Security and Privacy Issues in Deep Learning

Ho Bae[†], Seoul National University, Republic of Korea
Jaehee Jang[†], Seoul National University, Republic of Korea
Dahuin Jung, Seoul National University, Republic of Korea
Hyemi Jang, Seoul National University, Republic of Korea
Heonseok Ha, Seoul National University, Republic of Korea
Sungroh Yoon[*], Seoul National University, Republic of Korea
[*]E-mail: sryoon@snu.ac.kr
[*]: To whom correspondence should be addressed, [†]: These authors contributed equally to this work.

With the development of machine learning, expectations for artificial intelligence (AI) technology are increasing day by day. In particular, deep learning has shown enriched performance results in a variety of fields. There are many applications that are closely related to our daily life, such as making significant decisions in application area based on predictions or classifications, in which a deep learning (DL) model could be relevant. Hence, if a DL model causes mispredictions or misclassifications due to malicious external influences, it can cause very large difficulties in real life. Moreover, training deep learning models involves relying on an enormous amount of data and the training data often includes sensitive information. Therefore, deep learning models should not expose the privacy of such data. In this paper, we reviewed the threats and developed defense methods on the security of the models and the data privacy under the notion of SPAI: Secure and Private AI. We also discuss current challenges and open issues.

Additional Key Words and Phrases: Private AI, Secure AI, Machine Learning, Deep Learning, Homomorphic Encryption, Differential Privacy, Adversarial Example, White-box Attack, Black-box Attack

1. INTRODUCTION

Advances of deep learning (DL) algorithms have transformed the solution of data-driven problems in various applications in real life, including the use of large amounts of patient data for health prediction services [Shickel et al. 2017]; autonomous security audits from system logs [Buczak and Guven 2016]; and unmanned car driving powered by visual object detections [Ren et al. 2015]. However, the vulnerabilities of DL systems have been recently uncovered within a vast amount of literature. It is very dangerous that these applications are based on little understandings of security and privacy on DL systems.

Although many research studies have been published on both attacks and defense with deep learning security and privacy, they are still fragmented. Hence we review recent attempts toward Secure AI and Private AI. Addressing the need for robust ar-

This research was supported in part by Projects for Research and Development of Police science and Technology under Center for Research and Development of Police science and Technology and Korean National Police Agency funded by the Ministry of Science, ICT and Future Planning (PA-C000001), in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (grant number: 2016M3A7B4911115), in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) [2018R1A2B3001628], and in part by the Brain Korea 21 Plus Project (Electrical and Computer Engineering, Seoul National University) in 2018.

DOI: http://dx.doi.org/10.1145/0000000.0000000

tificial intelligence (AI) systems in security and privacy, we develop a perspective on SPAI: Secure and Private AI. Secure AI aims for AI systems that have high security guarantees; Private AI aims for AI systems that preserve the data privacy. Additionally, as a part of the effort to build the SPAI system, we review the fragmented findings and attempts to address the attacks and defenses in deep learning security and privacy.

Secure AI focuses on attacks and defense with respect to AI systems, which, in terms of DL, is a model. Based on the knowledge of the structure and parameters of the model, attacks on DL models usually attempt to subvert the learning process or induce false predictions on the purpose, by injecting adversarial samples. This type of attack, which can include gradient-based techniques [Biggio et al. 2013; Goodfellow et al. 2014b], is often called a white-box attack. In contrast, black-box attacks lead the target system to make false predictions, without any information about the underlying model. We observe that most of the attacks exploit the prediction confidence given by the targeted model without knowing the model's structure and parameters.

To defend from these attacks, methods such as adversarial training [Goodfellow et al. 2014b; Sun et al. 2018; Gu et al. 2018], gradient masking [Buckman et al. 2018; Dhillon et al. 2018; Song et al. 2017], GAN [Samangouei et al. 2018; Song et al. 2017] and statistical approaches [Steinhardt et al. 2017; Paudice et al. 2018b,a] have been proposed. Table III lists recent research on attacks with various models of deep learning, with their structures and parameters, together with the defense against these attacks.

On the other hand, Private AI aims for the AI systems that preserve data privacy. DL requires users to transfer some sensitive data to remote machines because of the computational cost or the need for collaborative training. In such situations, users lose control over the data after the transfer and have concerns about their data privacy being stolen between transfers, or the service holders that they upload their data to can misuse their data without consent. It was also claimed that only with the deployed DL model can the data used for training the model be inversed [Hitaj et al. 2017]. Table IV describes the potential privacy threats and their corresponding defence methods. Against such privacy threats, privacy-preserving techniques including fully homomorphic encryption (FHE) [Gilad-Bachrach et al. 2016; Hesamifard et al. 2017; Chabanne et al. 2017; Bourse et al. 2017; Sanyal et al. 2018], differential privacy [Abadi et al. 2016b; Chaudhuri and Monteleoni 2009; Dwork et al. 2006, 2010; Papernot et al. 2016a, 2018; McMahan et al. 2018; Ermis and Cemgil 2017], and secure multi-party computation (SMC) [Shokri and Shmatikov 2015; Aono et al. 2018], have been combined with the DL frameworks. Table V lists recent research on machine learning attacks and defenses to expose the privacy of the training and test data.

We review recent research on privacy and security issues associated with deep learning in several domains. Additionally, we taxonomize possible attacks and the state-of-the-art defense methods on Secure AI and Private AI. To the best of our knowledge, our work is the first attempt to taxonomize approaches to privacy in deep learning.

1.1. Adversarial Examples in Real World Setting

The first adversarial attack started with an image in a non-targeted manner. An alternative targeted attack soon developed maximizing the likelihood of the target class. A recent targeted attack performed on the Google Cloud Vision (GCV) API compromising commercial systems. This approach becomes problematic to the second service provider because of the use of a decision-making service with given prediction score. For example, a service provider that uses GCV API for the auto-driving will fail to stop an adversarily crafted stop sign. [Elsayed et al. 2018]. Finlayson et al. [2018] extended adversarial attacks to medical imaging tasks. Both white- and black-box attacks are presented fooling medical deep learning classifiers. As such, Finlayson et al. [2018]

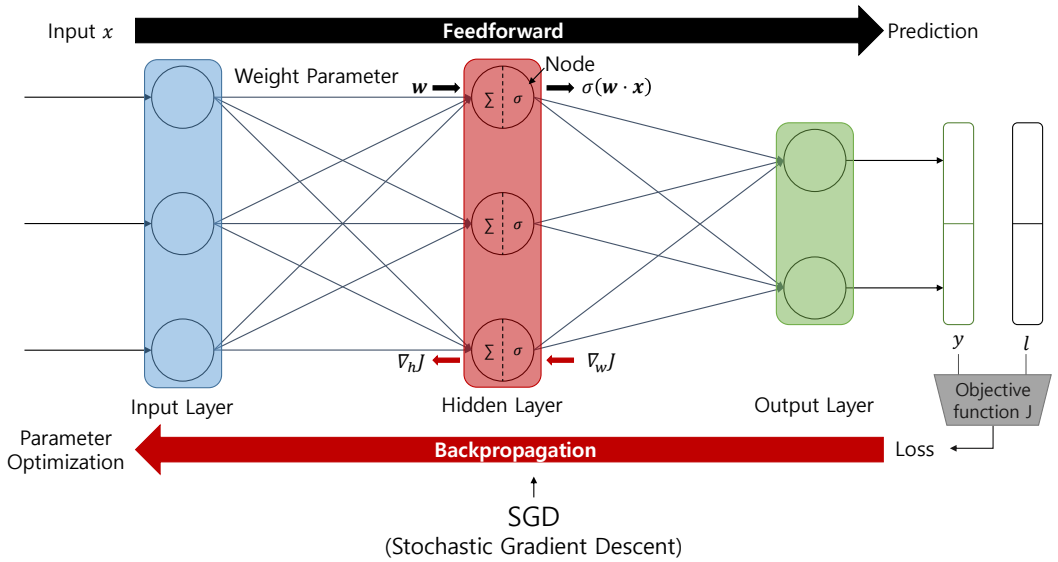


Fig. 1: General DNN training process

showed that there is potential harm in the medical domain that can be caused by such adversarial attacks. In addition to the medical domain, Carlini and Wagner [2018] proposed the first adversarial examples of automatic speech recognition. They applied a white-box iterative optimization-based attack and showed a 100% success rate, which showed that the feasibility of adversarial attacks on an image can be transferred to another domain. They reconstructed the waveform of input x to $x + \upsilon$ while exploiting conventional measure distortion, and they successfully produced speech to the desired phrase with 99.9% similarity given any audio waveform.

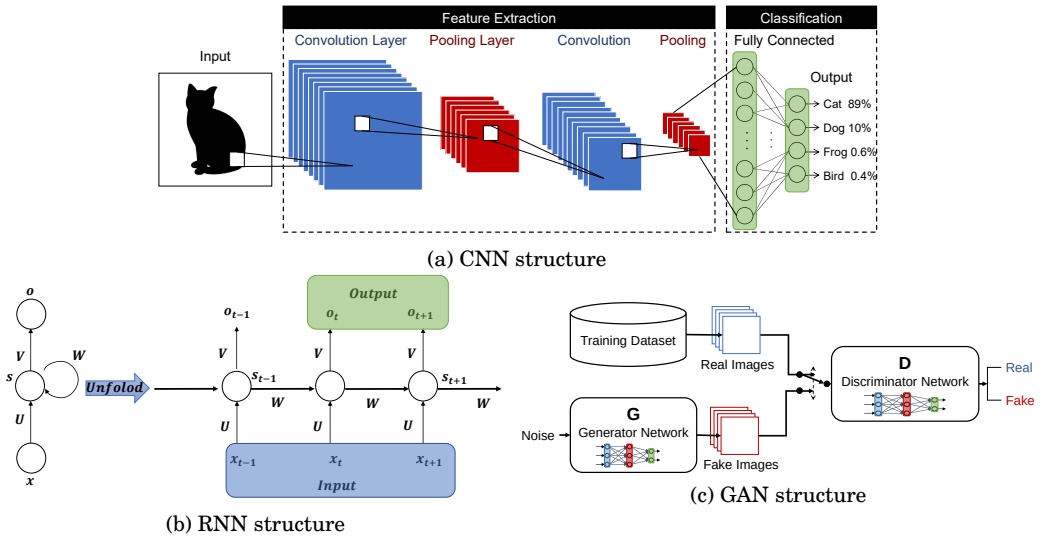
2. BACKGROUND

Behind the success of deep learning lies the advancements of deep neural networks (DNNs) trained with an extensive amount of data. In this section, we introduce the components and the training algorithm of a DNN. Further, we describe the recent DL models that are widely used. The building block of a neural network is an *artificial neuron*, which was designed to resemble a human neuron. However, because the actual biological activities inside human neurons are still uncovered, artificial neurons simply compute the weighted sum of the input and activations, as follows:

$$y = \sigma(\sum_{i=1}^{n} w_i x_i). \tag{1}$$

where x is the input, y is the output, σ is the activation function, and w are the weights. The artificial neurons are used as nodes to construct layers, and by piling up these layers deep neural networks (DNNs) are constructed. The activation functions are nonlinear functions such as the sigmoid, tanh and ReLU. The nonlinearity of the activation function piles up as the number of layers grows and enables DNNs to approximate target functions without any handcrafted feature selections.

2.1. Artificial Intelligence powered by Deep Learning



(a) CNN structure

(b) RNN structure

(c) GAN structure

Fig. 2: Different DNN model structures

2.1.1. ***Deep Learning Workflow—Training and Inference***. The workflow of the DL contains two phases: training and inference. DNNs learn new capabilities through the training phase from the existing data, and the learned capabilities are applied to unseen data at the inference phase.

The overview of the DNN training process is described in Figure 1. DNNs are trained by iterating feedforward and backpropagation until convergence. At the feedforward stage, the input propagates along the layers to computes the output. Then, to minimize the error between the output and the actual label, the gradient descent algorithm is used,

$$w \leftarrow w - \eta \nabla_w J(w) \tag{2}$$

where a loss function $J(w)$ is used for the weight parameters w and the learning rate η. Hence, at each backpropagation stage, each node computes the gradient and updates the weight parameters as described in Equation 2. However, it is highly inefficient to iterate the process for the full batch of the data (all instances in the data) since the training data required for the DNN training is enormous. Therefore, mini-batch stochastic gradient descent (mini-batch SGD or SGD) is widely used. After the model converges to a certain accuracy or loss value, the model is used for prediction at the inference stage. At the inference phase, the model only forward propagates the input and regards the output as a prediction.

2.1.2. ***Different Types of Deep Neural Network Models***. Different DNN model architectures are described in Fig. 2

— **Feed-forward Neural Network (FNN).** An FNN is the most basic structure of the DNNs. It contains multiple layers, and the nodes between layers are fully connected while the intra-layer nodes are not connected to one another.
— **Convolutional Neural Network (CNN).** A general architecture of CNNs is described in Fig. 2(a). A CNN consists of one or more convolutional layers, which use convolutional operations to compute layer-wise results. This operation allows the network to learn about spatial information and hence CNNs show outstanding per-

formances especially on vision applications [Krizhevsky et al. 2012; He et al. 2016; Huang et al. 2017a].

— **Recurrent Neural Network (RNN).** A recurrent neural network (RNN) is widely used to process sequential data. As illustrated in Fig. 2(b), an RNN updates the current hidden unit and calculates the output based on the current input and past hidden unit. There are well-known problems of RNNs such as the gradient vanishing problem, and some variants, such as Long short-term memory [Hochreiter and Schmidhuber 1997] and Gated recurrent unit [Cho et al. 2014] have been proposed to solve such problems.

— **Generative Adversarial Network (GAN).** A generative adversarial network (GAN) framework [Goodfellow et al. 2014a] consists of a discriminator D and a generator G. G generates fake data while D determines whether the generated data is real, as depicted in Fig. 2(c). Usually generators and discriminators are neural networks that can have various structures depending on the application. GANs are actively studied in various fields, such as image/speech synthesis and domain adaptation.

2.2. Privacy-preserving Techniques

2.2.1. ***Homomorphic Encryption.*** An encryption scheme that allows arbitrary computations on encrypted data without decrypting it or having access to any decryption key, is called homomorphic encryption (HE). In other words, the encryption scheme Enc satisfies the following equation:

$$\mathrm{Enc}(a) \diamond \mathrm{Enc}(b) = \mathrm{Enc}(a * b) \tag{3}$$

where $\mathrm{Enc} : \mathcal{X} \to \mathcal{Y}$ is a homomorphic ecnryption scheme with $\mathcal{X}$ a set of messages and $\mathcal{Y}$ a set of cyphertexts. a, b are messages in $\mathcal{X}$, and $*, \diamond$ are linear operations defined in $\mathcal{X}, \mathcal{Y}$, respectively.

Homomorphic cryptosystems in early stages were partial homomorphic cryptosystems [ElGamal 1985; Goldwasser and Micali 1982; Benaloh 1994; Paillier 1999], that showed either additive or multiplicative homomorphism [Gentry and Boneh 2009]. However, after the work by Gentry and Boneh [2009] using ideal lattices was introduced, various attempts on fully homomorphic encryption (FHE), which allows any computable function to be performed on the encrypted data, have been proposed [Van Dijk et al. 2010; Yagisawa 2015; Brakerski and Vaikuntanathan 2014; Bos et al. 2013; Hesamifard et al. 2017; Ducas and Micciancio 2015].

Although FHE can benefit many applications including cloud computing platforms and secure multi-party computation, the use of massive data inputs and computational workloads as well as the nonlinearity in DL models, is still a burden to be combined with deep learning.

2.2.2. ***Differential Privacy.*** Differential privacy is one of the state-of-the-art privacy preserving models [Dwork 2008]; it guarantees that an attacker cannot deduce any private information with high confidence from databases or released models. In other words, differential private algorithms prevent an attacker from knowing the existence of a particular record by adding noise to the query responses.

Here is the attack scenario that is assumed in differential privacy algorithms: An attacker is allowed to query two adjacent databases, which vary in at most one record. By sending the same query to both databases, the difference between the respective responses is considered to arise from "one record." For example, imagine that there is a database $\mathcal{D}$ on weights and one can query only the average value of all records. In this situation, it is impossible to grasp a specific person's weight. However, if a new

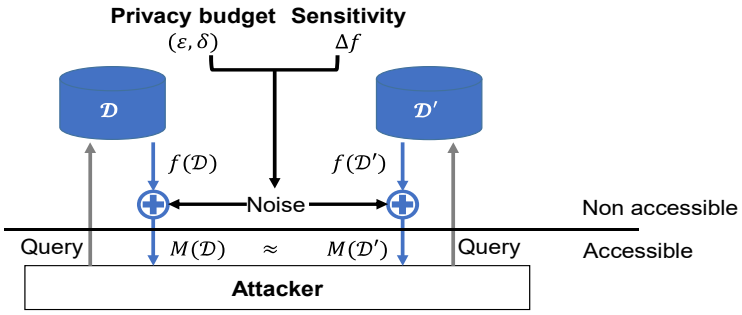


Fig. 3: Overview of the differential privacy framework

record is added and the attacker knows the former average weight, it is possible for the attacker to figure out the weight of the person added.

Differential privacy counters such privacy threats by adding noise to the response as follows:

$$M(\mathcal{D}) = f(\mathcal{D}) + n \tag{4}$$

where $M : \mathcal{D} \rightarrow \mathbb{R}$ is a randomized mechanism that applies the noise n to the query response; $\mathcal{D}$ is the target database, and f is the original query response, which is deterministic.

M gives ε-differential privacy if all adjacent $\mathcal{D}$ and $\mathcal{D}'$ satisfy the following:

$$\mathbf{Pr}[M(\mathcal{D}) \in S] \leq \exp(\varepsilon)\mathbf{Pr}[M(\mathcal{D}') \in S] \tag{5}$$

where $\mathcal{D}$ and $\mathcal{D}'$ are two adjacent databases, and $S \subseteq \mathrm{Range}(M)$ is a subset of $\mathbb{R}$. ε is the privacy budget that controls the privacy level; the smaller ε is determined, the more similar $M(\mathcal{D})$ and $M(\mathcal{D}')$ are required to be. These facts show that there is a trade-off between the data utility and the privacy level.

Since Equation. 5 is a strict condition, (ε, δ)-differential privacy introduces the δ term, which loosens the bound of error by the amount of δ. In other words, δ allows M to satisfy the differential privacy condition even if the probabilities are somewhat different. The definition of (ε, δ)-differential privacy holds when the following equation is satisfied:

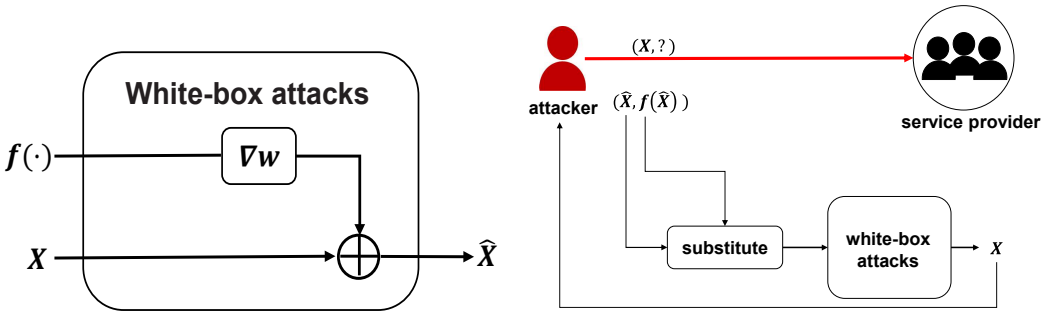
$$\mathbf{Pr}[M(\mathcal{D}) \in S] \leq \exp(\varepsilon)\mathbf{Pr}[M(\mathcal{D}') \in S] + \delta \tag{6}$$

where δ is another privacy budget which controls the privacy (confidence) levels.

Usually, the noise is sampled from the Laplace distribution or Gaussian distribution [Dwork 2008]. Each distribution depends on the sensitivity and privacy budgets. The sensitivity Δf [Dwork 2008] of the query response function f captures how much one record can affect the output, and it can be calculated as the maximum difference between responses on the adjacent databases:

$$\Delta f = \max_{\mathcal{D},\mathcal{D}'} |f(\mathcal{D}) - f(\mathcal{D}')|. \tag{7}$$

A larger sensitivity demands a larger amount of noise under the same privacy budget. There are some useful theories in which the composition of differential private mechanisms is also a differential private mechanism. Composition theorem [Dwork et al. 2006; Dwork and Lei 2009], advanced composition theorem [Kairouz et al. 2013; Dwork et al. 2010; Bun and Steinke 2016] and moment accountant [Abadi et al. 2016b] have been proposed.



(a) white-box attack (b) black-box attack

Fig. 4: Overview of (a) white-box attack scenario (b) black-box attack scenario.

Table I: Attack methods against Secure AI

Adversarial Attack Types ↓	White-box (Figure 4a)	Black-box (Figure 4b)	Training Phase	Inference Phase
Evasion	✓	✓		✓
Poisoning	✓		✓	

Table II: Secure vulnerability in AI

Attack Modes		Algorithms	Reference
Evasion	**White-box**		
		L-BFGS	[Szegedy et al. 2013]
		FGSM	[Goodfellow et al. 2014b]
		JSMA	[Papernot et al. 2016a]
		iFGSM	[Kurakin et al. 2016a]
		CW Attack	[Carlini and Wagner 2017b]
		UAP	[Moosavi-Dezfooli et al. 2017]
		Attacks on RL	[Huang et al. 2017b]
		ATN	[Baluja and Fischer 2017]
		AS attack	[Athalye and Sutskever 2017]
		Momentum iFGSM	[Dong et al. 2017]
		BPDA	[Athalye et al. 2018]
	Black-box		
		Hacking smart machines	[Ateniese et al. 2015]
		Adversarial examples Physical world	[Kurakin et al. 2016a]
		Autoregressive model	[Alfeld et al. 2016]
		Adversarial attacks on policies	[Huang et al. 2017b]
		Malware classification	[Grosse et al. 2016]
		Practical black-box attack	[Papernot et al. 2017]
		Membership training	[Long et al. 2017]
		Physical world AE	[Kurakin et al. 2016a]
		Policy induction attack	[Behzadan and Munir 2017]
		Human AE	[Elsayed et al. 2018]
Poisoning		Poisoning attack against SVM	[Biggio et al. 2012]
		Optimal teaching	[Patil et al. 2014]
		Optimal Training-set attacks	[Mei and Zhu 2015]
		Back-gradient optimization	[Muñoz-González et al. 2017]
		Generative poisoning attack	[Yang et al. 2017]
		Targeted backdoor attack	[Chen et al. 2017]

3. SECURE AI

Deep learning is applied to various fields ranging from autonomous driving to medical diagnosis. Hence, if the deep learning models are exposed to hostile influences that can destroy the training process or derive unintended behaviors from the pre-trained mod-

Table III: Corresponding defense methods to secure vulnerability in AI

Defense Modes		Algorithms	Reference
Evasion	**Gradient Masking**		
		Distillation defense	[Papernot et al. 2016b]
		AS attack	[Athalye and Sutskever 2017]
		Ensemble defense	[Carlini and Wagner 2017a]
		Efficient defense	[Zantedeschi et al. 2017]
		Ensemble adversarial learning	[He et al. 2017]
		Randomization	[Xie et al. 2017]
		Unified Embedding adversarial learning	[Na et al. 2017]
		Provable defenses	[Kolter and Wong 2017]
		Principled adversarial training	[Sinha et al. 2017]
		Input transformations adversarial learning	[Guo et al. 2017]
		Manifold defense	[Ilyas et al. 2017]
		Ensemble adversarial training	[Tramèr et al. 2017]
		Unified Embedding adversarial learning	[Na et al. 2017]
		Detecting perturbations	[Metzen et al. 2017]
		L1 based adversarial learning	[Sharma and Chen 2017]
		Pixel-defend	[Song et al. 2017]
		Defense-GAN	[Samangouei et al. 2018]
		Characterizing subspaces adversarial learning	[Ma et al. 2018]
		Stochastic activation pruning	[Dhillon et al. 2018]
		Thermometer defense	[Buckman et al. 2018]
	Adversarial Training		
		Harnessing adversarial examples	[Goodfellow et al. 2014b]
		Adversarial learning at scale	[Kurakin et al. 2016b]
		Adversary A3C for RL	[Gu et al. 2018]
		Speech recognition adversarial examples	[Sun et al. 2018]
		BPDA [Athalye et al. 2018]	
	GAN		
		Pixel-defend	[Song et al. 2017]
		Defense-GAN adversarial learning	[Samangouei et al. 2018]
	Statistical Approach		
		Certified defenses	[Steinhardt et al. 2017]
		Back-gradient optimization	[Paudice et al. 2018b]
		Anomaly detection	[Paudice et al. 2018a]
Poisoning		Certified defense	[Steinhardt et al. 2017]
		Influence functions	[Koh and Liang 2017]
		Anomaly detection	[Paudice et al. 2018b]
		Label flipping poisoning attack	[Paudice et al. 2018a]

Table IV: Potential privacy threats against private AI and the corresponding defense methods

Potential Threats by Role ↓	Homomorphic Encryption	Differential Privacy	Secure Multi-party Training
Model & Service Providers		✓	
Information Silos	✓	✓	✓
DL Service Users	✓		

els, they can result in terrible consequences in real life. For example, it was recently revealed that one can fool the autonomous driving system by jamming sensors [Yan et al. 2016]. Likewise, if someone can somehow change the input of the autonomous driving model to an adversarial example, it can even lead the passengers to death.

Hence, we suggest the concept of *Secure AI*, which means the AI system with security guarantees, in order to encourage the studies on the security of the AI systems. As deep learning is one of the state-of-the-art AI algorithms, we introduce and taxono-

Table V: A list of defending techniques of private AI in order of appearance: DL stands for deep learning, HE stands for homomorphic encryption, DP stands for differential Privacy and SMC stands for secure multi-party computation

Algorithm	HE	DP	SMC	Reference
CryptoNets	✓			[Gilad-Bachrach et al. 2016]
CryptoDL	✓			[Hesamifard et al. 2017]
Privacy-preserving classification	✓			[Chabanne et al. 2017]
TAPAS	✓			[Sanyal et al. 2018]
FHE-DiNN	✓			[Bourse et al. 2017]
DP-SGD		✓		[Abadi et al. 2016b]
DP LSTM		✓		[McMahan et al. 2018]
DPGAN		✓		[Xie et al. 2018]
DPGM		✓		[Acs et al. 2018]
DP Model Publishing		✓		[Yu et al. 2019]
Privacy-preserving logistic regression		✓		[Chaudhuri and Monteleoni 2009]
dPA		✓		[Phan et al. 2016]
dCDBN		✓		[Phan et al. 2017a]
AdLM		✓		[Phan et al. 2017b]
PATE		✓		[Papernot et al. 2016a]
Scalable private learning		✓		[Papernot et al. 2018]
Generating DP datasets		✓		[Triastcyn and Faltings 2018]
DSSGD			✓	[Shokri and Shmatikov 2015]
Privacy-preserving DL via additively HE	✓		✓	[Aono et al. 2018]
SecureML	✓		✓	[Mohassel and Zhang 2017]
MiniONN	✓		✓	[Liu et al. 2017]
DeepSecure			✓	[Rouhani et al. 2018]
Gazelle			✓	[Juvekar et al. 2018]

mize groups of studies on attacks on deep learning models and defenses against those attacks.

3.1. Security Attacks on Deep Learning Models

In this section, we will describe two major attacks on deep learning depending on which phase of the workflow of machine learning is interfered: poisoning attack and evasion attack as described in Table I. If the attack engages in the training phase and tries to destroy the model while training, it is called the poisoning attack and the example used in this attack is referred as an adversarial training example. On the other hand, adversarial (test) examples are used in the inference phase and intentionally lead the model to misclassify the input. This attack is called the evasion attack.

As well as phase of workflow, Attack scenarios on deep learning models can be differed by the amount of information that the attacker has about the model. If the attacker has full access to all information in the model, including the model structure and the values of all parameters, it shows high attack success rate that can not exist in reality. If the adversary has limited information about the model such as the predicted

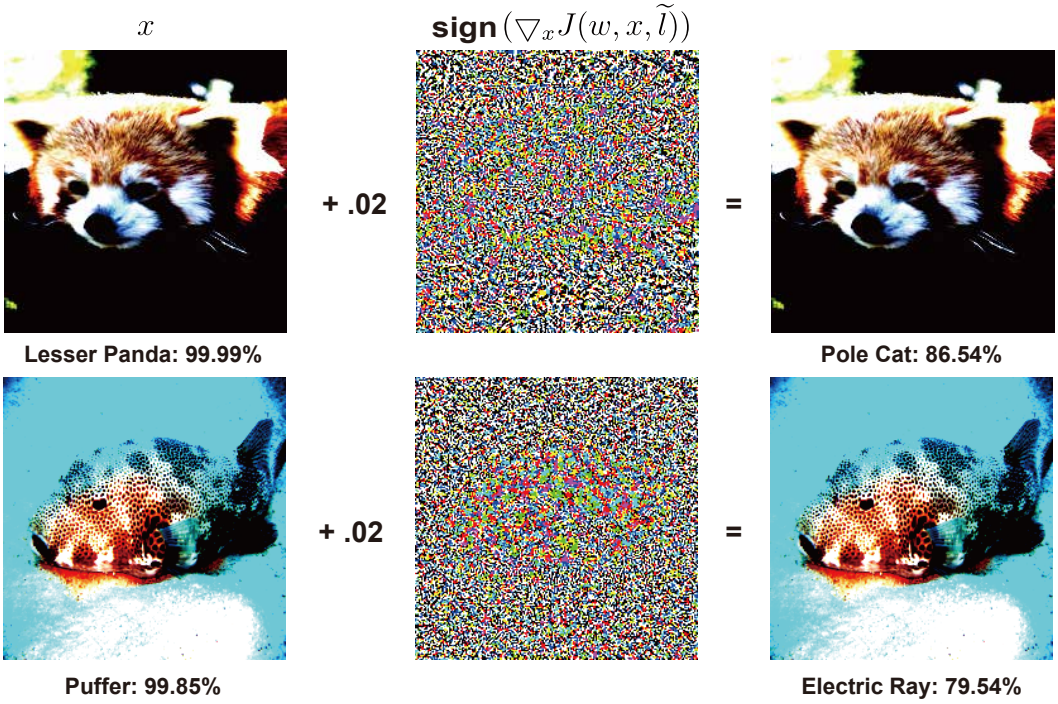


Fig. 5: Two adversarial examples generated by the fast gradient sign method Goodfellow et al. [2014b]. Left column: the original images. Middle column: the generated adversarial perturbations. Right column: the adversarial images into which the adversarial perturbation is added.

label of the input or limited authority, it is hard to attack and need alternative method like a substitute model or data.

Depending on the attacker's goal, moreover, the attacks can be divided into targeted and non-targeted attacks. That is, if the adversary aims to alter the classifier's output to some pre-specific target label, this attack is called a targeted attack; in the case of non-targeted attack, the adversary's goal is to make the classifier choose any incorrect label. Generally, a non-targeted attack shows higher success rate compared to a targeted one.

3.1.1. ***Evasion Attack.***

White-box Attack. The initial study on evasion attacks started from [Szegedy et al. 2013]. Szegedy et al. [2013] suggested the idea of using the limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm to generate an adversarial example. The authors propose a targeted attack method, which involves solving the simple box-constrained optimization problem of

$$\begin{aligned} &\textbf{minimize} \ \|n\|_2 \\ &\textbf{s.t.} \qquad f(x+n) = \tilde{l}, \end{aligned} \tag{8}$$

where $x \in \mathbb{R}^{I \times J \times K}$ is the untainted image ($I \times J \times K$ represents the height, width and channel of the image), and $\tilde{l} \in \{1...k\}$ is the target label; and n represents the minimum amount of noise needed to disassociate the image from its true label. This method is

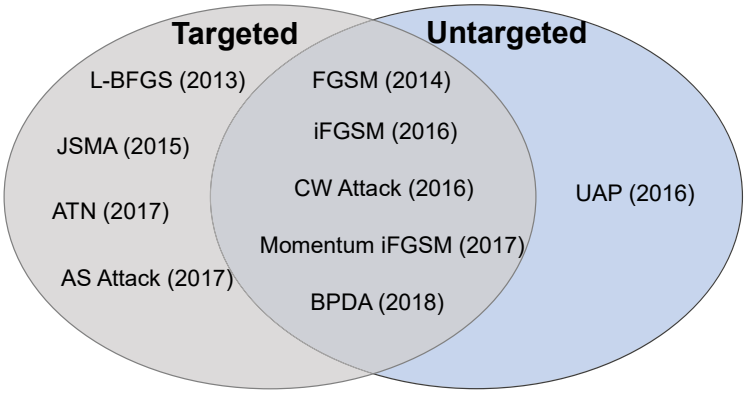


Fig. 6: White-box targeted vs. non-targeted attack methods. Abbreviation: L-BFGS [Szegedy et al. 2013] = Limited-memory Broyden–Fletcher–Goldfarb–Shanno, FGSM [Goodfellow et al. 2014b] = Fast Gradient Sign Method, JSMA [Papernot et al. 2016a] = Jacobian-based Saliency Map Attack, CW attack [Carlini and Wagner 2017b] = Carlini's and Wagner's attack, UAP [Moosavi-Dezfooli et al. 2017] = Universal Adversarial Perturbation, ATN [Baluja and Fischer 2017] = Adversarial transformation networks, AS attack [Athalye and Sutskever 2017] = Athalye's and Sutskever's attack, and BPDA [Athalye et al. 2018] = Backward Pass Differentiable Approximation.

designed to find the smallest perturbation needed for a successful attack. Sometimes it creates an inapplicable adversarial perturbation n, which performs only the role of blurring the image. This form of attack has a high misclassification rate but also a high computational cost since the adversarial examples are generated as a result of solving the optimization problem in Equation 8 via a box-constrained L-BFGS.

On the other hand, Carlini's and Wagner's attack (CW attack) [Carlini and Wagner 2017b] is based on the L-BFGS attack [Szegedy et al. 2013], and it modifies the optimization problem in Equation 8 as

$$\textbf{minimize}\ D\left(\tilde{x}, x\right) + c \cdot g\left(\tilde{x}\right) \tag{9}$$

where D is a distance metric that includes L_p, L_0, L_2, and L_∞, $g(\tilde{x})$ is an objective function in which $f(\tilde{x}) = \tilde{l}$ if and only if $g(\tilde{x}) \leq 0$ and $c > 0$ is a properly chosen constant. This modification enables Equation 9 to be solved by the existing optimization algorithms. The use of the Adam [Kingma and Ba 2014] optimizer enhances the effectiveness in finding adversarial examples quickly. For relaxation, they use the method of change of variables or projection into box constraints for each optimization step.

Papernot et al. [2016a] introduced a targeted attack method, that optimizes under the L_0 distance, which is known as the Jacobian-based Saliency Map Attack (JSMA). It constructs a saliency map based on the gradient derived from the feedforward propagation and modifies the input features that maximize the saliency map in a way that increases the probability to be classified as target label $\tilde{l}$.

In general, a deep learning model is described as non-linear and overfitting, but in [Goodfellow et al. 2014b], they introduce the fast gradient sign method (FGSM). Goodfellow et al. [2014b] assert that the main vulnerability of neural networks to adversarial perturbation is caused by their linear nature. Their method linearizes the cost function around the present value, and finds its maximum value from the following

closed-form equation as follows:

$$\tilde{x} = x + \upsilon \cdot \text{sign}(\nabla_x J(w, x, \tilde{l})) \tag{10}$$

where $\tilde{x}$ is the adversarial example; x is the untainted input to the model, and $\tilde{l}$ is the target label; υ decides how strong the adversarial perturbation is that is applied to the image, and J is the cost function to train the network. Although the proposed method can generate adversarial examples with relatively low computational costs, it shows a low success rate.

To overcome the shortcomings of the previous two ideas, various compromises have been made, and iterative FGSM [Kurakin et al. 2016a] is one of them. Iterative FGSM utilizes a specialized iterative optimization. It utilizes the FGSM for several steps but with a smaller step size. The clip function implements a per-pixel clipping of the image. Technically, the result will be in L_∞ ε-neighborhood of the original image. The detailed update rule is described as follows:

$$\tilde{x}_0 = x, \tilde{x}_{N+1} = \text{Clip}_{x,\upsilon}\left\{\tilde{x}_N + \upsilon \cdot \text{sign}(\nabla_x J(w, \tilde{x}_N, \tilde{l}))\right\} \tag{11}$$

where $\tilde{x}$ is the adversarial example iteratively optimized, and $\tilde{x}_N$ is the intermediate result in the N-th iteration. As a result, it showed improved performance in terms of the generation throughput and the success rate.

Using the iterative method proposed above, Dong et al. [2017] added a momentum term to improve the transferability of the generated adversarial examples as described in Figure 5. It was presented in the Adversarial Attacks and Defences Competition [Dong et al. 2017] at NIPS 2017, and it won the first place in both the non-targeted attack and targeted attack tracks. The main idea of the paper is as follows:

$$g_{N+1} = \mu \cdot g_N + \frac{\nabla_x J\left(f\left(\tilde{x}\right), \tilde{l}\right)}{\left\|\nabla_x J\left(f\left(\tilde{x}\right), \tilde{l}\right)\right\|_1}, \; x_{N+1} = \text{Clip}_{x,\upsilon}\left\{\tilde{x} + \upsilon \cdot \text{sign}(g_{N+1}))\right\}. \tag{12}$$

Compared to Equation 11, adding the g_N decay provides the momentum with a gradient.

An adversarial transformation network (ATN) [Baluja and Fischer 2017] is another targeted attack method. An ATN is a neural network trained to generate a targeted adversarial examples with minimal modification from the original input, making it hard to differentiate from the clean examples.

Beyond adding different noise values per input for misclassification, universal adversarial perturbations [Moosavi-Dezfooli et al. 2017] show the presence of universal (image-agnostic) perturbation vectors that cause all natural images in a dataset to be misclassified at a high probability. The main focus of the paper is to find a perturbation vector $n \in \mathbb{R}^{I \times J \times K}$ that tricks the samples in the dataset. Here, μ represents the dataset that contains all of the samples.

$$\hat{k}\left(x + n\right) \neq \hat{k}\left(x\right), \text{for most } \; x \sim \mu. \tag{13}$$

The noise n should satisfy the following conditions of $\|n\|_{_1} \leq \xi$, and the conditions of:

$$\underset{x \sim \mu}{\mathbf{P}}\left(\hat{k}\left(x + n\right) \neq k\left(\hat{x}\right)\right) \geq 1 - \delta, \tag{14}$$

where $\hat{k}$ is the classifier; ξ limits the value of the perturbation, and δ quantifies the specified fooling rate for all images.

In the case of most adversarial attacks, the efficacy of each attack can be decreased via transformations, such as viewpoint shift and camera noise. There is a very low per-

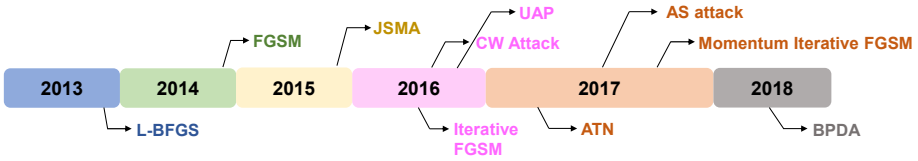


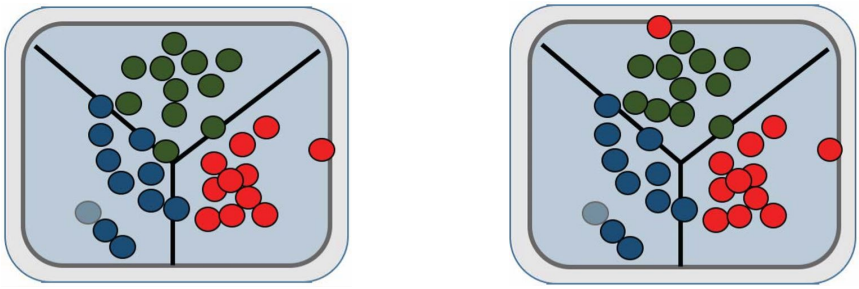
Fig. 7: Historical timeline of white-box attacks. (Abbreviations: L-BFGS [Szegedy et al. 2013] = Limited-memory Broyden–Fletcher–Goldfarb–Shanno, FGSM [Goodfellow et al. 2014b] = Fast Gradient Sign Method, JSMA [Papernot et al. 2016a] = Jacobian-based Saliency Map Attack, CW attack [Carlini and Wagner 2017b] = Carlini's and Wagner's attack, UAP [Moosavi-Dezfooli et al. 2017] = Universal Adversarial Perturbation, ATN [Baluja and Fischer 2017] = Adversarial transformation networks, AS attack [Athalye and Sutskever 2017] = Athalye's and Sutskever's attack, and BPDA [Athalye et al. 2018] = Backward Pass Differentiable Approximation.) Red: CW attack is an advanced idea of L-BFGS. Purple: Like mentioned on each title, FGSM is a basic idea of Iterative and Momentum Iterative FGSMs. Green: UAP and AS attack methods created the idea of generating a special perturbation that is robust to either image preprocessing or resource limitation. Pink: BPDA defeated a recently proposed large number of gradient masking defenses

centage of cases in which the image to which an adversarial noise is added directly applies to the classifier in the system of a physical world. They usually have some steps of preprocessing and angle changing for adjustment. Athalye and Sutskever [2017] proposes a method to overcome this current limitation by generating a perturbation that makes the input have a variety of distortions such as random rotation, or translation, and the addition of noise is implemented to be misclassified in a classifier. In addition, they use a visual difference for a boundary radius ball constraint instead of a distance in a texture space.

A backward pass differential approach attack method [Athalye et al. 2018] has been recently proposed, which is capable of preventing recent gradient masking defense methods. The authors claim that finding defenses that rely on recently suggested gradient masking methods can be circumvented by performing the backward pass with the identity function, which is for approximating true gradients.

Black-box attack. In the real world, accessing models or data sets that are used for training the model, or both are too difficult. Although there are a huge public data (Image, sound, video and etc.), an internal data used for training models from industries still secret. Moreover, models contained in mobile devices are not accessible to attackers. The Black-box attack assumes a situation similar to reality. The attacker has no information about the model and the dataset. The available information is the input format and the output label of a target model the same as when using a mobile application. A target model can be the models hosted by Amazon and Google. In the case of attacker wants to create an aggressive example exploiting gradients, the attacker needs to replace a model as the access to of the targeted model is restricted to the attackers. Black box attack is a replicating a target model without knowledge of the architecture of the target model.

According to Szegedy et al. [2013]; Goodfellow et al. [2014b], neural networks can attack other models regardless of the number of layers and the number of hidden nodes as long as the target task is the same. These authors considered this finding to be due to the neural networks's linear nature, in contrast to previous works in that the transferability is due to the nonlinearity of the neural network. Activation function



(a) Classifier of clean data (b) Classifier of poisoned data

Fig. 8: The functionality of poisoning a sample. (a) The decision boundary after training with normal data, (b) The decision boundary after injecting a poisoning sample.

of sigmoid of ReLU is well known to produce a non-linearity. The sigmoid has the advantage of nonlinearity, but it is tricky to use in learning. On the other hand, ReLU is widely used because it is easy to learn with, but non-linearity does not grow as with sigmoid. Thus, the replication of the target models can learn a similar decision boundary since the target task is the same.

Moreover, Papernot et al. [2016b] showed that transfer is possible between traditional machine learning techniques and neural networks using the experiment's intra-technique transferability; between the same algorithms but different initialization; cross-technique transferability; between different algorithm like SVM and neural networks. For example, Kurakin et al. [2016a] assumes the case in which a model obtain input by a camera or sensors that is not directly obtained, and as a result, it attacks a neural network using the pictures of the attacks. The attack still operated and thus, this approach showed robustness to transformation.

As described above, the transfer attack is possible by making a substitute model for a target. In the process of the substitution of a target model, it is possible to use an approximate architecture such as the CNN, RNN, and MLP exploiting the input format (image or sequence). The model can be trained by collecting data similar to the data obtained by learning the target from the public. However, The cost of collecting is enormous. Papernot et al. [2017] solved this issue using an initial synthetic dataset and Jacobian-based data augmentation method. If the dataset of the target is MNIST, then the initial synthetic dataset can be handcrafted digital digit images approximately 100 or the subset of a test set that is not used when training the targets. The label can be obtained by putting the data as a input in oracle which is the target model. After training the substitute using the input and label pair, the authors crafted an adversarial example using Goodfellow et al. [2014b] and Papernot et al. [2016b]. The results of the experiment on the transferability of the MNIST case showed about 90% success rate that corresponded to the epsilon range of 0.5–0.9. However, if an attacker wants to label its inputs by blowing queries to a service such as Google or Amazon, the attacker has a limited number of queries or a high probability of being caught by the detector due to the large number of instances. To release this problem, Papernot et al. [2016b] introduced reservoir sampling, and it effectively reduced the number of data instances needed to train the substitute.

3.1.2. ***Poisoning Attack.*** If the evasion attack is to avoid the decision boundary of the classifier at the test time, the poisoning attack intentionally inserts a malicious example into the training set at the training time in such a way as to interfere with the learning of the model or to attack at the test time more easily. There is a large number of poisoning attack methods that can be successfully applied to traditional machine

learning such as SVM or LASSO, but there are only a few for neural networks. The tradition poisoning attack can be expressed mathematically, but neural networks have been difficult to poison because of their complexity. A poisoning attack can be divided into a white-box and black-box attack similar to an evasion attack, but it will be expressed as strong adversaries and weak adversaries to make a concrete expression suitable for poisoning attack. The goal of adversaries could be to completely ruin the learning of the system or to make the backdoor to be recognized as a man of power with authentication when deployed.

Strong adversaries refer to adversaries with powerful permissions that can manipulate the parameter values of the model with direct access to the model and training data, similar to a white-box attack, or poison the training data to spoil the learning. Their main purpose is to subvert the training process by injecting malicious samples, but the accessibility can be different. The authors of Muñoz-González et al. [2017] presented two attack scenarios of strong adversaries, which are perfect-knowledge (PK) attacks and limited-knowledge (LK) attacks. As the terms suggest, a PK attack scenario is an unrealistic setting, and hence, it is only assumed for a worst-case evaluation of the attack. On the other hand, under LK attack scenarios, the typical knowledge that the attacker possesses, is described as $\theta = (\hat{\mathcal{D}}, \mathcal{X}, \mathcal{M}, \hat{w})$, where $\mathcal{X}$ is the feature representation and $\mathcal{M}$ is the learning algorithm. The hat symbol denotes limited knowledge of a given component; $\hat{\mathcal{D}}$ is the surrogate data available to the attacker, and $\hat{w}$ is the learned parameter from $\hat{\mathcal{D}}$.

$$\begin{aligned} \mathcal{D}_c^* \in \underset{\mathcal{D}'_c \in \phi(\mathcal{D}_c)}{\arg\max} \quad & \mathcal{A}(D'_c, \theta) = J(\hat{\mathcal{D}}_{\text{val}}, \hat{w}) \\ \textbf{s.t.} \quad & \hat{w} \in \underset{w' \in W}{\arg\min}\, J(\hat{\mathcal{D}}_{\text{tr}} \cup \mathcal{D}'_c, w') \end{aligned} \tag{15}$$

where the surrogate data $\hat{\mathcal{D}}$ is divided into the training data $\hat{\mathcal{D}}_{\text{tr}}$ and validation data $\hat{\mathcal{D}}_{val}$. $A(\mathcal{D}'_c, \theta)$ is an objective function that evaluates the impact of the adversarial examples on the clean examples, and it can be defined in terms of a loss function and $J(\hat{\mathcal{D}}_{\text{val}})$, which measures the performance of the surrogated model using $\hat{\mathcal{D}_{\text{val}}}$. The optimization problem comprised of bilevel optimization and the influence of $\mathcal{D}_c$ is propagated using $\hat{w}$. The goal of the optimization is to ruin the system, and the label of the poison is generic. If a specific target is required, the Equation 15 is changed to

$$\mathcal{A}(\mathcal{D}'_c, \theta) = -J(\hat{\mathcal{D}}'_{\text{val}}, \hat{w}) \tag{16}$$

where $\hat{\mathcal{D}}'_{\text{val}}$ is the manipulated validation set, which contains the same data as $\hat{\mathcal{D}}$ but with misclassified labels for the desired output. Muñoz-González et al. [2017] proposed the back-gradient optimization to solve the Equation 15 or 16 and generated poisoning examples, and compared with the previous gradient-based optimization methods. Since gradient-based optimization requires a strict convexity assumption of the objective function and Hessian-vector product, Muñoz-González et al. [2017] argue that such an approach is not applicable to complex learning algorithms including neural networks and deep learning architectures. In addition, Yang et al. [2017] introduce the possibility of applying the gradient-based method to DNNs, and they develop a generative method inspired by the concept of GAN [Goodfellow et al. 2014a]. Rather than computing the gradients directly, Goodfellow et al. [2014a] used an auto-encoder as a generator. Hence, the results show a speed up of the more than 200x compared to the gradient-based method.

A weak adversary has a capability that insider intruders can add a few poisoned samples without having authority for model or training. In contrast to previous studies

that assume strong adversaries, Chen et al. [2017] introduce three constraints: 1) no knowledge of the model, 2) injecting a small fraction of training data, and 3) poisoning data not detected by humans. This is based on situations similar to the real world. They proposed two methods, input-instance-key strategies and pattern key strategies, for weaker adversaries to break the security and obtain privilege of the face recognition system. The former is to make an image be a key image, and makes it recognized as a targeted label. In consideration of the situation of going through the camera, several random noise add to sample. In the latter case, three strategies exist: 1) blended injection strategy, 2) accessory injection strategy, and 3) blended accessory injection strategy. The first is to blend a kitty image or a random pattern onto the input image. However, it is unreasonable to add a specific pattern to the image captured by the camera in real situations, and thus, the second is to apply an accessory such as glasses or sunglasses to the input. It is easy to use at inference stage. When training, the parts other than the glasses have the same value as the input image, and the pixel values of the glasses are applied only to the glasses. The last method combines the first and the second. Unlike previous studies in which poisoning data accounted for 20 percent of the training data, only five poisoned samples were added when 600,000 training images were used for instance-key, and approximately 50 poisoned samples were used for pattern-key strategies. They were able to create a successful backdoor by adding a small fraction of poisoned samples.

Even if a small number of samples are placed in the training data, the attack may become useless when the data is preprocessed by the experts or crawled directly on the web and labeled by the experts. The above attacks can not guarantee success without inside intruders. In order to cope with this situation, Shafahi et al. [2018] adopts a method of changing the feature representation of the input, unlike the conventional method of changing the label. For example, when there is a image of a dog and a bird, it uses a gradient of the model to change the dog to have a bird-like feature representation. At this time, a picture of a dog is called a base image, and a picture of a bird is called a target image. The goal is to change the decision boundary by adding a perturbed base image to the training data using a gradient. As a result, the target image is misclassified as a class of base image and the target can be used as a key to exploit the model at will of the attacker. The authors experiment the attacks in two retraining situations: end-to-end learning which fine-tunes entire model and transfer learning which fine-tunes only final layer. The explained method using a base and a target image called a one-shot kill attack is successfully applied to transfer learning which change decision boundary dramatically, but are not applied to end-to-end learning which change the lower layer extracting fundamental features. From this, Shafahi et al. [2018] have succeeded in poisoning attack by proposing watermarking method of projecting target image to base image by adjusting opacity and multiple poison instances attacks using several target and base images to create attack sample efficiently.

3.2. Defense Techniques against Deep Learning Models

There are variety types of defense techniques against deep learning models. Defense techniques can be categorized in two big groups which are evasion and poisoning. Defense techniques against evasion attacks can be further categorized into two groups, namely, non-obfuscated gradient masking (which includes adversarial training), and obfuscated gradient masking. Defense techniques are used not limited to attacks but also used to improve prediction results. For example, Kurakin et al. [2016b] suggests that adversarial training can be employed in the scenario of a) when a model is overfitting, and b) when security against adversarial examples is a concern. For example, the recent work on speech data [Sun et al. 2018] trained the DNN using adversarial

examples along with the clean examples, to increase the robustness against evasion attacks.

3.2.1. ***Defense Techniques Against Evasion Attacks***. The basic idea of gradient masking is to have a method that augments the adversarial examples, which is created by making the gradients point slightly farther than a decision boundary with clean examples and a method that makes use of techniques that hand over incorrect or foggy gradients to an adversary; these are both currently under the category of gradient masking.

Non-Obfuscated Gradient Masking Currently, most representative studies of non-obfuscated gradient masking involve adversarial training [Szegedy et al. 2013], [Goodfellow et al. 2014b]. As a monumental article, it is still one of the significant methods among the various gradient masking methods. A robust model to an expected adversarial attack can be implemented by augmenting the training set with truly labeled adversarial examples. [Szegedy et al. 2013] showed that adversarial example of Equation 10 could be used as regularizer by training on a mixture of clean and adversarial examples. The regularizer with an objective function based on the FGSM methods by training with on the mixture examples is formalized as follows:

$$J(\theta, x, l) = \alpha J(\theta, x, l) + (1 - \alpha) \cdot J(\theta, x + \upsilon \cdot \text{sign}(\nabla_x J(w, x, l))). \tag{17}$$

The adversarial training procedure can be interpreted as minimizing the worst case error with a perturbed data by an adversary. The adversarial training can also be seen as learning to play an adversarial game with the model which is able to request labels on new points. Goodfellow et al. [2014b] explains generalization aspect of adversarial examples to support adversarial training. We often notice that an example generated for a model is often misclassified by other models. The behavior occurs due to extreme non-linearity and over-fitting cannot account for various different behaviors. For this, Goodfellow et al. [2013] suggested that the generative training could provide more constraint on the training process enforcing a model to distinguish real from fake data. The generative training limits the direction of perturbation by training across different models that have similar functions when trained to perform the same task.

Adversarial training originally developed for small model with MNIST that did not use batch normalization. Kurakin et al. [2016b] extended the original work to ImageNet [Deng et al. 2009] adding batch normalization step. The relative weight of adversarial examples are independently controlled in each batch with following loss function:

$$J = \frac{1}{(m-k) + \lambda k}\left(\sum_{\text{CLEAN}} (x_i|l_i) + \lambda \sum_{\text{ADV}} J(\tilde{x_i}|l_i)\right) \tag{18}$$

where $J(x|l)$ is a loss on a single example x with true class l; m is total number of training in the minibatch; k is the number of adversarial examples in the minibatch and λ is a parameter for the relative weight of adversarial examples in the loss.

In 2018, Tramèr et al. [2017] proposed a defense method with Ensemble adversarial training that is also robust to black-box attacks by containing adversarial examples generated from other models. The approach decouples adversarial example generation from the trained model to increase diversity of perturbations seen during training. Tramèr et al. [2017] introduces a connection between Ensemble adversarial training and multiple-source domain adaptation [Mansour et al. 2009; Zhang et al. 2012]. Assuming a target distribution takes a role of unseen black-box adversary, the output has bounded error on attacks from a future black-box adversary.

Table VI: Mapping continuous-valued input to quantized inputs, one-hot coding, and thermometer codes.

Real-valued	Quantized	One-hot	Thermometer
0.13	0.15	[0100000000]	[0111111111]
0.66	0.65	[0000001000]	[0000001111]
0.92	0.95	[0000000001]	[0000000001]

Obfuscated Gradient Masking The defense approaches against obfuscated gradient masking in general follow three types of obfuscated gradients, which are shattered gradients, stochastic gradients and vanishing/exploding gradient [Athalye et al. 2018].

Having shattered gradients means that incorrect gradients are achieved by making the model intentionally non-differentiable operationally or unintentionally numerically unstable. The purpose of a shattered gradient attack is to break this linearity with the consideration of the neural network, which in general, behaves in a largely linear manner [Athalye and Sutskever 2017]. In the case of images and other high dimensional space, the linearity will have a large effect on the model's prediction with small values of ϵ, making the model vulnerable to adversarial attacks. A recent defense algorithm over the shattered gradient technique is to exploit thermometer encoding [Buckman et al. 2018] neural networks to break the linearity. The method exploits non-differentiable and non-linear transformation to the input replacing one-hot encoding to thermometer encoding as shown in Table VI. With the input x, an index $j \in \{i, \cdots, k\}$, and the thermometer $\tau(j) \in \mathcal{R}^K$, the thermometer vector is formally defined as follows:

$$\tau(j)_l = \begin{cases} 1, & \text{if} \quad l \geq j \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

Then the thermometer (discretization) function f is defined pixel-wise for a pixel $i \in \{i, \cdots, n\}$ as:

$$f_{\text{therm}(x)_i} = \tau(b(x_i)) = \mathcal{C}(f_{\text{onehot}}(x_i)) \tag{20}$$

where $\mathcal{R}$ is the cumulative sum function, $C(c)_l = \sum_{j=0}^{l} c_l$, and b is a quantization function.

The stochastic gradients make a model obfuscated by test time randomness. The algorithm randomly drops some neurons of each layer to 0 considering their original output value, meaning that the network stochastically prunes a subset of the activations in each layer during the forward pass. The survived activations are scaled up to normalize the dynamic range of the inputs to the subsequent layer [Dhillon et al. 2018]. Similarly, Guo et al. [2017] proposed a transformation approaches under the baseline of image cropping, rescaling [Graese et al. 2016], bit-depth reduction [Xu et al. 2017], JPEG compression [Kinga and Adam 2015], and total variance minimization [Rudin et al. 1992]. Total variance minimization approach first drops pixels in a random manner, and reconstructs images by replacing small patches using minimum graph cuts in overlapping boundary regions to remove artificially crafted in the edge. The total variation minimization of an image z is formalized as follows:

$$\min_z ||(1 - \tilde{x}) \odot (z - x)||_2 + \lambda_{\text{TV}} \cdot \text{TV}_p(z). \tag{21}$$

where $\tilde{x}$ is a random set of pixels by sampling a Bernoulli random variable $x(i, j, k)$ for pixel location (i, j, k), $\odot$ denotes element-wise multiplication, TV denotes total vari-

ance, and $\mathrm{TV}_p(z)$ represents L_p total variation of z. Finally, Buckman et al. [2018] demonstrate thermometer code, which improves the robustness to adversarial attacks. Samangouei et al. [2018] proposed Defense-GAN, which is a similar defense method as PixelDefend, but it uses a GAN instead of a PixelCNN.

The vanishing/exploding gradients make a model unusable by deep computation. The basic idea is to purify adversarially perturbed images back to clean examples by exploiting a pixelCNN as a generative model. The purified image is then used for the unmodified classifier. A recent defense algorithm exploits PixelCNN [Oord et al. 2016] to build PixelDefend [Song et al. 2017] to approximate the training distribution. The PixelCNN is a generative model which is designed for images tracking likelihood over all pixels by factorizing it into a product of conditional distributions:

$$\mathrm{P}_{\mathrm{CNN}}(x) = \prod_i \mathrm{P}_{\mathrm{CNN}}(x_i | x_{1:(i-1)}). \tag{22}$$

The PixelDefend train a PixelCNN model on the CIFAR-10 dataset and use log-likelihood to approximate the true probability density and experimented against adversarial examples from RAND, FGSM, BIM, DeepFool and CW methods. The result showed that PixelDefend obtains accuracy above 70% for all attacking techniques, while maintaining the performance on clean images.

A robustness against iterative optimization attacks is a key idea for a good defense system that is built a based on machine learning. Nevertheless, the existing gradient-based defense algorithm is designed based on the gradient of the initial version, which makes vulnerability to gradient based attack. The attack methods attempts to search for a parameter υ such that image channel $c(x + \upsilon) \neq c^*(x)$ either maximizing or minimizing $||\upsilon||$. Athalye et al. [2018] exploits projected gradient descent to set υ and used l_2 Lagrangian relaxation Carlini and Wagner [2017a]. With the attack methods, Athalye et al. [2018] showed that most of the obfuscated gradient based defenses are vulnerable to iterative optimization attacks [Kurakin et al. 2016a; Madry et al. 2017; Carlini and Wagner 2017a] and become standard algorithm evaluating defenses.

3.2.2. ***Defense against Poisoning Attacks***. The framework proposed from [Steinhardt et al. 2017] takes the approach of removing outliers that are outside the applicable set. In binary classification, they aim to find the centroids of the positive and negative classes. Then, they remove points that are too far away from each corresponding centroid. To find these points, they make use of two methods: a sphere defense that removes points outside the spherical radius, and a slab defense that discards points that are too far away from the line in a complimentary way.

Koh and Liang [2017] uses influence functions to track model predictions and identify the most influential data points that are responsible for a given prediction. They show that approximations in functions can still provide important information in non-convex and non-differentiable models where the theory breaks down. They also claim that by using influence functions, the defender can check out only the data prioritized by its influence score. This method outperforms previous methods of identifying the greatest training loss for removing the tainted examples.

Paudice et al. [2018b] also suggests a defense mechanism to mitigate the effects of poisoning attacks on the basis of outlier detection. The attacker tries to have the greatest effect on the defender with a limited number of poisoning points. To mitigate this effect, they first divide the trustworthy dataset $\mathcal{D}$ into different classes, i.e., $\mathcal{D}_+$ and $\mathcal{D}_-$. Then, they use the curated data trains distance-based outlier detectors for each class. The outlier detection algorithm calculates the outlier score for each x in the original (total) data set. There are many ways to measure the outlier score, such as using SVM or LOF as a detector. The empirical cumulative distribution function

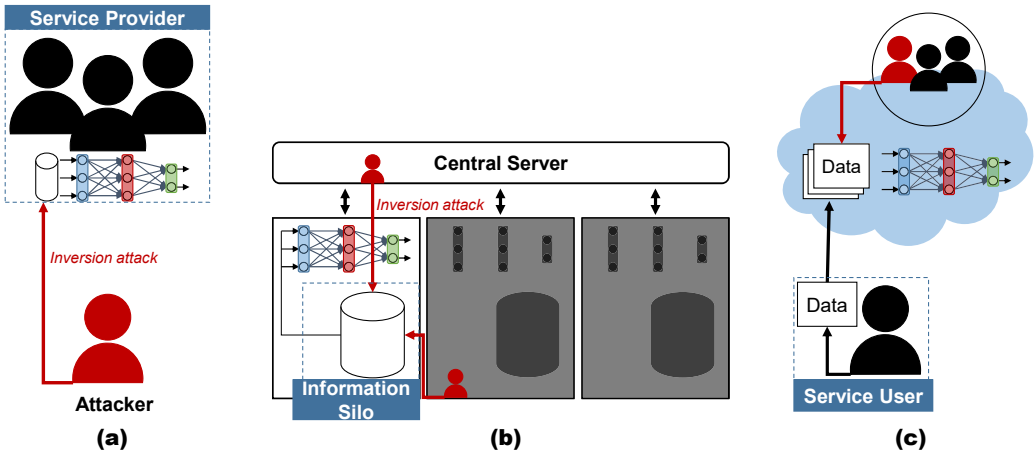


Fig. 9: Private AI: Potential threats in perspectives of (a) service provider, (b) information silo, (c) user.

(ECDF) of training instances is used to calculate the threshold for detecting outliers. By removing all of the samples that are expected to be contaminated, the defender can collect new data sets to retrain the learning algorithm.

Paudice et al. [2018a] chooses to re-label data points that are considered to be outliers instead of removing them. The label flipping attack is a special case of data poisoning that allows an attacker to control the label of a small number of training points. Paudice et al. [2018a] proposes a mechanism that considers the points farthest from the decision boundary to be malicious, and it reclassifies them. The algorithm reassigns the label of each instance of using a k-NN. For each sample of training data, they first find the closest k-NN using the Euclidean distance. If the number of data points with the most common label among k-NN is equal to or greater than a given threshold, the corresponding training sample is renamed to the most common label in the k-NN.

4. PRIVATE AI

Deep learning algorithms that account for most of the current AI systems rely highly on data. Hence, DL is always exposed to privacy threats, and it is imperative that the privacy of the training data be preserved. Hence we define *Private AI*, the AI system that preserves the privacy of the concerned data.

4.1. Potential Threats from Different Perspectives

4.1.1. ***Potential Threats in the Service Providers Perspective.***. When companies provide deep learning models and services to the public, there are potential risks in that the models leak private information even without revealing the original dataset. As shown in Figure 9a, a model inversion attack occurs when an adversary uses the pre-trained deep learning model to discover the data used in the model training. Such attacks seek to manipulate the correlation between the target, the unknown input and the model output.

Recent studies on inversion attacks show that a model inversion attack is possible, by recovering images used in training [Fredrikson et al. 2015] or performing a membership test to know whether an individual is in a dataset or not [Shokri et al. 2017]. Furthermore, deployed deep learning services are exposed to data integrity attacks as

well. Because the deployed service demands the user's data, adversaries can attempt to break the integrity of the data at the servicing server. If a data holder includes the collected data without an integrity check, the broken integrity can mislead or even ruin the model.

4.1.2. ***Privacy Violation in Information Silos****.* Information silos are a group of exclusive data management systems that are related. Data silos, often represented by hospitals or government agencies, have data of a similar nature and can derive productive output through collaborative data mining, but they do not share data by intellectual property or privacy breach issues. As shown in Figure 9b, secure multi-party computation (SMC) occurs when a set of parties like silos, with private inputs wish to compute some joint function of their inputs [Lindell 2005]. Similarly, the idea of secure multi-party training, which trains a joint deep learning model of private data input, has been emerging. In such training processes, the data privacy of each participant must be preserved in the face of adversarial behavior by other participants or by an external party.

Hitaj et al. [2017] shows that a distributed, federated, or decentralized deep learning approach is fundamentally broken and does not protect the training sets of honest participants from a GAN-based attack. The adversary based on GAN deceives a victim into releasing more accurate information on sensitive data.

4.1.3. ***Potential Threats in the User's Perspective****.* Because many deep learning-based applications have been introduced in industry, such service users are under serious threats of the invasion of privacy [Gilad-Bachrach et al. 2016; Sanyal et al. 2018]. Since deep learning models are too large and complicated [He et al. 2016; Huang et al. 2017a] to be computed on small devices such as mobile phones or smart speakers, most service providers require users to upload their sensitive data, such as their voice recordings or face images and compute on their (cloud) servers. The problem is that upon uploading, the users lose control of their data. In other words, the users cannot delete their data and cannot check how their data is used, as shown in Figure 9c. As the recent Facebook's privacy scandal suggests, even when there are some privacy policies, it is difficult to notice or restrain from excessive data exploitation. In addition, since hardware requirements for deep learning are enormous, Machine-Learning-as-a-Service (MLaaS) provided by Google, Microsoft, or Amazon has gained in popularity among deep learning-based service providers. Such remote servers even make it difficult to manage the users' data privacy.

4.2. Defense Techniques against Potential Threats

Unlike many attacks that are attempted in the domain of SecureAI, only a few attacks are attempted in the field of PrivateAI as well as defense with respect to privacy preserved deep learning. We observed that this finding is due to the nature of privacy preserving techniques. Exploiting traditional security to the field of deep learning require encryption and decryption phases, which make it impractical in a real world due to the enormous computational complexity. As a result, a homomorphic encryption is a one of the few security techniques that can be exploited in deep learning. As one further step to Private AI, the differential privacy technique is actively exploited in deep learning. In the following section, we detail Private AI, which adopts the most recent privacy-preserving methods.

4.2.1. ***Homomorphic Encryption on Deep Learning****.* CryptoNets [Gilad-Bachrach et al. 2016] took the initiative of applying neural networks for inferencing on the encrypted data. CryptoNets utilize the leveled HE scheme YASHE' [Bos et al. 2013] for the privacy-preserving inference on a pre-trained CNN model. It demonstrated over 99%

accuracy in detecting handwritten digits (MNIST data set [LeCun et al. 2010]). However, leveled HE leads to serious degradation in terms of the model accuracy and efficiency. Furthermore, because of the square activation function being replaced from nonpolynomial activation and the converted precision of the weights, the inferencing model obtains results that are quite different from the trained model. Hence, it is not suited for the recent complicated models [He et al. 2016; Huang et al. 2017a]. In addition, the latency of the computation is still of the order of hundreds of seconds, while Gilad-Bachrach et al. [2016] achieved a throughput of 50,000 predictions in an hour. Cryptonets' ability to batch images together can be useful in which applications where the same user wants to classify a large number of samples together. In the simplest case in which the user only wants a single image to be classified, this feature does not help.

In return, CryptoDL [Hesamifard et al. 2017] and Chabanne et al. [2017] attempted to improve CryptoNets by low degree polynomial approximations on activation functions. Chabanne et al. [2017] applied batch normalization to reduce the accuracy gap between the actual trained model and the converted model with an approximated activation function at the inference phase. The batch normalization technique also enabled fair predictions on a deeper model.

As a recent bootstrapping FHE technique was introduced [Chillotti et al. 2016], TAPAS [Sanyal et al. 2018] and FHE-DiNN [Bourse et al. 2017] were proposed. Since the method proposed by Chillotti et al. [2016] supports operations on binary data, both utilized the concept of Binary Neural Networks (BNNs) [Courbariaux et al. 2016]. FHE-DiNN [Bourse et al. 2017] utilized discretized neural networks with different weights and input dimensions to evaluate Chillotti et al. [2016] on DNNs. In comparison, TAPAS [Sanyal et al. 2018] binarized weights and enabled binary operations and sparsifications techniques. Both FHE-DiNN and TAPAS showed faster prediction than the approaches based on leveled HE. It is also notable that while leveled HE methods only support batch predictions, bootstrapping FHE-based methods enabled the predictions on single instances, which is more practical.

4.2.2. ***Secure Multi-Party Computation (SMC) on Deep Learning***. There are two major types of privacy-preserving deep learning algoirhtms related with multiple parties known so far: ones are the algorithms based on the conventional distributed deep learning algorithms [Dean et al. 2012; Abadi et al. 2016a; Lee et al. 2018] enables the parties to participate in training or testing deep learning models without revealing their data or models. The others are based on secure two-party computation (2PC) combined with homomorphic encryption (HE) and garbled circuit (GC). Such algorithms assume two parties which is a user who provides data and a server that operates deep learning based on the provided data. Modern cryptography techniques combined with MPC techniques such as oblivious transfer, they attempted to securely secure the data transfer process as well.

Distributed selective SGD (DSSGD) [Shokri and Shmatikov 2015] proposed collaborative deep learning protocols with different data holders to train joint deep learning models without sharing their training data. This approach is very similar to the prior distributed deep learning algorithms [Dean et al. 2012; Abadi et al. 2016a; Lee et al. 2018]. With the coordinated learning models and objectives, the participants train their local models and selectively exchange their gradients and parameters at every local SGD epoch asynchronously. On the other hand, since DSSGD assumes the parameter server [Li et al. 2014], Aono et al. [2018] pointed out that even with a few gradients, it is possible to restore the data used in training. Hence, to preserve the privacy against the honest-but-curious parameter server, LWE-based homomorphic encryption

was applied with exchanging weights and gradients. The improved privacy achieved by homomorphic encryption, however, trades off with the communication costs.

Secure two-party compuation (2PC) algorithms include SecureML [Mohassel and Zhang 2017], MiniONN [Liu et al. 2017], DeepSecure [Rouhani et al. 2018] and Gazelle [Juvekar et al. 2018]. SecureML [Mohassel and Zhang 2017] is the first privacy preserving method to train neural networks in multi-party computation settings. Using MPC and secret sharing, SecureML can train machine learning algorithms such as linear regression, logistic regression and neural networks. Although the authors of [Mohassel and Zhang 2017] attempts to speedup computation but SecureML still requires large amounts of communication. MiniONN [Liu et al. 2017] transforms the original neural network into *oblivious neural network* for training, using a simplified homomorphic encryption. MiniONN also utilized garbled circuit to approximate the non-linear activation functions. DeepSecure [Rouhani et al. 2018] computes encrypted data inference on DL model using Yao's garbled circuits [Yao 1986] and suggests some practical computing structure and security proof. As the authors of [Juvekar et al. 2018] pointed out, the aforementioned work showed that homomorphic encryption (HE) mainly shows strength in matrix-vector multiplications but restricted to linear operations. On the other hand, garbled circuits (GC) can cause serious communiation overhead while more suited in approximating non-linear functions in DNN models. Hence Gazelle [Juvekar et al. 2018] combines HE and GC that computes linear operations with HE and activation functions with GC.

4.2.3. ***Differential Privacy on Deep Learning.*** By applying differential privacy to the deep learning models, the training data can be protected from the inversion attacks when the model parameters are released. Hence, there are many studies that utilize the differential privacy to deep learning models. Such methods assume that the training datasets and parameters of the model are the database and the responses, respectively, and prove that their algorithms satisfy either Equation 5 or 6.

Depending on where the noise is added, such approaches can be divided into three groups: gradient-level [Abadi et al. 2016b; McMahan et al. 2018; Xie et al. 2018; Acs et al. 2018; Yu et al. 2019], objective-level [Chaudhuri and Monteleoni 2009; Phan et al. 2016, 2017a,b] and label-level [Papernot et al. 2016a, 2018; Triastcyn and Faltings 2018]. The gradient level approach injects noise into the gradients of the parameters in the training phase. The objective-level approach introduces the perturbed objective function by injecting the noise into the coefficients of the original objective function. The label-level approach introduces noise into the label in the knowledge transfer phase of the teacher student model.

The gradient-level approach [Abadi et al. 2016b] proposed a differential private SGD (DP-SGD) algorithm that adds noise to the gradients in the batch-wise updates. It is important to estimate the accumulated privacy loss as learning progress by batch. In particular, the authors of [Abadi et al. 2016b] proposed the moment accountant to track the cumulative privacy loss. The moment accountant algorithm considers privacy loss as a random variable and estimates the tail bound of it. The resulting bounds provide a tighter level of privacy than using the basic or strong composition theorems [Dwork et al. 2006, 2010]. McMahan et al. [2018] introduced user-level differentially private LSTM. In language modeling, it is difficult and ineffective to keep privacy as the word level. Therefore, McMahan et al. [2018] defined user-level adjacent datasets and ensured differential privacy for users. Xie et al. [2018] proposed a Differentially Private Generative Adversarial Network (DPGAN). They injects noise into the gradient of discriminator to get the differentially private discriminator and the generator which is trained with that discriminator also become differentially private based on the post-processing theory [Dwork et al. 2014].

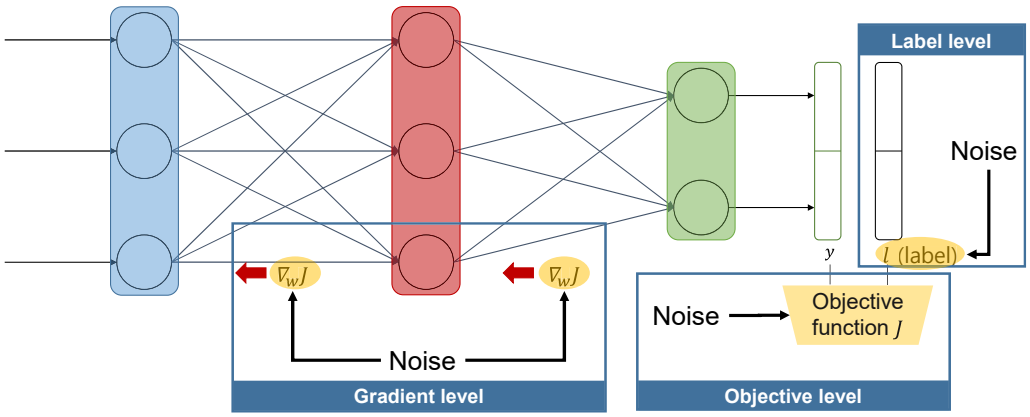


Fig. 10: Overview of differential privacy in deep learning framework

Acs et al. [2018] introduces a differentially private generative model which has mixture of k generative neural networks such as restricted Boltzamnn machine (RBM) [Goodfellow et al. 2016] and variational autoencoder (VAE) [Kingma and Welling 2013]. They applies differnetially private kernel k-means algorithm for clustering the original datasets and uses DP-SGD [Abadi et al. 2016b] to train the each neural networks. They extends the diffentially private k-means clustering [Blum et al. 2005] by applying random Fourier features [Chitta et al. 2012] and improves the accuracy of the trained model by carefully adjusting injected noised in DP-SGD framework.

Yu et al. [2019] introduces some techniques which can be utilized to DP-SGD [Abadi et al. 2016b]. Abadi et al. [2016b] assumes that the bathing method for mini-batch SGD is random sampling, however, in practice, random reshuffling is a widely used batching method. Yu et al. [2019] suggest privacy accounting methods for each case and analyze the characteristics. They also apply concentrated DP (CDP) [Bun and Steinke 2016] to achieve tighter estimation for a large number of iterations and dynamic privacy budget allocation mechanism to improve the performance.

The objective-level approach [Chaudhuri and Monteleoni 2009] disturbs the original objective function by adding noise to the coefficients. Then, the model trained on the disturbed objective function is differential private. Unlike the gradient-level approach, whose privacy loss is accumulated as training progresses, the privacy loss of the objective-level approach is determined at the building objective function and is independent of the epochs. To inject noise into the coefficients, the objective function should be a polynomial representation of the weights. If an objective function is not a polynomial form, the objective-level approach approximates it to the polynomial representation using approximation techniques such as Taylor or Chebyshev expansion. Then, the noise is added to each coefficient to obtain the disturbed objective function. Chaudhuri and Monteleoni [2009] proposed the differentially private logistic regression, whose parameters are trained based on the perturbed objective function. The functional mechanism is applied not only to logistic regression but also to various models such as auto-encoders [Bengio et al. 2009] and convolutional deep belief networks [Lee et al. 2009]. Phan et al. [2016] proposed deep private auto-encoder (dPA) and proved that the dPA is differential private based on the functional mechanism. Phan et al. [2017a] introduced the private convolutional deep belief network (pCDBN), and they utilized the Chebyshev expansion to approximate the objective function to the polynomial form. Phan et al. [2017b] developed a novel mechanism,

called Adaptive Laplace Mechanism (AdLM). The key concept is to add 'more noise' to the input features that are less relevant to the model output, and vice-versa. Phan et al. [2017b] injects noise from the Laplace distribution into the Layer-wise Relevance Propagation (LRP) [Bach et al. 2015a] to estimate the relevance between the output of the model and the input features. They apply an affine transformation based on the estimated relevance to distribute the noise adaptively. AdLM also applies a functional mechanism that perturbs the objective function. These differential private actions are processed before training the model.

The label-level approach injects noise into the knowledge transfer phase of the teacher-student framework. Papernot et al. [2016a] proposed the semi-supervised knowledge transfer model, which is called the Private Aggregation of Teacher Ensembles (PATE) mechanism. PATE is a type of teacher-student model, and its purpose is to train a differentially private classifier (student) based on an ensemble of non-private classifiers (teacher). Figure 10 shows the overview of the PATE approach. Each teacher model learns on disjoint training datasets, and the output of the teacher ensemble is determined by noisy aggregation of each teacher's prediction. The noisy aggregation introduces a noisy label that meets DP, and then, the student model learns the noisy label from the teacher ensemble as a target label. Because the student model cannot access the training data directly and the differential private noise is injected into the aggregation process, PATE ensures safety intuitively and in terms of the DP, respectively. PATE utilizes the *moment accountant* to trace the cumulated privacy budget in the learning process. Later, Papernot et al. [2018] extended the PATE to operate on a large scale environment by introducing a new noisy aggregation mechanism. They showed that the improved PATE outperforms the original PATE on all measures and has high utility with a low privacy budget. Triastcyn and Faltings [2018] applied the PATE to build the differential private GAN framework. The discriminator of GAN frameworks is a type of classifier that determines whether the input data is real or fake. By using PATE as a discriminator, the generator trained with the discriminator is also differential private.

5. DISCUSSION

5.1. Challenges and Future Research Directions

From Section 3, we confirmed that there exist attack methods that can fool or subvert the deep learning models. We reviewed two types of attack scenarios which are the white-box attack and black-box attack. In white-box attack scenarios, most adversaries generate adversarial examples by taking advantage of the gradients of the target DL model, and those examples showed very high misclassification rate. It is crucial for the success of such attacks to acquire the true gradients from the vanilla model, which is the model without any defense method applied, to find sparse or blind spots of the target model. Hence, to defend against such attack methods, many researchers proposed diverse gradient masking defense methods, and these methods showed decent achievements by involving more nonlinearity in a model or preventing the gradients of the model from being copied by an adversary. As the authors of [Athalye et al. 2018] suggest, proper gradient methods show powerful defense performance.

Therefore, it is believed to be beneficial if interpretable AI approaches [Simonyan et al. 2013; Bach et al. 2015b; Shrikumar et al. 2017] can be applied to such attack or defense methods. Interpretable AI analyzes the underlying functions of the deep learning model and determine the way that a deep learning model makes predictions. With deeper understanding of deep learning models, it will be feasible to make a system (model) robust to unseen attacks by identifying blind spots that should be considered and addressed.

From Section 3.1.2, we reviewed some poisoning attack methods on deep learning models. The recent approaches include outlier detections to eliminate [Paudice et al. 2018b] or re-label [Paudice et al. 2018a] the suspected poisoned examples. However, it is a concern that such actions might constrain the decision boundary of the models too much. As Figure 8 suggests, the elimination or re-labeling of some data points can vary a model's decision boundary a large amount. In addition, the degradation of the model accuracy might instead make the model susceptible to the other poisoning approaches. Hence, we need some metrics or evaluation methods to determine whether the model is defended to be safe and sound.

In addition, we reviewed the privacy-preserving deep learning models with the full homomorphic encryption cryptosystems applied in Section 4.2.1. Although the recent methods achieved a high prediction rate despite the strict encryption, the performance in accuracy falls behind the state-of-the-art model performances, and it is not compatible for deeper models. The main reason for this situation is that the FHE methods used in those papers do not include the nonlinear activation functions discussed in Section 2.1.1. Hence, current FHE-based prediction models use different models from the actual trained models. In other words, they train the unencrypted data on the unencrypted typical models, and then, the trained weights and biases are applied to a different model, in which the activation functions are replaced to simple activations such as square functions. A discrepancy between the training and inference models usually causes high degradation in the prediction accuracy. To overcome this deterioration, two approaches are possible: either train the same model from the beginning, or properly transfer the model. As the authors of [Hinton et al. 2015] suggest, the knowledge learned by a DL model can be distilled into another model.

In Section 4.2.3, a large number of attempts were confirmed using differential privacy to protect data privacy in deep learning training. Such methods add noise to gradients or objective functions to confuse the attacker, and give closed-form proof on the differential privacy bounds of the proposed methods. However, from the DL researchers' perspectives, such bounds are insufficient to give practical insights on whether such a privacy bound is strong enough or not. If differential privacy researchers can provide experiments on the assumed attack scenarios or some practical evaluations or metrics, it should be much more informative.

5.2. Practical Issues and Suggestions for Deployment

The deep learning model variants might pose threats on the model security and data privacy. Because deep learning models are very complicated, it is difficult to think of a new model structure. Hence, once a model structure is deployed, a large number of users add some variants for their uses and train further with their own data. In the case of U-Net [Ronneberger et al. 2015], which is a CNN model used for image segmentation in the biomedical field, there are several variants [Çiçek et al. 2016; Li et al. 2017; Jo et al. 2018] proposed. If such similar models are deployed in public, it is likely to be susceptible for the attacks reviewed in this paper. They might give clues in building substitute models in black-box attack scenarios, or induce easier inversion attacks based on the accumulated knowledge from the similar models. Hence, we must be careful when deploying models, especially when there is a large number of variants.

Practical considerations on the processing time and throughput are needed as well. Although FHE combined with deep learning predictions showed remarkable performances both in the privacy and utility, it lacks the considerations of practical implementations. Because predictions on FHE data and models are still too slow, parallel or distributed processing using GPUs or clusters is crucial. In particular, since GPUs have already achieved high computational speeds in deep learning training, combining GPU's high computing power with FHE model prediction is promising. Considering

those situations in which we need FHE on predictions when the computing resources of the user devices are insufficient, on-device encryption and decryption should be considered as well.

6. SUMMARY

Deep learning has become one of the inseparable technologies in our daily lives, and the problem of security and privacy of deep learning has become an issue that can no longer be overlooked. Therefore, we defined Secure AI and Private AI, and we reviewed the related attack and defense methods.

In Secure AI, we surveyed the two types of attacks: evasion attack and poisoning attack. We categorized the attack scenarios as white-box and black-box attacks, according to the amount of information and the authority of the model that the adversary possesses. In this process, we confirmed that many research studies have been conducted with advanced and varied attack methods. On the other hand, the studies on the defense techniques are in relatively early stages. In this paper, we introduce the related studies by classifying them as gradient masking, adversarial training and statistical approaches.

Furthermore, the risk of data privacy violations is always widespread due to the characteristics of deep learning, which highly relies on an extensive amount of data, and the era of the fourth industrial revolution, in which data itself is the enormous asset. In this paper, we describe the possible threats on the data privacy from the perspectives of deep learning models and service providers, information silos and deep learning-based service users. In addition, we name the deep learning-based approaches that are concerned with data privacy as Private AI. Unlike Secure AI, there are not many studies on privacy attacks using deep learning. Hence, we introduce recent studies on three defending techniques concerned with Private AI: homomorphic encryption, differential privacy, and secure multi-party training. Finally, open problems and directions for future work are discussed.

REFERENCES

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, and others. 2016a. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016b. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 308–318.

Gergely Acs, Luca Melis, Claude Castelluccia, and Emiliano De Cristofaro. 2018. Differentially private mixture of generative neural networks. *IEEE Transactions on Knowledge and Data Engineering* (2018).

Scott Alfeld, Xiaojin Zhu, and Paul Barford. 2016. Data Poisoning Attacks against Autoregressive Models.. In *AAAI*. 1452–1458.

Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, and others. 2018. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security* 13, 5 (2018), 1333–1345.

Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. 2015. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks* 10, 3 (2015), 137–150.

Anish Athalye, Nicholas Carlini, and David Wagner. 2018. Obfuscated gradients give

a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420* (2018).

Anish Athalye and Ilya Sutskever. 2017. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397* (2017).

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015a. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015), e0130140.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015b. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015), e0130140.

Shumeet Baluja and Ian Fischer. 2017. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv preprint arXiv:1703.09387* (2017).

Vahid Behzadan and Arslan Munir. 2017. Vulnerability of deep reinforcement learning to policy induction attacks. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*. Springer, 262–275.

Josh Benaloh. 1994. Dense probabilistic encryption. In *Proceedings of the workshop on selected areas of cryptography*. 120–128.

Yoshua Bengio and others. 2009. Learning deep architectures for AI. *Foundations and trends® in Machine Learning* 2, 1 (2009), 1–127.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 2013. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 387–402.

Battista Biggio, Blaine Nelson, and Pavel Laskov. 2012. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389* (2012).

Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. 2005. Practical privacy: the SuLQ framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 128–138.

Joppe W Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. 2013. Improved security for a ring-based fully homomorphic encryption scheme. In *IMA International Conference on Cryptography and Coding*. Springer, 45–64.

Florian Bourse, Michele Minelli, Matthias Minihold, and Pascal Paillier. 2017. *Fast Homomorphic Evaluation of Deep Discretized Neural Networks*. Ph.D. Dissertation. IACR Cryptology ePrint Archive.

Zvika Brakerski and Vinod Vaikuntanathan. 2014. Efficient fully homomorphic encryption from (standard) LWE. *SIAM J. Comput.* 43, 2 (2014), 831–871.

Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. 2018. Thermometer encoding: One hot way to resist adversarial examples. In *Submissions to International Conference on Learning Representations*.

Anna L Buczak and Erhan Guven. 2016. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials* 18, 2 (2016), 1153–1176.

Mark Bun and Thomas Steinke. 2016. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*. Springer, 635–658.

Nicholas Carlini and David Wagner. 2017a. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 3–14.

Nicholas Carlini and David Wagner. 2017b. Towards evaluating the robustness of

neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 39–57.

Nicholas Carlini and David Wagner. 2018. Audio Adversarial Examples: Targeted Attacks on Speech-to-Text. *arXiv preprint arXiv:1801.01944* (2018).

Hervé Chabanne, Amaury de Wargny, Jonathan Milgram, Constance Morel, and Emmanuel Prouff. 2017. Privacy-preserving classification on deep neural network. *IACR Cryptology ePrint Archive* 2017 (2017), 35.

Kamalika Chaudhuri and Claire Monteleoni. 2009. Privacy-preserving logistic regression. In *Advances in Neural Information Processing Systems*. 289–296.

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526* (2017).

Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachene. 2016. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 3–33.

Radha Chitta, Rong Jin, and Anil K Jain. 2012. Efficient kernel clustering using random fourier features. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 161–170.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 2016. 3D U-Net: learning dense volumetric segmentation from sparse annotation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 424–432.

Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2016. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830* (2016).

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, and others. 2012. Large scale distributed deep networks. In *Advances in neural information processing systems*. 1223–1231.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 248–255.

Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. 2018. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442* (2018).

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Xiaolin Hu, Jianguo Li, and Jun Zhu. 2017. Boosting adversarial attacks with momentum. arXiv preprint. *arXiv preprint arXiv:1710.06081* (2017).

Léo Ducas and Daniele Micciancio. 2015. FHEW: bootstrapping homomorphic encryption in less than a second. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 617–640.

Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*. Springer, 1–19.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 486–503.

Cynthia Dwork and Jing Lei. 2009. Differential privacy and robust statistics. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*. ACM, 371–380.

Cynthia Dwork, Aaron Roth, and others. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.

Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. 2010. Boosting and differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 51–60.

Taher ElGamal. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory* 31, 4 (1985), 469–472.

Gamaleldin F Elsayed, Shreya Shankar, Brian Cheung, Nicolas Papernot, Alex Kurakin, Ian Goodfellow, and Jascha Sohl-Dickstein. 2018. Adversarial examples that fool both human and computer vision. *arXiv preprint arXiv:1802.08195* (2018).

Beyza Ermis and Ali Taylan Cemgil. 2017. Differentially Private Variational Dropout. *arXiv preprint arXiv:1712.02629* (2017).

Samuel G Finlayson, Isaac S Kohane, and Andrew L Beam. 2018. Adversarial Attacks Against Medical Deep Learning Systems. *arXiv preprint arXiv:1804.05296* (2018).

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1322–1333.

Craig Gentry and Dan Boneh. 2009. *A fully homomorphic encryption scheme*. Vol. 20. Stanford University Stanford.

Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning*. 201–210.

Shafi Goldwasser and Silvio Micali. 1982. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*. ACM, 365–377.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*. Vol. 1. MIT press Cambridge.

Ian Goodfellow, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Multi-prediction deep Boltzmann machines. In *Advances in Neural Information Processing Systems*. 548–556.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014a. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014b. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

Abigail Graese, Andras Rozsa, and Terrance E Boult. 2016. Assessing threat of adversarial examples on deep neural networks. In *Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on*. IEEE, 69–74.

Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. 2016. Adversarial perturbations against deep neural networks for malware classification. *arXiv preprint arXiv:1606.04435* (2016).

Zhaoyuan Gu, Zhenzhong Jia, and Howie Choset. 2018. Adversary A3C for Robust Reinforcement Learning. (2018).

Chuan Guo, Mayank Rana, Moustapha Cissé, and Laurens van der Maaten. 2017. Countering Adversarial Images using Input Transformations. *arXiv preprint arXiv:1711.00117* (2017).

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. 2017. Adversarial example defenses: Ensembles of weak defenses are not strong. *arXiv preprint arXiv:1706.04701* (2017).

Ehsan Hesamifard, Hassan Takabi, and Mehdi Ghasemi. 2017. CryptoDL: Deep Neural Networks over Encrypted Data. *arXiv preprint arXiv:1711.05189* (2017).

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. 2017. Deep models under the GAN: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 603–618.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. 2017a. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Vol. 1. 3.

Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. 2017b. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284* (2017).

Andrew Ilyas, Ajil Jalal, Eirini Asteri, Constantinos Daskalakis, and Alexandros G Dimakis. 2017. The Robust Manifold Defense: Adversarial Training using Generative Models. *arXiv preprint arXiv:1712.09196* (2017).

YoungJu Jo, Hyungjoo Cho, Sang Yun Lee, Gunho Choi, Geon Kim, Hyun-seok Min, and YongKeun Park. 2018. Quantitative Phase Imaging and Artificial Intelligence: A Review. *arXiv preprint arXiv:1806.03982* (2018).

Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. 2018. Gazelle: A low latency framework for secure neural network inference. *arXiv preprint arXiv:1801.05507* (2018).

Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2013. The composition theorem for differential privacy. *arXiv preprint arXiv:1311.0776* (2013).

D Kinga and J Ba Adam. 2015. A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, Vol. 5.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730* (2017).

J Zico Kolter and Eric Wong. 2017. Provable defenses against adversarial examples via the convex outer adversarial polytope. *arXiv preprint arXiv:1711.00851* (2017).

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016a. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533* (2016).

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016b. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236* (2016).

Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist* 2

(2010).
Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. 2009. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 609–616.
Seil Lee, Hanjoo Kim, Jaehong Park, Jaehee Jang, Chang-Sung Jeong, and Sungroh Yoon. 2018. TensorLightning: A Traffic-efficient Distributed Deep Learning on Commodity Spark Clusters. *IEEE Access* (2018).
Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. 2014. Scaling Distributed Machine Learning with the Parameter Server.. In *OSDI*, Vol. 14. 583–598.
Xiaomeng Li, Hao Chen, Xiaojuan Qi, Qi Dou, Chi-Wing Fu, and Pheng Ann Heng. 2017. H-DenseUNet: Hybrid densely connected UNet for liver and liver tumor segmentation from CT volumes. *arXiv preprint arXiv:1709.07330* (2017).
Yehida Lindell. 2005. Secure multiparty computation for privacy preserving data mining. In *Encyclopedia of Data Warehousing and Mining*. IGI Global, 1005–1009.
Jian Liu, Mika Juuti, Yao Lu, and N Asokan. 2017. Oblivious neural network predictions via minionn transformations. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 619–631.
Yunhui Long, Vincent Bindschaedler, and Carl A Gunter. 2017. Towards Measuring Membership Privacy. *arXiv preprint arXiv:1712.09136* (2017).
Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Michael E Houle, Grant Schoenebeck, Dawn Song, and James Bailey. 2018. Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality. *arXiv preprint arXiv:1801.02613* (2018).
Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).
Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. 2009. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430* (2009).
H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning Differentially Private Recurrent Language Models. (2018).
Shike Mei and Xiaojin Zhu. 2015. Using Machine Teaching to Identify Optimal Training-Set Attacks on Machine Learners.. In *AAAI*. 2871–2877.
Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. 2017. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267* (2017).
Payman Mohassel and Yupeng Zhang. 2017. SecureML: A system for scalable privacy-preserving machine learning. In *2017 38th IEEE Symposium on Security and Privacy (SP)*. IEEE, 19–38.
Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. 2017. Universal adversarial perturbations. *arXiv preprint* (2017).
Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. 2017. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 27–38.
Taesik Na, Jong Hwan Ko, and Saibal Mukhopadhyay. 2017. Cascade Adversarial Machine Learning Regularized with a Unified Embedding. *arXiv preprint arXiv:1708.02582* (2017).
Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. 2016. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759* (2016).
Pascal Paillier. 1999. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic*

Techniques. Springer, 223–238.
Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2016a. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755* (2016).
Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016b. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277* (2016).
Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 506–519.
Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016a. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*. IEEE, 372–387.
Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016b. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 582–597.
Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. 2018. Scalable Private Learning with PATE. *arXiv preprint arXiv:1802.08908* (2018).
Kaustubh R Patil, Xiaojin Zhu, Łukasz Kopeć, and Bradley C Love. 2014. Optimal teaching for limited-capacity human learners. In *Advances in neural information processing systems*. 2465–2473.
Andrea Paudice, Luis Muñoz-González, Andras Gyorgy, and Emil C Lupu. 2018b. Detection of Adversarial Training Examples in Poisoning Attacks through Anomaly Detection. *arXiv preprint arXiv:1802.03041* (2018).
Andrea Paudice, Luis Muñoz-González, and Emil C Lupu. 2018a. Label Sanitization against Label Flipping Poisoning Attacks. *arXiv preprint arXiv:1803.00992* (2018).
NhatHai Phan, Yue Wang, Xintao Wu, and Dejing Dou. 2016. Differential Privacy Preservation for Deep Auto-Encoders: an Application of Human Behavior Prediction.. In *AAAI*, Vol. 16. 1309–1316.
NhatHai Phan, Xintao Wu, and Dejing Dou. 2017a. Preserving differential privacy in convolutional deep belief networks. *Machine Learning* 106, 9-10 (2017), 1681–1704.
NhatHai Phan, Xintao Wu, Han Hu, and Dejing Dou. 2017b. Adaptive laplace mechanism: differential privacy preservation in deep learning. In *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 385–394.
Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*. 91–99.
Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.
Bita Darvish Rouhani, M Sadegh Riazi, and Farinaz Koushanfar. 2018. Deepsecure: Scalable provably-secure deep learning. In *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*. IEEE, 1–6.
Leonid I Rudin, Stanley Osher, and Emad Fatemi. 1992. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena* 60, 1-4 (1992), 259–268.
Pouya Samangouei, Maya Kabkab, and Rama Chellappa. 2018. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint*

arXiv:1805.06605 (2018).

Amartya Sanyal, Matt J Kusner, Adrià Gascón, and Varun Kanade. 2018. TAPAS: Tricks to Accelerate (encrypted) Prediction As a Service. *arXiv preprint arXiv:1806.03461* (2018).

Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. 2018. Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks. *arXiv preprint arXiv:1804.00792* (2018).

Yash Sharma and Pin-Yu Chen. 2017. Attacking the Madry Defense Model with L_1-based Adversarial Examples. *arXiv preprint arXiv:1710.10733* (2017).

Benjamin Shickel, Patrick Tighe, Azra Bihorac, and Parisa Rashidi. 2017. Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis. *arXiv preprint arXiv:1706.03446* (2017).

Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 1310–1321.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 3–18.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. *arXiv preprint arXiv:1704.02685* (2017).

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).

Aman Sinha, Hongseok Namkoong, and John Duchi. 2017. Certifiable Distributional Robustness with Principled Adversarial Training. *arXiv preprint arXiv:1710.10571* (2017).

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. 2017. PixelDefend: Leveraging Generative Models to Understand and Defend against Adversarial Examples. *arXiv preprint arXiv:1710.10766* (2017).

Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. 2017. Certified defenses for data poisoning attacks. In *Advances in Neural Information Processing Systems*. 3517–3529.

Sining Sun, Ching-Feng Yeh, Mari Ostendorf, Mei-Yuh Hwang, and Lei Xie. 2018. Training Augmentation with Adversarial Examples for Robust Speech Recognition. *arXiv preprint arXiv:1806.02782* (2018).

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204* (2017).

Aleksei Triastcyn and Boi Faltings. 2018. Generating Differentially Private Datasets Using GANs. *arXiv preprint arXiv:1803.03148* (2018).

Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. 2010. Fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 24–43.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. 2017. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991* (2017).

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. 2018. Differentially Private Generative Adversarial Network. *arXiv preprint arXiv:1802.06739* (2018).

Weilin Xu, David Evans, and Yanjun Qi. 2017. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155* (2017).

Masahiro Yagisawa. 2015. Fully Homomorphic Encryption without bootstrapping. *IACR Cryptology ePrint Archive* 2015 (2015), 474.

Chen Yan, X Wenyuan, and Jianhao Liu. 2016. Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle. *DEF CON* (2016).

Chaofei Yang, Qing Wu, Hai Li, and Yiran Chen. 2017. Generative poisoning attack method against neural networks. *arXiv preprint arXiv:1703.01340* (2017).

Andrew Chi-Chih Yao. 1986. How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*. IEEE, 162–167.

Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. 2019. Differentially Private Model Publishing for Deep Learning. In *Differentially Private Model Publishing for Deep Learning*. IEEE, 0.

Valentina Zantedeschi, Maria-Irina Nicolae, and Ambrish Rawat. 2017. Efficient defenses against adversarial attacks. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 39–49.

Chao Zhang, Lei Zhang, and Jieping Ye. 2012. Generalization bounds for domain adaptation. In *Advances in neural information processing systems*. 3320–3328.