WASSERTRAIN: AN ADVERSARIAL TRAINING FRAMEWORK AGAINST WASSERSTEIN ADVERSARIAL ATTACKS

Qingye Zhao[1] *Xin Chen*[1⋆] *Zhuoyu Zhao*[2] *Enyi Tang*[1] *Xuandong Li*[1]

[1] State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]Henan University of Economics and Law, China
{qingyezhao, zhuoyuzhao, eytang}@foxmail.com, {chenxin, lxd}@nju.edu.cn

ABSTRACT

This paper presents an adversarial training framework WasserTrain for improving model robustness against the adversarial attacks in terms of the Wasserstein distance. First, an effective attack method WasserAttack is introduced with a novel encoding of the optimization problem, which directly finds the worst point within the Wasserstein ball while keeping the relaxation error of the Wasserstein transformation as small as possible. The proposed adversarial training framework utilizes these high-quality adversarial examples to train robust models. Experiments on MNIST show that the adversarial loss arising from adversarial examples found by our method is about three times as much as that found by the PGD-based attack method. Furthermore, within the Wasserstein ball with a radius of 0.5, the WasserTrain model achieves 31% adversarial robustness against WasserAttack, which is 22% higher than that on the PGD-based training model.

Index Terms— Wasserstein distance, adversarial attack, adversarial training

1. INTRODUCTION

As a large amount of work expands neural networks in safety-critical systems, the robustness issue of how to secure neural networks becomes essential [1, 2, 3]. Recent researches show that neural networks are vulnerable to adversarial attacks [4, 5, 6, 7]. To address this problem, adversarial training techniques are used to make models resistant to adversarial attacks. A lot of adversarial training methods have been proposed to train robust models in various scenes [8, 9, 10, 11]. Specifically, in [8, 9], models are trained resistant to adversarial attacks whose perturbations are bounded on ℓ_∞, ℓ_1, or ℓ_2 ball, generally, on ℓ_p ball. Beyond ℓ_p ball, space perturbations, such as translation and rotation, are considered in [10, 12], and models are trained to be robust against these kinds of attacks. In [13, 14], adversarial training methods are used to train models robust against the Wasserstein adversarial examples, which are measured with original images on the Wasserstein distance. The difference between these two works is that [13] considers the Wasserstein distance on the underlying data distribution, whereas [14] considers that on the image pairs.

In this paper, we consider the adversarial attacks in terms of the Wasserstein distance on image pairs. The Wasserstein distance can be understood as the transport cost of moving pixel mass for changing one image to another, and it is quite different from ℓ_p distance. Figure 1 illustrates the difference between ℓ_∞ ball and the Wasserstein ball. It can be observed that the distortion on ℓ_∞ ball is throughout the whole image. Contrarily, on the Wasserstein ball, the distortion is mainly on the subject of the image.

This paper proposes an adversarial training framework to train models robust against the Wasserstein adversarial attacks. The graphical illustration of the proposed WasserTrain is shown in Figure 2. WasserTrain is an iterative approach. It first carries out the adversarial attack on current model and original images to generate adversarial examples. Then the current model is trained and updated on the original images and adversarial examples to improve the robustness. The above process iterates until a robust model against Wasserstein attacks is learned.

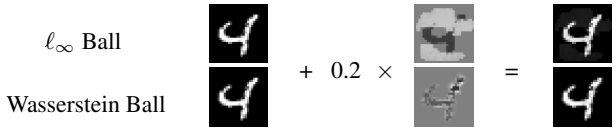

Fig. 1. An illustration of original images (left), perturbations (middle), and adversarial examples (right) within ℓ_∞ ball and the Wasserstein ball on MNIST dataset.

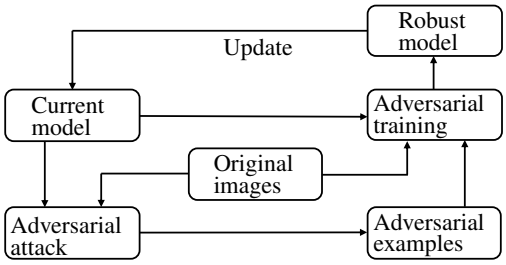

Fig. 2. Framework of WasserTrain.

The main contributions of this paper are as follows:

- We propose an effective attack method WasserAttack to produce high-quality adversarial examples.
- We construct an adversarial training framework WasserTrain to train models robust against the Wasserstein attacks based on WasserAttack.
- Experiments show WasserAttack produces adversarial examples with higher attack success rate and cross-entropy loss than the PGD-based attack method [14]. Moreover, the models trained by WasserTrain are much more robust than those trained by the PGD-based training method [14].

2. RELATED WORK

Existing work focusing on improving the model robustness falls into two categories: adversarial training and provable defenses [9, 15]. Adversarial training consists of two successive steps: generating adversarial examples and training the model to improve the robustness [8, 16]. The disadvantage of adversarial training is that it can

only make the models robust and not provide robustness guarantees. Provable defenses can provide robustness guarantees against any type of attack [17, 18]. Verification techniques including semi-definite programming relaxation [19], mixed-integer linear programming [20], Lagrangian relaxation [21] and relaxation with Lipschitz constant [22] have succeeded in producing provable certificates. Besides, Balunovic and Vechev propose a method with a combination of adversarial training and provable defenses [23]. Due to the prohibitive computational complexity, provable defenses can hardly scale up to practical networks.

The above work crafted adversarial training or provable defenses based on ℓ_p distance. In [13, 14, 24], the adversarial training is considered on the Wasserstein distance. Sinha et al. [13] present a training scheme using the worst-case perturbations to improve the network, where the Lagrangian relaxation is applied to transform the distribution distance bounded on Wasserstein distance into a distance penalty. Wong et al. [14] develop a procedure for projecting onto the Wasserstein ball, based on a modified Sinkhorn iteration, to generate adversarial examples and craft adversarial training. Wu et al. [24] introduce two faster and more accurate algorithms for Wasserstein constrained optimization.

3. OVERVIEW OF THE WASSERSTEIN ADVERSARIAL TRAINING

Most current adversarial training methods are trying to solve such a min-max problem [9, 23, 25]:

$$\min_{\theta} \mathop{\mathbb{E}}_{(x,y)\sim D} \max_{\tilde{x}\in B_\epsilon(x)} (L(\theta, \tilde{x}, y)), \tag{1}$$

where D denotes the distribution of input data pair (x, y), $B_\epsilon(x)$ denotes the robust region with radius ϵ around x, θ denotes the parameters of the neural network, and $\mathbb{E}(L(\theta, \tilde{x}, y))$ denotes the expected risk of the loss produced by the adversarial example $\tilde{x}$. Equation (1) is composed of two optimization problems: on the one hand, the inner maximization problem tries to find the most threatening adversarial example $\tilde{x}$ with the highest loss. On the other hand, the outer minimization problem tries to minimize the loss produced by $\tilde{x}$ to make the model robust. The adversarial training methods solve Equation (1) iteratively to get a robust model.

The vast majority of adversarial training methods in the previous work focus on the perturbations defined on the ℓ_p distance. However, because the Wasserstein distance is profoundly different from the ℓ_p distance, these methods can not be used to craft adversarial training against Wasserstein attacks naturally. Let $C \in \mathbb{R}_+^{n\times n}$ be a nonnegative cost matrix whose item C_{ij} denotes the cost of mass moving from x_i to $\tilde{x}_j$, the Wasserstein distance d_W between the original image x and the adversarial example $\tilde{x}$ is defined as follows:

$$\begin{cases} d_W(x, \tilde{x}) = \min\limits_{\Pi\in\mathbb{R}_+^{n\times n}} \langle \Pi, C\rangle, \\ \text{st. } \Pi 1 = x, \Pi^T 1 = \tilde{x}, \Pi_{i,j} \in [0, 1], \tilde{x}_j \in [0, 1], \end{cases} \tag{2}$$

where the minimization is over the transport plan Π, whose item Π_{ij} denotes how much the mass moves from x_i to $\tilde{x}_j$. The subjects in Equation (2) guarantee Π is a legal transport plan from x to $\tilde{x}$. That is to say, the sum of the pixel value of x is the same as that of $\tilde{x}$: $\sum_i x_i = \sum_j \tilde{x}_j$. Note that the Π_{ij} should be in $[0, 1]$ since the pixel value of a valid image is in $[0, 1]$, and the transport plan can not move the pixel mass less than 0 or more than 1. Based on the Wasserstein distance, the robust region of the Wasserstein ball around x with radius ϵ is defined as follows:

$$B_W(x, \epsilon) = \{x + \delta : d_W(x, x + \delta) \le \epsilon\}. \tag{3}$$

Then the adversarial training within the Wasserstein ball with radius ϵ is defined as the following min-max problem:

$$\begin{cases} \min\limits_{\theta} \mathop{\mathbb{E}}\limits_{(x,y)\sim D} \max\limits_{\Pi} (L(\theta, \tilde{x}, y)) \\ \text{st. } \Pi 1 = x, \Pi^T 1 = \tilde{x}, \Pi_{ij} \in [0,1], \tilde{x}_j \in [0,1], \langle \Pi, C\rangle \le \epsilon. \end{cases} \tag{4}$$

During the adversarial training, the inner maximization problem finds a transport plan Π for each x, which produces the highest loss in the Wasserstein ball with radius ϵ. The outer minimization problem minimizes the loss of $\tilde{x}$ corresponding to the transport plan Π to make the model robust.

Unfortunately, since it is intractable to find an exact transport plan Π and the corresponding adversarial example $\tilde{x}$, the Wasserstein adversarial training can not be processed directly [14, 25]. To generate Wasserstein adversarial examples, [14] develops a procedure for projecting onto the Wasserstein ball, based upon a modified version of the Sinkhorn iteration. However, due to the projection and clip operations, a significant precision error of the Wasserstein transformation appears in the process of generating adversarial examples. In other words, the equality $\sum_i x_i = \sum_j \tilde{x}_j$ is seriously unsatisfied. To alleviate this problem, we propose an effective attack method WasserAttack with a novel encoding, in which the Lagrangian relaxation and the change of variables technique are introduced to handle the constraints and eliminate the precision error.

4. WASSERTRAIN: WASSERSTEIN ADVERSARIAL TRAINING FRAMEWORK

The Wasserstein adversarial training is divided into two successive procedures: adversarial examples generation procedure and robust training procedure based on the generated adversarial examples. For the adversarial examples generation procedure, WasserAttack is presented with a relaxation encoding of the loss function. Then, the robust training procedure is processed by minimizing the mixed loss on adversarial examples and original images to improve the model robustness and natural accuracy at the same time.

4.1. Loss function encoding for generating Wasserstein adversarial examples

Since the inner maximization problem in Equation (4) is complicated to solve, the Lagrangian relaxation is used to encode the constraints as sub-loss functions. At the same time, the change of variables technique proposed in [4] is introduced to make the optimization process feasible.

Classification loss The first sub-loss l_0 is classification loss, guaranteeing the optimization process to produce the adversarial example $\tilde{x}$. Let $logit(\tilde{x})$ denotes the logit layer representation of $\tilde{x}$, y denote the original label index of x, l_0 is defined as follows:

$$l_0 = \max\{logit(\tilde{x})_{j\neq y}\} - logit(\tilde{x})_y. \tag{5}$$

Precision loss The constraints $\Pi 1 = x$ and $\Pi^T 1 = \tilde{x}$ in Equation (4) guarantee Π is a legal transport plan from x to $\tilde{x}$. For the constraint $\Pi^T 1 = \tilde{x}$, it is feasible to directly substitute $\Pi^T 1$ for $\tilde{x}$ in the loss function, since Π is an accurate representation of $\tilde{x}$. To handle the other one, the difference between $\Pi 1$ and x is encoded as sub-loss l_1 as follows:

$$l_1 = -\text{abs}(\Pi 1 - x). \tag{6}$$

Invalid pixel loss To handle the box constraints $\Pi_{i,j} \in [0, 1]$ and $\tilde{x}_j \in [0, 1]$, directly encoding is not proper, since some unexpected changes or getting stuck in extreme regions perhaps happen

due to the projection and clip operations. Instead, the change of variables technique is introduced, which can be considered as a smoothing method to eliminate the above problems. Let Φ be a free variable without the box constraints, Π is replaced with $\Pi = 0.5 * (\tanh(\Phi) + 1)$. However, the change of variables technique introduces a new problem. Note that the transformation between the transport plan Π and the adversarial example $\tilde{x}$ is $\Pi^T 1 = \tilde{x}$, so a single pixel value of $\tilde{x}$ is the sum of a column of Π, which means $\tilde{x}_j = \sum_i \Pi_{i,j}$. Due to $\Pi_{i,j} \in [0, 1]$ and the sum operation, the pixel value $\tilde{x}_j$ may be larger than 1, which is an invalid pixel. To handle the possible invalid pixels, the following sub-loss l_2 is considered:

$$l_2 = \min\{1 - \tilde{x}, 0\}. \tag{7}$$

Outside Wasserstein ball loss The constraint $\langle \Pi, C \rangle \leq \epsilon$ guarantees the optimization process to generate the adversarial example within the Wasserstein ball with radius ϵ. To satisfy this constraint, the risk of the adversarial example outside the Wasserstein ball is encoded as sub-loss l_3 as follows:

$$l_3 = \min\{\epsilon - \langle \Pi, C \rangle, 0\}. \tag{8}$$

Finally, let α, β, γ denote the Lagrange multipliers of l_1, l_2, l_3, respectively, the inner maximization problem in Equation (4) transforms to the following unconstrained problem:

$$\max_{\Phi} \ (l_0 + \alpha l_1 + \beta l_2 + \gamma l_3). \tag{9}$$

With the encoding of Equation (9), WasserAttack is presented to craft Wasserstein adversarial attacks. Furthermore, it acts as a subroutine to generate adversarial examples in the adversarial training to improve the model robustness effectively.

4.2. Robust training by minimizing mixed loss

Based on the adversarial examples generated from WasserAttack, the robust training procedure minimizes the adversarial loss, generally referred to as the cross-entropy loss, to improve the model robustness. [26] shows that there is a trade-off between accuracy and robustness, which means improving model robustness generally reduces natural accuracy. To better balance accuracy and robustness, the regular loss on original image pairs (x, y) is considered in the robust training procedure. Let $\omega \in (0, 1]$ denotes the relative weight between the adversarial loss on $(\tilde{x}, y)$ and the regular loss on (x, y) in the mixed loss, the outer minimization problem in Equation (4) transforms to the following problem:

$$\min_{\theta} \ \mathop{\mathbb{E}}_{(x,y)\sim D} (\omega L(\theta, \tilde{x}, y) + (1 - \omega) L(\theta, x, y)). \tag{10}$$

The larger the ω is, the more likely it is to consider the model robustness. When $\omega = 1$, the robust training procedure only minimizes the adversarial loss on the adversarial examples, and we can get the most robust model under this setting, whereas the natural accuracy may be affected severely.

4.3. Algorithm of WasserTrain

The adversarial training framework WasserTrain is summarized in Algorithm 1. In each iteration, the batch data (x, y) are sampled from the original image distribution. Then I-step iterative adversarial attacks are processed to generate adversarial examples $\tilde{x}$. The robust training procedure is processed on the mixed loss constructed from adversarial examples and original images. The above adversarial training process iterates M times to improve model robustness and generalization.

Algorithm 1 WasserTrain

Input: Model N with parameters θ, original input data distribution D, transport cost C, radius of Wasserstein ball ϵ, relative weight in the mixed loss ω, adversarial attacks iteration number I, adversarial training batch number M

Output: Robust model N with parameters θ

for $m = 0$ to $M - 1$ **do**
  Sample original image batch (x, y) from D
  $(\tilde{x}_0, y) = (x, y)$
  for $i = 0$ to $I - 1$ **do**
    Construct loss function L as Equation (9) from $x, \tilde{x}_i, y, C, \epsilon$, and θ
    Generate Φ by maximizing L
    $\tilde{x}_{i+1} \coloneqq (0.5 * (tanh(\Phi) + 1))^T 1$
  end for
  Construct mixed loss L as Equation (10) from $(\tilde{x}_I, y), (x, y)$, and ω
  Update θ by minimizing L
end for

Table 1. Model classification accuracies of original images.

Model	Acc on MNIST	Acc on CIFAR10
Standard	98.90%	94.70%
PGD-based training	97.05%	80.69%
WasserTrain	98.23%	86.37%

WasserTrain is different from the PGD-based approach in both adversarial attacks and adversarial training. For adversarial attacks, the PGD-based attack method develops an approximate projection operator onto the Wasserstein ball called projected Sinkhorn to find adversarial examples. However, due to the projection operation and clipping pixel values to $[0, 1]$, the requirement that the sum of the pixel value before and after Wasserstein transformation should be equal is seriously unsatisfied. Our WasserAttack directly finds the worst point within the Wasserstein ball, in which the Lagrangian relaxation and the change of variables technique are introduced to handle the constraints and eliminate the precision error. For adversarial training, the PGD-based training method processes adversarial training on the generated adversarial examples, while our WasserTrain defines a mixed loss on the original images and adversarial examples to improve the model accuracy and robustness together.

5. EXPERIMENTS

This section compares the proposed WasserAttack and WasserTrain with the PGD-based method [14] on MNIST [27] and CIFAR10 [28]. We would like to show: (i) WasserAttack produces adversarial examples with higher attack success rate, larger adversarial loss, and lower precision error than the PGD-based attack method; (ii) WasserTrain significantly improves the model robustness and outperforms the PGD-based training method against Wasserstein adversarial attacks.

The experiment settings are the same as those the PGD-based method uses in [14] as much as possible. For hyperparameters, we set $\alpha = \beta = \gamma = 10, \omega = 0.7, I = 200$, and $M = 50$. Table 1 summarizes the accuracies of three kinds of models on original images, including the original standard models and the models after adversarial training of the PGD-based training method and WasserTrain. It can be found that the accuracies of WasserTrain models are higher than those of the PGD-based training models, especially on CIFAR10 with 5.68% improvement. All experiments are conducted on the x86_64 Linux platform, with an Intel(R) Xeon(R) Gold 6146 CPU, 320GB RAM, and a Tesla V100 GPU.

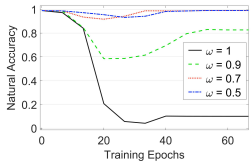

Fig. 3. Natural accuracy on original images along with training epochs of different ω.

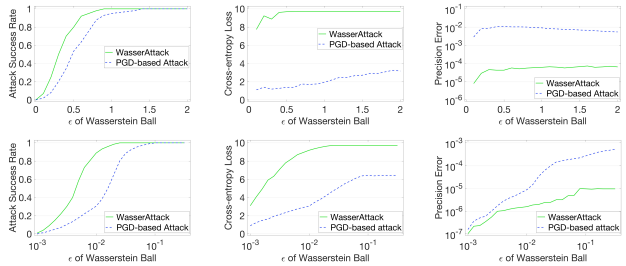

Fig. 4. Attack success rate, cross-entropy loss, and precision error of WasserAttack and the PGD-based attack over varying sizes of ϵ on the standard models. The upper subplots are on MNIST, and the lower ones are on CIFAR10.

5.1. Experimental settings

During adversarial training, we adopt the same settings of attack budget and attack iteration as those used by the PGD-based method in the literature. When generating adversarial examples for robust training, the adversarial attack is processed in a budget version. The radius of Wasserstein ball ϵ is in an adaptive scheme to avoid selecting a specific value. In detail, For MNIST, we first let $\epsilon = 0.1$ on the first iteration, and then increase it by a factor of 1.4 every 5 iterations. The adversarial attack terminates with an adversarial example or up to 50 iterations. The allowed range of ϵ is $[0.1, 2.1]$. For CIFAR10, a similar setting is used and we omit it here to save space.

Figure 3 displays the accuracy of WasserTrain model on original images in MNIST along with training epochs, where ω denotes the relative weight between the adversarial loss and the regular loss. We can find that for all ω, the natural accuracies fluctuate at the beginning of the training epochs, since the distribution of adversarial examples is different from that of original images. It can be observed that the natural accuracies are seriously affected when $\omega \geq 0.9$. Contrarily, when $\omega \leq 0.7$, the optimization process performs well, and increasing ω can accelerate the convergence process. In the experiments, we set $\omega = 0.7$ for better performance.

5.2. Wasserstein adversarial attacks evaluation

Three aspects are considered to evaluate the performance of adversarial attacks: **Attack success rate**: the percentage of the adversarial examples incorrectly classified. The higher the attack success rate is, the stronger the attack method is, or the less robust the model is. **Cross-entropy loss**: the attack effects of the adversarial examples, which is the higher the better for attack methods; **Precision error**: the ratio of difference of the sum of pixel value between original images and adversarial examples to the sum of pixel value of original images, which is the lower the better for attack methods.

Figure 4 displays the experiments results. It can be found that on MNIST WasserAttack achieves the 100% attack success rate within the Wasserstein ball with radius $\epsilon = 0.9$, whereas the PGD-based attack method achieves the same attack success rate with $\epsilon = 1.4$. In the middle subplot, the cross-entropy loss of WasserAttack is about three times higher than that of the PGD-based attack method. Moreover, as shown in the right subplot, the precision error of WasserAttack is about two orders of magnitude smaller than that of the PGD-based attack method, which means our loss function encoding method performs well on MNIST. On CIFAR10, we get similar results. The experiments results of adversarial attacks show that WasserAttack produces more threatening adversarial examples than the PGD-based attack method.

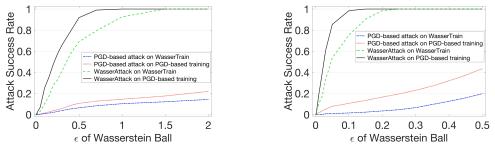

Fig. 5. Attack success rate of the PGD-based attack and WasserAttack on trained models after adversarial training. The left subplot is on MNIST, and the right one is on CIFAR10.

5.3. Wasserstein adversarial training evaluation

Based on the high-quality adversarial examples generated by WasserAttack, WasserTrain processes robust training to improve the model robustness. Figure 5 shows the experimental results. On MNIST, within the Wasserstein ball with radius $\epsilon = 0.5$, the attack success rate of WasserAttack is 91% on the PGD-based training model, whereas it is 69% on the WasserTrain model, which means WasserTrain can improve 22% model robustness compared with the PGD-based training method. On the other hand, with $\epsilon = 2$, the PGD-based attack method achieves 22% and 14% attack success rates on the PGD-based training and WasserTrain models, respectively, which means there is an 8% improvement of the WasserTrain model. On CIFAR10, we get similar results, and the model robustness improvement is more significant. For WasserAttack with $\epsilon = 0.05$, the attack success rate on the WasserTrain model is 53%, with 32% lower than that of 85% on the PGD-based training model. Besides, for the PGD-based attack with $\epsilon = 0.5$, the attack success rate on the WasserTrain model is 20%, half less than 43% on the PGD-based training model.

These results show that WasserTrain models are more robust than the PGD-based training models against adversarial attacks. Interestingly, it can be observed that the attack success rate of WasserAttack can always achieve 100% with increasing ϵ. To further improve the model robustness against WasserAttack, a possible solution is to conduct adversarial training within a larger Wasserstein ball than that of WasserAttack.

6. CONCLUSION

In this paper, we propose an adversarial training framework against Wasserstein adversarial attacks. To find the worst point in the adversarial examples generation procedure, a novel encoding of the optimization problem is introduced while keeping the relaxation error of the Wasserstein transformation as small as possible. The successive robust training procedure improves the model robustness based on the adversarial examples. Experiments show that WasserAttack generates adversarial examples with much higher quality than the PGD-based attack method. Compared with the PGD-based training method, WasserTrain significantly improves model robustness and outperforms against Wasserstein adversarial attacks.

Acknowledgments. We gratefully acknowledge the support from the National Natural Science Foundation of China under Grant 62172211, 62172210, 62172200, the Leading-edge Technology Program of Jiangsu Natural Science Foundation No. BK20202001.

7. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NeurIPS*, 2012, pp. 1106–1114.

[2] Tomas Mikolov, Daniel Povey, Lukás Burget, and Jan Cernocký, "Strategies for training large scale neural network language models," in *ASRU*, 2011, pp. 196–201.

[3] Matt Spencer, Jesse Eickholt, and Jianlin Cheng, "A deep learning network approach to ab initio protein secondary structure prediction," *IEEE/ACM Trans. Comput. Biology Bioinform.*, vol. 12, no. 1, pp. 103–112, 2015.

[4] Nicholas Carlini and David A. Wagner, "Towards evaluating the robustness of neural networks," in *SP*, 2017, pp. 39–57.

[5] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.

[6] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *CVPR*, 2016, pp. 2574–2582.

[7] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus, "Intriguing properties of neural networks," in *ICLR*, 2014.

[8] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh, "EAD: elastic-net attacks to deep neural networks via adversarial examples," in *AAAI*, 2018, pp. 10–17.

[9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, "Towards deep learning models resistant to adversarial attacks," in *ICLR*, 2018.

[10] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry, "Exploring the landscape of spatial robustness," in *ICML*, 2019, vol. 97 of *Proceedings of Machine Learning Research*, pp. 1802–1811.

[11] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *SP*, 2016, pp. 582–597.

[12] Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin T. Vechev, "An abstract domain for certifying neural networks," *Proc. ACM Program. Lang.*, vol. 3, no. POPL, pp. 41:1–41:30, 2019.

[13] Aman Sinha, Hongseok Namkoong, and John C. Duchi, "Certifying some distributional robustness with principled adversarial training," in *ICLR*, 2018.

[14] Eric Wong, Frank R. Schmidt, and J. Zico Kolter, "Wasserstein adversarial examples via projected sinkhorn iterations," in *ICML*, 2019, pp. 6808–6817.

[15] Wang Lin, Zhengfeng Yang, Xin Chen, Qingye Zhao, Xiangkun Li, Zhiming Liu, and Jifeng He, "Robustness verification of classification deep neural networks via linear programming," in *CVPR*, 2019, pp. 11418–11427.

[16] Taesik Na, Jong Hwan Ko, and Saibal Mukhopadhyay, "Cascade adversarial machine learning regularized with a unified embedding," in *ICLR*, 2018.

[17] Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake, "Evaluating robustness of neural networks with mixed integer programming," in *ICLR*, 2019.

[18] Meng Sha, Xin Chen, Yuzhe Ji, Qingye Zhao, Zhengfeng Yang, Wang Lin, Enyi Tang, Qiguang Chen, and Xuandong Li, "Synthesizing barrier certificates of neural network controlled continuous systems via approximations," in *DAC*, 2021, pp. 631–636.

[19] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang, "Semidefinite relaxations for certifying robustness to adversarial examples," in *NeurIPS*, 2018, pp. 10900–10910.

[20] Qingye Zhao, Xin Chen, Yifan Zhang, Meng Sha, Zhengfeng Yang, Wang Lin, Enyi Tang, Qiguang Chen, and Xuandong Li, "Synthesizing relu neural networks with two hidden layers as barrier certificates for hybrid systems," in *HSCC*, 2021, pp. 1–11.

[21] Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy A. Mann, and Pushmeet Kohli, "A dual approach to scalable verification of deep networks," in *UAI*, 2018, pp. 550–559.

[22] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane S. Boning, and Inderjit S. Dhillon, "Towards fast computation of certified robustness for relu networks," in *ICML*, 2018, pp. 5273–5282.

[23] Mislav Balunovic and Martin T. Vechev, "Adversarial training and provable defenses: Bridging the gap," in *ICLR*, 2020.

[24] Kaiwen Wu, Allen Wang, and Yaoliang Yu, "Stronger and faster wasserstein adversarial attacks," in *ICML*. PMLR, 2020, pp. 10377–10387.

[25] Shiqi Wang, Yizheng Chen, Ahmed Abdou, and Suman Jana, "Mixtrain: Scalable training of formally robust neural networks," *arXiv preprint arXiv:1811.02625*, 2018.

[26] Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao, "Is robustness the cost of accuracy? - A comprehensive study on the robustness of 18 deep image classification models," in *ECCV*, 2018, vol. 11216 of *Lecture Notes in Computer Science*, pp. 644–661.

[27] Yann LeCun, Corinna Cortes, and Christopher J.C. Burges, "The mnist database of handwritten digits," 1998.

[28] Alex Krizhevsky, "Learning multiple layers of features from tiny images," 2009.