ON EXTENSIONS OF CLEVER: A NEURAL NETWORK ROBUSTNESS EVALUATION ALGORITHM

ON EXTENSIONS OF CLEVER: A NEURAL NETWORK ROBUSTNESS EVALUATION ALGORITHM

Tsui-Wei Weng[1,3], Huan Zhang[2*], Pin-Yu Chen[3], Aurelie Lozano[3], Cho-Jui Hsieh[2], Luca Daniel[1]*

[1]Massachusetts Institute of Technology, Cambridge, MA 02139
[2]University of California, Los Angeles, CA 90095
[3]IBM Research, Yorktown Heights, NY 10598

ABSTRACT

CLEVER (Cross-Lipschitz Extreme Value for nEtwork Robustness) is an Extreme Value Theory (EVT) based robustness score for large-scale deep neural networks (DNNs). In this paper, we propose two extensions on this robustness score. First, we provide a new formal robustness guarantee for classifier functions that are twice differentiable. We apply extreme value theory on the new formal robustness guarantee and the estimated robustness is called second-order CLEVER score. Second, we discuss how to handle gradient masking, a common defensive technique, using CLEVER with Backward Pass Differentiable Approximation (BPDA). With BPDA applied, CLEVER can evaluate the *intrinsic* robustness of neural networks of a broader class – networks with non-differentiable input transformations. We demonstrate the effectiveness of CLEVER with BPDA in experiments on a 121-layer Densenet model trained on the ImageNet dataset.

***Index Terms*—** Adversarial Examples, Deep Learning, Robustness Evaluation

1. INTRODUCTION

It is well-known that deep neural networks (DNNs) are vulnerable to adversarial examples, and a small perturbation added to the input can mislead the network to classify in any desired class. There has been significant efforts developing verification techniques to prove that no adversarial perturbation δ exists if $\|\delta\|_p \leq r$ given an input $\boldsymbol{x_0}$ and a classifier function f. However, the verification problem is hard and generally intractable because a general neural network classifier is highly non-convex and non-smooth.

Alternatively, instead of verifying the exact robustness r, one idea is to provide a *lower bound* of r, which guarantees that no adversarial examples exist within an ℓ_p ball of radius ϵ. We call ϵ the *robustness lower bound* of the input image $\boldsymbol{x_0}$ on classifier function f. CLEVER (Cross-Lipschitz Extreme Value for nEtwork Robustness) [1] is the first attack-agnostic robustness score to estimate the robustness lower bound ϵ for large-scale DNNs, e.g. modern ImageNet networks such as ResNet, Inception, etc. It is based on a theoretical analysis of formal robustness guarantee with Lipschitz continuity assumption. The authors of [1] propose a sampling based approach with Extreme Value Theory to estimate the local Lipschitz constant, and empirically, this estimation aligns well with other robustness evaluation metrics, for example, the distortion of adversarial perturbation found by strong attacks.

In this work, we provide two extensions of CLEVER. First, we derive a new robustness guarantee for classifier functions that are twice differentiable, and we estimate the theoretical bounds via extreme value theory. Second, we extend CLEVER to be capable of evaluating the robustness of networks with non-differentiable input transformations, making it available for a wider class of neural networks deployed with gradient masking based defense.

2. RELATED WORK

Evaluating the robustness of a neural network can be done by crafting adversarial examples with a specific attack algorithm [2, 3, 4, 5]. However, this methodology has a major drawback as the resilience of a network to existing attacks is not guaranteed to be extended to subsequent attacks. In fact, many defensive methods have been shown either partially or completely broken after stronger and adaptive attacks are proposed [6, 7, 8, 9]. Thus, it is of great importance to provide an attack-agnostic robustness evaluation metric.

On the other hand, existing formal verification methods that solves the exact minimum adversarial distortion r (which is independent of attack algorithm) are quite expensive – verifying a small network with only a few hundred neurons on one input example can take a few hours [10], and in fact, even finding a non-trivial lower bound for r can be hard, and so far only results on CIFAR and MNIST networks are available [11, 12]. [1] presents a framework to estimate local Lipschitz constant using extreme value theory, and then obtain an attack-agnostic robustness score (CLEVER) based on first-order Lipschitz continuity condition. CLEVER can scale to ImageNet networks.

*Equally contributed. Codes: https://github.com/huanzhang12/CLEVER.

Recently, Goodfellow [13] raises concerns on CLEVER in the case of networks with gradient masking, a defensive technique that obfuscates model gradients to prevent gradient based attacks. One of the main objective of this work is to show that such concerns can be safely eliminated with the BPDA technique proposed in [6]. Moreover, we also experimentally show how CLEVER can successfully handle networks with non-differentiable input transformations, including the stair-case function example in [13].

3. EXTENDING CLEVER WITH SECOND ORDER APPROXIMATION

3.1. Background and definitions

Let $\boldsymbol{x_0} \in \mathbb{R}^d$ be the input of a K-class classifier $f : \mathbb{R}^d \to \mathbb{R}^K$, the predicted class of $\boldsymbol{x_0}$ is $c(\boldsymbol{x_0}) = \arg\max_{1 \le i \le K} f_i(\boldsymbol{x_0})$. Given $\boldsymbol{x_0}$ and c, we say $\boldsymbol{x_a} := \boldsymbol{x_0} + \boldsymbol{\delta}$ is an adversarial example if there exists a $\boldsymbol{\delta} \in \mathbb{R}^d$ makes $c(\boldsymbol{x_a}) \neq c(\boldsymbol{x_0})$ while $\|\boldsymbol{\delta}\|_p$ is small. A successful *untargeted attack* is to find a $\boldsymbol{x_a}$ such that $c(\boldsymbol{x_a}) \neq c(\boldsymbol{x_0})$ while a successful *targeted attack* is to find a $\boldsymbol{x_a}$ such that $c(\boldsymbol{x_a}) = t$ given a target class $t \neq c(\boldsymbol{x_0})$. On the other hand, the definition of norm-bounded robustness ϵ is the following: given a target class t, ϵ is the *targeted* robustness of $\boldsymbol{x_0}$, if

$$g_t(\boldsymbol{x_0} + \boldsymbol{\delta}) \ge 0, \forall \, \|\boldsymbol{\delta}\|_p \le \epsilon, \tag{1}$$

where $g_t(\boldsymbol{x}) := f_c(\boldsymbol{x}) - f_t(\boldsymbol{x})$. Similarly, ϵ is the *untargeted* robustness if (1) holds for all classes $t \neq c(\boldsymbol{x_0})$.

3.2. Robustness for continuously differentiable classifiers

In [1], the authors have shown that if the classifier function f has continuously differentiable components f_i, the targeted robustness is

$$\epsilon = \min(\frac{g_t(\boldsymbol{x_0})}{L_q^t}, R), \tag{2}$$

where L_q^t is the local Lipschitz constant for the function $g_t(\boldsymbol{x})$ within a local region $\boldsymbol{x} \in \mathbb{B}_p(\boldsymbol{x_0}, R)$ and $1/p + 1/q = 1, \, 1, \le p, q \le \infty$. A simple proof of this guarantee is based on the mean value theorem on the first order expansion of $g_t(\boldsymbol{x_0} + \boldsymbol{\delta})$:

$$\exists s \in [0, 1], \;\; g_t(\boldsymbol{x_0} + \boldsymbol{\delta}) = g_t(\boldsymbol{x_0}) + \nabla g_t(\boldsymbol{x_0} + s\boldsymbol{\delta})^\top \boldsymbol{\delta}. \tag{3}$$

With Hölder's inequality,

$$\begin{aligned} g_t(\boldsymbol{x_0} + \boldsymbol{\delta}) &= g_t(\boldsymbol{x_0}) + \nabla g_t(\boldsymbol{x_0} + s\boldsymbol{\delta})^\top \boldsymbol{\delta} \\ &\ge g_t(\boldsymbol{x_0}) - \|\nabla g_t(\boldsymbol{x_0} + s\boldsymbol{\delta})\|_q \|\boldsymbol{\delta}\|_p \\ &\ge g_t(\boldsymbol{x_0}) - \max_{\boldsymbol{x} \in \mathbb{B}_p(\boldsymbol{x_0}, R)} \|\nabla g_t(\boldsymbol{x})\|_q \cdot \|\boldsymbol{\delta}\|_p \\ &= g_t(\boldsymbol{x_0}) - L_q^t \cdot \|\boldsymbol{\delta}\|_p. \end{aligned}$$

Thus, the targeted robustness bound (2) is obtained by requiring the lower bound of $g_t(\boldsymbol{x_0} + \boldsymbol{\delta})$ to be non-negative. The authors of [1] further extend their analysis to neural networks with ReLU activations, which is a special case of *non-differentiable* functions.

3.3. Robustness for twice differentiable classifiers

In this work, we provide formal robustness guarantees when classifier functions f are twice differentiable – for example, neural networks with twice differentiable activations such as tanh, sigmoid, softplus, etc. For a twice-differentiable function $g_t(\boldsymbol{x}) := f_c(\boldsymbol{x}) - f_t(\boldsymbol{x})$, there exists $s \in [0, 1]$ such that

$$g_t(\boldsymbol{x_0} + \boldsymbol{\delta}) = g_t(\boldsymbol{x_0}) + \nabla g_t(\boldsymbol{x_0})^\top \boldsymbol{\delta} + \frac{1}{2} \boldsymbol{\delta}^\top \boldsymbol{H}(\boldsymbol{x_0} + s\boldsymbol{\delta}) \boldsymbol{\delta}, \tag{4}$$

where $\boldsymbol{H}(\boldsymbol{x_0} + s\boldsymbol{\delta})$ is the Hessian of g_t at $\boldsymbol{x_0} + s\boldsymbol{\delta}$. This is analogous to the Mean Value Theorem in the first order case, but extended with a second order term. This expansion of $g_t(\boldsymbol{x_0} + \boldsymbol{\delta})$ can be used to derive the targeted robustness of $\boldsymbol{x_0}$ in the following Theorem:

Theorem 3.1 (Formal robustness guarantee)**.** *Given an input $\boldsymbol{x_0}$ and a K-class classifier f, the targeted robustness of $\boldsymbol{x_0}$ is*

$$\epsilon = \min(\frac{-b + \sqrt{b^2 + 2a\gamma}}{a}, R) \tag{5}$$

where $a = \max_{\boldsymbol{x} \in \mathbb{B}_p(\boldsymbol{x_0}, R)} \|\boldsymbol{H}(\boldsymbol{x})\|_{p,q}$, $b = \|\nabla g_t(\boldsymbol{x_0})\|_p$, and $\gamma = g_t(\boldsymbol{x_0})$.

Proof. By holder's inequality and the definition of induced norm, we have

$$|\nabla g_t(\boldsymbol{x_0})^\top \boldsymbol{\delta}| \le \|\nabla g_t(\boldsymbol{x_0})\|_q \|\boldsymbol{\delta}\|_p$$

and

$$\begin{aligned} |\boldsymbol{\delta}^\top \boldsymbol{H}(\boldsymbol{x_0} + s\boldsymbol{\delta}) \boldsymbol{\delta}| &\le \|\boldsymbol{H}(\boldsymbol{x_0} + s\boldsymbol{\delta}) \boldsymbol{\delta}\|_q \|\boldsymbol{\delta}\|_p \\ &\le \|\boldsymbol{H}(\boldsymbol{x_0} + s\boldsymbol{\delta})\|_{p,q} \|\boldsymbol{\delta}\|_p \|\boldsymbol{\delta}\|_p \\ &\le \max_{\boldsymbol{x} \in \mathbb{B}_p(\boldsymbol{x_0}, R)} \|\boldsymbol{H}(\boldsymbol{x})\|_{p,q} \|\boldsymbol{\delta}\|_p^2. \end{aligned}$$

Let $a = \max_{\boldsymbol{x} \in \mathbb{B}_p(\boldsymbol{x_0}, R)} \|\boldsymbol{H}(\boldsymbol{x})\|_{p,q}$, $b = \|\nabla g_t(\boldsymbol{x_0})\|_p$, and $\gamma = g_t(\boldsymbol{x_0})$, we get a lower bound of $g_t(\boldsymbol{x_0} + \boldsymbol{\delta})$:

$$\begin{aligned} g_t(\boldsymbol{x_0} + \boldsymbol{\delta}) &= g_t(\boldsymbol{x_0}) + \nabla g_t(\boldsymbol{x_0})^\top \boldsymbol{\delta} + \frac{1}{2} \boldsymbol{\delta}^\top \boldsymbol{H}(\boldsymbol{x_0} + s\boldsymbol{\delta}) \boldsymbol{\delta} \\ &\ge g_t(\boldsymbol{x_0}) - b\|\boldsymbol{\delta}\|_p - \frac{1}{2} a \|\boldsymbol{\delta}\|_p^2. \end{aligned} \tag{6}$$

If we can guarantee (6) ≥ 0, then we can guarantee $g_t(\boldsymbol{x_0} + \boldsymbol{\delta}) \ge 0$, which is the definition of targetted robustness in (1). Thus, the condition of (6) ≥ 0 gives

$$\|\boldsymbol{\delta}\|_p \le \frac{-b + \sqrt{b^2 + 2a\gamma}}{a}.$$

□

3.4. Sampling via Extreme Value Theory

Theorem 3.1 needs the value $a := \max_{\boldsymbol{x} \in \mathbb{B}_p(\boldsymbol{x_0}, R)} \|\boldsymbol{H}(\boldsymbol{x})\|_{p,q}$, which is the maximum subordinate norm of the Hessian matrix within $\boldsymbol{x} \in \mathbb{B}_p(\boldsymbol{x_0}, R)$. When $p = q = 2$, it becomes the

well-known spectral norm, and can be evaluated efficiently on a single point $\boldsymbol{x}$ using power iteration or Lanczos method. Under the framework of CLEVER, we apply extreme value theory to estimate a by sampling different $\boldsymbol{x} \in \mathbb{B}_p(\boldsymbol{x_0}, R)$ and running power iterations on each sampled point. In this paper, we focus on the case of $p = q = 2$ only (ℓ_2 robustness). After we get an estimate of a, a second order robustness lower bound can be estimated at point $\boldsymbol{x_0}$ using (5). The estimated bound of (2) is named *1st-order* CLEVER while the estimated bound of (5) is called *2nd-order* CLEVER.

4. CLEVER WITH GRADIENT MASKING BASED DEFENSE

4.1. Gradient Masking

Gradient masking [14] is a popular defending method against adversarial examples where the model does not provide useful gradients for generating adversarial examples. Typical gradient masking techniques include adding non-differentiable layers [15] (bit-depth reduction, JPEG compression, etc) to the network, numerically making the gradient vanish (Defensive Distillation [16]), and modifying the optimization landscape of the loss function in a local region [14] of each data point. These methods typically prevent gradient-based adversarial attacks by providing non-informative gradients. However, many of the gradient masking techniques have been shown ineffective as a defense. Notably, Defensive Distillation can be bypassed by attacking the logit (unnormalized probability) layer values to avoid the saturated softmax functions; many non-differentiable transformation functions can be bypassed using the Backward Pass Differentiable Approximation (BPDA) [6]; the modifications in local landscape of the loss function can be escaped by adding a small random noise when performing the attack [14].

When CLEVER is evaluated, we always use the logit layer values, thus we are not subject to the saturation of the sigmoid units. Additionally, during the sampling processes, we evaluate gradients using a large number of randomly perturbed images, thus CLEVER is likely to escape the region of masked gradients in local loss landscape. The remaining concern is thus whether CLEVER can be evaluated on networks with a non-differentiable layer as a defense. For example, if the input image is quantized via bit-depth reduction, a staircase function is applied to the network and thus its gradient cannot be computed via automatic differentiation. We will formally discuss this situation in the next section.

4.2. Apply Backward Pass Differentiable Approximation (BPDA) to CLEVER

For a neural network classifier $f(\boldsymbol{x})$, we can apply a non-differentiable transformation $h(\boldsymbol{x})$ to the input $\boldsymbol{x}$ and then feed the data after transformation into f. The function $f(h(\boldsymbol{x}))$ thus becomes non-differentiable, and gradient based adversarial attacks fail to find successful adversarial examples. An example of $h(\boldsymbol{x})$ is a staircase function, as suggested in [13]. This transformation also hinders the direct use of CLEVER to evaluate the robustness of $f(h(\boldsymbol{x}))$.

To handle non-differentiable transformations, we use the Backward Pass Differentiable Approximation (BPDA) [6] technique. The intuition behind BPDA is that although $h(\boldsymbol{x_0})$ is non-differentiable (e.g., bit-depth reduction, JPEG compression, etc), it usually holds that $h(\boldsymbol{x_0}) \approx \boldsymbol{x_0}$. Thus, in backpropagation, we can assume that

$$\nabla_x f(h(\boldsymbol{x}))|_{\boldsymbol{x}=\boldsymbol{x_0}} \approx \nabla_x f(\boldsymbol{x})|_{\boldsymbol{x}=h(\boldsymbol{x_0})}. \tag{7}$$

To evaluate CLEVER for a network with an input transformation h (for example, a staircase function), $\boldsymbol{x}$ is sampled within an ℓ_p ball around $\boldsymbol{x_0}$. Then, a transformation $h(\boldsymbol{x_0})$ is applied, such that $\hat{\boldsymbol{x}} = h(\boldsymbol{x})$. Then, the backpropagation procedure computes $\nabla_{\hat{\boldsymbol{x}}} f(\hat{\boldsymbol{x}})$. We simply collect $\nabla_{\hat{\boldsymbol{x}}} f(\hat{\boldsymbol{x}})$ as the gradient, and compute its norm as a sample for Lipschitz constant estimation.

4.3. CLEVER is a White-Box Evaluation Tool

CLEVER is intended to be a tool for network designers and to evaluate network robustness in the "white-box" setting in which we know how a (defended) neural network processes the input. In this case, we can deal with the non-differentiable transformation h with BPDA, and evaluate the *intrinsic* robustness of the model, without the "False Sense of Security [6]" provided by gradient masking.

In black-box attack setting, the gradient of $f(h(\boldsymbol{x}))$ must be evaluated via finite differences [17], thus a non-differentiable $g(\boldsymbol{x})$ prevents gradient based attacks in black-box settings because the estimated gradient becomes infinite (i.e., the value of $f(g(\boldsymbol{x}))$ is unlikely to change when $\boldsymbol{x}$ is changed by a small amount). Goodfellow [13] raises concerns on the effectiveness of CLEVER in this setting, but this setting is different from our intended usage of CLEVER. Most importantly, CLEVER computes gradients using back-propagation via automatic differentiation in the white-box setting, rather than using finite differences. Despite the limited numerical precision on digital computers, CLEVER is not subject to the same numerical issues as in the black-box attack setting. Unless backpropagation fails, CLEVER is able to estimate a reasonable robustness score reflecting the intrinsic model robustness.

5. EXPERIMENTS

5.1. Experiments on 1st Order and 2nd Order Bounds

We compute the targeted robustness bounds for a 7-layer CNN model with tanh activations (which is twice differentiable) on CIFAR dataset with a validation accuracy of 72.6%. We calculated both Eq. (2) and (5) via sampling with extreme

Table 1. Comparison of 1st order and 2nd order ℓ_2 CLEVER with least-likely target labels on a 7-layer $\tanh$ CIFAR CNN. The average distortion found by CW-ℓ_2 attack is 0.310.

Least-likely Target	1st order	2nd order
avg ℓ_2 CLEVER	0.057	0.051
% of images with larger score	54	46
avg % of increase on the score	47%	44%

Table 2. Comparison of 1st order and 2nd order ℓ_2 CLEVER with runner-up target labels on a 7-layer $\tanh$ CIFAR CNN. The average distortion found by CW-ℓ_2 attack is 0.101.

Runner-up Target	1st order	2nd order
avg ℓ_2 CLEVER	0.024	0.026
% of images with larger score	18	82
avg % of increase on the score	77%	58%

value theory, and we denote the estimated scores as "1st order" and "2nd order" CLEVER scores respectively in the Tables. In particular, we follow the sampling procedure proposed in [1] to estimate the Lipschitz constant by fitting the samples with maximum likelihood estimation on Reversed Weibull distribution and calculate the estimated robustness scores of (2). For the "2nd order" bound (5), we also use sampling and extreme value theory to calculate the estimated bounds, as describe in Section 3.4. For fair comparison, we use the same number of samples ($N_b = 100$ and $N_s = 200$) for both estimated bounds and we compare their average as well as the percentage of image examples that the score is larger than the other. For each image, we select three attack target classes: least likely, random and runner-up. The results are summarized in Tables 1, 2 and 3. We observe that the 1st order and 2nd order average CLEVER scores usually stay close, indicating that both scores agree with each other.

Since CLEVER is a score of estimated lower bound, we desire the score is not trivially small, but smaller than the upper bound found by adversarial attacks (in our case the CW ℓ_2 attack). As shown in Tables 1, 2 and 3, all CLEVER scores are less then CW ℓ_2 distortion. Second order CLEVER can sometimes give a better result than its first order counterpart, indicating that second order approximation is probably more accurate for these examples. The "avg. % of increase on the score" rows in tables report the improvement of score when one method is better than the other; for example, in runner-up target, second order CLEVER increases the score for 82% of the examples, and the average improvement of score comparing to first order CLEVER is 58%.

5.2. Experiments on Networks with Input Transformation as a Gradient Masking based Defense

We conduct experiments on a 121-layer DenseNet [18] network pretrained on ImageNet dataset[1]. We employ two non-differentiable input transfomrations that mask gradients: bit-depth reduction (reducing each color channel from 8-bit to 3-bit, setting all lower bits to 0) and JPEG compression (quality set to 75%). We compute ℓ_2 CLEVER (first order) scores for the network with and without input transformations, with CLEVER parameter $N_b = 200$ and $N_s = 1024$. We randomly choose 100 images from the ImageNet validation set, and select three attack target classes for each image (least likely, random and runner-up). Misclassified images are skipped.

[1]model available at https://github.com/pudae/tensorflow-densenet

Table 3. Comparison of 1st order and 2nd order ℓ_2 CLEVER with random target labels on a 7-layer $\tanh$ CIFAR CNN. The average distortion found by CW-ℓ_2 attack is 0.264.

Random Target	1st order	2nd order
avg ℓ_2 CLEVER	0.049	0.036
% of images with larger score	76	24
avg % of increase on the score	55%	68%

Table 4. ℓ_2 robustness CLEVER scores with and without input transformations on a 121-layer Densenet model, for three different target classes. The average adversarial distortion of CW ℓ_2 attack for the same set of images are 0.2058, 0.52788 and 0.66114, for runner-up, random and least-likely target classes, respectively.

Target Class	Runner-up	Random	Least Lilely
No transformation	0.14229	0.35632	0.44725
Bit-depth reduction	0.10223	0.26224	0.34722
JPEG compression	0.11539	0.27804	0.36275

Table 5.2 compares the ℓ_2 CLEVER scores for three target classes, for the original model, and for bit-depth reduction or JPEG compression as input transformations. BPDA is used to compute CLEVER when an input transformation is applied. Not surprisingly, the CLEVER scores for networks with input transformation as a gradient masking method do not noticeably increase, indicating that these transformations do not increase the model's intrinsic robustness; in other words, with BPDA applied, we can still obtain similar gradients as the original model, thus it is expected that CLEVER scores do not change too much in this situation.

6. CONCLUSIONS

CLEVER [1] is a first-order approximation based robustness score. We move one step further to give a second order formal guarantee for DNN robustness. We show that it improves the estimated robustness lower bound for some examples, and in many cases both first and second order CLEVER scores are coherent. Additionally, we successfully apply Backward Pass Differentiable Approximation (BPDA) to compute CLEVER scores for a network with non-differentiable input transformations, including staircase functions. Our discussions and results remedy the concerns raised in [13].

7. ACKNOWLEDGEMENT

Tsui-Wei Weng and Luca Daniel acknowledge partial support of MIT IBM Watson AI Lab.

8. REFERENCES

[1] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel, "Evaluating the robustness of neural networks: An extreme value theory approach," *Sixth International Conference on Learning Representations (ICLR)*, 2018.

[2] Nicholas Carlini and David Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 39–57.

[3] Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya Nori, and Antonio Criminisi, "Measuring neural net robustness with constraints," in *Advances in Neural Information Processing Systems*, 2016, pp. 2613–2621.

[4] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh, "Ead: elastic-net attacks to deep neural networks via adversarial examples," *arXiv preprint arXiv:1709.04114*, 2017.

[5] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.

[6] Anish Athalye, Nicholas Carlini, and David Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," *35th International Conference on Machine Learning (ICML)*, 2018.

[7] Anish Athalye and Nicholas Carlini, "On the robustness of the cvpr 2018 white-box adversarial example defenses," *arXiv preprint arXiv:1804.03286*, 2018.

[8] Nicholas Carlini and David Wagner, "Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples," *arXiv preprint arXiv:1711.08478*, 2017.

[9] Nicholas Carlini and David Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," *arXiv preprint arXiv:1705.07263*, 2017.

[10] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer, "Reluplex: An efficient smt solver for verifying deep neural networks," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 97–117.

[11] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel, "Towards fast computation of certified robustness for relu networks," *35th International Conference on Machine Learning (ICML)*, 2018.

[12] Matthias Hein and Maksym Andriushchenko, "Formal guarantees on the robustness of a classifier against adversarial manipulation," in *Advances in Neural Information Processing Systems*, 2017, pp. 2263–2273.

[13] Ian Goodfellow, "Gradient masking causes clever to overestimate adversarial perturbation size," *arXiv preprint arXiv:1804.07870*, 2018.

[14] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel, "Ensemble adversarial training: Attacks and defenses," *Sixth International Conference on Learning Representations (ICLR)*, 2018.

[15] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten, "Countering adversarial images using input transformations," *arXiv preprint arXiv:1711.00117*, 2017.

[16] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 582–597.

[17] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh, "ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 15–26.

[18] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, vol. 1, p. 3.