The Dark Clouds in Adversarial Training

Anonymous Author(s)
Affiliation
Address
email

Abstract

Graph Convolutional Networks (GCNs) have attracted more and more attentions in recent years. A typical GCN layer consists of a linear feature propagation step and a nonlinear transformation step. Recent works show that a linear GCN can achieve comparable performance to the original non-linear GCN while being much more computationally efficient. In this paper, we dissect the feature propagation steps of linear GCNs from a perspective of continuous graph diffusion, and analyze why linear GCNs fail to benefit from more propagation steps. Following that, we propose Decoupled Graph Convolution (DGC) that decouples the terminal time and the feature propagation steps, making it more flexible and capable of exploiting a very large number of feature propagation steps. Experiments demonstrate that our proposed DGC improves linear GCNs by a large margin and makes them competitive with many modern variants of non-linear GCNs.

1 Introduction

In recent years deep learning has witnessed a rapid development, but the existence of adversarial examples [13] alerts us that there are still some black clouds in modern neural networks. Crafted by adding imperceptible perturbations to the input images, adversarial examples can dramatically degrade the performance of accurate deep models. This venerability has become a major security issue of neural networks and raise a huge concern in both the academy and the industry [3].

Though many methods failed to address this issue, researchers do find that Adversarial Training (AT) [6, 10] still remains an effective approach [1]. In practice, adversarially trained models have shown good robustness under various attack [15], and the recent state-of-the-art defense algorithms are all variants of adversarial training [5]. Therefore, it is widely believed that we have already found the cure to adversarial attack, *i.e.,* adversarial training, based on which we can build trustworthy models. However, in this work, we challenge this common belief by revealing that a certain type of adversarial data could lead to unexpected failure of the whole adversarial training progress, which suggests that adversarial training is still exposed to the risk of losing accuracy and robustness all together.

Adversarial training is typically formulated as an adversarial minimax game, where the inner loop tries to find the worst-case adversarial example maximizing the loss, while the outer loop drives model parameters to minimize the loss. According to the Danskin's Theorem [2], we need to reach the exact maximum in the inner loop in order to estimate the outer-loop gradient. Instead, the common practice adopts an attack algorithm (*e.g.,* PGD [10]) that terminates after a few fixed steps, *e.g.,* 10, which could lead to biased gradient estimate for the outer loop. Nevertheless, recent works tend to believe that adversarial training is still quite reliable even if the inner loop is not solved exactly (or up to a high precision) [10]. For example, DAT [17] observes that using a weaker (*i.e.,* more inexact) inner-loop adversary at the beginning of training can yield even better robustness, and FAT [20] shows that an early-stopped adversary can improve the accuracy without sacrificing robustness.

In this work, we challenge the common belief above by designing a special kind of adversarial examples generated by **Manually-Assigned Deceiving Attack (MADA**, meaning problems in Chinese), which can also be seen as inexact solutions to the inner loop, as they are also generated by small

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

perturbations that increase the loss. However, unlike previous **benign adversarial examples** that assist the training and help improve model robustness, applying MADA examples to the outer loop will crush the training progress and lead to models no better (or even worse) than random guess.

Up to our knowledge, we are the first to spot this kind of examples and we name them as **fatal adversarial examples**. Previously, adversarial training is widely believed to be a robust and reliable training diagram, while the existence of fatal adversarial examples alerts us to the risks of solving the inner loop inexactly. Researcher used to believe that AT has expelled the dark clouds of adversarial examples, while fatal adversarial examples now become the dark clouds in AT itself.

In view of the analysis above, MADA examples can also be regarded as a novel kind of the recently proposed "unlearnable examples" [8], which add small perturbations to personal data to prevent them from being freely exploited by deep learning models. Previous to our work, the error-minimizing examples adopted by [8] can also lead to training collapse by hiding data with nearly zero loss. However, these samples are easily detected by inspecting their loss values, and also easily alleviated by adversarial training that pushes up the loss again. In comparison, our MADA examples enjoy better "unlearnability" as 1) they lead to lower accuracy on protected data; 2) they enjoy better transferability across different models; 3) they also increase the loss values, which makes them harder to be detected; and 4) they cannot be fully alleviated by adversarial training.

We summarize our main contributions as follows:

- We spot the existence of fatal adversarial examples with our proposed MADA algorithm, which resemble benign adversarial examples but will break down the training progress. As a result, it poses a serious challenge to the existing adversarial training methods.
- By comparing MADA to canonical targeted attack, we notice that the success of MADA can be attributed to a consistent bias added to the adversarial examples. Based on this, we propose different variants of MADA, including both deterministic and stochastic ones, and evaluate several detection and defense strategies to eliminate them.
- Our MADA examples can serve as a more advanced kind of unlearnable examples for preserving personal data. It outperforms previous error-min examples with better attack rate, better transferability, and harder to be detected or alleviated.

2 Related Work

Convergence of AT. Madry *et al.* [10] viewed adversarial defense as solving a minimax problem and proposed Projected Gradient Descent (PGD) for inner maximization. However, PGD cannot guarantee to find its global maximum. Wang *et al.* [17] proposed a First-Order Stationary Condition (FOSC) to characterize the local convergence of inner maximization, based on which they empirically observe that high convergence quality adversarial examples are not necessary and can even be harmful in in the early training stages. Zhang *et al.* [20] further showed that friendly training adversary that early-stops after misclassification could obtain comparable robustness and improve the model accuracy. These previous methods all advocate the idea that adversarial training can work well (or even better) without strong convergence of the inner maximization, while we challenge this belief by discovering the existence of fatal adversarial examples.

Targeted Attack (TA) for AT. Targeted attack generates adversarial examples such that they are misclassified to the target class (different to original label). While iterative untargeted attack (*e.g.,* PGD [10]) is more popular in solving the inner loop of adversarial training, some recent works found that targeted attack can achieve comparable, and sometimes better, performance. These methods mainly differ by their assignment of the target class, *e.g.,* random class [19], Least Likely (LL) class [9], or Most Confusing (MC) class [16]. And we want to highlight that adversarial training with these previous targeted attack methods all yields good performance, in other words, they are all *benign adversarial examples*. On the contrary, our proposed Manually-Assigned Deceiving Attack can craft fatal examples that totally breaks down the training progress.

Certified Robustness. Although adversarial training is the state-of-the-art defense in practice, its robustness is not theoretically guaranteed [14]. There is an another line of works that train provably robust models by maximizing the certified radius provided by robust certification methods [12, 18, 11, 4]. In our work, the existence of fatal adversarial examples also highlight the necessity of provably robust models as it shows adversarial training can be risky without guarantees.

Unlearnable Examples. Recently, Huang *et al.* [8] found that, if we invert the inner maximization and instead optimize examples to minimize the loss, *i.e.,* adding the so-called *error-min noise*, we

can make the resulting examples un-exploitable by the neural network training (the outer loop). Thus they name these samples as *unlearnable examples* and apply them to protect personal data from being exploited by deep models. Unlike error-min noises, our MADA noises are crafted by targeted attack that will likely increase the original loss, just like classical (benign) adversarial examples. Nevertheless, MADA examples will also lead to training collapse, and thus it also belongs to unlearnable examples. We conduct a comprehensive comparison of the two methods in Section **??**.

3 The Fall of Adversarial Training

In this section, we reveal the risk of adversarial training by designing a kind of fatal adversarial examples that could lead to training collapse.

3.1 Preliminaries

3.1.1 Adversarial Training

Given a C-class dataset $\mathcal{S} = \left\{\left(\mathbf{x}_i^0, y_i\right)\right\}_{i=1}^n$ with $\mathbf{x}_i^0 \in \mathbb{R}^d$ as a clean data example in the d-dimensional input space $\mathcal{X}$ and $y_i \in \mathcal{Y} = \{0, \cdots, C-1\}$ as its associated label, the objective of adversarial training is to solve the following minimax optimization problem:

$$\min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} \max_{\left\|\mathbf{x}_i - \mathbf{x}_i^0\right\|_p \leq \varepsilon} \ell\left(h_{\boldsymbol{\theta}}\left(\mathbf{x}_i\right), y_i\right), \tag{1}$$

where $h_{\boldsymbol{\theta}} : \mathbb{R}^d \to \mathbb{R}^C$ is the DNN classifier, $\mathbf{x}_i$ is the adversarial example of $\mathbf{x}_i^0$, $\ell\left(h_{\boldsymbol{\theta}}\left(\mathbf{x}_i\right), y_i\right)$ is the loss function on the adversarial pair $(\mathbf{x}_i, y_i)$, and ε is the maximum perturbation constraint *w.r.t.* ℓ_p norm. In order to solve the minimax problem above, we need to compute the gradient of $\boldsymbol{\theta}$ *w.r.t.* the max loss $\phi_{\boldsymbol{\theta}}(\mathbf{x}_i^0, y)$ for each data pair $(\mathbf{x}_i^0, y)$, *i.e.,*

$$\nabla_{\boldsymbol{\theta}} \phi_{\boldsymbol{\theta}}(\mathbf{x}_i^0, y) := \nabla_{\boldsymbol{\theta}} \max_{\left\|\mathbf{x}_i - \mathbf{x}_i^0\right\|_p \leq \varepsilon} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y), \tag{2}$$

while the max loss $\phi_{\boldsymbol{\theta}}$ typically does not have a closed form. Luckily, the Danskin's Theorem [2] offers an approach for estimating this gradient. In fact, under mild conditions (see Appendix **??**), the max loss is differentiable at $\boldsymbol{\theta}$, and its gradient is equivalent to the gradient of ℓ at the maximizer $\mathbf{x}_i^*$,

$$\nabla_{\boldsymbol{\theta}} \phi_{\boldsymbol{\theta}}(\mathbf{x}_i^0, y) = \nabla_{\boldsymbol{\theta}} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i^*), y), \text{ where } \mathbf{x}_i^* = \operatorname*{argmax}_{\left\|\mathbf{x}_i - \mathbf{x}_i^0\right\|_p \leq \varepsilon} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y). \tag{3}$$

As a result, we can solve the minimax problem (Eq. (1)) by alternating between the inner loop and the outer loop: 1) we first solve the inner maximization problem and find the optimal adversarial example $\mathbf{x}_i^*$ for each $\mathbf{x}_i^0$; and then 2) we update $\boldsymbol{\theta}$ *w.r.t.* the outer minimization loss of these adversarial examples $\{\mathbf{x}_i^*\}_{i=1}^n$. Therefore, how well the inner maximization problem is solved directly affects the performance of the outer minimization, *i.e.,* the robustness of the classifier.

3.1.2 Practical (Benign) Training Adversaries

From the above analysis, in order to obtain unbiased estimate of the gradient of $\boldsymbol{\theta}$, we need to solve the inner loop exactly and find the global maximizer $\mathbf{x}_i^*$ for each sample $\mathbf{x}_i^0$. However, in practice, the DNN classifier $h_{\boldsymbol{\theta}}$ is typically highly non-concave, making it hard to reach the global maximum. In practice, adversarial training has to settle down with the inexact first-order training adversaries [14].

FGSM. FGSM is the one-step gradient ascent for the linearized loss at $\mathbf{x}_i^0$ for ℓ_∞ attack [6]

$$\mathbf{x}_i = \mathbf{x}_i^0 + \varepsilon \cdot \operatorname{sign}(\nabla_{\mathbf{x}_i} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i^0, y))). \tag{4}$$

Projected Gradient Descent (PGD). PGD is a multi-step adversary that perturbs a clean example $\mathbf{x}_i^0$ for a number of steps K with smaller step size α. After each step of perturbation, PGD projects the adversarial example back onto the ε-ball of $\mathbf{x}_i^0$, if it goes beyond the ε-ball [10]:

$$\mathbf{x}_i^k = \Pi\left(\mathbf{x}_i^{k-1} + \alpha \cdot \operatorname{normalize}\left(\nabla_{\mathbf{x}_i} \ell\left(h_{\boldsymbol{\theta}}\left(\mathbf{x}_i^{k-1}\right), y\right)\right)\right) \tag{5}$$

where $\Pi(\cdot)$ is the projection function, $\operatorname{normalize}$ re-scales the gradient to the unit ball under ℓ_p norm, and $\mathbf{x}_i^k$ is the adversarial example at the k-th step.

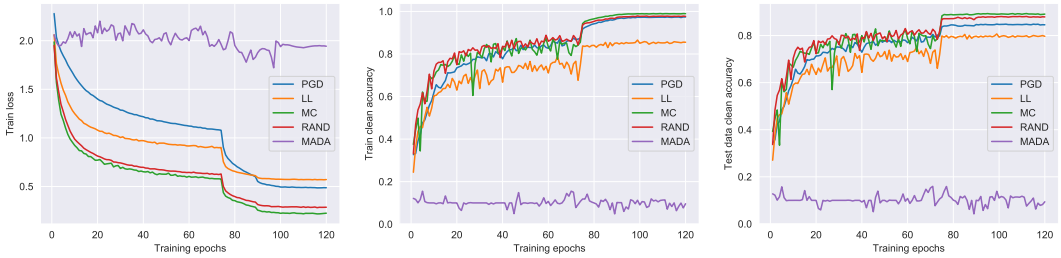

Figure 1: Adversarial training with different inner-loop solvers (training adversaries) under the same perturbation budge ($\varepsilon = 8/255$ under ℓ_∞ norm) with ResNet-18 backbone on CIFAR-10.

Targeted Attack. Different from PGD that directly maximizes the loss of $\mathbf{x}_i$ with the original label y (known as untargeted attack), targeted attack first selects a (wrong) target class $y' \neq y$, and push $\mathbf{x}_i$ to the target class by minimizes the loss $\ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i, y'))$. Previous works have considered various strategies for selecting the target class. [19] chose y' uniformly at random (RAND); [9] adopted the Least Likely (LL) class, $y' = \operatorname{argmax}_{y' \neq y} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i^0, y))$; while [16] used the Most Confusing (MC) class, $y' = \operatorname{argmin}_{y' \neq y} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i^0, y))$.

3.2 The Training Adversary That Kills Training

As the exact global maximum is hard to reach, we have to rely on practical inexact training adversaries to solve the inner maximization. Previous works suggest that although a too large perturbation constraint ε could lead to training collapse [20], within a moderate ε (*e.g.,* $\varepsilon \leq 8/255$ under ℓ_∞ norm), a weak training adversary as introduced above could also solve the minimax problem well. Does this really mean that inexact inner maximization already suffices? Our work provides a negative answer to this question by showing that certain training adversary can inevitably kill the training.

Manually Assigned Deceiving Attack (MADA). We design a special kind of targeted attack that assigns target class with a predefined assignment $\mathcal{T} : \mathcal{Y} \to \mathcal{Y}$ that maps from the original class y to the target class y'. Notably, for each data pair $(\mathbf{x}_i^0, y_i)$, the assignment is only dependent on the label y_i, and independent of the input $\mathbf{x}_i^0$ or the classifier $h_{\boldsymbol{\theta}}$. For simplicity, we first consider the following plus-one assignment $\mathcal{T}_+$ that shifts each class to the next one, *i.e.,*

$$y_i' = \mathcal{T}_+(y_i) = y_i + 1 \pmod{C} = \begin{cases} y_i + 1, & y_i = 0, \dots, C-2; \\ 0, & y_i = C-1. \end{cases} \tag{6}$$

Then, we generate adversarial examples by minimizing the loss to the target class, *i.e.,*

$$\mathbf{x}_i^* = \operatorname*{argmin}_{\|\mathbf{x}_i - \mathbf{x}_i^0\|_p \leq \varepsilon} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i'), \text{ where } y_i' = \mathcal{T}_+(y_i), \tag{7}$$

which we name as *plus-one MADA*. Specifically, we can solve the minimization with first-order optimization methods analogous to FGSM (Eq. (4)) and PGD (Eq. (5)).

Comparing Training Adversaries. In Figure **??**, we compare the training progress of adversarial training with different training adversaries, including PGD [10], RAND [19], LL [9], MC [16], and our MADA. For a fair comparison, all training adversaries have the same budgets and only differ by their loss functions. Our experiments are conducted on CIFAR-10 with ResNet-18 [7], ℓ_∞-norm attack and perturbation limits $\varepsilon = 8/255$. Detailed experimental setup is included in Section **??**.

Benign Training Adversary. From Figure 1, we can see that AT with canonical untargeted attack (PGD) and targeted attack (RAND, LL & MC) all goes on smoothly and ends up with models of good (natural and robust) accuracy. It is widely believed that learning with adversarially augmented samples will help improve model robustness [6, 10]. Now we know that at least it is true for these *benign training adversaries*. Although designed to degrade the model accuracy, a benign adversary will not interrupt the training progress, but help prepare the models for possible attack instead.

Fatal Training Adversary. However, we show that the common belief on the stability and reliability of AT could be wrong if we adopt *fatal training adversaries* like MADA. From Figure **??**, we notice that MADA will totally kill the training and lead to classifiers with around 10% accuracy, which is basically random guess. Notably, MADA adopts the same perturbation budget and optimizer as other targeted attack methods and the only difference lies the choice of the assignment $\mathcal{T}$. In Figure **??**, we

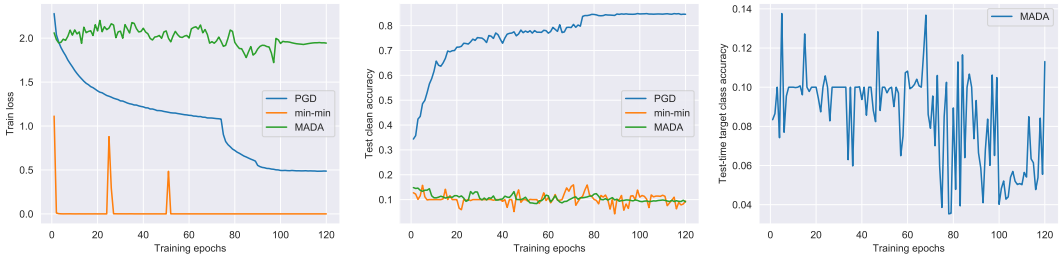

Figure 2: Comparing two fatal adversaries, EMA and MADA, for adversarial training with ResNet-18 backbone on CIFAR-10.

further compare the adversarial loss of four targeted attack methods. We can see that they all increase the loss with similar values, making them hardly distinguishable. However, like an undercover spy, MADA will unexpectedly kill the training process while others don't.

3.3 The Risk in Adversarial Training

Different from the minimax formulation (Eq. (1)) for exact adversarial training, we formalize the practical adversarial learning algorithms using a training adversary $\mathcal{A}$ as follows:

$$\underbrace{\min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} \ell\left(h_{\boldsymbol{\theta}}\left(\mathbf{x}_i\right), y_i\right),}_{\text{standard training}} \text{ where } \underbrace{\mathbf{x}_i = \mathcal{A}(\mathbf{x}_i^0, y_i; h_{\boldsymbol{\theta}}), \text{ s.t. } \left\|\mathbf{x}_i - \mathbf{x}_i^0\right\|_p \leq \varepsilon,}_{\text{adversarial data augmentation}} \tag{8}$$

which consists of two phases: 1) the training adversary $\mathcal{A}$ generates an adversarially augmented example $\mathbf{x}_i$ from the data pair $(\mathbf{x}_i^0, y_i)$ with imperceptible noise; and 2) the model takes the augmented pair $(\mathbf{x}_i, y_i)$ and performs standard training.

The Risk of Adversary Attack. Previous practice shows that we can learn a good model from samples generated by any adversary, no matter weak or strong. However, the existence of fatal training adversaries like MADA highlights that (inexact) adversarial training could be unreliable. Consider the case if a hacker could break in a learning system and replace canonical training adversaries with MADA maliciously, for example, by editing or redirecting the adversary. Then, even if the learner checks the perturbation constraint, or examine the loss and adversarial loss (under protection), he or she will find everything seems OK. However, once the learner starts training, he or she will inevitably run into a crushed model. It highlights that adversarial training still has some unsolved security issues in itself and there are risks if we perform inexact inner maximization.

Comparison to EMA. Instead of common inner maximization, the recently proposed *Error-Minimizing Adversary (EMA)* [8] generates adversarial examples by minimizing the loss, *i.e.,*

$$\mathbf{x}_i = \mathcal{A}_{\text{EMA}}(\mathbf{x}_i^0, y_i; h_{\boldsymbol{\theta}}) = \underset{\left\|\mathbf{x}_i - \mathbf{x}_i^0\right\|_p \leq \varepsilon}{\operatorname{argmin}} \ell(h_{\boldsymbol{\theta}}(\mathbf{x}), y). \tag{9}$$

From Figure 2, we can see that like MADA, EMA will also lead to training collapse, thus it can also be regarded as a fatal training adversary. In fact, EMA can be seen a special case of MADA if it takes the identity assignment $\mathcal{T}(y) = y$. However, the poisoning mechanisms of EMA and (plus-one) MADA (Eq. (7)) actually differ a lot. As shown in Figure **??** (x), EMA cheats the model by making all examples $\{(\mathbf{x}_i, y_i)\}$ having nearly zero loss. In this way, the model can hardly receive any learning signals from the data and the gradient saturates. Nevertheless, this also makes it easily detected by inspecting the loss value, as its loss is significantly lower than the loss of clean data. On the contrary, by adding a consistent bias disturbing the classification, the training loss of MADA remain as high as the beginning stage, while the loss of canonical adversaries gradually decay as training continues. As MADA is also a kind of targeted attack that increases the inner loss (Figure **??** (x)), it is harder than EMA to detect this fatal training adversary by inspecting the loss values.

4 Application to Data Protection

We have shown that certain training adversaries like MADA can corrupt the training progress if it is applied to adversary attack (Section 3.3). However, as the saying goes, technology is a double-edged

sword. Despite its danger from a trainer's perspective, from a user's perspective, it can be used to protect personal data from being freely exploited by deep models instead. In this section, we introduce the application of MADA to the scenario of data protection.

4.1 Problem Setup

5 Experiments

5.1 Benchmarking Adversary Attack for Adversarial Training

Network. ResNet-18, WideResNet-34.

CIFAR-10.

CIFAR-100.

SVHN.

5.2 Understanding How MADA Works

MADA fools the model by creating disturbing examples with a consistent bias on the labels. Anytime the accuracy becomes better, MADA will adversarially adds add class-$(y+1)$ features to class-y samples. Then, training with these samples will encourage the model to predict class-$(y+1)$ features to class y. As a result, the model is confused and its performance is degraded. This negative feedback loop described above prevents the model to learn anything better than random guess. In contrast, previous adversaries will not suffer from this problem because their adversarial examples do not have a consistent bias.

5.3 Defending Fatal Adversaries

Setup. Different datasets (Table). Comparing error-min and MADA. Detection with loss value.

5.4 Data Protection

References

[1] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, 2018.

[2] Dimitri P Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.

[3] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

[4] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. *ICML*, 2019.

[5] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. *ICML*, pages 2206–2216, 2020.

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

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

[8] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. *ICLR*, 2021.

[9] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *ICLR*, 2017.

[10] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[11] Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. *ICML*, 2018.

[12] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *ICLR*, 2018.

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

[14] Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. *ICLR*, 2018.

[15] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[16] Jianyu Wang and Haichao Zhang. Bilateral adversarial training: Towards fast training of more robust models against adversarial attacks. *ICCV*, 2019.

[17] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. *ICML*, 2019.

[18] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. *ICML*, 2018.

[19] Cihang Xie and Alan Yuille. Intriguing properties of adversarial training at scale. *ICLR*, 2020.

[20] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Lizhen Cui, Masashi Sugiyama, and Mohan Kankanhalli. Attacks which do not kill training make adversarial learning stronger. *ICML*, 2020.