Received October 13, 2020, accepted December 9, 2020, date of publication December 15, 2020, date of current version January 8, 2021.

Digital Object Identifier 10.1109/ACCESS.2020.3045078

Privacy and Security Issues in Deep Learning: A Survey

XIMENG LIU[1,2], (Member, IEEE), LEHUI XIE[1,2], YAOPENG WANG[1,2], JIAN ZOU[1,2], JINBO XIONG[3], (Member, IEEE), ZUOBIN YING[4], (Member, IEEE), AND ATHANASIOS V. VASILAKOS[1,5,6], (Senior Member, IEEE)

[1]College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China
[2]Fujian Provincial Key Laboratory of Information Security of Network Systems, Fuzhou University, Fuzhou 350108, China
[3]Fujian Provincial Key Laboratory of Network Security and Cryptology, College of Mathematics and Informatics, Fujian Normal University, Fuzhou 350117, China
[4]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798
[5]School of Electrical and Data Engineering, University of Technology Sydney, Sydney, NSW 2007, Australia
[6]Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, 97187 Luleå, Sweden

Corresponding author: Jian Zou (zoujian@fzu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant U1804263 and Grant 61702105.

ABSTRACT Deep Learning (DL) algorithms based on artificial neural networks have achieved remarkable success and are being extensively applied in a variety of application domains, ranging from image classification, automatic driving, natural language processing to medical diagnosis, credit risk assessment, intrusion detection. However, the privacy and security issues of DL have been revealed that the DL model can be stolen or reverse engineered, sensitive training data can be inferred, even a recognizable face image of the victim can be recovered. Besides, the recent works have found that the DL model is vulnerable to adversarial examples perturbed by imperceptible noised, which can lead the DL model to predict wrongly with high confidence. In this paper, we first briefly introduces the four types of attacks and privacy-preserving techniques in DL. We then review and summarize the attack and defense methods associated with DL privacy and security in recent years. To demonstrate that security threats really exist in the real world, we also reviewed the adversarial attacks under the physical condition. Finally, we discuss current challenges and open problems regarding privacy and security issues in DL.

INDEX TERMS Deep learning, DL privacy, DL security, model extraction attack, model inversion attack, adversarial attack, poisoning attack, adversarial defense, privacy-preserving.

I. INTRODUCTION

Internet of Things (IoT) is a network of physical devices embedded with sensors, software, and connectivity that can communicate over the network with other interconnected devices. With large numbers of IoT devices, a colossal amount of data is generated for usage. Fueled by the vast quantities of data, algorithmic breakthroughs, availability of computational resources, Deep Learning (DL), is part of a broader family of Machine Learning (ML), has been extensively applied in various fields such as image classification [1], speech recognition [2], [3], facial recognition [4], [5], medical diagnosis [6], credit risk assessment [7], Artificial Intelligence (AI) game [8], [9]. While connected sensors, found in everything from surveillance cameras to industrial plants to fitness trackers, collect troves of sensitive data, has driven interest in DL, a significant portion of data poses potential privacy and security questions [10]. On the one hand, companies such as Google, Amazon, and Facebook take advantage of the massive amounts of data collected from their users and the vast computational power of GPU farms to deploy DL on a large scale. If such data is a collection of users' private data, including online behaviors, private images, interests, geographical positions, and more, companies will have access to sensitive information that could potentially be mishandled. Further, recent researches showed that the adversary can duplicate the parameters/hyparameters of the model deployed in the cloud to provide Machine Learning as a Service (MLaaS). The intellectual property of the DL model (e.g., parameters, architecture) and the sensitive training datasets are referred to as DL privacy in this paper. On the other hand, due to the defects of the DL

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

model itself, the adversary can craft a sample to mislead the DL model or lead the learner to train a bad model. For example, the autopilot system recognizes the stop sign collected from the sensor (images using the camera) as a limit 45 sign [11], the adversary also can manipulate the training data by tampering with sensors to significantly decrease overall performance, cause targeted misclassification or insert backdoors [12]–[14]. Therefore, the unique security challenges threaten the availability of DL models, especially in security and safety critical applications, such as autonomous driving, face recognition, intrusion detection. These security threats are referred as to DL security in this paper. All in all, the application of DL faces many privacy and security challenges.

Currently, the privacy issues of DL have been revealed, and various attacks have been proposed. The attacks that invade the privacy of the model fall into two categories: model extraction attack and model inversion attack. In model extraction attacks, the adversary aims to duplicate the parameters/hyperparameters of the model that is deployed to provide cloud-based ML services [15], [16], which compromise the ML algorithms confidentiality and intellectual property of the service provider. In model inversion attacks, the adversary aims to infer sensitive information by utilizing available information. Shokri *et al.* [17] first proposed a membership inference attack again ML models, which can infer whether a sensitive record was used as a part of the training data when the ML model is overfitted. Later, Long *et al.* [18] suggested that even a well-generated model on training data also can be attacked. Besides, there are a large number of model inversion attacks, which are deployed in various scenarios, their target model, model learning approach, and assumptions are all different. A summary of the model inversion attack is provided in Table 1. The security threats of DL can be categorized into two types: adversarial attacks and poisoning attacks. In adversarial attacks, Szegedy *et al.* [19] firstly pointed out that Deep Neural Network (DNN) is vulnerable to adversarial attacks in the form of perturbations that are invisible to the human visual system by adding them to the original image. Such attacks can make a neural network classifier output wrongly predictions with high confidence, as shown in Figure 1. Since the concept of adversarial examples was proposed, a large number of adversarial attacks were discovered, which can be further categorized into white-box attack and black-box attack, as shown in Table 6. The adversarial attacks have evolved from early white-box attack to black-box attack. In white-box setting, the adversary has total knowledge of the target model, such as model architecture, parameter, training data. Instead, in black-box settings, the adversary has no knowledge of the model, such as model architecture parameters, training data. The adversary crafts an adversarial example by sending a series of queries, which is more practical in real scenarios. In poisoning attacks, the adversary aims to pollute the training data by injecting malicious samples, modifying data such that the learner train a bad classifier, which would misclassify malicious samples or activities crafted by the adversary at

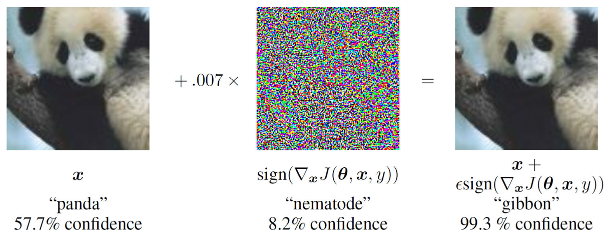

$+ .007 \times$
$=$
$\boldsymbol{x}$
"panda"
57.7% confidence
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

FIGURE 1. A demonstration of an adversarial sample [21]. The panda image is recognized as a gibbon with high confidence by the classifier after adding adversarial perturbations.

the testing stage. For example, Muñoz-González *et al.* [20] crafted poisoning samples that look like real data points to reduce the accuracy of the classifier.

To solve privacy and security issues in DL, a variety of approaches have been proposed, as shown in Figure 2. There are currently four mainstream technologies in the aspect of DL privacy for privacy-preserving, namely differential privacy, homomorphic encryption, secure multi-party computation, and trusted execution environment. The differential privacy aims to prevent the adversary from inferring whether a particular instance was used to train the target model. The homomorphic encryption and secure multi-party computation scheme focus on preserving the privacy of the training and testing data. The trusted execution environment aims to use hardware to create a secure and isolated environment to protect training code and sensitive data. However, these methods significantly increase the computational overhead and require customization for specific neural network models. At present, there is still no universal approach to address DL privacy issues. With respect to DL security, a large number of defenses have been proposed to defend the adversarial attack, which can be categorized into three categories: input pre-processing, malware detection, and improving the robustness of the model, as shown in Table 7. The purpose of pre-processing aims to reduce the influence of immunity on the model, which is done by performing some operations such as image transformation, randomization, denoising, which usually do not require modification and retraining of the model. The second category aims to improve the robustness of the model by introducing regulation, adversarial training, feature denoising, which requires modification and retraining of the model. The third category aims to detect the adversarial by introducing a detection mechanism before the input and first layer of the model, including stateful detection, image transformation detection, and adaptive denoising detection. Although a variety of defensive mechanisms have been proposed, to our best knowledge, there still is no defense method that can completely defend against adversarial examples. At present, adversarial training is considered to be the most effective method to defend against adversarial examples. For poisoning attacks, there are two typical methods to defend against poisoning attacks. The first one is the outlier detection mechanism, which removes outliers outside the applicable set. The second one is to improve the robustness of the neural network to resist the pollution of poisoning samples.

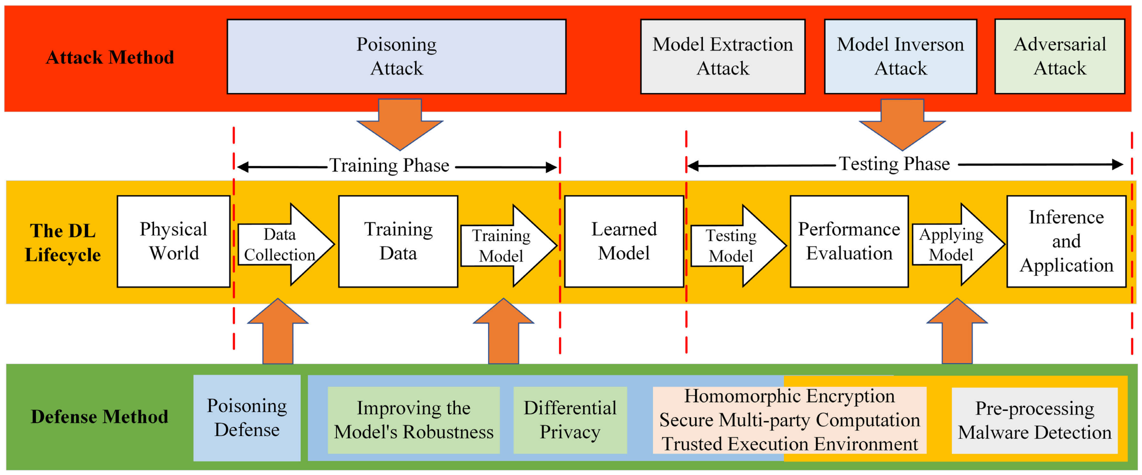


FIGURE 2. An overview of attacks and defenses in DL. The top of the image describes the threat of the existing privacy and security in DL. The middle part shows the lifecycle of DL, which involves two major phases (training phase, testing/inference phase). The bottom of the image shows the defense methods at different stages of the DL lifecycle, where homomorphic encryption, secure multi-party computation, and trusted execution environment can be used to preserve the DL privacy both at training and testing phases.

A. MOTIVATION

Privacy and security issues in DL have been becoming a hot topic in recent years. In this paper, we present a comprehensive survey on the privacy and security issues of DL. To date, there are a few review and survey papers associated with privacy and security in DL have been published. Akhtar *et al.* [22] reviewed the adversarial example attacks and defenses on DL in the field of computer vision. Tanuwidjaja *et al.* [23] and Boulemtafes *et al.* [24] studied several privacy-preserving techniques on DL. Yuan *et al.* [25] presented a review on adversarial examples for DL, in which they summarize the adversarial example generation methods and discuss the corresponding defense methods. The above review works all focus on the adversarial attacks or cryptographic primitives-based privacy-preserving techniques. Liu *et al.* [26] analyzed security threats and defenses on ML and provided a more comprehensive literature review from a data driven view. Papernot *et al.* [27] systematically studied the security and privacy of ML, but they don't involve many DL models that are widely used.

B. MAIN CONTRIBUTIONS

The differences between our paper and existing review are summary as follows:

1) Instead of focusing on one phase and a few types of attacks and defenses, all aspects of privacy and security in DL are systematically reviewed in this paper. All types of attacks and defenses are reviewed with respect to the life cycle of DL (training phase, testing phase), as shown in Figure 2.
2) According to the life cycle of the DL model, the adversary types and goals of the DL model are introduced, and four types of attacks regarding the privacy and security of the DL are reviewed, including model extraction attacks, model inversion attacks, adversarial attacks, and poisoning attacks.
3) This paper not only reviews the privacy-preserving technology based on cryptography in DL but also study the privacy and intellectual property protection technology based on the trusted execution environment and digital watermark.
4) This paper systematically reviews the privacy and security defenses that are representative of DL in recent years. This paper also compared the advantages and disadvantages of these defenses and the analysis of their effectiveness.
5) The current challenges and open problems regarding privacy and security issues in DL are discussed, including the deflect of current privacy-preserving techniques, attacks in real-world, effective, and low overhead defense methods.

The rest of this paper is organized as follows. In Section II, we first discuss four types of attacks and introduce the cryptography technologies for preserving privacy in DL. In Section III, we detail the recent attack and privacy-preserving techniques in DL. In Section IV, we review the representative attack and defense method. In Section V, we discussed future challenges and research directions for security and privacy in DL. In Section VI, we draw a conclusion.

II. PRELIMINARIES

In this section, we begin with an overview of attacks and defenses in the DL. Then, we discuss the adversary's capabilities and four types of attacks based on the DL lifecycle. Besides, we also briefly introduce the cryptographic tools used to preserve the privacy of data in DL.

A. OVERVIEW OF ATTACKS AND DEFENSES

The DL life cycle can roughly fall into two phases (training phase, testing/inference phase). Figure 2 shows the existing threats and defense strategies regarding privacy and security issues according to the DL life cycle. The privacy threat of DL can be divided into two categories, model extraction attack, and model inversion attack. There are four mainstream defenses against them, namely, differential privacy, homomorphic encryption, secure multi-party computation, and trusted execution environment. Differential privacy defends against model inversion attacks by injecting noise during the training phase. Homomorphic encryption, secure multi-party computing, and trusted execution environment can be used to protect DL privacy during the training and testing phases. The threat of DL security falls into two categories, adversarial attacks, and poisoning attacks. Adversarial defenses are being developed along with three main directions, pre-processing, malware detection, and improving model robustness. Pre-processing and malware detection attempt to reduce the effect of the adversarial perturbation on the classification or detect the adversarial examples during the testing/inference phase. Improving the model's robustness aims to essentially enhance the DL model to resist adversarial examples. The poisoning defenses attempt to filter out the poisoning sample during the training phase.

B. ADVERSARIAL CAPABILITIES

1) TRAINING STAGE

- **Data Injection.** The adversary has no knowledge of the target model and training data. The adversary only injects poisoning samples into the training data to change the distribution of the training data such that the learner trains a bad model.
- **Data Modification.** The adversary has no knowledge of the target model, but can directly access the training data. The adversary attempts to pollute the training data by modifying the training data before it is used for training the target model.
- **Logic Corruption.** The adversary has complete knowledge of the target model and can modify the learning algorithm. This type of attack is tricky and is hard to defend.

2) TESTING STAGE

- **White-box.** In white-box settings, the adversary has complete knowledge of the target model, including model architecture, model parameters, and training data. The adversary only identifies the model's vulnerability by utilizing available information, and then launches an attack against the target model such as adversarial attack (see Section II-C3), model extraction attack (see Section II-C1), model inversion attack (see Section II-C2). Moreover, in adversarial attacks, the adversary also has complete knowledge about the defense mechanisms against adversarial attacks.
- **Black-box.** In black-box settings, the adversary does not know the target model, including model architecture, model parameters, and training data. The adversaries only identify the model's vulnerability by utilizing knowledge about output responded from the target model and then launched an attack against the target model by sending a series of queries to the target model and observing corresponding outputs. Such attacks include model extraction attack, model inversion attack, and adversarial attack.
- **Gray-box.** In gray-box settings, the adversary has complete knowledge of the target model, including model architecture, model parameters, and training data. However, unlike the white-box setting, the adversary does not know the defense mechanism against the adversarial attack. The gray-box setting usually is used to evaluate the defense against the adversarial attack.

C. ADVERSARIAL TYPES AND GOALS

1) MODEL EXTRACTION ATTACK

In model extraction attacks, the adversary aims to steal parameters of the target model with black-box access to the target model, which compromises the ML algorithm confidentiality and intellectual property of the learner.

2) MODEL INVERSION ATTACK

In model inversion attacks, the adversary aims to utilize model predictions to expose the privacy of sensitive records that were used as part of the training set. For example, Shokri *et al.* [17] proposed a membership inference attack that can infer whether a given record was a part of the training data.

3) ADVERSARIAL ATTACK

In adversarial attacks, the adversary aims to craft an adversarial example by utilizing the knowledge about the target model, which leads the target model to predict falsely with high confidence. An adversarial sample is a kind of modified image generated by adding imperceptible noise, which can cause the classifier to make wrong predictions with high confidence. Moreover, an adversarial sample crafted on one model is likely to be effective for other models, which is known as transferability. According to the goal of the adversary, the adversarial attack falls into two categories:

- **Non-targeted Attack.** The adversary crafts adversarial examples x_{adv} to cause the target model f to misclassify the input with high confidence, but does not require the prediction to be specified class, that is, $f(x_{adv}) \neq y_{true}$, where $f(x_{adv})$ can be any class except the correct class y_{true}.
- **Targeted Attack.** The adversary crafts adversarial examples x_{adv} to cause the target model f to misclassify the input with high confidence into a particular class t specified by the adversary, that is, $f(x_{adv}) = t$, where t is not correct class y_{true}.

4) POISONING ATTACK

In poisoning attacks, the adversary aims to poison the training data such that the learner train a bad classifier, which would misclassify malicious sample or activities crafted by the adversary at the testing stage. The adversary could inject malicious samples, modify data labels, and corruption in the training data. Depending on the adversary' goals, the poisoning attack falls into three categories:

- **Accuracy Drop Attack.** The adversary aims to disrupt the training process by injecting malicious samples to reduce the performance of the target model at the testing stage.
- **Target Misclassfication Attack.** The adversary aims to enforce test samples to be misclassified at the testing stage.
- **Backdoor Attack.** The adversary aims to install a backdoor with a specific mark so that the target model has a target output for that particular input.

D. CRYPTOGRAPHIC TOOLS

1) DIFFERENTIAL PRIVACY

The concept of Differential Privacy (DP) was proposed by Dwork *et al.* [28] that aim to guarantee an algorithm to learn statistical information of the population without disclosing information about individuals.

A randomized mechanism M is considered to provide ϵ-DP if, for all datasets D and D' that only differ on one record and all subsets S of M, satisfy the following:

$$\Pr[M(\mathcal{D}) \in S] \leq \exp(\varepsilon) \Pr\left[M\left(\mathcal{D}'\right) \in S\right] \tag{1}$$

where ϵ is the privacy budget parameter that decides the privacy level. The definition of the randomized mechanism M usually is as follows:

$$M(\mathcal{D}) = f(\mathcal{D}) + n \tag{2}$$

where the randomized mechanism M takes a dataset as input and add noise n to the original query response $f(D)$. The noise usually samples from the Gaussian distribution or Laplace distribution. That is, even if an adversary knows the whole dataset D except for a single record, he/she cannot infer much about the unknown record from the output $M(D)$.

Due to the definition of ϵ-DP is strict, the (ϵ,δ)-DP were introduced, which loosens the bound of the error by the amount δ. The definition of (ϵ,δ)-DP is as follows:

$$\Pr[M(\mathcal{D}) \in S] \leq \exp(\varepsilon) \Pr\left[M\left(\mathcal{D}'\right) \in S\right] + \delta \tag{3}$$

where δ is a small real number, which also controls the privacy budget like ϵ. The sensitivity Δf of the function f characterize how much changing any one of the records in the datasets causes the output of the function to change:

$$\Delta f = \max \left\| f(D) - f\left(D'\right)\right\|_1 \tag{4}$$

where $\|\cdot\|_1$ denote the l_1-norm and *max* function apply on all datasets D and D'.

2) HOMOMORPHIC ENCRYPTION

The Homomorphic Encryption (HE) is a form of the encryption scheme that allows computation on ciphertexts, generating an encrypted result which, when decrypted, matches the result of the operations as if they had performed on the plaintext. The definition of encryption function Enc has followed the equation:

$$\text{Enc}(a) * \text{Enc}(b) = \text{Enc}(a * b) \tag{5}$$

where Enc: $x \to y$ is a HE scheme that map plaintext x to ciphertext y. $*$ is mathematical operation that can be performed on plaintext and ciphertext.

The HE exists in partial and full forms. The partial homomorphic systems only support certain operations on encrypted data, typically additive [29]–[32] or multiplication [33], [34]. Since neural networks need to perform a large number of additions and multiplications, The partial homomorphic encryption systems (e.g, Paillier [29]) that support additive operations are better suitable for DL complex computation than that, only support multiplication operation (e.g., RSA [33], ElGamal [34]). The Fully Homomorphic Encryption (FHE) system, which allows all operations on encrypted data, was first proposed in 1978. Until 2009, Gentry *et al.* [35] first constructed a feasible FHE scheme using lattice-based cryptography, which supports both addition and multiplication operations on ciphertexts. Since then, several FHE schemes were proposed [36]–[41]. Although theoretically, FHE can perform all operations on encrypted data, the actual scheme proposed now still has many limitations when applied in DL, such as only support integer type data, computational complexity (e.g., it requires 29.5s to run secure integer multiplication computation with a standard PC [41]). Therefore, the existing (fully) HE schemes still require a lot of custom work for each DL model to be fitted to the HE environment and improve the efficiency of computation.

3) SECURE MULTI-PARTY COMPUTATION

The problem of secure computation was first proposed by Yao [42] in 1982, which is also known as the millionaire problem: two rich men Alice and Bob met on the street, how to compare who is richer without exposing their wealth.

Later, the millionaire problem was expanded by Goldeich *et al.* [43] to become a very active research field in modern cryptography, namely Secure Multi-party Computation (SMC) whose purpose is to address the problem of joint computing that preserves participant's data privacy in a flock of the non-trusted participant.

Formally, in SMC, for a given objective function f, a group of participants, $p_1, p_2, p_3, \ldots, p_n$, each participants his own private data, $d_1, d_2, d_3, \ldots, d_n$, respectively. Then, all participants want to jointly compute $f(d_1, d_2, d_3, \ldots, d_n)$ on those private data. At the end of computing, each participant has no knowledge of other participant about their private data.

There are two kinds of secure multi-party computation: secure Two-Party Computation (2PC) and secure Multi-Party Computation (MPC), which are quite different in the protocols. Garbled Circuit (GC) [42] and Oblivious Transfer (OT) [44] protocol are used in the 2PC. GC transforms a function into a safe boolean circuit, and its input and output are encrypted data, which can be decrypted by its decoding

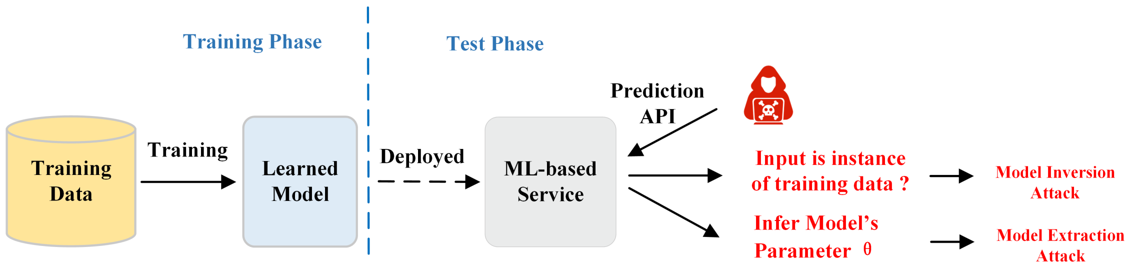


FIGURE 3. An overview of the model extracion attack and model inversion attack.

table. OT is used to transfer the information obviously. The most protocol commonly used in MPC are secret sharing [45], which divides an input from each participant into several parts and distributes them to each participant. Each participant calculates a given function together.

By its nature, SMC can be used to jointly train a global DL model without revealing the data privacy of each participant, which helps to break the island of information.

III. PRIVACY

In this section, we review and summarize the representative existing privacy threats (model extraction attack, model inversion attack) in DL and privacy-preserving technologies, including DP, HE, SMC, trusted execution environment.

A. ATTACKS

The attacks that invade privacy falls into two categories: model extraction and model inversion attack, an overview of model extraction and model inversion attack is shown in Figure 3. The main difference between the two is that the former focuses on the private information on the model (e.g., model parameter, model architecture). The latter focuses on the sensitive record of the training data.

1) MODEL EXTRACTION ATTACK

Tramer *et al.* [15] introduced a model extraction attack that aims to duplicate the parameters of ML models deployed to provide cloud-based ML services. The general idea is to build model equations from the output obtained by sending a plurality of queries. However, it cannot be extended to some scenarios where the attacker does not have access to the probabilities returned for each class and only effective for specific ML models, such as decision tree, logistic regression, neural network. To avoid overfitting on training data, the regulation term is usually used in ML algorithms, where the hyperparameters are introduced to balance regulation terms and loss function. Wang *et al.* [16] proposed hyperparameter stealing attacks to steal the hyperparameter of the target model. Because the goal of learning of the ML algorithm is to achieve the minima of the objective function where the gradient of the objective function is close to 0, therefore, based on this observation, the adversary can establish several linear equations by executing a large collection of queries. Finally, the hyperparameters can be estimated by utilizing linear least squares. They empirically demonstrated that their attack could accurately steal hyperparameters with less than 10^{-4} error in regression algorithms.

As a countermeasure to possible intellectual property thefts, watermarks for DNN has been developed that embed watermarks into the DL model. Wang *et al.* [46] showed that the watermarking scheme proposed by Uchida *et al.* [47] increases the standard deviation of the distribution of the weights as the embedded watermark length increases. Based on this observation, they proposed an algorithm to detect the presence of a watermarking and then remove the watermarking by available knowledge. However, the removability and overwriting of the watermarking are often considered when embedding into DL models. Therefore, assuming that the watermarking might not be removed, Hitaj *et al.* [48] designed two novel evasion attacks that allow an adversary to run an MLaaS with stole ML models and still go undetected by the legitimate owners of those ML models.

2) MODEL INVERSION ATTACK

Shokri *et al.* [17] proposed a Membership Inference Attack (MIA) again ML models. The adversary trains an attack model to distinguish the difference between the target model's behavior on the sample for its trained sample and its untrained sample. That is, the attack model is a classification model. To construct such an attack model under a black-box setting, the author invented a new technique called shadow training that builds multiple shadow models to simulate the target model. Because the target model's data distribution is unknown, they utilize the input/output pairs of the shadow model to train an attack model. The experiments demonstrated that an adversary could successfully perform an MIA with black-box access to the target model. However, Long *et al.* [18] pointed out that MIA, proposed by Shokri *et al.* [17], works effectively when the model is highly overfitted on its training data. The prediction (probability) of the overfitted model for the query on the training sample is significantly different from the prediction (probability) of other queries. In contrast, a well-generalized model behaves similarly to training data and test data. Therefore, not all data is vulnerable to member inference attacks under well-generated models. To solve this challenge, Long *et al.* [18] presented a Generalized Membership Inference Attack (GMIA) against a well-generalized model. The general idea behind this attack is to identify vulnerable target records that are vulnerable to member inference attacks to perform inference attacks.

The MIA proposed by Shokri *et al.* [17] depends on too many assumptions, such as using multiple shadow models, having the knowledge of the target model, and having a data distribution same as the target model's training data. Salem *et al.* [49] relaxed these assumptions and proposed three adversaries that are very broadly applicable at a low cost. The first adversary utilizes only one shadow model instead of multiple shadow models to mimic the target model. For the second adversary, the adversary does not have direct access to the training data of the target model and its architecture. The third adversary has a minimal set of assumptions. It is not necessary to build any shadow model and know the structure of the target model, which is more practical in real-scenario.

Hitaj *et al.* [50] proposed an MIA to perform white-box attacks in the context of collaborative DL models. They constructed a generator and used it to form a Generative Adversarial Network (GAN) [51]. After training, GAN can generate synthetic data similar to the training data of the target model. However, the limitation of this approach is that all training data belonging to the same category are required to be visually similar, so they cannot be distinguished under the same distribution. Similarly, an MIA against collaborative learning models was proposed by Melis *et al.* [52]. Collaborative learning is a learning technique in which two or more participants jointly train a joint model by training locally and periodically exchanging model updates, where each participant has its training datasets. However, these updates can leak unintended information about the training data of the participants. In some non-numeric data scenarios such as natural language processing, the embedding layer firstly is used to transform the input into a low-dimensional vector representation, where the update of the given matrix is only related to whether the word appears in the batch. Therefore, this character directly reveals whether the word appears in the training batches and can be used to design a property inference attack.

Hayes *et al.* [53] presented the first MIA on generative models, which utilizes GAN [51] to infer whether a data item was part of the training data by learning statistical difference in distribution. Hayes *et al.* [53] observed that the discriminator would place a higher confidence value on samples that appeared in the training data when the target model is highly overfitted. Based on this observation, they proposed the white-box and black-box MIA. In white-box settings, the adversary can directly utilize the discriminator to infer whether a sample was used to train the model by discriminating its output's confidence. In black-box settings, the adversary does not know the target model's parameter, thereby only locally train a GAN by using queries from the target model. The experiments demonstrated that white-box attacks are 100% successful at determining whether a data record was used to train the target model, and the black-box ones succeed with 80% accuracy.

Nasr *et al.* [54] noticed that a black-box attack might not be effective against well-generalized DNN. The parameters of the model are visible to curiosity adversaries in some scenarios. Therefore, Nasr *et al.* [54] proposed an MIA again DL models under the white-box setting. Due to the distribution of the model's gradients between the sample in the training data and not in the training data is likely to be distinguishable and thereby can be exploited to perform MIA, even though the DL models are well-generated. They successfully launched MIA against well-generalized federated learning models in many scenarios, such as training and fine-tuning, updating models, which showed that even a well-generated model on training data can be attacked.

The model inversion attacks in DL discussed above are deployed in various scenarios, their target model, model learning approach, and assumptions are all different. To intuitively understand the differences between these attacks, we compared those algorithms and provide a summary of the MIA in Table 1.

B. DEFENSES

To date, a variety of different methods for protecting privacy in DL have been proposed, which can roughly fall into four categories: DP, HE, SMC, and trusted execution environment. In this section, we review and summarize the representative methods of preserving privacy, as shown in Table 2, 3, 4, 5. For a comprehensive study, we also briefly describe another privacy-preserving method in DL.

1) DIFFERENTIAL PRIVACY

Depending on where the noise is added, DP approaches can be classified into three categories: gradient perturbation, objective perturbation, and label perturbation, as shown in Figure 4.

1) **Gradient Perturbation.** The gradient perturbation is done by injecting noise to the gradients of the parameters during the training stage.
2) **Objective Perturbation.** The objective perturbation is done by injecting noise to the objective function and solving a precise solution to the new problem.
3) **Label Perturbation.** The label perturbation is done by injecting noise to the label during the knowledge transfer process of the teacher-student network.

a: GRADIENT PERTURBATION

Abadi *et al.* [55] proposed a Differentially Private Stochastic Gradient Descent (DPSGD) algorithm that can train DNN with non-convex objectives. The main idea is to inject noise into the gradient at each step of the stochastic gradient descent process. Besides, Abadi *et al.* [55] developed a stronger accounting method called moment accountant to obtain the tail bound. The moment accountant utilizes the moments bound combined with Markov inequality to track the cumulative privacy loss, which provides tighter privacy loss analysis than composition theorems [56]. Xie *et al.* [57]

TABLE 1. A summary of membership inference attacks. ✓ means the adversary needs the information while empty indicates the information is not necessary. The attack strength (higher for more stars) is based on the impression of the reviewed paper.

Method	White-box/ Black-box	Target Model	Target Model Learning Approach	Shadow Model	Dataset Distribution	Model Architecture	Performance
[17]	Black-box	Classifier	Independent learning	✓	✓	✓	⋆
[18]	Black-box	Neural Network	Independent learning		✓	✓	⋆⋆⋆
[49]	Black-box	Convolutional Neural Network	Independent learning	✓		✓	⋆⋆⋆
	Black-box	Convolutional Neural Network	Independent learning	✓			⋆⋆⋆
	Black-box	Convolutional Neural Network	Independent learning				⋆⋆
[50]	White-box	Generative Adversarial Networks	Collaborative leraning				⋆⋆⋆
[52]	White-box	Classifier	Collaborative learning				⋆⋆
[53]	White-box	Generative Adversarial Networks	Independent learning				⋆⋆⋆
	Black-box	Generative Adversarial Networks	Independent learning				⋆
	Black-box	Generative Adversarial Networks	Independent learning		✓		⋆
[54]	White-box	Classifier	Federated learning				⋆⋆⋆
	White-box	Classifier	Federated learning				⋆⋆
	White-box	Classifier	Independent learning				⋆⋆

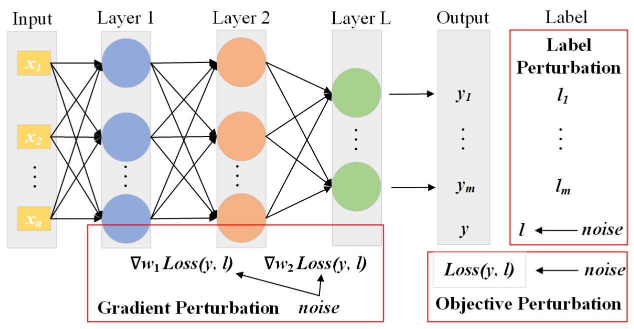


FIGURE 4. An overview of DP framework.

presented a Differentially Private Generative Adversarial Network (DPGAN) that guarantees (ϵ, θ)-DP by perturbing the gradient of the discriminator during the training procedure. According to the post-processing theory [58], the output of the differentially private discriminator will not invade privacy, which means that the generator is also differentially private. Acs *et al.* [59] designed a novel Differentially Private Generative Model (DPGM) that rely on a mixture of k generative neural networks, such as restricted Boltzmann Machines [60] and variational Autoencoders [61]. The dataset is clustered by utilizing the differential private kernel k-means [62], and then each cluster is assigned to k generative neural networks. Finally, the DP-SGD [55] is used to train k generative neural networks, and these generative neural networks can generate synthetic high-dimensional data while keeping provable privacy.

b: OBJECTIVE PERTURBATION

Phan *et al.* [63] introduced a Deep Private Autoencoder (DPA) that enforce ϵ-DP by perturbing the objective functions of the traditional deep auto-encoder. Phan *et al.* [64] proposed the Private Convolutional Deep Belief Network (PCDBN) to enforce ϵ-DP perturbing the polynomial forms that approximate the non-linear objective function by utilized the Chebyshev expansion in convolutional deep belief network. Phan *et al.* [65] proposed Adaptive Laplace Mechanism (AdLM), a novel mechanism to guarantee DP in DNN. This approach not only adaptively adds noise from the input features to the model output, but also easily extends to various differential DNN. Unlike gradient perturbation, which accumulate privacy loss as training progresses, the privacy loss due to objective perturbation is determined at the time the objective function is built and is independent of epoch.

c: LABEL PERTURBATION

Papernot *et al.* [66] demonstrated a privacy-preserving approach, Private Aggregation of Teacher Ensembles (PATE), which transfers the knowledge of an ensemble of ''teacher'' models to a ''student'' model. An ensemble of teachers models learns on disjoint subsets of the sensitive data, and then a student model is trained on public data labeled using the ensemble of teacher model. Because the student model cannot directly access the sensitive data and the differential private noise is injected into the aggregation process, and, thereby, the privacy of the sensitive data is protected. Besides, the moment accountant [55] is introduced to trace the cumulated privacy budget in the learning process. However, the performance of PATE is evaluated on simple classification tasks (e.g., MNIST [67]). Later, Papernot *et al.* [68] proposed a new aggregation mechanism that successfully extends the PATE to the large scale learning task. Besides, it is empirically shown that the improved PATE guarantees tighter DP and has higher utility than the original PATE. Furthermore, The PATE was applied to construct the differential private GAN framework [69]. Because the discriminator is differential privacy, so is the generator trained with discriminator [58]. The disadvantage of this method is that it requires additional training of a teacher model to teach the student model.

2) HOMOMORPHIC ENCRYPTION

In DL, the HE is mainly used to protect testing inputs and results and train neural network models. The main adverse effects of applying HE are the reduction of efficiency, the high computation cost of the ciphertext, and the sharp increase in the amount of data after encryption.

Xie *et al.* [70] theoretically demonstrated that the activation function of the neural network can be approximated by a polynomial, therefore, inferencing over encrypted data can be practical. Subsequently, Gilad *et al.* [71] presented a method,

TABLE 2. A summary of DP techniques.

Method	Scheme	Advantage	Shortcoming	Accuracy (%)
DPSGD [55]	Gradient perturbation	Track the cumulative privacy loss and automate analysis of the privacy loss	Not apply for complex DL models	97.00 (MNIST)
DPGAN [57]		Solve the privacy issues in GAN	Not apply for complex dataset	-
DPGM [59]		Solve the privacy issue in autoencoders	Restricted to specific autoencoders	-
DPA [63]	Objective perturbation	Not depend on the number of training epochs in consuming privacy budget	Apply particularly for autoencoder and objective function	-
PCDBN [64]		Not depend on the number of training epochs in consuming privacy budget	Affect the accuracy of the model for complex learning task	91.71 (MNIST)
AdLM [65]		Not depend on the number of training epochs in consuming privacy budget	Affect the accuracy of the model for complex learning task	93.66 (MNIST)
PATE [66]	Label perturbation	Apply for any DL model	Need to train an additional teacher model and not apply for large scale learning tasks	98.10 (MNIST)
Improved PATE [68]		Extend PATE [66] to large scale learning tasks	Need to train an additional teacher model	98.50 (MNIST)

TABLE 3. A summary of HE techniques.

Method	Scheme	Training/ Inference	Advantage	Shortcoming	Performance		
					Communication Cost(Mbytes)	Runtime (s)	Accuracy (%)
[81]	Additively HE	Training	Allow multiple parties to jointly train a global model without disclosing participant's private data	Protecting privacy against an honest-but-curious parameter server introduces the communication cost	1	8100	97.00 (MNIST)
Improved CryptoNet [73]	Additively HE	Training and inference	Extend CryptoNets [71] to deeper neural networks and non-linear layers	Training on encrypted data is feasible only if data is small or network is shallow	-	0.29	96.92 (MNIST)
CryptoDL [75]	Leveled HE	Inference	Approximate the activation with low-degree polynomials	The smae efficiency for single and batch instance predicion	336.7	320	99.52 (MNIST)
TAPAS [77]	FHE	Inference	Accelerate encrypted data computation	Not support non-binary quantized neural networks	-	147	98.60 (MNIST)
FHE-DiNN [76]	FHE	Inference	Improve efficiency at the cost of increasing storage	Not support non-binary quantized neural networks	-	1.64	96.35 (MNIST)

TABLE 4. A summary of SMC techniques.

Method	Scheme	Training/ Inference	Advantage	Shortcoming	Performance		
					Communication Cost (Mbyte)	Runtime (s)	Accuracy (%)
SecureML [82]	Secret-sharing and GC	Training and Inference	Allow data owner to outsource training to two servers	Not support the convolutional neural network	-	4.88	93.40 (MNIST)
MiniONN [83]	Secret-sharing and GC	Inference	Transform the neural network model into an oblivious neural network without any modifications to pretrained model	The higher computation and communication cost than GAZZLLE [85] and SecureNN [86] in inference phase	47.60	1.04	97.60 (MNIST)
DeepSecure [84]	HE and GC	Training and inference	The high throughout when performating batch prediction	The low efficiency when executing single instance prediction	791	9.67	98.95 (MNIST)
GAZELLE [85]	HE and GC	Inference	The faster prediction than DeepSecure [84] and MiniONN [83]	Not apply for large input size	8	0.2	-
SecureNN [86]	Serect-sharing	Training and inference	Allow multiple parties to jointly train or testing a model without disclosing private data	Only support three-party or four-party	7.93	0.1	99.15 (MNIST)

TABLE 5. A summary of trusted execution environment techniques. The performance (higher runtime for more ⋆) is based on the impression of the reviewed paper.

Method	Scheme	Training/Inference	Advantage	Shortcoming	Performance	
					Runtime	Accuracy(%)
[88]	SGX	Training	Allow multiple parties to jointly train a shared model without declosing private data.	Preventing the trained model to leak outside the SGX enclave introduces large overhead	⋆⋆	98.7 (MNIST)
Chiron [90]	Royan	Training and inference	Support launching multiple enclaves to increase performance	Not allow data owner to serve as a service provider	⋆ ⋆ ⋆	84.63 (CIFAR-10)
Myelin [92]	SGX and TVM compiler	Training and inference	Support multithreading inside the enclave	Need to compile a given model and retrain	⋆⋆	84.40 (CIFAR-10)
Deepenclave [94]	SGX	Inference	Not be constrained by SGX's limited memory	The final output of the model is plaintext	⋆ ⋆ ⋆	-
Slalom [96]	SGX and GPU	Inference	Combine trusted hardware and non-trusted hardware (GPU)	Only apply for network models established through TensorFlow	⋆	70.6 (ImageNet)
MLcapsule [98]	SGX	Inference	Support offline deployment	Not protect model architecture	⋆ ⋆ ⋆	-

CryptoNets, that can perform inference on encrypted data, which utilizes leveled HE scheme proposed by Bos *et al.* [72] to perform privacy-preserving inference on a pre-trained Convolutional Neural Network (CNN) model. However, the derivative function of the activation function using the square function approximation is unbounded, which will lead to strange phenomena when training the network on encrypted data, especially for deeper neural networks. Therefore, it is not suitable for a deeper neural network. Chabanne *et al.* [73] improved CryptoNets by combining the polynomial approximation with batch normalization [74]. Hesamifard *et al.* [75] presented CryptoDL, a new approximation method for activation functions (such as ReLU, Sigmoid, and Tanh) commonly used on CNN. This method has low-degree polynomials, which significantly improve the computational efficiency. Compared with CryptoNets [71], the CryptoDL not only has a lower communication overhead, but is also independent of the dataset. However, their method is based on a batch operation, so the same efficiency for single and batch instance prediction.

Because HE schemes applied in the neural network significantly increase complexity when the depth of the neural network increases. Bourse *et al.* [76] and Sanyal *et al.* [77] attempted to improve the efficiency of HE used in the neural network. Bourse *et al.* [76] proposed FHE-DiNN, a linear complexity framework for FHE evaluation of neural network. FHE-DiNN leverage bootstrapping technique [78] to reach strictly linear complexity in the depth of the neural network. Sanyal *et al.* [77] noted that the FHE scheme proposed by Chillotti *et al.* [78] only supports operations on binary data, which can be utilized to compute all of the operations of the binary neural network (BNN) [79]. Therefore, Sanyal *et al.* [77] design Tricks to Accelerate (encrypted) Prediction As a Service (TAPAS), which speeds up binary operation in BNN. The experiments showed that TAPAS drastically shortens the time of the evaluation step.

3) SECURE MULTI-PARTY COMPUTATION

There are two main application scenarios for SMC in DL. The first scenario is that a data holder does not want to expose all the training data to a server to train/infer the DL model. He/She hopes to distribute the training/testing data to multiple servers, and train/infer the DL model together. Each server will not understand the training/testing data of other servers. The second scenario is that multiple data holders want to jointly train a shared DL model on aggregate training data while keeping the privacy of their training data.

Shokri *et al.* [81] implemented a practical system for collaborative DL, which enabled multiple participants to jointly learn neural network models based on their inputs without sharing those inputs and sacrificing the accuracy of models. The participants train their local model, after each round of local training, they asynchronously selectively share their gradients they computed for some of the parameters. However, Ano *et al.* [80] pointed out that even a small portion of gradients stored on cloud servers also could be utilized to extract local data. Hence, Ano *et al.* [80] used the additive HE scheme to protect privacy against an honest-but-curious parameter server, but it increases the communication overheads between the learning participants and the cloud server.

Mohasse *et al.* [82] proposed SecureML, a novel and efficient protocol for privacy-preserving ML. They implemented the first privacy-preserving system for training neural networks in multiparty computation settings using OT, secret sharing, and Yao' GC protocol. However, the disadvantage of this scheme is that only a simple neural network can (without any convolutional layer) be implemented. Liu *et al.* [83] proposed a framework, MiniONN, that transforms an existing neural network to an oblivious neural network and supports privacy-preserving predictions with reasonable efficiency. They used GC to approximate all nonlinear activation functions. Their experiment demonstrated that MiniONN outperforms SecureML [82] and Cryptonets [71] in terms of latency and message size. Rouhani *et al.* [84] presented DeepSecure, a scalable privacy-preserving framework for DL, which jointly compute DL inference over their private data between distributed client and cloud servers. They used Yao' GC [42] and OT protocol to preserve the privacy of the client data during the data transfer process and provided a security proof of DeepSecure in the honest-but-curious adversary model.

The major bottleneck of GC is the huge communication cost. Juvekar *et al.* [85] and Wagh *et al.* [86] attempted to improve SecureML. Juvekar *et al.* [85] pointed out that HE is suitable for calculating linear functions, while GC is suitable for non-linear functions such as ReLU, MaxPool. Therefore, Juvekar *et al.* [85] made optimal use of HE and GC to achieve large computational and communication gains, that is, using a simple linear size circuit to compute non-linear functions and using HE to compute linear function. Wagh *et al.* [86] developed SecureNN, a novel three-party or four-party secure computation protocols, which are more communication efficient for non-linear activation functions. The overall execution time of this method is 3.68X faster than Gazelle proposed by Juvekar *et al.* [85] in the inference phase.

4) TRUSTED EXECUTION ENVIRONMENT

The Trusted Execution Environment (TEE) is an independent operating environment of the main processor that provides a secure execution environment for authorized and trusted applications. In addition, the TEE ensures the integrity and confidentiality of the data and codes loaded inside, and provides access control to the resources of the trusted application. As a result, there are some solutions that utilize TEE to tackle privacy and security issues in DL [87].

Ohrimenko *et al.* [88] proposed a privacy-preserving system for multi-party ML on an untrusted platform, which is based on Software Guard Extensions (SGX) [89] that is a set of x86 instructions used to protected memory regions called enclaves, whose contents are unable to be accessed by any process outside the enclave itself, including higher privilege levels process. Each participant uploads their training model and encrypted training data and then verifies whether their training code is executed in enclaves. After remote attestation was successful, the participant uploads their encryption key to decrypt their training data to train a shared model. Hunt *et al.* [90] developed a system, Chiron, where the data holders can collaboratively train an ML model on MLaaS while keeping their training data private. The source code is executed in a Ryoan [91] sandbox that provides a distributed sandbox that leverages hardware enclaves (e.g., SGX), without revealing the training code. The Chiron improves the efficiency of training by launching multiple enclaves, each enclaves running a part of the training data. However, the Chiron only allows the data owner to query a trained model via a simple interface when the training is finished. Therefore, the Chiron is not suitable for data owners who want to provide MLaaS.

Hynes *et al.* [92] developed Myelin, a privacy-preserving framework for DL, which supports both privacy-preserving during the training and testing phases. The Myelin utilizes TVM compiler [93] to compile a given model into a TVM-generated library that only includes the numerical operations needed for this model. After the compilation is complete, data owners can deploy an SGX [89] enclave and load the compiled library into the enclave to train the model without revealing their training data. Gu *et al.* [94] presented Deepenclave, a privacy-preserving system for DL inference using SGX [89]. The general idea of Deepenclave is to partition a given DNN model into FrontNet and BackNet. The FrontNet is located in a trusted environment, the BackNet is located in an untrusted environment and therefore protected by SGX [89]. The user's encrypted input is fed into the FrontNet, which is stored in a trusted environment. The intermediate output of the FrontNet inside the enclave is computed. Once computing is finished, the intermediate output is delivered to BackNet to compute the final output. Zeiler *et al.* [95] indicated that the shadow layers of DNN respond to low-level information of the input, such as edges, corners, whereas deep layers represent more abstract information associated with the final output. Since the first few layers of the DNN operate in a TEE, the privacy of the input sensitive information is guaranteed, but the final output of the model is plaintext.

The TEE uses hardware and software protection to isolate sensitive computing from untrusted software stacks and resolve the integrity and privacy of outsourced ML computing. However, these isolation guarantees also pay a considerable price for performance. Therefore, Tramer *et al.* [96] explored a pragmatic solution to improve the execution efficiency of DNN in TEE, which is based on the effective outsourcing scheme of matrix multiplication. They proposed a framework called Slalom, which uses Freivalds's algorithm [97] to verify the correctness of the linear operator. It encrypts the input through a pre-calculated blinding factor to protect privacy, ensuring that the DNN execution part is safely outsourced from TEE to a co-located, untrusted but faster device, such as GPUs. Hanzlik *et al.* [98] proposed MLCapsule, a system that can safely execute ML algorithms in offline deployed clients. Because the data is stored locally, and the protocol is transparent, the user's data security is guaranteed. The ML model is protected by TEE, MLCapsule calculates the ML model through the enclave in TEE. At the same time, the encrypted information sent by the service provider can only be decrypted by the enclave. To realize this method, the author proposed to encapsulate the standard ML layer in the MLCapsule layer and execute it in TEE. These MLCapsule layers can decrypt the network weights sent by the service provider, and through the layer-by-layer merge, to achieve large-scale network encapsulation. MLCapsule can also integrate advanced defense mechanisms against attacks on ML models, and the computational cost is meager.

5) MISCELLANEOUS DEFENSE

Tramer *et al.* [15] and Lee *et al.* [99] suggested that the efficiency of the model extraction attack can be decreased by omitting the confidence value or adding smart noise to the predicted probabilities. However, Juuti *et al.* [100] shown that model extraction is effective, even omitting prediction probabilities. Juuti *et al.* [100] proposed PRADA, a detection mechanism of model extraction attacks, to detect malware attack process by analyzing the distribution of user's queries when the adversary launch attack. Zhang *et al.* [101] presented a privacy-preserving approach that solves a regularized empirical risk minimization in distributed ML while providing $\alpha(t)$-DP for the final trained output. Hamm *et al.* [102] proposed a method to train a global differential private classifier by transferring the knowledge of the local classifier. Because the global classifier cannot directly access the sensitive training data, the ϵ-DP is provided. Long *et al.* [103] proposed a measuring mechanism, Differential Training Privacy (DTP), to estimate whether there is a risk of privacy leakage when the classifier is released. Although several approaches have been proposed to preserve privacy, it is not understood the correction, implementation, and privacy guarantee of these methods. Carlini *et al.* [104] explored the effectiveness of DP schemes against attacks. Rahman [105] and Jayaraman [106] attempted to analysis privacy cost of DP implementations. The difference is that Rahman *et al.* [105] evaluate not only DP but also its relaxed variants. Jia *et al.* [107] proposed MemGuard with formal utility loss guarantess against MIA under the black-box setting, which randomly inject noise to the confdence score predicted by the target classifer for query data sample.

Motivated by digital watermarking, researchers embed watermarking into DNN to protect the intellectual property of deep neural networks. After embedding watermarks to DNN models, once the models are stolen, the ownership can be verified by extracting watermarks from those models. Uchida *et al.* [47] first proposed a framework to embed watermarks into the parameters of DNN models via parameter regularizer during the training phase. Its disadvantage is that extracting the watermark from the model is needed to access all parameters of the model, which is impractical in real scenarios. Later, Zhang *et al.* [108] proposed a verification framework that can quickly verify the ownership of remotely deployed MLaaS with only given black-box access to the model. There are also several methods for the black-box watermark that target the zero bit (see reference [109]–[111] for details).

IV. SECURITY

In this section, we review and summarize the representative adversarial attacks and poisoning attacks in recent literature,

TABLE 6. An overview of adversarial attacks. The perturbation norm l_p is used to make adversarial examples imperceptible. The attack strength (higher for more ⋆) is based on the impression of the reviewed literature.

Threat Model	Method	Target/Non-target	Perturbation Norm	Type	Strength
White-box	L-BFGS [19]	Targeted	l_0	Optimization-based attack	⋆⋆
	FGSM [21]	Non-targeted	l_∞	Gradient-based attack	⋆
	BIM [113]	Non-targeted	l_∞	Gradient-based attack	⋆⋆⋆
	PGD [114]	Non-targeted	l_∞	Gradient-based attack	⋆⋆⋆
	JSMA [115]	Targeted	l_0	Gradient-based attack	⋆
	C&W [117]	Targeted	l_0, l_2, l_∞	Optimization-based attack	⋆⋆⋆
	Deepfool [118]	Non-targeted	l_2, l_∞	Gradient-based attack	⋆⋆
	UAP [119]	Non-targeted	l_2, l_∞	Optimization-based attack	⋆⋆⋆
	Obfucated Gradient Attack [123]	Targeted	l_∞	Optimization-based attack	⋆⋆⋆⋆
Black-box	One Pixel Attack [125]	Non-targeted	l_0	Optimization-based attack	⋆
	EOT [124]	Targeted	l_2	Transfer-based attack	⋆⋆⋆
	Boundary Attack [138]	Targeted, Non-targeted	l_2, l_∞	Decision-based attack	⋆⋆
	Biased Boundary Attack [139]	Targeted, Non-targeted	l_2, l_∞	Decision-based attack	⋆⋆⋆
	ZOO [134]	Targeted, Non-targeted	l_2	Confidence-based attack	⋆⋆⋆
	AutoZOOM [137]	Targeted, Non-targeted	l_2	Confidence-based attack	⋆⋆⋆

as well as defense strategies against adversarial attacks and poisoning attacks.

A. ADVERSARIAL ATTACKS

An overview of adversarial attacks is shown in Figure 5. Since the concept of the adversarial examples was proposed by Szegedy *et al.* [19], the researcher proposed several adversarial attack algorithms in recent literature. Due to high activity in this research direction, more attacks are likely to emerge in the future. Therefore, in this section, we discuss the representative white-box attack and black-box. Besides, we also provide a summary of the adversarial attack in Table 6.

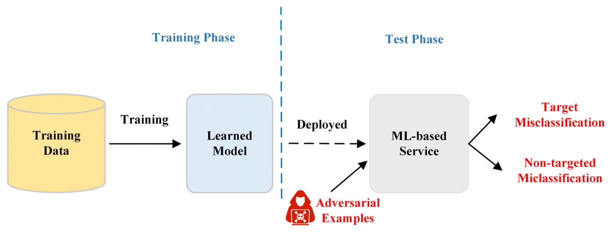


FIGURE 5. An overview of adversarial attacks.

1) WHITE-BOX

a: L-BFGS

Szegedy *et al.* [21] first demonstrated that the neural network is vulnerable to adversarial examples crafted by adding a small perturbation to benign input. The perturbation is imperceptible to the human visual system and can lead the model to predict wrongly with high confidence. The adversarial examples generated by solving the following equation:

$$\min_{\delta} ||\delta||_p \quad \text{s.t.}\, f(x+\delta) = t, \quad x+\delta \in [0,1]^m \tag{6}$$

As this is a hard problem, the authors turned the above equation into a convex optimization objective by using box-constrained L-BFGS algorithm [112]:

$$\min_{\delta} c\cdot|\delta| + J\,(x+\delta, t) \quad \text{s.t.}\, x+\delta \in [0,1]^m \tag{7}$$

where x represents the original image; J is the loss function of the model (e.g., cross-entropy); c is a hyperparameter, that is, the algorithm uses a linear search to find a constant that can produce the smallest distance adversarial sample; t is target label that is different from correct label y; δ means a perturbation.

b: FAST GRADIENT SIGN METHOD

Szegedy *et al.* [19] deemed that the existence of adversarial samples is caused by the nonlinearity and overfitting of neural network models. However, Goodfellow *et al.* [21] demonstrated that even a simple linear model is vulnerable to adversarial samples. They proposed the first Fast Gradient Sign Method (FGSM), an untargeted attack algorithm. Formally, the formulation of FGSM is as follows:

$$\eta = \epsilon\, \text{sign}(\nabla_x J(x, y_{true})) \tag{8}$$

where $\nabla_x J(x, y_{true})$ indicates the gradient of the adversarial loss $J(x, y_{true})$, sign$(\cdot)$ means the gradient direction. The adversarial perturbation η denotes the one-step gradient direction against the adversarial loss $J(x, y_{true})$, and ϵ controls the magnitude of the perturbation.

c: BASIC ITERATIVE METHOD/PROJECTED GRADIENT DESCENT

Kurakin *et al.* [113] extended the FGSM attack algorithm [21] with multiple small-step iterations and proposed Basic Iterative Method (BIM). In each iteration, they clip the pixel values to ensure that they are all in the ϵ-neighbourhood of the original image:

$$\begin{aligned} x_0^{adv} &= x, \\ x_{N+1}^{adv} &= Clip_{x,\epsilon}\{x_N^{adv} + \alpha\, \text{sign}(\nabla_x J(x, y_{true}))\}. \end{aligned} \tag{9}$$

Later, Mary *et al.* [114] extended the iterative attack proposed in [113] by iteratively applying Projected Gradient Descent (PGD) to search for a perturbation that can approximate the p-norm ball around an input. In [114], the authors proposed robust adversarial training which is based on optimizing the aforementioned saddle point formulation and uses PGD as a reliable first-order adversary.

d: JACOBIAN BASED SALIENCY MAP ATTACK

While most of the attacks focus on the ℓ_2 or ℓ_∞ norms, Papernot *et al.* [115] proposed Jacobian based Saliency Map Attack (JSMA), which uses the ℓ_0 norm to control the perturbation on a few pixels in the image, rather than on the whole image. In this attack, Papernot et. al. used the Jacobian matrix to compute the forward derivative of the DNN:

$$\nabla F(x) = \frac{\partial F(x)}{\partial x} = \left[\frac{\partial F_j(x)}{\partial x_i}\right]_{i \in 1..M_{in},\, j \in 1..M_{out}} \tag{10}$$

where M_{in} represents the number of input layers, and M_{out} represents the number of output layers. Then calculate the corresponding adversarial saliency map S through the forward derivative, and select the input feature $x[i]$ corresponding to the higher $S\left(x, y_{target}\right)[i]$ in the adversarial saliency map as the perturbation points. The algorithm sequentially selects the most efficient pixels in the adversarial saliency map and modifies these perturbation points until the maximum number of pixels allowed to change in the adversarial image is reached or the fooling succeeds.

e: C&W ATTACK

To demonstrate that defensive distillation [116] does not significantly increase the robustness of neural networks, Carlini and Wagner [117] proposed an optimization-based adversarial attack (C&W attack), which makes the perturbations imperceptible by restricting their ℓ_0, ℓ_2 and ℓ_∞ norm, its core formulation is as follows:

$$\min_{\delta} \|\delta\|_p + c \cdot f(x + \delta) \tag{11}$$

where δ denotes the adversarial perturbation, corresponding to the difference between the original image and the adversarial sample. The smaller the part, the less likely it is to be detected. And in their implementations, they used a modified binary search to choose the constant c. This $f(\cdot)$ denotes the objective function. They provide seven candidate functions. One of the practical functions in their experiments is as follows:

$$f\left(x'\right) = \max\left(\max\left\{Z\left(x'\right)_i : i \neq t\right\} - Z\left(x'\right)_t, -k\right) \tag{12}$$

where $Z(\cdot)$ denotes the softmax function of the model, k is a constant to control the confidence with the misclassification. This kind of attack is now used as a benchmark for many adversarial defense methods.

f: DEEPFOOL ATTACK

Moosavi-Dezfooli *et al.* [118] proposed a classifier-based linearized iterative adversarial attack (Deepfool), which generates a minimal adversarial perturbation sufficient to change the classification label. In the binary classification problem, the original image x iteratively advances in the direction perpendicular to the boundary of the classifier model $f(x)$. At each step, the perturbations are accumulated to form the final perturbation to the image. However, most neural networks are highly non-linear, so the problem extends from two-class to multi-class. The multi-classification problem can be regarded as a collection of multiple binary classification problems, that is, finding the minimum distance between the original sample and the boundary of the convex region where it is located, and approaching the classification boundary through multiple iterations, making the attack successful.

g: UNIVERSAL ADVERSARIAL PERTURBATIONS

Unlike previous attack methods, such as FGSM [21], Deepfool [118], which fool neural networks with a single perturbed image, Moosavi-Dezfooli *et al.* [119] proposed Universal Adversarial Perturbation attack (UAP) to find a universal perturbation that can be applied to all samples in the training data to fool the network. The algorithm is to find the universal perturbation δ that satisfies the following constraint:

$$\mathbb{P}_{x \sim \mu}(\hat{k}(x + \delta) \neq \hat{k}(x)) \geq 1 - \eta \quad \text{s.t. } \|\delta\|_p \leq \epsilon, \tag{13}$$

where μ denotes a distribution of images in $\mathbb{R}^d$ and $\hat{k}$ define a classification function that outputs an estimated label $\hat{k}(x)$ for each image $x \in \mathbb{R}^d$. The hyperparameter ϵ limits the magnitude of the universal perturbation δ, and η quantifies the fooling rate for all images $x \sim \mu$.

The algorithm gradually builds a universal perturbation through an iterative approach. In each iteration, the algorithm uses the DeepFool attack [118] to sequentially push all the images in the distribution μ to their respective decision boundaries, and project the updated disturbance to the ℓ_p ball of radius ϵ. The experiments show that only 4% of image perturbation can achieve 80% fool accuracy.

h: OBFUSCATED GRADIENT ATTACK

Many defenses rely on obfuscated gradients [120]–[122], which make it difficult for an attacker to obtain an effective gradient and defend against iterative optimization-based attacks. This phenomenon is considered to provide a false sense of security and leads to improper evaluation of adversarial defenses. Athalye *et al.* [123] found that defenses relying on this phenomenon can be circumvented. The authors identify three types of obfuscated gradients: 1) shattered gradients are non-existent or incorrect gradients, either intentionally caused by non-differentiable manipulations or unintentionally caused by numerical instability. 2) stochastic gradients depend on test-time randomness. 3) vanish/explode gradients in very deep calculations leading to unusable gradients. Correspondingly, Athalye *et al.* introduced three techniques to overcome the obfuscated gradients caused by these phenomenons: 1) using Backward Pass Differentiable Approximation (BPDA) to address shattered gradients. 2) applying Expectation Over Transformation (EOT) [124] to compute gradients of randomized defenses. 3) resolving vanishing/explosive gradients by re-parameterization and optimizing on the space where the gradients do not explode/vanishing.

2) BLACK-BOX

a: ONE PIXEL ATTACK

Su *et al.* [125] proposed one pixel attack based on the differential evolution algorithm [126]–[128], which is an extreme adversarial attack method, and only changing one pixel can make the network model misclassified. The algorithm iteratively modifies a single pixel and generates a sub-image, compares it with the parent image, and retains the sub-image with the best attack effect according to the selection criteria to achieve the adversarial attack. One pixel attack can be achieved by modifying a few different pixels, such as modifying 1, 3, or 5 pixels, and the success rates are 73.8%, 82.0%, and 87.3%, respectively.

b: EOT ATTACK

Previous works have shown that under image transformations in the real world [129], [130], such as the change of angle and viewpoint, the adversarial examples generated by these standard techniques (e.g., FGSM [21]) fail to maintain their adversarial properties [131], [132]. To address this issue, Athalye *et al.* [124] proposed the EOT attack algorithm. The core formula is as follows:

$$\arg\max_{x'} \mathbb{E}_{t\backsim T}[\log P(y_t|t(x'))-\lambda\|LAB(t(x'))-LAB(t(x))\|_2] \tag{14}$$

where x' denotes adversarial sample; x denotes original image; LAB [133] denotes a color space, and the ℓ_2 distance in this color space is the perceived distance; T denotes image transformation distribution.

The basic idea of the algorithm is to transform the distribution T that can model perceptual distortions, such as random rotation, transformation, or noise. EOT can not only simulate simple transformations, but also performs operations such as the three-dimensional rendering of textures.

c: ZEROTH ORDER OPTIMIZATION

Inspired by the C&W algorithm [117], Chen *et al.* [134] proposed the Zeroth Order Optimization (ZOO) method, which performs a black box attack against the target DNN by sending a lot of queries and observing responding output confidence values. ZOO uses zeroth-order optimization to approximate the network gradient while using dimensionality reduction, hierarchical attack, and importance sampling techniques to improve attack efficiency. The optimization scheme of the ZOO is consistent with the C&W algorithm, but the difference is that it is a black-box attack and cannot obtain the model gradient. ZOO uses the symmetric difference quotient [135] to compute the approximate gradient and the Hessian matrix. On the premise of obtaining the gradient and the Hessian matrix, the optimal perturbation is generated by the Stochastic coordinate descent method and using ADAM method [136] to improve the convergence efficiency.

d: AUTOENCODER-BASED ZEROTH ORDER OPTIMIZATION METHOD

Tu *et al.* [137] proposed a generic query-efficient black-box framework, called Autoencoder-based Zeroth Order Optimization Method (AutoZOOM), which can efficiently generate adversarial samples under the black-box setting. AutoZOOM leverages an adaptive stochastic gradient estimation strategy to stabilize the number of queries and the amount of perturbation, and simultaneously, train the automatic encoder offline with the unlabeled data, thereby speeding up the generation of adversarial samples. Compared with standard ZOO [134], AutoZOOM can reduce the number of queries while keeping attack effective and adversarial sample visual quality.

e: BOUNDARY ATTACK

Brendel *et al.* [138] pointed out that most methods used to generate adversarial perturbations rely either on detailed model information (gradient-based attacks) or on confidence scores such as class probabilities (score-based attacks). However, the network model information required for the attack cannot be obtained in a real scenario, and neither of these two methods is suitable for a real scenario. Therefore, Brendel *et al.* [138] proposed boundary attack, which only depend on class label. The algorithm starts with an already adversarial sample and then walks randomly along the decision boundary: 1) keeping in the adversarial area. 2) reducing the distance to the target image. Finally, the adversarial samples are iteratively generated.

f: BIASED BOUNDARY ATTACK

The boundary attack [138] uses an unbiased sampling method that sample perturbation candidates from a multi-dimensional normal distribution. Although this method is flexible, it is not efficient for robust models to craft adversarial examples. Brunner *et al.* [139] redefined the boundary attack as a biased sampling framework to improve attack efficiency. The three biases are as followed:

1) **Low Frequency Perturbations.** Since the perturbations generated by typical attack methods are high-frequency perturbations, most of the defense methods also defend against high-frequency perturbations. Based on this observation, they used low-frequency Perlin noise [140] to bypass the detection mechanism.
2) **Regional Masking.** They used the regional mask to update the areas where the difference between the sample and the original image is significant, and not update the extremely similar parts, thereby, effectively reducing the search space.
3) **Gradients from Surrogate Models.** The adversarial samples are transferability. That is, the gradient of the surrogate model is also helpful for attacking the target model. Therefore, they used the gradient of the surrogate model to guide the update direction of the boundary attack, which improves the attack efficiency.

To a certain extent, the above improvements improve the efficiency of the algorithm. However, the gradient of the surrogate model relies on the transferability of the model. Later, Chen *et al.* [141] further improved the boundary attack by utilizing Monte Carlo estimation to determine the direction of the gradient, which does not rely on the transferability of the model.

B. ADVERSARIAL EXAMPLES IN REAL WORLD

The adversarial attacks mentioned above are mostly in experimental settings. However, there have been a number of adversarial samples that have been applied in real-life applications such as road sign, objection detection, face recognition. In this section, we will present some real-life adversarial samples.

1) ROAD SIGN

Based on previous attack algorithms [117], [142], Evtimov *et al.* [11] proposed a general attack algorithm (robust physical perturbation) for generating visually adversarial perturbations with robustness under different physical conditions (e.g., distance, angle, distortion). The robust physical perturbations successfully deceive the road sign recognition system in a real driving environment. To confirm that robust physical perturbations are generalizable, they affixed graffiti generated with robust physical perturbations to a microwave oven and successfully mislead the Inception-v3 classifier [143] to recognize the microwave oven as a mobile phone. Lu et al [144] conducted experiments on a sample of physical confrontations between road sign images and detectors and showed that YOLO [145] and Faster-RCNN [146] and other detectors are not currently spoofed by the attack proposed by Evtimov *et al.* [11]. However, Eykholt *et al.* [147] claimed to be able to generate a small sticker to spoof the YOLO detector [40] and also to spoof the Faster-RCNN [146]. Chen *et al.* [148] further used the EOT technique [124], [149] to make the attack more robust and successfully mislead the Faster-RCNN detector [146].

2) OBJECTION DETECTION

Thys *et al.* [150] proposed a dynamic person target detection attack method based on the YOLO (You Only Look Once, You Only Look Once) [145] model. As shown in Figure 6, they successfully bypassed the detection model detection by optimizing the image to generate an adversarial patch and placing it in the center of the human body. They divided the optimized target loss function into three parts, namely, L_{nps}, L_{tv} and L_{obj}. L_{nps} indicates whether the color of the current patch can be applied to real life; L_{tv} reflects the smoothness of the image; and L_{obj} is the maximum target detection confidence level in the image. During the optimization process, the neural network model parameters were kept constant and only the adversarial patches were changed. And after each modified patch is rotated, scaled, and other basic transformations, it is applied to the dataset image again to improve the robustness of the adversarial patch so that it can successfully mislead the detection model.

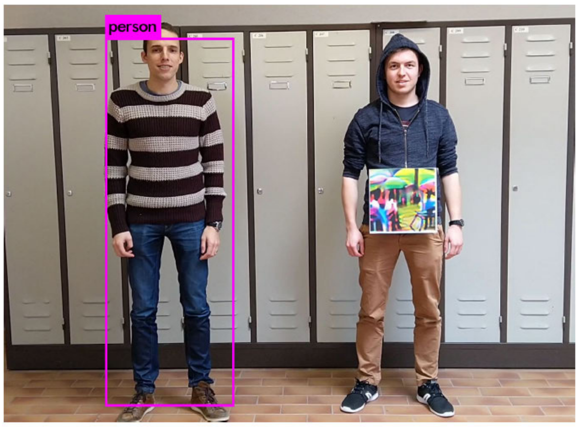


FIGURE 6. An example of objection detection [150].

3) CYBER SECURITY

Papernot et al [151] proposed a realistic adversarial sample attack on cyberspace, training an agent model on a synthetic dataset for generating adversarial samples and launching an attack against the remotely hosted neural networks of MetaMind, Amazon and Google. The results showed that the model's misclassification rates were 84.24%, 96.19%, and 88.94%, respectively. Similarly, Liu et al [142] also exploit the transferability of adversarial samples to carry out attacks, the basic idea of which is to generate an adversarial sample capable of making multiple models misclassified at the same time, which is used to carry out transfer attacks. This approach enabled a black-box attack on a large dataset ImageNet [8], and successfully attacked Clarifai, a commercial company providing state-of-the-art image classification services at the time.

In contrast to transfer attacks, Li *et al.* [152] improved on single-pixel and boundary attacks, respectively. Based on the single-pixel attack [125], it improves efficiency by gradually increasing the number of pixel modifications and incorporating the idea of semantic segmentation. In contrast, semantic segmentation and greedy ideas are introduced in the boundary attack [138] to improve efficiency. Li *et al.* [152] also conducted black-box attacks on computer vision-related services (e.g., image classification, object recognition, and illegal image detection) provided by five major cloud service providers, Amazon, Microsoft, Google, Baidu, and Alibaba, respectively, with a success rate of almost 100%.

4) FACE RECOGNITION

Sharif *et al.* [153] developed a systematic method of attacking against face recognition systems by simply adding a pair of eyeglass frames to make the face recognition system recognize errors. Zhou *et al.* [154] studied an interesting example of a real-world adversarial attack and found that infrared light can also be used to interfere with face recognition systems.

An attacker can install an LED light on the brim of a hat and use it to shine on the face to produce a purple light that is invisible to the human eye, but it can be captured by the camera sensor to evade detection by the facial recognition system.

C. ADVERSARIAL DEFENSES

Several defenses against adversarial attacks were proposed in recent studies, which grew along with three main directions: pre-processing, improving the model robustness, and malware detection, as shown in Figure 7.

1) **Pre-processing.** Pre-processing attempts to reduce the impact of the adversarial perturbation by performing some operation (e.g., denoising, randomization, reconstruction, scaling) on the input image. This defense mechanism is deployed before input and the first layer of the model, which usually requires no modification to the model and can be directly extended to the pre-trained model.
2) **Improving the Robustness of the Model.** Improving the robustness of the model aims to enhance the model's ability to resist adversarial samples by modifying the model architecture, training algorithms, regularization. The retraining and adversarial training usually entail significant computational overhead.
3) **Malware Detection.** Malware detection is deployed between input and the first layer of the model to detect the input to determine whether the input is an adversarial sample. The corresponding measures would be taken immediately to block this attack process if the user input is a malicious sample.

In this section, we describe these defense stragies from the three direction mentioned above.

1) PRE-PROCESSING

a: RANDOMIZATION

Wang *et al.* [155] proposed a novel adversarial defense approach that prevents adversaries from constructing impactful adversarial samples by randomly nullifying features within samples, which makes a DNN model non-deterministic and significantly reduce the effectiveness of this adversarial perturbation. Prakash *et al.* [156] proposed a method termed as pixel deflection to defend against adversarial examples, which consists of two parts: redistributing pixel values and wavelet-based denoising. The pixel deflection first makes full use of CNN's resistance to the natural noise by randomly replacing some pixels with randomly selected pixels from a small neighborhood. Then, the thresholding process in the wavelet domain is taken to effectively soften the corruption caused by redistributing pixels and some of the adversarial changes. The experiments demonstrated that combining these techniques can effectively reduce the impact of adversarial perturbation on the classifiers. Similarly, Ho *et al.* [157] proposed Pixel Redrawn (PR) as a defense against the adversarial examples, which redraws every pixel value of training images into a different pixel value. First, a prediction model is trained to generate a prediction image, and the range of the value of the image pixel value is divided into sections. The original image is fed into the prediction model to obtain a prediction image, and the interval of each pixel value of the prediction image is obtained. Then the random value in the interval is used to replace the pixel value of the original image. Finally, the modified image is fed into the classifier. Experimental results on several benchmark datasets [67], [158], [159] showed that the PR method not only relieves overfitting but it also boosts the robustness of the neural network.

b: IMAGE TRANSFORMATION

Dziugaite *et al.* [160] first demonstrated that the JPEG compression could reduce the classification errors caused by the adversarial examples generated by FGSM algorithm [21]. However, the defense effect of JPEG compression decreases with the increase in the magnitude of the perturbation. Das *et al.* [161] further studied that an important capability of JPEG compression is that it can remove high-frequency signal components inside the square blocks of the image, which is equivalent to selectively blur the image, which can eliminate adversarial perturbation on the image. Therefore, Das *et al.* [161] proposed a JPEG compression preprocessing module that can be quickly built on a trained network model to protect a model from multiple types of adversarial attacks. However, Guo *et al.* [129] found that total variance minimization [162] and image quilting [163] are stronger defense than deterministic denoising procedures (e.g., JPEG compression [160], bit depth reduction [164], non-local means [165]). Based on simple image transformations, Raff *et al.* [166] proposed to combinate a series of simple defenses (e.g, bit depth reduction [164], JPEG compression [160], wavelet denoising [167], mean filtering [168], non-local mean [165]) to build a strong defense mechanism to resist adversarial samples, and obfuscated gradient was taken into account [123]. The basic intuition is to stochastically select several transforms from a large number of random transforms, and apply each transform in a random order before the image is fed into the network. The method is also robust in large-scale datasets [169].

c: DENOISING NETWORK

The traditional denoising encoder [170] is a popular denoising model, but it cannot completely remove adversarial perturbations. To address this problem, Liao *et al.* [171] designed a High-level representation Guided Denoiser (HGD) as a defense against adversarial examples, which used U-net [172] as the denoising network model. The U-net [172] network model differs from traditional autoencoders in two ways. The first is that the denoising network does not use pixel-level construction loss function, but use the difference between top-level outputs of the target model induced by original and adversarial examples as the loss function. The second difference is that network learns adversarial perturbations rather than constructing the entire image. However,

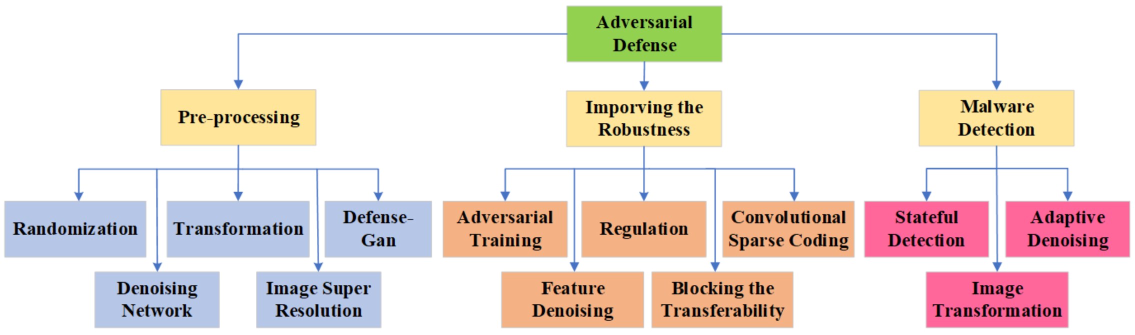


FIGURE 7. An overview of the adversarial defense.

Athalye *et al.* [173] pointed out that the HGD method cannot effectively prevent white-box attacks.

d: GAN-BASED DEFENSE

Samangouei *et al.* [122] proposed a defensive framework based on GAN [51]. The main idea is to train an adversarial generation network using the original dataset, and utilize the generator's expression to reconstruct a clean image that is similar to the original image but does not contain adversarial perturbations. An overview of the defense framework is shown in Figure 8. An adversarial example is reconstructed by the adversarial generative network, a reconstructed image similar to the original image is obtained, and the reconstructed image is fed into the target network model for classification. The introduction of random seeds makes the entire network model difficult to attack. However, this method cannot effectively prevent white-box attacks on the CIFAR-10 dataset [159], and Athalye *et al.* [123] used BPDA technology to attack this defense mechanism on the MNIST dataset [67]. However, the success rate was only 48%.

Likewise, based on bidirectional generative adversarial networks [174], [175], Bao *et al.* [176] proposed a novel defense approach, Featured Bidirectional Generative Adversarial Network (FBGAN), which can learn the latent semantic features of the image that is unchanged after the image is perturbed. After the bidirectional mapping, the adversarial data can be reconstructed to denoised data by extracting semantic features, which can be fed into any pre-trained classifier. The experiments showed that the FBGAN is effective for any pre-trained classifier under the white-box and gray-box attack.

e: IMAGE SUPER-RESOLUTION

Mustafa *et al.* [177] hypothesized that a generated-well image super-resolution model is enough to project the off-the-manifold adversarial samples into the natural image manifold. Therefore, Mustafa *et al.* [177] proposed a defense mechanism, deep image restoration networks, to defend against a wide range of recently proposed adversarial attacks. First, the adversarial perturbation is suppressed by wavelet domain filtering [178]. Second, the image super-resolution model [179] was used to enhance the visual quality of the image. Their method can easily complement the existing defense mechanism without retraining the model while improving the classification accuracy. The disadvantage is that it depends on the expressive power of the super-resolution model.

2) IMPROVING THE MODEL ROBUSTNESS

a: ADVERSARAL TRAINING

Goodfellow *et al.* [21] firstly proposed adversarial training to enhance the robustness of the model, Kurakin *et al.* [113] used the batch normalization [74] method and successfully extended it to the Inception-v3 model [143] and the ImageNet dataset [1]. However, its disadvantage is that it can only defend against single step attacks [21], and it can not defend against iterative attacks [114]. Chang *et al.* [180] proposed a training method based on dual adversarial samples, which can resist both single-step adversarial samples and iterative adversarial samples. Much adversarial training can only defend against specific adversarial attacks [21], [181], [182]. Madry *et al.* [114] proposed PGD attack for adversarial training. However, Madry *et al.* [114] only conducted adversarial training on the MNIST [67] and CIFAR-10 datasets [159]. Subsequently, Kannan *et al.* [183] successfully extended it to the ImageNet dataset [1]. They formed a pair of similar samples, and the degree of similarity of the model output of the paired samples is used as part of the loss function. Their method is robust on the ImageNet dataset [1] and exceeds the best performing integrated adversarial training method at that time [184]. Adversarial training is currently considered to be the most effective method to defend against adversarial samples. Its main disadvantage is that it is computationally expensive, and the improvement of adversarial training is still in progress.

b: REGULATION

The neural network model cannot learn the robust features, and slight changes in the image cause the classifier to decide wrongly. To address this problem, Liu *et al.* [185] proposed a feature prioritization model based on non-linear attention modules and L_2 feature regularization to make model classification depend on robust features. The attention module encourages the model to rely heavily on the robust features by assigning larger weights to them while suppressing non-robust features. The L_2 regularizer prompts the

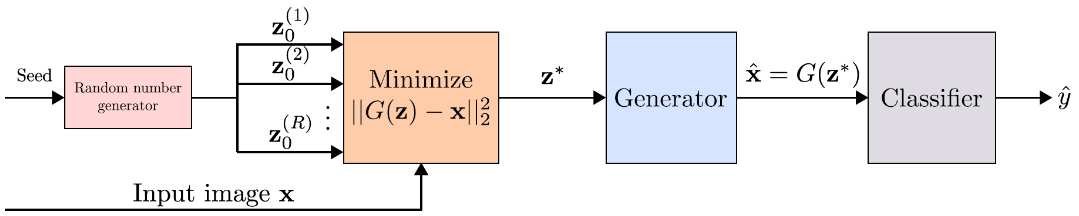


FIGURE 8. An overview of Defense-Gan framework [122].

extraction of similar essential features of the original image and the adversarial examples, effectively ignoring the added perturbations.

c: FEATURE DENOISING

Xie *et al.* [186] suggested that adversarial perturbations are amplified layer by layer in the network, resulting in a large amount of noise in the network's feature map. The features of the natural image mainly focus on the semantic features in the image, and the features of the adversarial examples are also activated in the semantically irrelevant areas. Therefore, Xie *et al.* [186] developed a new network architecture that improves the model robustness by performing feature denoising. Although the denoising module cannot improve the classification accuracy in the original dataset, the combination of the denoising module and adversarial training can significantly improve the robustness of the model under white-box and black-box attacks. In Competition on Adversarial Attacks and Defenses (CAAD) 2018, Their method still achieved a classification accuracy of 50.6% against 48 unknown attacks.

d: CONVOLUTIONAL SPARSE CODING

Based on convolutional sparse coding [187], [188], Sun *et al.* [189] proposed a novel defense method, which projects adversarial examples into a stratified low-dimensional quasi-natural image space, where the adversarial examples are similar to the natural image without adversarial perturbations. In the training phase, they introduced a novel Sparse Transformation Layer (STL) between the input and the first layer of the neural network to efficiently project images into quasi-natural image space and train the classification model with the image projected to the quasi-natural space. In the testing phase, the image that projects the original input to quasi-natural space is fed into the classification model. Compared with other adversarial defense methods for unknown attacks, Their method is more robust in terms of the size of adversarial perturbations, various image resolutions, and dataset size.

e: BLOCKING THE TRANSFERABILITY

The adversarial samples crafted by particular network models are likely to mislead other classifiers with different architectures or trained with different training data, which is known as transferability. To address this problem, Hosseini *et al.* [190] proposed an empty label approach to defending against adversarial sample transfer attacks under the black-box settings. The main idea is to add a NULL label to the output class and train the classifier to project the adversarial into the NULL label. The advantage of their method is the ability to classify the adversarial example into the NULL classes, rather than other error classes, effectively preventing the transfer problem of the adversarial samples, while also maintaining the accuracy of the model.

3) MALWARE DETECTION

a: STATEFUL DETECTION

Up to date, the defenses against white-box adversarial examples have proven difficult to achieve, and white-box attacks are not practical in real-world scenarios. The ML services provided by cloud platforms are generally based on queries. Therefore, Chen *et al.* [191] firstly proposed a black-box defense method based on stateful detection. Compared with the stateless defense currently researched, their method enhances the capabilities of the defender. An overview of the defense framework is shown in Figure 9. First, to compress the storage of user query records, similar encoding is used for compression encoding. Then, the user's query input is compared with previous records, and the distance d is calculated using the k nearest neighbor algorithm. If d is less than the threshold value, the user is considered to be performing a malicious attack. Under black-box attacks NES [192] and border attacks [138] setting, the experimental analysis showed that query-based black-box attacks usually require hundreds of thousands to millions of queries, which could easily trigger their defense mechanisms. Even if the defense mechanism is not triggered, the storage services required for the attack will consume many resources. The disadvantage of stateful detection methods is that it cannot defend against transfer attacks that do not require any query. However, this method can combine with adversarial training to defend against transfer attacks, which can compensate for the deficiency of stateful detection and make it perform better in the case of black-box attacks.

b: IMAGE TRANSFORMATION

Tian *et al.* [193] showed that adversarial samples are usually susceptible to image transformations, such as rotating and shifting, but the natural image is usually immune to these transformations. Based on this observation, they proposed a novel adversarial example detection mechanism that can effectively detect the adversarial attack. First, a set of transformation operations is performed on an input image

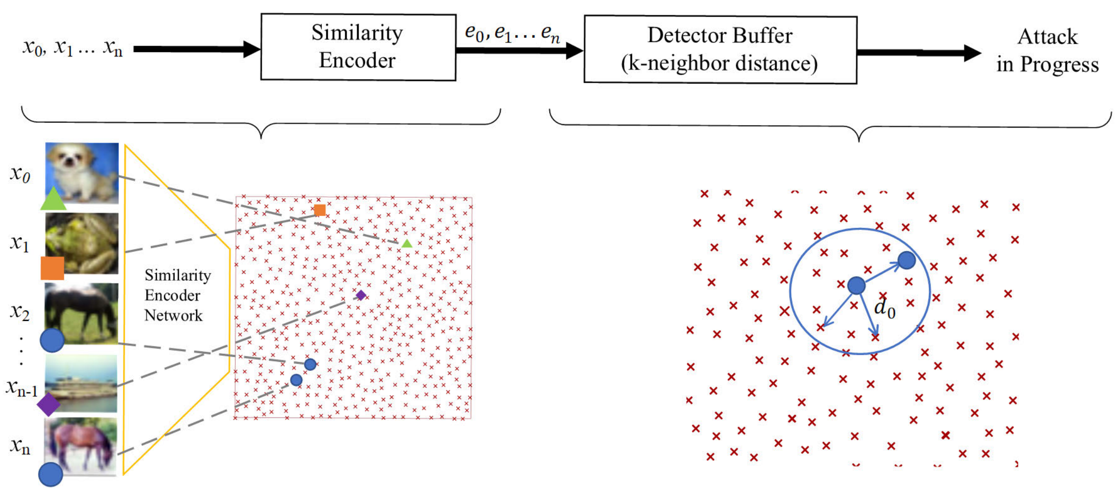


FIGURE 9. An overview of stateful detection framework [191].

to generate multiple transformed images. Then, the classification results of these transformed images are fed into the classifier to train a classifier that predicts whether the input image had been perturbed by an adversary or not. To defend against more complex white-box attacks, they also introduced randomness during the transformation process. The experiments showed that the detection rate of the adversarial samples crafted by the C&W [117] algorithm reaches 99%, and the detection rate of their method reaches more than 70% under white-box attacks setting. This approach is relatively simple, requiring only a simple transformation of the input image, however, the angle of transformation often affects the performance of the defense and may fail in the face of stronger adversarial attacks [123].

c: ADAPTIVE DENOISING

The traditional denoising has a significant effect on large noise. However, in the case of small noise, denoising can also make the image blurred, resulting in low classification performance. To address this problem, Liang *et al.* [194] introduced two typical image processing techniques: scalar quantization and smoothing spatial filter, to reduce the impact of adversarial perturbation on the classifier. The cross-entropy is employed as a metric to implement an adaptive noise reduction for different kinds of images. They first utilize cross-entropy to adaptively quantize the interval size and then determine whether spatial smoothing filtering is required. A classifier, respectively, classifies the denoised image and the original image. If the prediction of the classifier on the denoised image and the original image is consistent, the original image is considered to be a normal sample. Otherwise, it is considered to be an adversarial example. However, their method works poorly against attacks that modify only a fraction of the image pixels.

D. POISONING ATTACKS

An overview of poisoning attacks is shown in Figure 10. The adversary can reduce the performance of the model or manipulate the prediction of the model by injecting malicious samples into the model training data during the training phase. A large number of poisoning attacks were used in DL, which can be divided into three categories: accuracy drop attack, targeted misclassification attack, and backdoor attack. A summary of poisoning attacks and corresponding defense strategies is provided in Table 9.

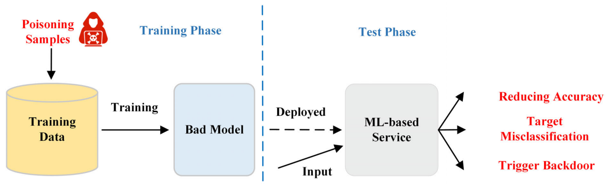


FIGURE 10. An overview of poisoning attacks.

1) ACCURACY DROP ATTACK

Muñoz-González *et al.* [197] proposed a novel poisoning attack based on gradient-based optimization, which targets a wider class of ML algorithms, such as DL architectures. The algorithm calculates the gradient by back propagation while reversing the underlying learning process, it traces the parameters executed during the learning and updates the entire sequence. The algorithm only requires the learning algorithm to update its parameters in a smooth manner during training (for example, gradient descent) to track these changes backward properly. They also demonstrated that attacks designed for specific learning algorithms are still effective for different learning algorithms.

The attacks that create poisoning samples based on back-gradient optimization is computationally intensive and inefficient. Inspired by GAN [49], Yang *et al.* [198] proposed a general method to accelerate the generation of poisoning data using generators and discriminators. The autoencoder is used as the generator in GAN to generate poisoned data. The target model is treated as a discriminator in GAN, which receives the poisoned data, calculates the loss, and then sends the calculated gradient back to the generator. The experiments showed that this algorithm has more than 230 times higher

TABLE 7. A summary of adversarial defense. The defense strength evaluates how powerful a defense is against different adversarial attacks (stronger for more ⋆), (⋆) means a defense was broken.

Type	Method	Advantage	Shortcoming	Strength
Pre-processing	Random Feature Nullification [155]	Increase the difficulty of crafting examples	Need to modify model architecture	⋆
	Pixel Deflection [156]	Simple, generic, low computional cost	Sensitive to the size of the perturbation and might be defeated by the strong attack (e.g., PGD attack [114])	⋆
	Pixel Redrawn [157]			⋆
	JPEG [161]			⋆
	TVM [129]			⋆⋆
	Defense-gan [122]			⋆
	Barrage Defense [166]	Not depend on obfucased gradient	Need a amount of simple defense methods	⋆ ⋆ ⋆
	FBGAN [176]	Generate clean examples with semantic features	Depend on the reconsturction accuracy of the FBGAN	⋆⋆
	HGD [171]	Effectively remove perturbations	Depend on the model's representation	⋆
	Image Super-resolution [177]	Improve the accuracy of clean examples	Depend on the representation of the image super-resolution.	⋆⋆
Improving the Robustness of the Model	Adversarial Training at Scale [113]	Extend adversarial training [21] to ImageNet [1] dataset	Not defend iterative attack and increase the overhead of training	⋆⋆
	Feature Prioritization and Regularization [185]	Defend white-box iterative attack and improve the decision boundary	The huge computational overhead of adversarial training	⋆ ⋆ ⋆
	e2SAD [180]			⋆ ⋆ ⋆
	PGD Adversarial Training [114]			⋆ ⋆ ⋆
	Logit Pairing [183]			⋆⋆
	Feature Denoising [186]			⋆ ⋆ ⋆
	BANG [195]	Not depend on data augumentation and adversarial example	Need adjust the hypeparameter with different dataset	⋆⋆
	Stratified Convolutional Sparse Coding [189]	Not need to modify original model	Need to train a additional network	⋆ ⋆ ⋆
	Block Transferablility [190]	Block transfer attack	Only apply for transfer attack.	⋆⋆
Malware Detection	Image Transformation [193]	Simple	The angle of trainsformation affect the performance	⋆
	Stateful Detection [191]	Block process of attack	Not defend transfer attack and need to store user query record	⋆ ⋆ ⋆
	I-defender [196]	Not need any knowledge of attack method	Might be bypassed by strong attack	⋆⋆
	Adaptive Noise Reduction [194]	Adaptive to remove perturbation	Not apply for attack that only modify a fraction of pixel	⋆⋆

computational efficiency than the direct gradient poisoning algorithm.

Similarity, Muñoz-González *et al.* [20] proposed to use GAN to generate poisoning samples that look like real data points, but it leads to reducing the accuracy of the classifier when it is used in training. They proposed a model called pGAN, which mainly consists of three parts: generator, discriminator, and target classifier. The purpose of the generator is to generate poisoning points to maximize the error of the target classifier, but minimize the ability of the discriminator to distinguish the poisoning points from the original data points. The purpose of the classifier is to minimize some of the loss functions evaluated on the training data that contains a small portion of poisoning points. The generator is utilized to maximize the convex combination of the discriminator and the classifier's loss on the poisoning data points to ensure that the model has a mechanism to control the detectability of the generated poisoning points. The aggressiveness of the attack can be controlled by the parameters of the weighted sum of the two losses. The pGAN can be utilized for system testing of ML classifiers at different risk levels by controlling the trade-off between effectiveness and detectability of attacks.

2) TARGETED MISCLASSIFICATION ATTACK

The concept of targeted misclassification attack was first proposed by Koh and Liang [199]. DL system performs well in various applications, but, DL models are extremely poor in interpretability. Therefore, Koh and Liang [199] attempted to use a classic influence function from robust statistics [200] to explain the prediction of the black-box model. The influence functions track the prediction of the model to identify the training points that are most relevant to a given prediction. The adversary observes the changes predicted by the model through up weighting a training point and perturbing a training input and leverage Euclidean distance to find the training point most relevant to the test point. A poisoning sample is iteratively generated by modifying the influence of the training point on the test point. Later, Shafahi *et al.* [201] proposed a new poisoning attack, clean-label attack, which generates poisoned instances via feature collisions. The clean-label attacks can control the behavior of the classifier on a specific test instance while keeping the performance of the entire classifier. Besides, the adversary is not required to have the ability to control the label of the training data. Compared with the poisoning attack proposed by Koh and Liang [199],

TABLE 8. An overview of poisoning attacks. The attack strength (higher for more stars) is based on the impression of the reviewed paper.

Method	Type	Advantage	Shortcoming	Strength
Back-Gradient Optimization [197]	Reducing Accuracy Attack	Simple and Effective	Limited to the model trained with a large number of training points	⋆
Auto-Encoder Generator [198]		Accelerate the generation rate of the poisoned data	Need train an extra model as generator	⋆⋆
GAN Poisoning Attack [20]		Manipulate a trade-off between effectiveness and detectability of the attack	Need train an extra GAN (generator and discriminator)	⋆⋆⋆⋆
Influence Function [199]	Target Classification Attack	Only the final fully connected layer of the network was retrained	Low attack success rate and need the full knowledge of the model and data	⋆⋆
Targeted Clean-Label Poisoning Attacks [201]		No need to control over the labeling of training data	Need has knowledge of the model and its parameters	⋆⋆⋆
Transferable Clean-Label Poisoning Attacks [12]		No need to access to the target model and dataset	Difficult to achieve the desired results under the end-to-end training model	⋆⋆⋆⋆
BadNet [13]	Backdoor Attack	Simple and Effective	Need to tamper with the original training process	⋆⋆
Trojaning Attack [14]		Generate stealthy semantic trojan trigger	Unequally distributed mispredicted results	⋆⋆⋆
Targeted Backdoor Attacks [202]		About 50 poisoned samples, the attack success rate can reach more than 90%	Auxiliary pristine data can reduce the attack effect	⋆⋆⋆⋆
Invisible Backdoor Attacks [203]		Invisible to detector and human body inspection	Not practical under the data collection strategy	⋆⋆⋆⋆

the clean-label attack has better performance on the same dog-vs-fish classification task. Under the end-to-end training scenario, only 50 poisoned instances are needed to achieve an attack success rate of 60%. Furthermore, Zhu *et al.* [12] proposed transferable clean-label poisoning attacks under the black box model through the convex polytope strategy, which has a higher attack success rate than feature collisions [201] in the black-box model. The adversary has no knowledge of the parameter of the target model but has knowledge of the similar training data as the target model. Then, the adversary can train a substitute model on this training data and optimizes a novel objective that makes the poisons form a convex polytope to wrap the target image. In this way, a linear classifier that is overfitted on the poisoning dataset can classify the target image into the same class as the poisoning data.

3) BACKDOOR ATTACK

Due to the expensive cost of training models, several users outsource the training process to the cloud servers or rely on pre-trained models and then fine-tune specific tasks. Gu *et al.* [13] proposed BadNet, a maliciously trained backdoor network, which performs well in the training and verification phases, but can mislead specific data during the testing phase. The adversary selects a backdoor trigger composed of pixels and related color intensity, which can be of any shape, such as square. The algorithm assumes that the adversary can control the entire training process (e.g., parameters, learning rate) and then use the poisoning training data to construct a backdoor network sensitive to specific backdoor triggers. The backdoor triggers lead the neural network to misclassify the backdoor data into the target label specified by the adversary. The experiments showed that a backdoor network could attack the network model trained on the MNIST datasets [67] with a success rate of over 99% without affecting the performance of the neural network. Liu *et al.* [14] proposed a trojaning attack that does not depend on the original training data. The trojaning attack takes the existing model and target prediction output as input, and modifies the model to generate a small part of the input data, called the trojan trigger. Any valid input with a trojan trigger of the model will cause the mutated model to output a specific prediction category. The trojaning attack is implemented in three steps: 1) trojan trigger generation. The trojan trigger is a special input that triggers the behavior of the trojan neural network to be misbehaving. 2) training data generation. As the trojaning attack has no knowledge of the original training data, it is necessary to derive a set of training data to retrain the model. 3) retraining model. Due to the retraining of the entire model is very expensive, they used the trigger and the reverse-engineered images to retrain a part of the model.

Chen *et al.* [202] proposed targeted backdoor attacks that can be applied to very weak threat models: 1) the adversary has no knowledge of the target model. 2) the adversary can inject a small portion sample into the training data. 3) the backdoor key is hard to be noticed by human visual system. The attack strategy is divided into two steps. First, a poisoning sample is generated and added to the training data. Then, a sample backdoor is created to make the neural network misclassify the sample as the target label. The previous backdoor attacks [13], [14], [202] mainly focused on the establishment of triggers. Furthermore, Li *et al.* [203] focused on how to make triggers invisible and proposed an invisible backdoor attack, which makes the backdoor attacks hardly perceptible for the human vision system, while ensuring that the neural network can still recognize backdoor triggers. They used the Perceptual Adversarial Similarity Score (PASS) to define people's ability to recognize triggers and used ℓ_2 and ℓ_0 regularization to hide the trigger in the entire image, making the trigger less obvious.

E. POISONING DEFENSES

1) DEFENSE AGAINST ACCURACY DROP/TARGETED MISCLASSIFICATION ATTACK

For the almost infinite possible attack space, it is impossible to draw conclusions based on experience alone, whether the defense against the known attack set can defend against future attacks. Therefore, Steinhardt *et al.* [204] proposed a framework to solve this problem by removing outliers and keeping them out of the feasible set. For binary classification, a natural defense strategy is to find the centroids of positive and negative classes and remove points that are too far from the corresponding centroids. There are two ways to implement it. The sphere defense removes points outside the radius of sphere, and the slab defense, which first projects points onto a straight line between the centroids and then discards points that are too far away from the straight line.

The so-called ''optimal'' attacks choose poisoning samples to maximize the damage to target models [205]–[207]. However, such attacks usually only focus on the learning algorithm while ignoring the data preprocessing step. Base on this observation, Paudice *et al.* [208] proposed a defense strategy based on data pre-filtering with outlier detection to mitigate the effects of optimal poisoning attacks. The method uses distance-based anomaly detection to detect poisoned samples using a small number of trusted data points. They split a small fraction of trusted data D into different categories, i.e., D_+ and D_-. Then, these curated data were used to train a distance-based outlier detector for each category. Next, the Empirical Cumulative Distribution Function (ECDF) was used to calculate the outlier detection threshold based on the outliers score. Finally, these samples were filtered by the threshold to get clean datasets to retrain the model.

2) DEFENSE AGAINST BACKDOOR ATTACKS

Liu *et al.* [209] proposed a ''fine-pruning'' method, that is, a method combining pruning and fine-tuning. The pruning defense resists backdoor behavior by removing neurons that are dormant for clean inputs in the backdoor network. For pruning defense, they developed a stronger pruning-aware attack, which evades pruning defense by focusing clean and backdoor behavior on the same set of neurons. In order to defend against stronger pruning-aware attack, they proposed a fine-tuning defense strategy, which locally retrains the network on clean training data, However, since the accuracy of the backdoored DNN on clean input does not depend on the weight of the backdoor neurons, the defense effect of fine-tuning defense is not significant. Finally, by combining the advantages of pruning and fine-tuning, they putted forward the method of fine-pruning. The idea behind fine-tuning first prunes the DNN returned by the attacker and then fine-tuning the pruned network. In some cases, this method can even reduce the success rate of backdoor attacks to 0%.

Chen *et al.* [210] hypothesized that the features activated by the activation function between standard and backdoor samples are different from the neural network. Hence, Chen *et al.* [210] proposed an Activation Clustering (AC) method to detect poisoning training samples. The AC method analyzes the activations of the last hidden layer of the neural network to determine whether the input is poisoned. The AC method is also the first method to detect poisoning data for inserting backdoors and repair models that do not require verified and trusted datasets. They have shown that the effectiveness of the AC method at detecting and repairing backdoors. In addition, the experiment also demonstrated that the AC method is robust to multimodal classes and complex poisoning schemes.

V. FUTURE RESEARCH WORK

A. DL PRIVACY

1) LIGHTWEIGHT PRIVACY-PRESERVING TECHNIQUES

The recent work has proposed several regarding the privacy-preserving DL technique to protect the privacy of sensitive data. However, there is still a lot of work to do before it is applied in practical application. The biggest deflect of privacy-preserving DL techniques are computation cost. Due to the non-linear operation of DL, the computation cost is enormous, which seriously reduces the availability of the DL. An important challenge regarding privacy-preserving DL techniques is to decrease the overhead of privacy-preserving.

2) INTELLECTUAL PROPERTY PROTECTION OF DL MODEL

A well-performing ML model requires massive amounts of training data, a large amount of hardware resources, and a lot of time for parameter tuning. Therefore, the labeled training dataset, model architecture, and model parameters have been considered as a commercial intellectual property and therefore need to be protected. Currently, there are only a few works of intellectual property protection for watermark-based machine learning models [47], [108], [109], and the effectiveness is still difficult to secure. For neural network models, a more effective and secure intellectual property protection method is still to be solved.

3) GENERIC PRIVACY-PRESERVING TECHNIQUES

Most current privacy-preserving technologies can only make privacy predictions during the testing phase, and only a few solutions can be trained on encrypted data. Moreover, the inference privacy-preserving models are trained on the unencrypted data on unencrypted typical models, and then, the trained weights and biases are applied to different models in which the activation function is replaced with a simple activation function, such as a square function. Differences between the trained and inferred models typically result in a high degree of degradation in performance of the model. Therefore, the current privacy-preserving techniques still require a lot of custom work for each DL model. A general privacy-preserving framework is a challenge to be solved in the future.

B. DL SECURITY

1) ADVERSARIAL EXAMPLES IN REAL WORLD

The adversarial examples was first proposed for the image classification task [19], but recent works have found adversarial for cyberspace attack [151], [152], stop sign recognition [211], objection detection [150], [212], semantic segmentation [213], [214], face recognition [153], authorship

TABLE 9. An overview of poisoning defenses. The defense strength (higher for more stars) is based on the impression of the reviewed paper.

Method	Type	Advantage	Shortcoming	Strength
Certified Defenses [204]	Defense against Accuracy Drop/Misclassification Attack	Detect potential attacks	Need the trusted dataset to train the detector	⋆⋆
Anomaly Detection [208]		Mitigate the effects of optimal poisoning attacks	Need the trusted dataset to train the detector	⋆⋆⋆
Fine-Pruning [209]	Defense against Backdoor	Just need to fine-tuning	Affect the accuracy of the trained model	⋆⋆⋆
Activation Clustering [210]		No affect the accuracy of the model	Need access to the trojaned samples	⋆⋆⋆⋆

recognition [215]. Although there are already adversarial samples for different application scenarios, there is still a large gap with the application scenarios in the real world. For example, the adversarial example is sensitive to physical environments such as light, angle, transformation. The robust adversarial examples against real physical environments raised the interest of a large number of researchers.

2) ROBUSTNESS ADVERSARIAL EXAMPLES

The adversarial attacks have evolved from gradient-based attacks [21], [114] to decision-based attacks [192], [216], and the amount of information required for their attacks has gradually decreased and has been successfully applied to real scenarios. The algorithm for generating adversarial samples has made great progress in recent years, but there are still many deficiencies and limitations. For example, there are too many queries and the attacks are not stable enough on different application scenarios. In the future, the more efficient, smaller disturbance, and stable black box attack algorithms are still the focus of research.

3) DEFENSES AGAINST ADVERSARIAL EXAMPLES

As mentioned in this paper, the majority of defense strategies are only effective against specific known attacks and are not well generalized in unknown attacks. To the best of our knowledge, there is still no defensive method against adversarial examples that can completely defend against white-box attacks. Adversarial training is considered to be the most effective method of defending against adversarial attacks, but it has the drawback of huge computational overhead. In short, how to train a robust model with negligible overhead remains a hot topic for future research.

4) SYSTEMATIC EVALUATION ADVERSARIAL DEFENSES

Adaptive adversary have (rightfully) become the de facto standard for evaluating adversarial defenses. However, Tramer *et al.* [217] showed that the evaluation of adaptive adversary in many published papers is incomplete or flawed. How to properly perform an adaptive attack assessment against adversarial instance defenses is particularly important. A system evaluation criterion and metrics for defense mechanisms is still a problem to be solved.

5) WHY ARE THE REASONS FOR THE PRESENCE OF ADVERSARIAL EXAMPLES?

Why are DL models so vulnerable to adversarial samples? There is some discussion in the current research, but there is still a lack of consensus. Ilyas *et al.* [218] pointed out that models trained on the same dataset tend to learn similar non-robust features, which accounts for the transferability of adversarial examples. However, why DL model tend to learn non-robust features and how to make them learn robust features is still an open question.

VI. SUMMARY

DL has been extensively applied in a variety of application domains such as speech recognition, medical diagnosis, but the recent security and privacy issues of DL have raised concerns of the researcher. One of the keys to the rise of DL is to rely on the vast quantities of data, which is also accompanied by the risk of privacy leakage. In this paper, we first describe the potential risks of DL and then reviewed the two types of attack: model extraction attack and model inversion attack in DL and four typical defense technologies for protecting the data privacy of user: DP, HE, SMC, and TEE. We then investigated two types of attacks: adversarial attacks and poisoning attacks. In adversarial attacks, we reviewed the representative black box and white box attack in recent studies, and reviewed the adversarial attacks under the physical condition. Regarding the defense methods of security, we describe the defense approach from three aspects: pre-processing, improving model robustness, and malware detection. Finally, the unresolved problems and the direction of future work are discussed.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[2] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[3] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 30–42, Jan. 2012.

[4] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1988–1996.

[5] Y. Sun, D. Liang, X. Wang, and X. Tang, "DeepID3: Face recognition with very deep neural networks," 2015, *arXiv:1502.00873*. [Online]. Available: https://arxiv.org/abs/1502.00873

[6] D. Shen, G. Wu, and H.-I. Suk, "Deep learning in medical image analysis," *Annu. Rev. Biomed. Eng.*, vol. 19, pp. 221–248, Jun. 2017.

[7] L. Yu, S. Wang, and K. Lai, "Credit risk assessment with a multistage neural network ensemble learning approach," *Expert Syst. Appl.*, vol. 34, no. 2, pp. 1434–1444, Feb. 2008.

[8] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, ''Mastering the game of go with deep neural networks and tree search,'' *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[9] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, and J. Oh, ''Grandmaster level in StarCraft II using multi-agent reinforcement learning,'' *Nature*, vol. 575, no. 7782, pp. 350–354, Nov. 2019.

[10] J. Xiong, R. Bi, M. Zhao, J. Guo, and Q. Yang, ''Edge-assisted privacy-preserving raw data sharing framework for connected autonomous vehicles,'' *IEEE Wireless Commun.*, vol. 27, no. 3, pp. 24–30, Jun. 2020.

[11] 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 Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1625–1634.

[12] C. Zhu, W. R. Huang, A. Shafahi, H. Li, G. Taylor, C. Studer, and T. Goldstein, ''Transferable clean-label poisoning attacks on deep neural nets,'' 2019, *arXiv:1905.05897*. [Online]. Available: https://arxiv.org/abs/1905.05897

[13] T. Gu, B. Dolan-Gavitt, and S. Garg, ''Badnets: Identifying vulnerabilities in the machine learning model supply chain,'' 2017, *arXiv:1708.06733*. [Online]. Available: https://arxiv.org/abs/1708.06733

[14] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, ''Trojaning attack on neural networks,'' in *Proc. NDSS*, 2018, pp. 1–17.

[15] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, ''Stealing machine learning models via prediction apis,'' in *Proc. 25th USENIX Secur. Symp. (USENIX Secur.)*, 2016, pp. 601–618.

[16] B. Wang and N. Z. Gong, ''Stealing hyperparameters in machine learning,'' in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 36–52.

[17] 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.

[18] Y. Long, V. Bindschaedler, L. Wang, D. Bu, X. Wang, H. Tang, C. A. Gunter, and K. Chen, ''Understanding membership inferences on well-generalized learning models,'' 2018, *arXiv:1802.04889*. [Online]. Available: https://arxiv.org/abs/1802.04889

[19] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, ''Intriguing properties of neural networks,'' 2013, *arXiv:1312.6199*. [Online]. Available: https://arxiv.org/abs/1312.6199

[20] L. Muñoz-González, B. Pfitzner, M. Russo, J. Carnerero-Cano, and E. C. Lupu, ''Poisoning attacks with generative adversarial nets,'' 2019, *arXiv:1906.07773*. [Online]. Available: https://arxiv.org/abs/1906.07773

[21] I. J. Goodfellow, J. Shlens, and C. Szegedy, ''Explaining and harnessing adversarial examples,'' 2015, *arXiv:1412.6572*. [Online]. Available: https://arxiv.org/abs/1412.6572

[22] N. Akhtar and A. Mian, ''Threat of adversarial attacks on deep learning in computer vision: A survey,'' *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[23] H. C. Tanuwidjaja, R. Choi, and K. Kim, ''A survey on deep learning techniques for privacy-preserving,'' in *Proc. Int. Conf. Mach. Learn. Cyber Secur.* Cham, Switzerland: Springer, 2019, pp. 29–46.

[24] A. Boulemtafes, A. Derhab, and Y. Challal, ''A review of privacy-preserving techniques for deep learning,'' *Neurocomputing*, vol. 384, pp. 21–45, Apr. 2020.

[25] X. Yuan, P. He, Q. Zhu, and X. Li, ''Adversarial examples: Attacks and defenses for deep learning,'' *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2805–2824, Sep. 2019.

[26] Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, and V. C. M. Leung, ''A survey on security threats and defensive techniques of machine learning: A data driven view,'' *IEEE Access*, vol. 6, pp. 12103–12117, 2018.

[27] N. Papernot, P. McDaniel, A. Sinha, and M. P. Wellman, ''SoK: Security and privacy in machine learning,'' in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Apr. 2018, pp. 399–414.

[28] C. Dwork, ''Differential privacy: A survey of results,'' in *Proc. Int. Conf. Theory Appl. Models Comput.* Berlin, Germany: Springer, 2008, pp. 1–19.

[29] P. Paillier, ''Public-key cryptosystems based on composite degree residuosity classes,'' in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1999, pp. 223–238.

[30] X. Liu, R. H. Deng, W. Ding, R. Lu, and B. Qin, ''Privacy-preserving outsourced calculation on floating point numbers,'' *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 11, pp. 2513–2527, Nov. 2016.

[31] X. Liu, K.-K.-R. Choo, R. H. Deng, R. Lu, and J. Weng, ''Efficient and privacy-preserving outsourced calculation of rational numbers,'' *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 1, pp. 27–39, Jan. 2018.

[32] X. Liu, R. H. Deng, K.-K.-R. Choo, and J. Weng, ''An efficient privacy-preserving outsourced calculation toolkit with multiple keys,'' *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 11, pp. 2401–2414, Nov. 2016.

[33] R. L. Rivest, A. Shamir, and L. Adleman, ''A method for obtaining digital signatures and public-key cryptosystems,'' *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[34] T. Elgamal, ''A public key cryptosystem and a signature scheme based on discrete logarithms,'' *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985.

[35] C. Gentry, ''Fully homomorphic encryption using ideal lattices,'' in *Proc. 41st Annu. ACM Symp. Theory Comput.*, New York, NY, USA, 2009, pp. 169–178.

[36] C. Gentry, A. Sahai, and B. Waters, ''Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based,'' Cryptol. ePrint Arch., Berlin, Germany, Tech. Rep. 2013/340, 2013.

[37] A. Lopez-Alt, E. Tromer, and V. Vaikuntanathan, ''On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption,'' Cryptol. ePrint Arch., New York, NY, USA, Tech. Rep. 2013/094, 2013.

[38] J. Fan and F. Vercauteren, ''Somewhat practical fully homomorphic encryption,'' Cryptol. ePrint Arch., Berlin, Germany, Tech. Rep. 2012/144, 2012.

[39] Z. Brakerski, ''Fully homomorphic encryption without modulus switching from classical GapSVP,'' Cryptol. ePrint Arch., Berlin, Germany, Tech. Rep. 2012/078, 2012.

[40] J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig, ''Improved security for a ring-based fully homomorphic encryption scheme,'' Cryptol. ePrint Arch., Berlin, Germany, Tech. Rep. 2013/075, 2013.

[41] X. Liu, R. H. Deng, K.-K.-R. Choo, Y. Yang, and H. Pang, ''Privacy-preserving outsourced calculation toolkit in the cloud,'' *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 5, pp. 898–911, Sep. 2020.

[42] A. C.-C. Yao, ''How to generate and exchange secrets,'' in *Proc. 27th Annu. Symp. Found. Comput. Sci. (SFCS)*, Oct. 1986, pp. 162–167.

[43] O. Goldreich, S. Micali, and A. Wigderson, *How to Play Any Mental Game, or a Completeness Theorem for Protocols with Honest Majority*. New York, NY, USA, 2019, pp. 307–328, doi: 10.1145/3335741.3335755.

[44] M. O. Rabin, ''How to exchange secrets with oblivious transfer,'' IACR Cryptol. ePrint Arch., Tech. Rep. 2005/187, 2005.

[45] A. Shamir, ''How to share a secret,'' *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[46] T. Wang and F. Kerschbaum, ''Attacks on digital watermarks for deep neural networks,'' in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 2622–2626.

[47] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, ''Embedding watermarks into deep neural networks,'' in *Proc. ACM Int. Conf. Multimedia Retr.*, 2017, pp. 269–277.

[48] D. Hitaj and L. V. Mancini, ''Have you stolen my model? Evasion attacks against deep neural network watermarking techniques,'' 2018, *arXiv:1809.00615*. [Online]. Available: http://arxiv.org/abs/1809.00615

[49] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, ''ML-leaks: Model and data independent membership inference attacks and defenses on machine learning models,'' 2018, *arXiv:1806.01246*. [Online]. Available:https://arxiv.org/abs/1806.01246

[50] B. Hitaj, G. Ateniese, and F. Perez-Cruz, ''Deep models under the GAN: Information leakage from collaborative deep learning,'' in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 603–618.

[51] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, ''Generative adversarial nets,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[52] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, ''Exploiting unintended feature leakage in collaborative learning,'' in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 691–706.

[53] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro, ''LOGAN: Membership inference attacks against generative models,'' *Proc. Privacy Enhancing Technol.*, vol. 2019, no. 1, pp. 133–152, Jan. 2019.

[54] M. Nasr, R. Shokri, and A. Houmansadr, ''Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning,'' in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 739–753.

[55] M. Abadi, A. Chu, I. Goodfellow, H. B. Mcmahan, I. Mironov, K. Talwar, and L. Zhang, ''Deep learning with differential privacy,'' in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 308–318.

[56] C. Dwork, G. N. Rothblum, and S. Vadhan, ''Boosting and differential privacy,'' in *Proc. IEEE 51st Annu. Symp. Found. Comput. Sci.*, Oct. 2010, pp. 51–60.

[57] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou, "Differentially private generative adversarial network," 2018, *arXiv:1701.04722*. [Online]. Available: https://arxiv.org/abs/1701.04722

[58] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.

[59] G. Acs, L. Melis, C. Castelluccia, and E. De Cristofaro, "Differentially private mixture of generative neural networks," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1109–1121, Jun. 2019.

[60] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[61] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*. [Online]. Available: https://arxiv.org/abs/1312.6114

[62] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.

[63] N. Phan, Y. Wang, X. Wu, and D. Dou, "Differential privacy preservation for deep auto-encoders: An application of human behavior prediction," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 1309–1316.

[64] N. Phan, X. Wu, and D. Dou, "Preserving differential privacy in convolutional deep belief networks," *Mach. Learn.*, vol. 106, nos. 9–10, pp. 1681–1704, Oct. 2017.

[65] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive laplace mechanism: Differential privacy preservation in deep learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2017, pp. 385–394.

[66] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," 2016, *arXiv:1610.05755*. [Online]. Available: https:/arxiv.org/abs/1610.05755

[67] Y. LeCun and C. Cortes. (2010). *MNIST Handwritten Digit Database*. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[68] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Ú. Erlingsson, "Scalable private learning with PATE," 2018, *arXiv:1802.08908*. [Online]. Available: https://arxiv.org/abs/1802.08908

[69] A. Triastcyn and B. Faltings. (2018). *Generating Differentially Private Datasets Using GANs*. [Online]. Available: https://openreview.net/forum?id=rJv4XWZA-

[70] P. Xie, M. Bilenko, T. Finley, R. Gilad-Bachrach, K. Lauter, and M. Naehrig, "Crypto-nets: Neural networks over encrypted data," 2014, *arXiv:1412.6181*. [Online]. Available: https://arxiv.org/abs/1412.6181

[71] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 201–210.

[72] J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," in *Proc. IMA Int. Conf. Cryptogr. Coding*. Berlin, Germany: Springer, 2013, pp. 45–64.

[73] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network," *IACR Cryptol. ePrint Arch.*, vol. 2017, p. 35, Mar. 2017.

[74] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[75] E. Hesamifard, H. Takabi, and M. Ghasemi, "CryptoDL: Deep neural networks over encrypted data," 2017, *arXiv:1711.05189*. [Online]. Available: https://arxiv.org/abs/1711.05189

[76] F. Bourse, M. Minelli, M. Minihold, and P. Paillier, "Fast homomorphic evaluation of deep discretized neural networks," in *Proc. Annu. Int. Cryptol. Conf.* Cham, Switzerland: Springer, 2018, pp. 483–512.

[77] A. Sanyal, M. J. Kusner, A. Gascón, and V. Kanade, "TAPAS: Tricks to accelerate (encrypted) prediction as a service," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 4490–4499.

[78] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachene, "Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds," in *Proc. Int. Conf. theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2016, pp. 3–33.

[79] M. Courbariaux and Y. Bengio, "Binarynet: Training deep neural networks with weights and activations constrained to $+1$ or -1," 2016, *arXiv:1602.02830*. [Online]. Available: https://arxiv.org/abs/1602.02830

[80] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 5, pp. 1333–1345, May 2018.

[81] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2015, pp. 1310–1321.

[82] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 19–38.

[83] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via MiniONN transformations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 619–631.

[84] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: Scalable provably-secure deep learning," in *Proc. 55th Annu. Design Autom. Conf.*, 2018, pp. 1–6.

[85] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "GAZELLE: A low latency framework for secure neural network inference," in *Proc. 27th USENIX Secur. Symp. (USENIX Secur.)*, 2018, pp. 1651–1669.

[86] S. Wagh, D. Gupta, and N. Chandran, "SecureNN: Efficient and private neural network training," *IACR Cryptol. ePrint Arch.*, vol. 2018, p. 442, Feb. 2018.

[87] X. Liu, R. H. Deng, P. Wu, and Y. Yang, "Lightning-fast and privacy-preserving outsourced computation in the cloud," *Cybersecurity*, vol. 3, no. 1, pp. 1–21, Dec. 2020.

[88] O. Ohrimenko, F. Schuster, C. Fournet, A. Mehta, S. Nowozin, K. Vaswani, and M. Costa, "Oblivious multi-party machine learning on trusted processors," in *Proc. 25th USENIX Secur. Symp. (USENIX Secur.)*, 2016, pp. 619–636.

[89] F. McKeen, I. Alexandrovich, A. Berenzon, C. V. Rozas, H. Shafi, V. Shanbhogue, and U. R. and Savagaonkar, "Innovative instructions and software model for isolated execution," in *Proc. 2nd Int. Workshop Hardw. Architectural Support Secur. Privacy*. New York, NY, USA: ACM, 2013.

[90] T. Hunt, C. Song, R. Shokri, V. Shmatikov, and E. Witchel, "Chiron: Privacy-preserving machine learning as a service," 2018, *arXiv:1803.05961*. [Online]. Available: https://arxiv.org/abs/1803.05961

[91] T. Hunt, Z. Zhu, Y. Xu, S. Peter, and E. Witchel, "Ryoan: A distributed sandbox for untrusted computation on secret data," *ACM Trans. Comput. Syst.*, vol. 35, no. 4, pp. 1–32, 2018.

[92] N. Hynes, R. Cheng, and D. Song, "Efficient deep learning on multi-source private data," 2018, *arXiv:1807.06689*. [Online]. Available: https://arxiv.org/abs/1807.06689

[93] T. Chen, T. Moreau, Z. Jiang, L. Zheng, E. Yan, H. Shen, M. Cowan, L. Wang, Y. Hu, L. Ceze, C. Guestrin, and A. Krishnamurthy, "TVM: An automated end-to-end optimizing compiler for deep learning," in *Proc. 13th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, Carlsbad, CA, USA, Oct. 2018, pp. 578–594. [Online]. Available: https://www.usenix.org/conference/osdi18/presentation/chen

[94] Z. Gu, H. Huang, J. Zhang, D. Su, A. Lamba, D. Pendarakis, and I. Molloy, "Securing input data of deep learning inference systems via partitioned enclave execution," 2018, *arXiv:1807.00969*. [Online]. Available: https://arxiv.org/abs/1807.00969

[95] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 818–833.

[96] F. Tramer and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," in *Proc. 7th Int. Conf. Learn. Represent. (ICLR)*, New Orleans, LA, USA, May 2019.

[97] R. Freivalds, "Probabilistic machines can use less running time," in *Proc. IFIP Congr.*, vol. 839, 1977, p. 842.

[98] L. Hanzlik *et al.*, "MLCapsule: Guarded offline deployment of machine learning as a service," 2018, *arXiv:1808.00590*. [Online]. Available: https://arxiv.org/abs/1808.00590

[99] T. Lee, B. Edwards, I. Molloy, and D. Su, "Defending against machine learning model stealing attacks using deceptive perturbations," 2018, *arXiv:1806.00054*. [Online]. Available: https://arxiv.org/abs/1806.00054

[100] M. Juuti, S. Szyller, S. Marchal, and N. Asokan, "PRADA: Protecting against DNN model stealing attacks," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Jun. 2019, pp. 512–527.

[101] T. Zhang and Q. Zhu, "A dual perturbation approach for differential private ADMM-based distributed empirical risk minimization," in *Proc. ACM Workshop Artif. Intell. Secur.*, 2016, pp. 129–137.

[102] J. Hamm, Y. Cao, and M. Belkin, "Learning privately from multiparty data," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 555–563.

[103] Y. Long, V. Bindschaedler, and C. A. Gunter, "Towards measuring membership privacy," 2017, *arXiv:1712.09136*. [Online]. Available: https://arxiv.org/abs/1712.09136

[104] N. Carlini, C. Liu, U. Erlingsson, J. Kos, and D. Song, ''The secret sharer: Evaluating and testing unintended memorization in neural networks,'' in *Proc. 28th USENIX Secur. Symp. (USENIX Secur.)*, 2019, pp. 267–284.

[105] M. A. Rahman, T. Rahman, R. Laganière, N. Mohammed, and Y. Wang, ''Membership inference attack against differentially private deep learning model,'' *Trans. Data Privacy*, vol. 11, no. 1, pp. 61–79, 2018.

[106] B. Jayaraman and D. Evans, ''Evaluating differentially private machine learning in practice,'' in *Proc. 28th USENIX Secur. Symp. (USENIX Secur.)*, 2019, pp. 1895–1912.

[107] J. Jia, A. Salem, M. Backes, Y. Zhang, and N. Z. Gong, ''MemGuard: Defending against black-box membership inference attacks via adversarial examples,'' in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 259–274.

[108] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, ''Protecting intellectual property of deep neural networks with watermarking,'' in *Proc. Asia Conf. Comput. Commun. Secur.*, May 2018, pp. 159–172.

[109] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, ''Turning your weakness into a strength: Watermarking deep neural networks by backdooring,'' in *Proc. 27th USENIX Secur. Symp. (USENIX Secur.)*, 2018, pp. 1615–1631.

[110] E. Le Merrer, P. Pérez, and G. Trédan, ''Adversarial frontier stitching for remote neural network watermarking,'' *Neural Comput. Appl.*, vol. 32, no. 13, pp. 9233–9244, Jul. 2020.

[111] B. D. Rouhani, H. Chen, and F. Koushanfar, ''Deepsigns: A generic watermarking framework for ip protection of deep learning models,'' 2018, *arXiv:1804.00750*. [Online]. Available: https://arxiv.org/abs/1804.00750

[112] R. Fletcher, *Practical Methods of Optimization*. Hoboken, NJ, USA: Wiley, 2013.

[113] A. Kurakin, I. Goodfellow, and S. Bengio, ''Adversarial machine learning at scale,'' 2016, *arXiv:1611.01236*. [Online]. Available: https://arxiv.org/abs/1611.01236

[114] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, ''Towards deep learning models resistant to adversarial attacks,'' 2017, *arXiv:1706.06083*. [Online]. Available: https://arxiv.org/abs/1706.06083

[115] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, ''The limitations of deep learning in adversarial settings,'' in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Mar. 2016, pp. 372–387.

[116] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, ''Distillation as a defense to adversarial perturbations against deep neural networks,'' in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2016, pp. 582–597.

[117] N. Carlini and D. Wagner, ''Towards evaluating the robustness of neural networks,'' in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.

[118] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, ''DeepFool: A simple and accurate method to fool deep neural networks,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2574–2582.

[119] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, ''Universal adversarial perturbations,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1765–1773.

[120] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, ''Countering adversarial images using input transformations,'' 2017, *arXiv:1711.00117*. [Online]. Available: http://arxiv.org/abs/1711.00117

[121] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, ''Mitigating adversarial effects through randomization,'' 2017, *arXiv:1706.06083*. [Online]. Available: https://arxiv.org/abs/1706.06083

[122] P. Samangouei, M. Kabkab, and R. Chellappa, ''Defense-GAN: Protecting classifiers against adversarial attacks using generative models,'' 2018.

[123] A. Athalye, N. Carlini, and D. Wagner, ''Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,'' in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, Jul. 2018, pp. 274–283. [Online]. Available: https://arxiv.org/abs/1802.00420

[124] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, ''Synthesizing robust adversarial examples,'' in *Proc. PMLR*, 2018, pp. 284–293.

[125] J. Su, D. V. Vargas, and K. Sakurai, ''One pixel attack for fooling deep neural networks,'' *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 828–841, Oct. 2019, doi: 10.1109/TEVC.2019.2890858.

[126] R. Storn and K. Price, ''Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces,'' *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[127] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, ''Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems,'' *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Dec. 2006.

[128] S. Das and P. N. Suganthan, ''Differential evolution: A survey of the State-of-the-Art,'' *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[129] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, ''Countering adversarial images using input transformations,'' 2017, *arXiv:1711.00117*. [Online]. Available: https://arxiv.org/abs/1711.00117

[130] D. Duy Thang and T. Matsui, ''Image transformation can make neural networks more robust against adversarial examples,'' 2019, *arXiv:1901.03037*. [Online]. Available: http://arxiv.org/abs/1901.03037

[131] Y. Luo, X. Boix, G. Roig, T. A. Poggio, and Q. Zhao, ''Foveation-based mechanisms alleviate adversarial examples,'' 2015, *arXiv:1511.06292*. [Online]. Available: https://arxiv.org/abs/1511.06292

[132] J. Lu, H. Sibai, E. Fabry, and D. Forsyth, ''NO need to worry about adversarial examples in object detection in autonomous vehicles,'' 2017, *arXiv:1707.03501*. [Online]. Available: http://arxiv.org/abs/1707.03501

[133] K. McLaren, ''XIII—The development of the CIE 1976 (L∗ a∗ b∗) uniform colour space and colour-difference formula,'' *J. Soc. Dyers Colourists*, vol. 92, no. 9, pp. 338–341, 1976.

[134] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, ''ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models,'' in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, 2017, pp. 15–26.

[135] P. D. Lax and M. S. Terrell, *Calculus With Applications*. New York, NY, USA: Springer, 2014.

[136] D. P. Kingma and J. Ba, ''Adam: A method for stochastic optimization,'' 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[137] C.-C. Tu, P. Ting, P.-Y. Chen, S. Liu, H. Zhang, J. Yi, C.-J. Hsieh, and S.-M. Cheng, ''AutoZOOM: Autoencoder-based zeroth order optimization method for attacking black-box neural networks,'' in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 742–749.

[138] W. Brendel, J. Rauber, and M. Bethge, ''Decision-based adversarial attacks: Reliable attacks against black-box machine learning models,'' 2017, *arXiv:1712.04248*. [Online]. Available: https://arxiv.org/abs/1712.04248

[139] T. Brunner, F. Diehl, M. T. Le, and A. Knoll, ''Guessing smart: Biased sampling for efficient black-box adversarial attacks,'' in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 4958–4966.

[140] K. Perlin, ''An image synthesizer,'' *ACM SIGGRAPH Comput. Graph.*, vol. 19, no. 3, pp. 287–296, 1985.

[141] J. Chen, M. I. Jordan, and M. J. Wainwright, ''HopSkipJumpAttack: A query-efficient decision-based attack,'' in *Proc. IEEE Symp. Secur. Privacy (SP)*, Los Alamitos, CA, USA, May 2020, pp. 1277–1294, doi: 10.1109/sp40000.2020.00045.

[142] Y. Liu, X. Chen, C. Liu, and D. Song, ''Delving into transferable adversarial examples and black-box attacks,'' 2016, *arXiv:1611.02770*. [Online]. Available: http://arxiv.org/abs/1611.02770

[143] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, ''Rethinking the inception architecture for computer vision,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2818–2826.

[144] J. Lu, H. Sibai, E. Fabry, and D. Forsyth, ''Standard detectors aren't (currently) fooled by physical adversarial stop signs,'' 2017, *arXiv:1710.03337*. [Online]. Available: http://arxiv.org/abs/1710.03337

[145] J. Redmon and A. Farhadi, ''YOLO9000: Better, faster, stronger,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 7263–7271.

[146] S. Ren, K. He, R. Girshick, and J. Sun, ''Faster R-CNN: Towards real-time object detection with region proposal networks,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[147] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, D. Song, T. Kohno, A. Rahmati, A. Prakash, and F. Tramer, ''Note on attacking object detectors with adversarial stickers,'' 2017, *arXiv:1712.08062*. [Online]. Available: http://arxiv.org/abs/1712.08062

[148] S.-T. Chen, C. Cornelius, J. Martin, and D. H. P. Chau, ''ShapeShifter: Robust physical adversarial attack on faster R-CNN object detector,'' in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Cham, Switzerland: Springer, 2018, pp. 52–68.

[149] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, ''Adversarial patch,'' 2017, *arXiv:1712.09665*. [Online]. Available: https://arxiv.org/abs/1712.09665

[150] S. Thys, W. V. Ranst, and T. Goedemé, ''Fooling automated surveillance cameras: Adversarial patches to attack person detection,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2019.

[151] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, ''Practical black-box attacks against machine learning,'' in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, 2017, pp. 506–519.

[152] X. Li, S. Ji, M. Han, J. Ji, Z. Ren, Y. Liu, and C. Wu, "Adversarial examples versus cloud-based detectors: A black-box empirical study," *IEEE Trans. Dependable Secure Comput.*, early access, Sep. 24, 2019, doi: 10.1109/TDSC.2019.2943467.

[153] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 1528–1540.

[154] Z. Zhou, D. Tang, X. Wang, W. Han, X. Liu, and K. Zhang, "Invisible mask: Practical attacks on face recognition with infrared," 2018, *arXiv:1803.04683*. [Online]. Available: http://arxiv.org/abs/1803.04683

[155] Q. Wang, W. Guo, K. Zhang, A. G. Ororbia, II, X. Xing, X. Liu, and C. L. Giles, "Adversary resistant deep neural networks with an application to malware detection," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 1145–1153.

[156] A. Prakash, N. Moran, S. Garber, A. DiLillo, and J. Storer, "Deflecting adversarial attacks with pixel deflection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8571–8580.

[157] J. Ho and D.-K. Kang. (2018). *Pixel Redrawn for a Robust Adversarial Defense*. [Online]. Available: https://openreview.net/forum?id=r1ez_sRcFQ

[158] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*. [Online]. Available: https://arxiv.org/abs/1708.07747

[159] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.

[160] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy, "A study of the effect of JPG compression on adversarial images," 2016, *arXiv:1608.00853*. [Online]. Available: https://arxiv.org/abs/1608.00853

[161] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and H. D. Chau, "Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression," 2017.

[162] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D*, vol. 60, nos. 1–4, pp. 259–268, Nov. 1992. [Online]. Available: https://doi.org/10.1016/0167-2789(92)90242-F

[163] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. 28th Annu. Conf. Comput. Graph. Interact. Techn.*, 2001, pp. 341–346.

[164] W. Xu, D. Evans, and Y. Qi, "Feature squeezing mitigates and detects carlini/wagner adversarial examples," 2017.

[165] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jun. 2005, pp. 60–65.

[166] E. Raff, J. Sylvester, S. Forsyth, and M. McLean, "Barrage of random transforms for adversarially robust defense," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 6528–6537.

[167] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Trans. Image Process.*, vol. 1, no. 2, pp. 205–220, Apr. 1992.

[168] T. Huang, G. Yang, and G. Tang, "A fast two-dimensional median filtering algorithm," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-27, no. 1, pp. 13–18, Feb. 1979.

[169] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[170] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.

[171] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided denoiser," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1778–1787.

[172] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput.-Assist. Intervent*. Cham, Switzerland: Springer, 2015, pp. 234–241.

[173] A. Athalye and N. Carlini, "On the robustness of the CVPR 2018 white-box adversarial example defenses," 2018, *arXiv:1804.03286*. [Online]. Available: https://arxiv.org/abs/1804.03286

[174] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," 2016, *arXiv:1606.00704*. [Online]. Available: https://arxiv.org/abs/1606.00704

[175] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," 2016, *arXiv:1605.09782*. [Online]. Available: https://arxiv.org/abs/1605.09782

[176] R. Bao, S. Liang, and Q. Wang, "Featurized bidirectional gan: Adversarial defense via adversarially learned semantic inference," 2018, *arXiv:1805.07862*. [Online]. Available: https://arxiv.org/abs/1805.07862

[177] A. Mustafa, S. H. Khan, M. Hayat, J. Shen, and L. Shao, "Image super-resolution as a defense against adversarial attacks," *IEEE Trans. Image Process.*, vol. 29, pp. 1711–1724, 2020.

[178] S. G. Chang, B. Yu, and M. Vetterli, "Adaptive wavelet thresholding for image denoising and compression," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1532–1546, 2000.

[179] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jul. 2017, pp. 136–144.

[180] T.-J. Chang, Y. He, and P. Li, "Efficient two-step adversarial defense for deep neural networks," 2018, *arXiv:1810.03739*. [Online]. Available: https://arxiv.org/abs/1810.03739

[181] R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári, "Learning with a strong adversary," 2015, *arXiv:1511.03034*. [Online]. Available: http://arxiv.org/abs/1511.03034

[182] U. Shaham, Y. Yamada, and S. Negahban, "Understanding adversarial training: Increasing local stability of supervised models through robust optimization," *Neurocomputing*, vol. 307, pp. 195–204, Sep. 2018.

[183] H. Kannan, A. Kurakin, and I. Goodfellow, "Adversarial logit pairing," 2018, *arXiv:1803.06373*. [Online]. Available: https://arxiv.org/abs/1803.06373

[184] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," 2017, *arXiv:1705.07204*. [Online]. Available: https://arxiv.org/abs/1705.07204

[185] C. Liu and J. JaJa, "Feature prioritization and regularization improve standard accuracy and adversarial robustness," 2018, *arXiv:1810.02424*. [Online]. Available: https://arxiv.org/abs/1810.02424

[186] C. Xie, Y. Wu, L. V. D. Maaten, A. L. Yuille, and K. He, "Feature denoising for improving adversarial robustness," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 501–509.

[187] F. Heide, W. Heidrich, and G. Wetzstein, "Fast and flexible convolutional sparse coding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5135–5143.

[188] B. Choudhury, R. Swanson, F. Heide, G. Wetzstein, and W. Heidrich, "Consensus convolutional sparse coding," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 4280–4288.

[189] B. Sun, N.-H. Tsai, F. Liu, R. Yu, and H. Su, "Adversarial defense by stratified convolutional sparse coding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 11447–11456.

[190] H. Hosseini, Y. Chen, S. Kannan, B. Zhang, and R. Poovendran, "Blocking transferability of adversarial examples in black-box learning systems," 2017, *arXiv:1703.04318*. [Online]. Available:https://arxiv.org/abs/1703.04318

[191] S. Chen, N. Carlini, and D. Wagner, "Stateful detection of black-box adversarial attacks," in *Proc. 1st ACM Workshop Secur. Privacy Artif. Intell.*, 2019, pp. 30–39.

[192] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," 2018, *arXiv:1804.08598*. [Online]. Available: https://arxiv.org/abs/1804.08598

[193] S. Tian, G. Yang, and Y. Cai, "Detecting adversarial examples through image transformation," in *Proc. 32nd AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, Feb. 2018, pp. 4139–4146.

[194] B. Liang, H. Li, M. Su, X. Li, W. Shi, and X. Wang, "Detecting adversarial image examples in deep neural networks with adaptive noise reduction," *IEEE Trans. Dependable Secure Comput.*, early access, Oct. 5, 2019, doi: 10.1109/TDSC.2018.2874243.

[195] A. Rozsa, M. Gunther, and T. E. Boult, "Towards robust deep neural networks with BANG," in *Proc. WACV*, 2018, pp. 803–811.

[196] Z. Zheng and P. Hong, "Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7913–7922.

[197] 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.*, 2017, pp. 27–38.

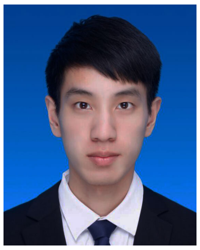
[198] C. Yang, Q. Wu, H. Li, and Y. Chen, "Generative poisoning attack method against neural networks," 2017, *arXiv:1703.01340*. [Online]. Available:https://arxiv.org/abs/1703.01340

[199] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *Proc. 34th Int. Conf. Mach. Learn. (JMLR)*, vol. 70, 2017, pp. 1885–1894.

[200] R. D. Cook and S. Weisberg, "Characterizations of an empirical influence function for detecting influential cases in regression," *Technometrics*, vol. 22, no. 4, pp. 495–508, Nov. 1980.

[201] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6103–6113.
[202] 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*. [Online]. Available: https://arxiv.org/abs/1712.05526
[203] S. Li, M. Xue, B. Z. H. Zhao, H. Zhu, and X. Zhang, "Invisible backdoor attacks on deep neural networks via steganography and regularization," *IEEE Trans. Dependable Secure Comput.*, early access, Sep. 3, 2020, doi: 10.1109/TDSC.2020.3021407.
[204] J. Steinhardt, P. W. W. Koh, and P. S. Liang, "Certified defenses for data poisoning attacks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3517–3529.
[205] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proc. ICML*, 2012, pp. 1467–1474.
[206] S. Mei and X. Zhu, "Using machine teaching to identify optimal training-set attacks on machine learners," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2871–2877.
[207] 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.
[208] A. Paudice *et al.*, "Detection of adversarial training examples in poisoning attacks through anomaly detection," 2018, *arXiv:1802.03041*. [Online]. Available: https://arxiv.org/abs/1802.03041
[209] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*. Cham, Switzerland: Springer, 2018, pp. 273–294.
[210] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. and Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," 2018, *arXiv:1811.03728*. [Online]. Available: https://arxiv.org/abs/1811.03728
[211] 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 models," 2017, *arXiv:1707.08945*. [Online]. Available: https://arxiv.org/abs/1707.08945
[212] X. Wei, S. Liang, N. Chen, and X. Cao, "Transferable adversarial attacks for image and video object detection," 2018, *arXiv:1811.12641*. [Online]. Available: https://arxiv.org/abs/1811.12641
[213] J. H. Metzen, M. C. Kumar, T. Brox, and V. Fischer, "Universal adversarial perturbations against semantic image segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2755–2764.
[214] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 1369–1378.
[215] E. Quiring, A. Maier, and K. Rieck, "Misleading authorship attribution of source code using adversarial learning," in *Proc. 28th USENIX Secur. Symp. (USENIX Secur.)*, 2019, pp. 479–496.
[216] M. Cheng, T. Le, P.-Y. Chen, J. Yi, H. Zhang, and C.-J. Hsieh, "Query-efficient hard-label black-box attack: An optimization-based approach," 2018, *arXiv:1807.04457*. [Online]. Available: https://arxiv.org/abs/1807.04457
[217] F. Tramer, N. Carlini, W. Brendel, and A. Madry, "On adaptive attacks to adversarial example defenses," 2020, *arXiv:2002.08347*. [Online]. Available: http://arxiv.org/abs/2002.08347
[218] 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.*, 2019, pp. 125–136.

XIMENG LIU (Member, IEEE) received the B.Sc. degree in electronic engineering and the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 2010 and 2015, respectively. He is currently a Full Professor with the College of Mathematics and Computer Science, Fuzhou University, China. He is also a Research Fellow with the School of Information System, Singapore Management University, Singapore. He has published over 100 research articles, including IEEE Transactions on Information Forensics and Security, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Computers, IEEE Transactions on Industrial Informatics, IEEE Transactions on Services Computing, and IEEE Transactions on Cloud Computing. His research interests include cloud security, applied cryptography, and big data security.



LEHUI XIE received the B.Sc. degree from the College of Computer and Information Science, Fujian Agriculture and Forestry University, Fuzhou, China, in 2019. He is currently pursuing the master's degree with Fuzhou University. His current research interest includes privacy and security in machine learning.

YAOPENG WANG received the B.Sc. degree from the College of Mathematics and Computer Science, Yantai University, Yantai, China, in 2018. He is currently pursuing the master's degree with Fuzhou University. His current research interest includes privacy and security in machine learning.

JIAN ZOU received the B.S. degree in mathematics and applied mathematics from Central China Normal University, in 2009, and the Ph.D. degree in information security from the Institute of Software, Chinese Academy of Sciences, Beijing, China, in 2015. He is currently a Lecturer with Fuzhou University. His main research interests include block cipher, hash function, and post-quantum cryptography.

JINBO XIONG (Member, IEEE) received the M.S. degree in communication and information systems from the Chongqing University of Posts and Telecommunications, China, in 2006, and the Ph.D. degree in computer system architecture from Xidian University, China, in 2013. He is currently a Visiting Scholar with the University of North Texas, USA, and an Associate Professor with the Fujian Provincial Key Laboratory of Network Security and Cryptology and the College of Mathematics and Informatics, Fujian Normal University. His research interests include cloud data security, privacy protection, and mobile Internet security. He has published more than 40 publications and one monograph and holds eight patents in these fields. He is a member of ACM.

ZUOBIN YING (Member, IEEE) was born in Anhui, China, in 1982. He received the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 2016. He is currently a Lecturer with the School of Computer Science and Technology, Anhui University, China. His research interests include cloud security and applied cryptography.

ATHANASIOS V. VASILAKOS (Senior Member, IEEE) is currently with the University Technology Sydney, Australia, with the Fuzhou University, Fuzhou, China, and with the Luleå University of Technology, Luleå, Sweden. He served or is serving as an Editor for many technical journals, such as the IEEE Transactions on Network and Service Management, IEEE Transactions on Cloud Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Computers, IEEE Transactions on NanoBioscience, IEEE Transactions on Information Technology in Biomedicine, ACM TAAS, and the IEEE Journal on Selected Areas in Communications.