI systems throughout their life cycle. The framework is designed to help organizations identify, assess, prioritize, and mitigate risks related to AI technologies effectively. The framework emphasizes the importance of establishing clear governance structures and processes for managing AI risks within organizations. This includes defining roles and responsibilities, establishing risk management policies and procedures, and ensuring accountability and oversight at all levels of the organization. The framework provides guidance on identifying and cataloging the risks associated with AI systems, including technical, operational, legal, ethical, and societal risks. This involves conducting thorough assessments of AI system components, data sources, algorithms, models, and deployment environments to identify potential vulnerabilities and threats. The framework outlines methodologies and techniques for assessing the likelihood and impact of identified risks on AI systems and their stakeholders. This includes quantitative and qualitative risk assessment methods, such as risk matrices, scenario analysis, and probabilistic modeling, to prioritize risks based on their severity and potential consequences. The framework offers strategies and best practices for mitigating and controlling AI risks to an acceptable level. This includes implementing technical controls, security measures, and safeguards to reduce the likelihood of adverse events, as well as developing contingency plans and response procedures to mitigate the impact of incidents when they occur. It emphasizes the importance of effective communication and collaboration among stakeholders throughout the risk management process. This includes transparently communicating AI risks, vulnerabilities, and mitigation strategies to decision-makers, users, regulators, and other relevant parties to build trust and confidence in AI systems. It advocates for continuous monitoring and evaluation of AI systems to detect emerging risks, monitor changes in the threat landscape, and assess the effectiveness of risk mitigation measures over time. This involves implementing monitoring tools, metrics, and feedback mechanisms to track AI system performance, security posture, and compliance with risk management objectives. Finally, the framework recommends documenting risk management activities, decisions, and outcomes in comprehensive risk registers, reports, and documentation repositories. This enables organizations to maintain a clear audit trail of their risk management efforts, demonstrate compliance with regulatory requirements, and facilitate learning and knowledge sharing across the organization. The NIST AI Risk Management framework provides a structured approach for organizations to systematically identify, assess, prioritize, and mitigate risks associated with AI systems, helping them build trust, resilience, and confidence in their AI capabilities.

C. IBM RESEARCH - ADVERSARIAL ROBUSTNESS TOOLBOX

Adversarial robustness toolbox (ART) [156] is an open-source Python library developed by IBM Research to help researchers and practitioners evaluate and improve the robustness of ML models against adversarial attacks. The toolbox provides a comprehensive set of tools and techniques for generating adversarial examples, evaluating model robustness, and implementing defense mechanisms against adversarial attacks. ART includes various algorithms for generating adversarial examples, which are carefully crafted inputs designed to deceive machine learning models and cause misclassifications. These algorithms include fast gradient sign method (FGSM), projected gradient descent (PGD), deepfool, carlini-wagner, and others. Users can choose from a range of attack strategies based on their specific requirements and use cases. It supports adversarial training, a defense mechanism aimed at enhancing model robustness by incorporating adversarial examples into the training process. Adversarial training involves augmenting the training dataset with adversarial examples or incorporating adversarial perturbations during model training to improve the model's ability to withstand adversarial attacks. The toolbox provides metrics and evaluation techniques for assessing the robustness of machine learning models against adversarial attacks. Users can measure model performance under attack using metrics such as accuracy, robustness, perturbation magnitude, and success rate of adversarial attacks. ART also supports adversarial evaluation on different datasets and benchmarking against state-of-the-art models. It offers a range of defense mechanisms and countermeasures for mitigating the impact of adversarial attacks on machine learning models. These defenses include adversarial training, input preprocessing techniques (e.g., feature squeezing, spatial smoothing), model distillation, randomization, and adversarial detection methods (e.g., detection using statistical tests, density estimation, or generative models). ART

supports model interpretability techniques to help users understand and interpret model behavior, especially in the presence of adversarial attacks. These techniques include visualizing model decision boundaries, feature importance, and adversarial perturbations to gain insights into how models make predictions and identify vulnerabilities to adversarial manipulation. The toolbox is compatible with popular deep learning frameworks such as TensorFlow, Keras, PyTorch, and scikit-learn, making it easy to integrate with existing machine learning pipelines and workflows. Users can seamlessly incorporate ART into their projects and leverage its functionalities without significant modifications to their codebase. In general, the Adversarial Robustness Toolbox (ART) provides a comprehensive set of tools and techniques for assessing, improving, and defending against adversarial attacks in machine learning models. By using ART, developers and researchers can better understand the vulnerabilities of their models, develop more robust machine learning systems, and enhance the security and reliability of AI applications in practice. Organizations can as well use ART to gain assurance of the defense posture of their AI and ML applications against AML attacks.

D. MICROSOFT - COUNTERFIT

Microsoft Counterfit [158] is another open-source AML testing toolkit developed by Microsoft Research to help developers and security professionals evaluate the robustness of ML models against AML attacks. The tool is designed to simulate various attack scenarios and generate adversarial examples to assess the security and resilience of ML models in real-world settings. Counterfit provides a range of attack algorithms and techniques for generating adversarial examples that can fool ML models into making incorrect predictions. These attacks include gradient-based methods like FGSM (Fast Gradient Sign Method) and PGD (Projected Gradient Descent), optimization-based attacks like C&W (Carlini and Wagner), and model inversion attacks. Users can configure various parameters and settings for adversarial attacks, such as the attack type, attack strength, targeted or untargeted attacks, and constraints on adversarial perturbations. Counterfit offers flexibility in customizing attack scenarios to suit different use cases and requirements. The toolkit enables users to evaluate the robustness of ML models against adversarial attacks by generating adversarial examples and measuring model performance under attack. Users can assess metrics such as accuracy, robustness, success rate of attacks, and perturbation magnitude to quantify the impact of adversarial manipulation on model behavior. Counterfit supports testing and validation of adversarial defense mechanisms and countermeasures designed to enhance the security and resilience of ML models against adversarial attacks. Users can evaluate the effectiveness of defenses such as adversarial training, input preprocessing, model distillation, and adversarial detection methods. Counterfit is integrated with Azure AI services, allowing users to test ML models deployed on Azure cloud infrastructure. Users can leverage Counterfit to assess the security posture of ML models hosted on Azure ML, Azure Kubernetes Service (AKS), or other Azure services and environments. The toolkit provides a user-friendly command-line interface (CLI) and interactive dashboard for easy configuration, execution, and visualization of adversarial attacks and model evaluation results. Users can interact with Counterfit using simple commands and workflows, making it accessible to both developers and security practitioners. Counterfit is open-source and actively maintained by Microsoft Research, with contributions from the developer and cybersecurity communities. Users can collaborate, share knowledge, and contribute to the development of Counterfit through GitHub repositories, forums, and community-driven initiatives. To summarize, Microsoft Counterfit is a powerful and versatile toolkit for assessing the security and robustness of ML models against adversarial attacks. By using Counterfit, developers and security professionals can identify vulnerabilities, test defense mechanisms, and improve the resilience of ML systems against emerging cyber threats in today's increasingly adversarial landscape.

E. IBM - AI PRIVACY TOOLKIT

The AI Privacy Toolkit (APT) [145] by IBM is an open-source toolkit designed to help developers and data scientists address privacy concerns and comply with regulations when working with AI and ML technologies. The toolkit provides a set of tools, libraries, and resources to assist in implementing privacy-preserving techniques and evaluating privacy risks associated with AI and ML models. APT offers a variety of privacy-preserving techniques and algorithms that developers can integrate into their AI and ML pipelines. These techniques include differential privacy, federated learning, homomorphic encryption, secure multiparty computation, and data anonymization methods. The toolkit provides tools and libraries for assessing the privacy risks associated with AI and ML models. Developers can use these tools to analyze model architectures, data flows, and model outputs to identify potential privacy vulnerabilities and compliance issues. APT includes tools for anonymizing and de-identifying sensitive data to protect individual privacy. Developers can use these tools to mask or remove personally identifiable information (PII) from datasets while preserving the utility and integrity of the data for analysis and model training. It also offers metrics and evaluation criteria for quantifying the privacy risks and vulnerabilities of AI and ML models. Developers can use these metrics to assess the effectiveness of privacy-preserving techniques and compare the privacy performance of different models and algorithms. The toolkit provides guidance and resources on privacy regulations, standards, and best practices for AI and ML applications. Developers can use these resources to ensure compliance with regulations such as the General Data Protection Regulation (GDPR), Health Insurance Portability and Accountability Act (HIPAA), and other privacy laws and regulations. APT includes tools for interpreting and

TABLE 8. Applicability of framework and tools on AML attacks.

S.No.	Framework / Tool	Purpose	Attacks Coverage	Strengths	Limitations
1.	MITRE ATLAS Framework [16]	Designed to provide a structured approach to evaluate and improve the resilience of ML models against adversarial attacks. Includes methodologies, tools, and datasets to facilitate research and development in AML.	Evasion, Poisoning and Privacy Attacks	Comprehensive framework covering a wide range of adversarial techniques and defenses and provides guidelines and resources for both offensive (attack) and defensive (defense) strategies.	Requires a deep understanding of AML to effectively utilize the framework.
2.	NIST AI Risk Management Framework [159]	NIST AI Risk Management Framework provides guidance on managing risks associated with the deployment and operation of AI systems.	Evasion, Poisoning and Privacy Attacks	Systematic approach to assessing and managing risks across the AI lifecycle.	Requires adaptation to specific organizational contexts, emerging AI threats and regulatory environments.
3.	Adversarial Robustness Toolbox (ART) [156]	ART provides a collection of tools and techniques to test and improve the robustness of ML models against adversarial attacks.	Evasion, Poisoning and Privacy Attacks	Offers a wide range of attack methods and defense techniques (e.g., adversarial training, defensive distillation). Includes functionalities for evaluating model robustness, generating adversarial examples, and implementing defenses.	Requires expertise in machine learning and adversarial techniques for effective use and the performance impact of defenses on model accuracy and efficiency needs careful consideration.
4.	Microsoft - Counterfit [157]	Counterfit is a tool designed for adversarial simulation and testing of AI systems. It provides capabilities to generate and deploy adversarial attacks against ML models to assess their robustness and security posture.	Evasion and Poisoning Attacks	User-friendly interface and tools for creating and executing adversarial attacks and supports a variety of attack scenarios and techniques, including evasion and poisoning attacks.	Limited to Microsoft Azure environment and integration with specific Microsoft tools.
5.	AI Privacy Toolkit [145]	Focuses on preserving privacy and confidentiality in AI systems. Provides methods and tools to assess and mitigate privacy risks associated with AI technologies, including techniques for data anonymisation, differential privacy, and model privacy	Privacy Attacks	Offers practical solutions and algorithms for enhancing privacy in AI applications. Supports compliance with privacy regulations and standards.	Requires expertise in data privacy regulations and techniques for effective implementation. The applicability of privacy-preserving methods may vary depending on the specific use case and data environment.

explaining the decisions and predictions of AI and ML models. Developers can use these tools to understand how models make predictions and assess the potential impact of model decisions on individual privacy rights. It also integrates with popular AI development platforms and frameworks, such as TensorFlow, PyTorch, and scikit-learn, making it easy to incorporate privacy-preserving techniques into AI and ML workflows. Developers can seamlessly integrate APT into their existing development environments and workflows without significant modifications. Therefore, the AI Privacy Toolkit (APT) by IBM provides a comprehensive set of tools, libraries, and resources to help developers and data scientists address privacy concerns and compliance requirements when working with AI and ML technologies. By using APT, organizations can enhance the security and privacy of their AI applications while ensuring compliance with privacy regulations and standards.

With this, the researchers conclude the enumeration of beneficial open-source security and privacy tools that organizations can leverage to secure their ML models and AI applications. They strongly advocate for organizations to remain vigilant against AML attacks. Through thorough security testing and validation processes, organizations can fortify the security and privacy posture of their ML models

and AI applications, thereby safeguarding sensitive data. By integrating these frameworks and tools into coding and testing pipelines, developers can swiftly identify and rectify issues without manual intervention. This streamlined approach not only expedites the development cycle but also empowers organizations to deliver AI solutions to market expeditiously, granting them a competitive advantage in the industry. The benefits and limitations of each framework and tool are highlighted in Table 8.

VII. KEY RESEARCH CHALLENGES AND LIMITATIONS

Research on adversarial machine learning (AML) attacks poses several key research challenges, including:

- *Understanding Adversarial Attacks*: A thorough understanding of various adversarial attack tactics and techniques and their implications on ML models is essential. It required delving into the intricacies of each attack, including evasion, poisoning, and privacy attacks, to accurately analyze their impact and devise effective countermeasures.
- *Designing Robust Countermeasures*: Designing robust defense mechanisms to mitigate the effects of AML attacks is another significant challenge. It is difficult to test countermeasures such as adversarial training, training data sanitization, differential privacy, and model robustness enhancements on real-world ML models as they require computational power and real datasets. However, it's important to test these to bolster the resilience of ML models against adversarial manipulation and gain security and privacy assurance.
- *Evaluating Adaptive Attacks and Countermeasures*: Evaluating the effectiveness of both AML attacks and defense strategies requires rigorous experimentation and analysis. It requires assessing the countermeasures under various adaptive attack scenarios.
- *Evolving AML Threat Landscape*: The threat landscape surrounding AML attacks has evolved rapidly, especially after the release of commercial large language models (LLMs). One of the key factors contributing to this changing landscape is the increasing sophistication of adversaries, who continuously devise novel attack techniques to exploit vulnerabilities in ML models. Stronger attack algorithms are constantly being developed.

These present a significant challenge for researching on AML. It's a difficult task of keeping pace with emerging threats and understanding their implications on ML systems. However, this further necessitates ongoing research efforts to explore novel AML attack tactics and techniques, evaluating the effectiveness of defense mechanisms, and improving their efficacy to mitigate the impact.

While this paper aims to provide a comprehensive overview of AML, it is important to acknowledge certain limitations inherent in the scope and methodology of this study.

- *Scope*: The scope of this research has been limited to identifying and discussing key concepts of AML. Due to the vastness of the field and the rapid pace of advancements, it is not possible to cover every aspect of AML comprehensively.
- *Practical evaluation of attack countermeasures*: The focus of this paper has been primarily on the three types of AML attacks - evasion, poisoning and privacy and their defense measures. The research performed a theoretical analysis of each defense measures and identified its strength and weaknesses. However, these have not been practically tested.
- *Various ML architectures*: The proposed countermeasures for addressing AML attacks may not be applicable to all ML models and may require adaptation based on specific type and constraints.
- *Evolving AML Attack Techniques*: While this paper provides insights into some tactics and techniques used by attackers to execute an AML attack, the field of AML is constantly evolving. Continuous monitoring and adaptation are essential to stay abreast of emerging attacker techniques in AML.

Despite these limitations, this paper provides a valuable contribution to the understanding of key research challenges in AML and lays the groundwork for future research endeavors in this area. By acknowledging these limitations, researchers can work towards addressing them and advancing the state-of-the-art in AML research.

VIII. CONCLUSION AND FUTURE SCOPE OF RESEARCH

With the rising utilization of ML models and AI applications, particularly in critical sectors like autonomous transportation and healthcare, AML attacks have emerged as a significant area of concern. These sophisticated attacks have the potential to disrupt ML models and AI applications, leading to severe health and safety implications. This study systematically surveyed AML attacks based on the attacker goals and suggested countermeasures for various types, including evasion, poisoning, and privacy attacks. Table 9 summarizes the AML attack types, goals and countermeasures recommended in this research.

This research also offered recommendations for the secure deployment of ML models within organizational settings, outlining detailed mitigation strategies against evasion, poisoning, and privacy attacks. While some mitigation strategies proved effective, many were susceptible to stronger variants of attacks. These were carefully analysed and their merits and limitations addressed. Furthermore, the researchers evaluated specific open-source security and privacy frameworks and tools to assist organizations in testing and securing their ML models and AI applications. Given the observable trend among threat actors towards designing more potent and stealthier AML attacks, maintaining robust

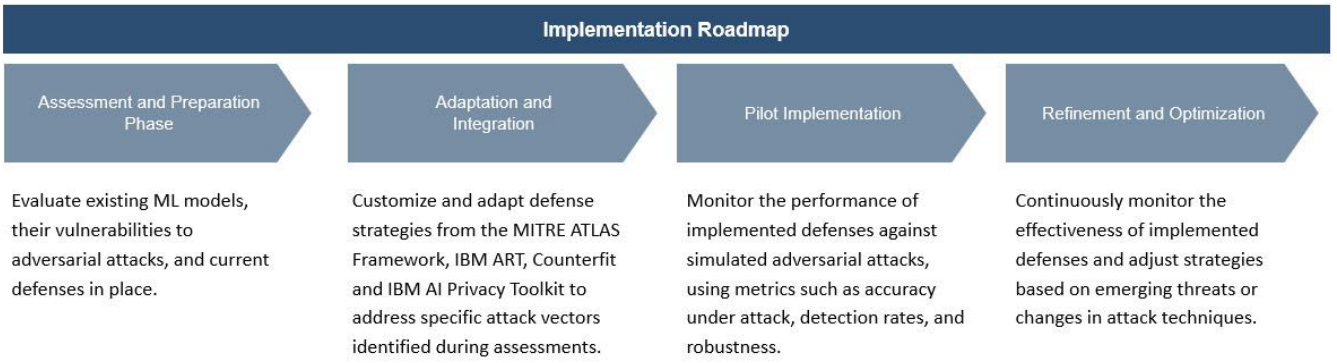

FIGURE 10. Implementation roadmap.

TABLE 9. Summary of AML attacks.

S.No.	Attack Type	Attacker Goal	Countermeasures
1.	Evasion Attacks	To manipulate input data in a way that causes a ML model to make incorrect predictions.	Adversarial Training, Formal Verification
2.	Poisoning Attacks	To manipulate the training data used to train a model, compromising its performance or introducing vulnerabilities.	Training Data Sanitization, Adversarial Training
3.	Privacy Attacks	To extract sensitive information from models or training data, compromising confidentiality of the system.	Differential Privacy, Privacy Auditing, PATE

security and privacy controls based on AI and ML security frameworks and tools is imperative. The researchers intend to further explore the practical implementation of the proposed countermeasures against AML attacks using the identified set of frameworks and tools as a future scope of this research. Figure 10 illustrates the roadmap for implementing countermeasures against AML attacks using the tools discussed.

REFERENCES

[1] S. Raza, M. Garg, D. J. Reji, S. R. Bashir, and C. Ding, "Nbias: A natural language processing framework for BIAS identification in text," *Expert Syst. Appl.*, vol. 237, Mar. 2024, Art. no. 121542.

[2] W. Wang, Z. Chen, X. Chen, J. Wu, X. Zhu, G. Zeng, P. Luo, T. Lu, J. Zhou, Y. Qiao, and J. Dai, "VisionLLM: Large language model is also an open-ended decoder for vision-centric tasks," in *Advances in Neural Information Processing Systems*, vol. 36, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., Red Hook, NY, USA: Curran Associates, 2023, pp. 61501–61513.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[4] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017, *arXiv:1706.06083*.

[5] A. Mitra, A. Jain, A. Kishore, and P. Kumar, "A comparative study of demand forecasting models for a multi-channel retail company: A novel hybrid machine learning approach," *Operations Res. Forum*, vol. 3, no. 4, p. 58, Sep. 2022.

[6] R. S. Siva Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia, "Adversarial machine learning-industry perspectives," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, May 2020, pp. 69–75.

[7] D. Pavithra, "A study on machine learning algorithm in medical diagnosis," *Int. J. Adv. Res. Comput. Sci.*, vol. 9, no. 4, pp. 42–46, Aug. 2018.

[8] M. Elbattah and O. Molloy, *Analytics Using Machine Learning-Guided Simulations With Application to Healthcare Scenarios*. New York, NY, USA: Taylor & Francis, 2018.

[9] M. Hashem Eiza and Q. Ni, "Driving with sharks: Rethinking connected vehicles with vehicle cybersecurity," *IEEE Veh. Technol. Mag.*, vol. 12, no. 2, pp. 45–51, Jun. 2017.

[10] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning visual classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1625–1634.

[11] *Cylance Anti-Virus Products Suspectible to Concatenation Bypass*. Accessed: Feb. 29, 2024. [Online]. Available: https://www.kb.cert.org/vuls/id/489481/

[12] (2019). *Adversarial Machine Learning Against Tesla's Autopilot*. [Online]. Available: https://www.schneier.com/blog/archives/2019/04/adversarial_mac.html

[13] Business Insider. *Hackers Steered a Tesla Into Oncoming Traffic by Placing Three Small Stickers on the Road | Business Insider India*. Accessed: May 18, 2024. [Online]. Available: https://www.businessinsider.in/finance/hackers-steered-a-tesla-into-oncoming-traffic-by-placing-three-small-stickers-on-the-road/articleshow/68678701.cms

[14] *ProofPoint Evasion*. Accessed: May 18, 2024. [Online]. Available: https://atlas.mitre.org/studies/AML.CS0008

[15] *Adversarial Machine Learning: Attacks From Laboratories to the Real World*. Accessed: Dec. 22, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/9426997

[16] MITRE | ATLAS. *Tay Poisoning*. Accessed: Dec. 22, 2023. [Online]. Available: https://atlas.mitre.org/studies/AML.CS0009/

[17] *Tay Poisoning*. Accessed: May 18, 2024. [Online]. Available: https://atlas.mitre.org/studies/AML.CS0009

[18] G. Apruzzese, H. S. Anderson, S. Dambra, D. Freeman, F. Pierazzi, and K. A. Roundy, "'Real attackers don't compute gradients': Bridging the gap between adversarial ML research and practice," 2022, *arXiv:2212.14315*.

[19] R. S. S. Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia, "Adversarial machine learning - industry perspectives," 2020, *arXiv:2002.05646*.

[20] F. Boenisch, V. Battis, N. Buchmann, and M. Poikela, "'I never thought about securing my machine learning systems': A study of security and privacy awareness of machine learning practitioners," in *Proc. Mensch Comput.*, 2021, pp. 520–546, doi: 10.1145/3473856.3473869.

[21] Z. Sun, R. Sun, L. Lu, and A. Mislove, "Mind your weight(s): A large-scale study on insufficient machine learning model protection in mobile apps," in *Proc. 30th USENIX Security Symp.*, 2021, pp. 1955–1972.

[22] L. Bieringer, K. Grosse, M. Backes, B. Biggio, and K. Krombholz, "Mental models of adversarial machine learning," 2021, *arXiv:2105.03726*.

[23] K. Grosse, L. Bieringer, T. R. Besold, B. Biggio, and K. Krombholz, "Machine learning security in industry: A quantitative survey," 2022, *arXiv:2207.05164*.
[24] R. R. Wiyatno, A. Xu, O. Dia, and A. de Berker, "Adversarial examples in modern machine learning: A review," 2019, *arXiv:1911.05268*.
[25] J. Gilmer, R. P. Adams, I. Goodfellow, D. Andersen, and G. E. Dahl, "Motivating the rules of the game for adversarial example research," 2018, *arXiv:1807.06732*.
[26] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognit.*, vol. 84, pp. 317–331, Dec. 2018.
[27] A. Kurakin et al., "Adversarial attacks and defences competition," in *Adversarial Machine Learning*. Cham, Switzerland: Springer, 2018, pp. 195–231.
[28] N. Carlini and D. Wagner, "Defensive distillation is not robust to adversarial examples," 2016, *arXiv:1607.04311*.
[29] S. Guo, J. Zhao, X. Li, J. Duan, D. Mu, and X. Jing, "A black-box attack method against machine-learning-based anomaly network flow detection models," *Secur. Commun. Netw.*, vol. 2021, pp. 1–13, Apr. 2021.
[30] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "BadNets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47230–47244, 2019.
[31] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, "Trojaning attack on neural networks," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2018.
[32] P. Kairouz et al., "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*.
[33] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, 2015.
[34] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2014.
[35] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 3–18.
[36] I. Dinur and K. Nissim, "Revealing information while preserving privacy," in *Proc. 22nd ACM SIGMOD-SIGACT-SIGART Symp. Princ. Database Syst.*, Jun. 2003, pp. 202–210.
[37] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," 2016, *arXiv:1611.02770*.
[38] *NIST AI 100: Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations*. Accessed: Dec. 22, 2023. [Online]. Available: https://doi.org/10.6028/NIST.AI.100-2e2023.ipd
[39] M. Kearns and M. Li, "Learning in the presence of malicious errors," in *Proc. 20th Annu. ACM Symp. Theory Comput. (STOC)*, New York, NY, USA, 1988, pp. 267–280.
[40] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma, "Adversarial classification," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2004, pp. 99–108.
[41] D. Lowd and C. Meek, "Adversarial learning," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*. New York, NY, USA: Association for Computing Machinery, Aug. 2005, pp. 641–647.
[42] N. Carlini and D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, May 2018, pp. 1–7.
[43] R. Jia and P. Liang, "Adversarial examples for evaluating reading comprehension systems," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Copenhagen, Denmark, 2017, pp. 2021–2031.
[44] S. Li, A. Neupane, S. Paul, C. Song, S. V. Krishnamurthy, A. K. Roy Chowdhury, and A. Swami, "Adversarial perturbations against real-time video classification systems," 2018, *arXiv:1807.00458*.
[45] X. Wei, J. Zhu, S. Yuan, and H. Su, "Sparse adversarial perturbations for videos," in *Proc. 33rd AAAI Conf. Artif. Intell. 31st Innov. Appl. Artif. Intell. Conf. 9th AAAI Symp. Educ. Adv. Artificial Intell.* AAAI Press, 2019, pp. 8973–8980.
[46] *Microsoft Edge AI Evasion*. Accessed: May 18, 2024. [Online]. Available: https://atlas.mitre.org/studies/AML.CS0011
[47] *Face Identification System Evasion via Physical Countermeasures*. Accessed: May 18, 2024. [Online]. Available: https://atlas.mitre.org/studies/AML.CS0012
[48] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019.
[49] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, New York, NY, USA, Nov. 2017, pp. 3–14.
[50] F. Tramer, "Detecting adversarial examples is (nearly) as hard as classifying them," in *Proc. 39th Int. Conf. Mach. Learn.*, vol. 162, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., Jul. 2022, pp. 21692–21702.
[51] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 274–283.
[52] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Francisco, CA, USA, May 2019, pp. 656–672.
[53] J. Cohen, E. Rosenfeld, and Z. Kolter, "Certified adversarial robustness via randomized smoothing," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1310–1320.
[54] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *Computer Aided Verification*, R. Majumdar and V. Kuncak, Eds., Cham, Switzerland: Springer, 2017, pp. 97–117.
[55] G. Singh, T. Gehr, M. Puschel, and M. Vechev, "An abstract domain for certifying neural networks," in *Proc. ACM Program. Lang.*, Jan. 2019.
[56] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Formal security analysis of neural networks using symbolic intervals," in *Proc. 27th USENIX Secur. Symp. (USENIX Security)*. Baltimore, MD, USA: USENIX Association, 2018, pp. 1599–1614.
[57] A. Fromherz, K. Leino, M. Fredrikson, B. Parno, and C. Pasareanu, "Fast geometric projections for local robustness certification," in *Proc. Int. Conf. Learn. Represent.*, 2021.
[58] C. Li, A. Dakkak, J. Xiong, and W.-m. Hwu, "The design and implementation of a scalable DL benchmarking platform," 2019, *arXiv:1911.08031*.
[59] N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, Ú. Erlingsson, A. Oprea, and C. Raffel, "Extracting training data from large language models," in *Proc. 30th USENIX Secur. Symp.*, 2021, pp. 2633–2650.
[60] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *6th Int. Conf. Learn. Represent. (ICLR)*, Vancouver, BC, Canada, Apr. 2018.
[61] B. Balle, G. Cherubin, and J. Hayes, "Reconstructing training data with informed adversaries," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2022, pp. 1138–1156.
[62] *Huawei AI Security Whitepaper*. Accessed: Dec. 22, 2023. [Online]. Available: https://www-file.huawei.com/-/media/corporate/pdf/trust-center/ai-security-whitepaper.pdf
[63] M. Bak, V. I. Madai, M.-C. Fritzsche, M. T. Mayrhofer, and S. McLennan, "You Can't have AI both ways: Balancing health data privacy and access fairly," *Frontiers Genet.*, vol. 13, Jun. 2022.
[64] N. Papernot and P. McDaniel, "On the effectiveness of defensive distillation," 2016, *arXiv:1607.05113*.
[65] Y. Ji, B. Bowman, and H. H. Huang, "Securing malware cognitive systems against adversarial attacks," in *Proc. IEEE Int. Conf. Cognit. Comput. (ICCC)*, Jul. 2019, pp. 1–9.
[66] A. Alotaibi and M. A. Rassam, "Adversarial machine learning attacks against intrusion detection systems: A survey on strategies and defense," *Future Internet*, vol. 15, no. 2, p. 62, Jan. 2023.
[67] R. Perdisci, D. Dagon, W. Lee, P. Fogla, and M. Sharif, "Misleading worm signature generators using deliberate noise injection," in *Proc. IEEE Symp. Secur. Privacy (S&P)*, Berkeley, CA, USA, May 2006, pp. 15–31.
[68] A. Paudice, L. Mu noz-González, and E. C. Lupu, "Label sanitization against label flipping poisoning attacks," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, C. Alzate, A. Monreale, H. Assem, A. Bifet, T. S. Buda, B. Caglayan, B. Drury, E. García-Martín, R. Gavalda, S. Kramer, N. Lavesson, M.l Madden, I. Molloy, M.-I. Nicolae, and M. Sinn, Eds. Cham, Switzerland: Springer, 2018, pp. 5–15.

[69] S. Venkatesan, H. Sikka, R. Izmailov, R. Chadha, A. Oprea, and M. J. de Lucia, "Poisoning attacks and data sanitization mitigations for machine learning models in network intrusion detection systems," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Nov. 2021, pp. 874–879.

[70] C. Sabottke, O. Suciu, and T. Dumitras, "Vulnerability disclosure in the age of social media: Exploiting Twitter for predicting real-world exploits," in *Proc. 24th USENIX Security Symp. (USENIX Security)*, Washington, DC, USA, 2015, pp. 1041–1056.

[71] G. Severi, J. Meyer, S. Coull, and A. Oprea, "Explanation-guided backdoor poisoning attacks against malware classifiers," in *Proc. USENIX Security Symp.*, 2021.

[72] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, and F. Roli, "Is feature selection secure against training data poisoning?" in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1689–1698.

[73] J. Geiping, L. H. Fowl, W. R. Huang, W. Czaja, G. Taylor, M. Moeller, and T. Goldstein, "Witches' brew: Industrial scale data poisoning via gradient matching," in *Proc. Int. Conf. Learn. Represent.*, 2021.

[74] X. Chen, A. Salem, D. Chen, M. Backes, S. Ma, Q. Shen, Z. Wu, and Y. Zhang, "BadNL: Backdoor attacks against NLP models with semantic-preserving improvements," in *Proc. Annu. Comput. Secur. Appl. Conf.*, Dec. 2021, pp. 554–569.

[75] S. Li, H. Liu, T. Dong, B. Z. H. Zhao, M. Xue, H. Zhu, and J. Lu, "Hidden backdoors in human-centric language models," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2021, pp. 3123–3140.

[76] E. Wallace, T. Zhao, S. Feng, and S. Singh, "Concealed data poisoning attacks on NLP models," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Human Lang. Technol.*, 2021.

[77] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 19–35.

[78] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. Roli, "Towards poisoning of deep learning algorithms with back-gradient optimization," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, Nov. 2017, pp. 27–38.

[79] J. Steinhardt, P. W. W. Koh, and P. S. Liang, "Certified defenses for data poisoning attacks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2017.

[80] I. Diakonikolas, G. Kamath, D. Kane, J. Li, J. Steinhardt, and A. Stewart, "Sever: A robust meta-algorithm for stochastic optimization," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1596–1606.

[81] E. Rosenfeld, E. Winston, P. Ravikumar, and Z. Kolter, "Certified robustness to label-flipping attacks via randomized smoothing," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 8230–8241.

[82] Y. Ma, X. Zhu, and J. Hsu, "Data poisoning against differentially-private learners: Attacks and defenses," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019.

[83] P. Benz, C. Zhang, S. Ham, G. Karjauv, A. Cho, and I. S. Kweon, "The triangular trade-off between accuracy, robustness, and fairness," in *Proc. Workshop Adversarial Mach. Learn. Real-World Comput. Vis. Syst. Online Challenges (AML-CV)*, 2021.

[84] A. Levine and S. Feizi, "Deep partition aggregation: Provable defenses against general poisoning attacks," in *Proc. 9th Int. Conf. Learn. Represent.*, May 2021.

[85] J. Hayase, W. Kong, R. Somani, and S. Oh, "Spectre: Defending against backdoor attacks using robust statistics," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 18–24.

[86] N. Peri, N. Gupta, W. R. Huang, L. Fowl, C. Zhu, S. Feizi, T. Goldstein, and J. P. Dickerson, "Deep k-NN defense against clean-label data poisoning attacks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2020, pp. 55–70.

[87] L. Batina, S. Bhasin, D. Jap, and S. Picek, "CSI NN: Reverse engineering of neural network architectures through electromagnetic side channel," in *Proc. USENIX Secur. Symp.*, 2019, pp. 515–532.

[88] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Francisco, CA, USA, May 2019, pp. 707–723.

[89] X. Hu, X. Lin, M. Cogswell, Y. Yao, S. Jha, and C. Chen, "Trigger hunting with a topological prior for trojan detection," in *Proc. Int. Conf. Learn. Represent.*, 2022.

[90] Y. Liu, W.-C. Lee, G. Tao, S. Ma, Y. Aafer, and X. Zhang, "ABS: Scanning neural networks for back-doors by artificial brain stimulation," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, Nov. 2019, pp. 1265–1282.

[91] X. Huang, M. Alzantot, and M. Srivastava, "NeuronInspect: Detecting backdoors in neural networks via output explanations," 2019, *arXiv:1911.07399*.

[92] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "DeepInspect: A black-box trojan detection and mitigation framework for deep neural networks," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 4658–4664.

[93] X. Xu, Q. Wang, H. Li, N. Borisov, C. A. Gunter, and B. Li, "Detecting AI trojans using meta neural analysis," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2021, pp. 103–120.

[94] D. Wu and Y. Wang, "Adversarial neuron pruning purifies backdoored deep models," in *Proc. Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, 2021, pp. 16913–16925.

[95] Y. Zeng, S. Chen, W. Park, Z. Mao, M. Jin, and R. Jia, "Adversarial unlearning of backdoors via implicit hypergradient," in *Proc. Int. Conf. Learn. Represent.*, 2022.

[96] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "FLTrust: Byzantine-robust federated learning via trust bootstrapping," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2021.

[97] R. Guerraoui, A. Guirguis, J. Plassmann, A. Ragot, and S. Rouault, "GARFIELD: System support for Byzantine machine learning (Regular Paper)," in *Proc. 51st Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2021, pp. 39–51.

[98] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "Distributed momentum for Byzantine-resilient stochastic gradient descent," in *Proc. ICLR*, 2021.

[99] J. Sun, A. Li, L. DiValentin, A. Hassanzadeh, Y. Chen, and H. Li, "FL-WBC: Enhancing robustness against model poisoning attacks in federated learning from a client perspective," in *Proc. NeurIPS*, 2021.

[100] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. ICML*, 2018.

[101] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. AISTATS*, 2020, pp. 2938–2948.

[102] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. USENIX Secur.*, 2020.

[103] V. Shejwalkar and A. Houmansadr, "Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2021.

[104] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in *Proc. Asian Conf. Mach. Learn.*, vol. 20, Taiwan, C.-N. Hsu and W. S. Lee, Eds., Nov. 2015, pp. 97–112.

[105] L. Fowl, P.-Y. Chiang, M. Goldblum, J. Geiping, A. Bansal, W. Czaja, and T. Goldstein, "Preventing unauthorized use of proprietary data: Poisoning for secure dataset release," 2021, *arXiv:2103.02683*.

[106] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017, *arXiv:1712.05526*.

[107] A. Salem, R. Wen, M. Backes, S. Ma, and Y. Zhang, "Dynamic backdoor attacks against machine learning models," 2020, *arXiv:2003.03675*.

[108] S. Li, M. Xue, B. Zhao, H. Zhu, and X. Zhang, "Invisible backdoor attacks on deep neural networks via steganography and regularization," *IEEE Trans. Dependable Secure Comput.*, vol. 18, pp. 2088–2105, 2021.

[109] Y. Liu, X. Ma, J. Bailey, and F. Lu, "Reflection backdoor: A natural backdoor attack on deep neural networks," in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Cham, Switzerland: Springer, 2020, pp. 182–199.

[110] E. Wenger, J. Passananti, A. N. Bhagoji, Y. Yao, H. Zheng, and B. Y. Zhao, "Backdoor attacks against deep learning systems in the physical world," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6202–6211.

[111] C. Shi, T. Zhang, Z. Li, H. Phan, T. Zhao, Y. Wang, J. Liu, B. Yuan, and Y. Chen, "Audio-domain position-independent backdoor attack via unnoticeable triggers," in *Proc. 28th Annu. Int. Conf. Mobile Comput. Netw.*, New York, NY, USA, Oct. 2022, pp. 583–595.

[112] *VirusTotal Poisoning*. Accessed: May 18, 2024. [Online]. Available: https://atlas.mitre.org/studies/AML.CS0002

[113] S. Bahadoripour, H. Karimipour, A. N. Jahromi, and A. Islam, "An explainable multi-modal model for advanced cyber-attack detection in industrial control systems," *Internet Things*, vol. 25, Apr. 2024, Art. no. 101092.

[114] *National Security Commission on Artificial Intelligence*, NSCAI, Washington, DC, USA, 2021.

[115] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," 2018, *arXiv:1811.03728*.
[116] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *Proc. 21st Int. Symp. Res. Attacks, Intrusions, Defenses*, in Lecture Notes in Computer Science, M. Bailey, S. Ioannidis, M. Stamatogiannakis, and T. Holz, Eds., Cham, Switzerland: Springer, 2018, pp. 273–294.
[117] D. Alistarh, Z. Allen-Zhu, and J. Li, "Byzantine stochastic gradient descent," in *Proc. NeurIPS*, 2018.
[118] P. Blanchard, E. M. E. Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. NeurIPS*, 2017, pp. 118–128.
[119] M. Rigaki and S. Garcia, "A survey of privacy attacks in machine learning," *ACM Comput. Surv.*, vol. 56, no. 4, pp. 1–34, Apr. 2024.
[120] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig, "Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays," *PLoS Genet.*, vol. 4, no. 8, Aug. 2008, Art. no. e1000167.
[121] F. Tramer, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *Proc. USENIX Secur.*, 2016.
[122] C. Dwork, A. Smith, T. Steinke, and J. Ullman, "Exposed! A survey of attacks on private data," *Annu. Rev. Statist. Appl.*, vol. 4, no. 1, pp. 61–84, Mar. 2017.
[123] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, Oct. 2015.
[124] N. Haim, G. Vardi, G. Yehudai, M. Irani, and O. Shamir, "Reconstructing training data from trained neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022.
[125] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Commun. ACM*, vol. 64, no. 3, pp. 107–115, Feb. 2021.
[126] G. Brown, M. Bun, V. Feldman, A. Smith, and K. Talwar, "When is memorization of irrelevant training data necessary for high-accuracy learning?" in *Proc. 53rd Annu. ACM SIGACT Symp. Theory Comput.*, New York, NY, USA, Jun. 2021.
[127] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici, "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers," *Int. J. Secur. Netw.*, vol. 10, no. 3, pp. 137–150, 2015.
[128] S. Mahloujifar, E. Ghosh, and M. Chase, "Property inference from poisoning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2022, pp. 1120–1137.
[129] *ChatGPT Plugin Privacy Leak*. Accessed: Dec. 22, 2023. [Online]. Available: https://atlas.mitre.org/studies/AML.CS0021/
[130] H. Chaudhari, J. Abascal, A. Oprea, M. Jagielski, F. Tramèr, and J. Ullman, "SNAP: Efficient extraction of private properties with poisoning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2023, pp. 400–417.
[131] M. Chase, E. Ghosh, and S. Mahloujifar, "Property inference from poisoning," 2021, *arXiv:2101.11073*.
[132] M. Jagielski, J. Ullman, and A. Oprea, "Auditing differentially private machine learning: How private is private SGD?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 22205–22216.
[133] B. Jayaraman and D. Evans, "Evaluating differentially private machine learning in practice," in *Proc. 28th USENIX Security Symp. (USENIX Security)*. Berkeley, CA, USA: USENIX Association, 2019, pp. 1895–1912.
[134] K. Bayoudh, R. Knani, F. Hamdaoui, and A. Mtibaa, "A survey on deep multimodal learning for computer vision: Advances, trends, applications, and datasets," *Vis. Comput.*, vol. 38, no. 8, pp. 2939–2970, Aug. 2022.
[135] H. Chang, T. Duy Nguyen, S. K. Murakonda, E. Kazemi, and R. Shokri, "On adversarial bias and the robustness of fair machine learning," 2020, *arXiv:2006.08669*.
[136] F. Zhang, P. P. K. Chan, B. Biggio, D. S. Yeung, and F. Roli, "Adversarial feature selection against evasion attacks," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 766–777, Mar. 2016.
[137] K. W. McClintick, J. Harer, B. Flowers, W. C. Headley, and A. M. Wyglinski, "Countering physical eavesdropper evasion with adversarial training," *IEEE Open J. Commun. Soc.*, vol. 3, pp. 1820–1833, 2022.
[138] H. Dang, Y. Huang, and E.-C. Chang, "Evading classifiers by morphing in the dark," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017.
[139] C. Song, Z. Wang, and H. Li, "Feedback learning for improving the robustness of neural networks," in *Proc. 18th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2019, pp. 686–693.
[140] W. Wang, A. Levine, and S. Feizi, "Improved certified defenses against data poisoning with (deterministic) finite aggregation," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 162, Baltimore, MD, USA, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., Jul. 2022, pp. 22769–22783.
[141] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2018.
[142] Z. Xiang, D. J. Miller, and G. Kesidis, "Post-training detection of backdoor attacks for two-class and multi-attack scenarios," in *Proc. 10th Int. Conf. Learn. Represent.*, 2022.
[143] E. M. E. Mhamdi, S. Farhadkhani, R. Guerraoui, A. Guirguis, L.-N. Hoang, and S. Rouault, "Collaborative learning in the jungle (decentralized, Byzantine, heterogeneous, asynchronous, and nonconvex learning)," in *Proc. NeurIPS*, 2021.
[144] N. Trcka, M. Moulin, S. Bopardikar, and A. Speranzon, "A formal verification approach to revealing stealth attacks on networked control systems," in *Proc. 3rd Int. Conf. High Confidence Networked Syst.*, Apr. 2014, pp. 67–76.
[145] *IBM/AI-Privacy-Toolkit: A Toolkit for Tools and Techniques Related to the Privacy and Compliance of AI Models*. Accessed: Mar. 1, 2024. [Online]. Available: https://github.com/IBM/ai-privacy-toolkit
[146] S. Calzavara, P. Ferrara, and C. Lucchese, "Certifying machine learning models against evasion attacks by program analysis," *J. Comput. Secur.*, vol. 31, no. 1, pp. 57–84, 2023.
[147] J. Richter-Brockmann, J. Feldtkeller, P. Sasdrich, and T. Güneysu, "VERICA—Verification of combined attacks: Automated formal verification of security against simultaneous information leakage and tampering," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2022, pp. 255–284, Aug. 2022.
[148] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in Byzantium," in *Proc. Int. Conf. Mach. Learn.*, 2018.
[149] S. Zhao, Q. Zhao, C. Zhao, H. Jiang, and Q. Xu, "Privacy-enhancing machine learning framework with private aggregation of teacher ensembles," *Int. J. Intell. Syst.*, vol. 37, no. 11, pp. 9904–9920, Nov. 2022.
[150] W. Watkins, H. Wang, S. Bae, H.-H. Tseng, J. Cha, S. Y.-C. Chen, and S. Yoo, "Quantum privacy aggregation of teacher ensembles (QPATE) for privacy-preserving quantum machine learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2024, pp. 6875–6879.
[151] Q. Zhang, J. Ma, J. Lou, L. Xiong, and X. Jiang, "Towards training robust private aggregation of teacher ensembles under noisy labels," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2020, pp. 1103–1110.
[152] P. Gohari, B. Chen, B. Wu, M. Hale, and U. Topcu, "Privacy-preserving kickstarting deep reinforcement learning with privacy-aware learners," 2021, *arXiv:2102.09599*.
[153] *MITRE | ATLAS*. Accessed: Mar. 10, 2024. [Online]. Available: https://atlas.mitre.org/
[154] *AI Risk Management Framework*. Accessed: May 16, 2024. [Online]. Available: https://www.nist.gov/itl/ai-risk-management-framework
[155] S. Dahmen-Lhuissier. *Securing Artificial Intelligence (SAI)*. Accessed: May 18, 2024. [Online]. Available: https://www.etsi.org/technologies/securing-artificial-intelligence
[156] *Trusted-AI/Adversarial-Robustness-Toolbox: Adversarial Robustness Toolbox (ART)—Python Library for Machine Learning Security—Evasion, Poisoning, Extraction, Inference—Red and Blue Teams*. [Online]. Available: https://github.com/Trusted-AI/adversarial-robustness-toolbox
[157] *Azure/Counterfit: A CLI That Provides a Generic Automation Layer for Assessing the Security of ML Models*. Accessed: Oct. 12, 2023. [Online]. Available: https://github.com/Azure/counterfit
[158] R. S. S. Kumar. (2021). *AI Security Risk Assessment Using Counterfit*. [Online]. Available: https://www.microsoft.com/en-us/security/blog/2021/05/03/ai-security-risk-assessment-using-counterfit/
[159] Apostol Vassilev. (2023). *NIST Artificial Intelligence (AI) 100-2 E2023 (Draft), Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations*. [Online]. Available: https://csrc.nist.gov/pubs/ai/100/2/e2023/ipd

JASMITA MALIK (Student Member, IEEE) received the B.E. degree in computer systems engineering from Middlesex University, Dubai, United Arab Emirates, and the M.Sc. degree in network security with a focus in cybersecurity from Heriot-Watt University, Dubai. She is currently pursuing the Ph.D. degree in computer science and information systems in the area of AI assurance and cybersecurity from the Birla Institute of Technology and Science, Dubai Campus, United Arab Emirates. She is also the Information Security Manager with Majid Al Futtaim (MAF), a leading multi-national lifestyle conglomerate in the EMEA region. In MAF, she is the In-Charge of the Enterprise Information Security and IT Risk Management Program. She is passionate about enabling organizations to take a holistic approach to security by embedding risk management practices with ongoing business processes and working toward overall technological resilience. She is an Information Security and Data Privacy Practitioner with more than eight years of experience delivering and advising executive leadership on large-scale business continuity and ICT recovery, cybersecurity, data protection, and resilience programs. She has worked across industries, such as fintech, banking, retail, real estate, and hospitality. She holds several industry-based certifications in cybersecurity, risk management, and business continuity. Over the years, she has received several accolades for accelerating the IT and cybersecurity posture of different organizations. She was awarded the Cyber Strategist Award by CXOInsights Middle-East and was shortlisted for Women in IT Awards in Asia. She is an active researcher on new-age AI and ML-driven cybersecurity and incident response solutions.

RAJA MUTHALAGU received the B.E. and M.E. degrees in electronics and communication engineering from Anna University, Chennai, in 2005 and 2007, respectively, and the Ph.D. degree in wireless communication from the National Institute of Technology (NIT), Tiruchirappalli, India, in 2014. He was a Postdoctoral Research Fellow with the Air Traffic Management Research Institute, Nanyang Technological University, Singapore, from 2014 to 2015. He is currently an Associate Professor with the Birla Institute of Technology and Science, Pilani, Dubai Campus, Dubai, United Arab Emirates. He has published more than 55 research articles in reputed journals and conferences. His current research interests include wireless communications, signal processing, aeronautical communications, cyber security, and applying intelligent techniques for detecting and mitigating a security attack in the IoT, SDN, and other computer networks. He was a recipient of the Canadian Commonwealth Scholarship Award for Graduate Student Exchange Program from the Department of Electrical and Computer Engineering, University of Saskatchewan, Saskatoon, Canada, in 2010.

PRANAV M. PAWAR (Member, IEEE) received the degree in computer engineering from Dr. Babasaheb Ambedkar Technological University, Maharashtra, India, in 2005, the master's degree in computer engineering from Pune University, in 2007, and the Ph.D. degree in wireless communication from Aalborg University, Denmark, in 2016. He is an IBM DB2 and IBM RAD certified professional and completed NPTEL certification in different subjects. He received recognition from Infosys Technologies Ltd. for his contribution to the Campus Connect Program and also received different funding for research and attending conferences at the international level. From 2006 to 2007, he was a System Executive with POS-IPC, Pune. He was an Associate Professor with the Department of Information Technology, STES's Smt. Kashibai Navale College of Engineering, Pune, from 2008 to 2018; and MIT ADT University, Pune, from 2018 to 2019. He is currently an Assistant Professor with the Department of Computer Science, Birla Institute of Technology and Science (BITS), Dubai. Before joining BITS, he was a Postdoctoral Fellow with Bar-Ilan University, Israel, from March 2019 to October 2020, in the areas of wireless communication and deep learning. He has published more than 40 papers at national and international levels. His research interests include energy-efficient MAC for WSN, QoS in WSN, wireless security, green technology, computer architecture, database management systems, and bioinformatics. His Ph.D. thesis received a nomination for the Best Thesis Award from Aalborg University. He was a recipient of the Outstanding Postdoctoral Fellowship from Israel Planning and Budgeting Committee.