2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C) | 978-1-6654-7836-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/QRS-C55045.2021.00012

ADVRET: An Adversarial Robustness Evaluating and Testing Platform for Deep Learning Models

Fei Ren
Institution of Computer Application CAEP
MianYang, China
1356781673@qq.com

Yonghui Yang*
Institution of Computer Application CAEP
MianYang, China
younphy@163.com

Chi Hu
Institution of Computer Application CAEP
MianYang, China
huchi_nudt@126.com

Yuyao Zhou
Institution of Computer Application CAEP
MianYang, China
zhouyy211816@126.com

Siyou Ma
Institution of Computer Application CAEP
MianYang, China
infogc@163.com

***Abstract*—Recent studies have shown that deep learning models such as image classification and object detection models are vulnerable to attacks from adversarial examples. These malicious examples may trigger potential security threats and cause damage in security-related fields. In this paper, in order to find model robustness weakness, we obtain the adversarial robustness metric against several adversarial attacks, including gradient-based attacks, optimization-based attacks, etc. The metric quantizes the adversarial robustness of deep learning models and can be used as a guide for extra model training. Meanwhile, we design a testing platform to achieve model adversarial robustness evaluation. Utilizing the front and back end separation strategy, the testing platform consist of 2 modules: one is multiple adversarial example attacks executor and the other is HCI module. In addition, the tool allows parallel comparison among different adversarial attack algorithms under given conditions and measures the quality of generated adversarial examples utilizing FID(Fréchet Inception Distance) metrics. Besides, we developed ERFGSM(edge-RFGSM), a new gradient-based attack using edge information through canny operator.**

***Index Terms*—robustness, platform, adversarial example, ERFGSM, evaluation**

I. Introduction

In recent years, deep neural networks have achieved great success in areas such as image classification and image recognition. Common classifier models contain ResNet, VggNet, AlexNet, MobileNet, etc. Common target detection models are mainly divided into one-stage models and two-stage models. The representative one-stage models are SSD and YOLO, which are usually much faster than two-stage models. The representative two-stage model is RCNN, which is slower but performs better in terms of accuracy. These deep learning models are used in a wide range of applications, such as supermarket shopping, face recognition, autonomous driving, and even medical education.

* means corresponding author. CAEP denotes for China Academy of Engineering Physics.

However, the security of deep learning models has come into problem with the discovery of adversarial examples. Contaminated images generated by algorithms can be used to attack deep learning models. These images are so carefully designed that the human eye cannot perceive the differences. Most of the current adversarial example research is focused on deep classification networks, while less research has been conducted on adversarial example generation algorithms for deep target detection networks. However, many detection models have still been successfully attacked, which could lead to catastrophic consequences in security-critical areas. So it is essential to verify that deep learning networks are robust to adversarial examples. Evaluations can be done by performing a variety of adversarial example attacks on the network. There are several main categories of adversarial example algorithms: gradient-based, optimization-based, boundary decision-based, search-based, and patch-based. The characteristics of the adversarial examples generated by different types of adversarial algorithms vary. Therefore the adversarial robustness needs to be measured from the aspect of multiple algorithms due to the strengths and weaknesses of the adversarial example algorithms themselves.

There are several works that implement most existing adversarial attacks and integrate them in a library. For example Cleverhans [1], FoolBox [2], AdverTorch [3], and Adversarial Robustness Toolkit [4]. These libraries provide interfaces for programmers to implement one or more specific adversarial attacks. In order to test and evaluate model effectiveness under adversarial attacking, and to reduce the probability that adversarial examples deceived models successfully, researchers construct model adversarial robustness evaluation platforms, such as AdvBox [5], Composite Adversarial Attacks(CAA, Alibaba) [6], Adversarial Robustness Benchmark [7], DeepSec [8]. All of them provide an evaluation mechanism by collecting original examples, implementing attacks quickly, generating adversarial examples, test models, and show evaluation re-

2693-9371/21/$31.00 ©2021 IEEE
DOI 10.1109/QRS-C55045.2021.00012

sults. For example, CAA automatically selects attackers, tunes hyper-parameters of the attackers, and composites multiple attackers to achieve stronger attack performance for effectively evaluating model adversarial robustness.

To this end, this paper designs a platform that integrates a variety of adversarial example attack methods. Users can upload a customized model via a web-based platform and select multiple attack methods to attack the model. We obtain a visual analysis of the attack results. It also allows for a side-by-side comparison of different adversarial example algorithms in some ways of time cost, quality (measured using FID [9]), attack success rate, etc. We also proposed a new gradient-based algorithm called ERFGSM which can utilize edge information to craft more inconspicuous adversarial examples.

II. Adversarial Algorithms

The platform implements a variety of adversarial example algorithms, divided into several types: gradient-based, optimisation-based and other methods. The gradient-based adversarial algorithms include FGSM, BIM, PGD, RFGSM, MI-FGSM, FFGSM; The optimisation-based adversarial algorithms include C&W and the other methods include DeepFool, OnePixel, etc. The following contents introduce several algorithms used in our platform.

A. Gradient-based Methods

FGSM (Fast Gradient Sign Method) is a lightening single-step algorithm proposed by Goodfellow, Shlens, and Szegedy(2014) [10]. The method first feeds legitimate examples into a machine learning model and calculates the gradient of the model loss function over the legitimate examples. The gradient implies the direction of perturbation. The attacker then adds a fixed size of noise to the original examples in the direction of this gradient. Due to the simplicity of modifying perturbation in a single pass, the method achieves fast generation of adversarial examples. Finally, adding the generated adversarial examples to the training dataset and labelling them with the target category enables the model's classification plane to be shifted and flipped, thus interfering with the model's normal recognition of all input data. The formula is as follow:

$$x' = x + \epsilon \cdot \mathrm{sgn}(\nabla_x \ell(f(x), y)) \tag{1}$$

where x is the original image as input and x' represents the adversarial example. ϵ is a hyper parameter which determines the attack strength. y is the output of model and f represents a DNN model which outputs a score vector: $f(x) \in [0,1]^m$ and ℓ is the loss function for training.

BIM(or iterative-FGSM) is an iterative adversarial attack method proposed by Kurakin,Goodfellow,and Bengio (2016) [11]. Different from FGSM, it utilizes multiple iterative gradients instead of one. To achieve this, Kurakin,Goodfellow,and Bengio (2016) divide FGSM into several steps with an hyper parameter t. They also use α as a replacement of ϵ to get more precise perturbation. The formula is as follow:

$$x' = clip_{(x,\epsilon)}\left\{x'_t + \alpha \cdot \mathrm{sgn}(\nabla_{x'_t} \ell(f(x'_t), y))\right\} \tag{2}$$

where $clip_{(x,\epsilon)}\{x'_t\}$ denotes $\min(\max(x', x-\epsilon), x+\epsilon)$. $x'_0 = x$ and x'_t denotes the adversary after t-steps. Interestingly, if we add a randomized noise to the input at the beginning, it becomes a new method called PGD.

RFGSM adds a random initialization before computing the gradient. It is designed to avoid gradient masking effect by Tramer [12].

$$\begin{aligned} x'_0 &= x + \alpha \cdot \mathrm{sgn}\left(\mathcal{N}\left(\mathbf{0}^n, \mathbf{I}^n\right)\right) \\ x'_{t+1} &= \mathrm{clip}_{(x,\epsilon)}\left\{x'_t + (\epsilon-\alpha) \cdot \mathrm{sgn}\left(\nabla_{x'_t} \ell\left(f\left(x'_t\right), y\right)\right)\right\} \end{aligned} \tag{3}$$

where $clip_{(x,\epsilon)}\{x'_t\}$ denotes $\min(\max(x', x-\epsilon), x+\epsilon)$ and $\mathcal{N}\left(\mathbf{0}^n, \mathbf{I}^n\right)$ is a normal distribution. x'_t denotes the adversary after t-steps and α is a single step size.

MI-FGSM is actually FGSM with momentum algorithm (Dong. 2018) [13] In order to control the iterations more finely and find the best perturbations, Dong draws on the idea of momentum and add a momentum approach to the BIM base.

$$\begin{aligned} g_{t+1} &= \mu \cdot g_t + \frac{\nabla_{x'_t} \ell\left(f\left(x'_t\right), y\right)}{\left\|\nabla_{x'_t} \ell\left(f\left(x'_t\right), y\right)\right\|_1} \\ x'_{t+1} &= \Pi_{\mathcal{B}(x,\epsilon)}\left\{x'_t + \alpha \cdot \mathrm{sgn}\left(g_{t+1}\right)\right\} \end{aligned} \tag{4}$$

where $\Pi_{\mathcal{B}(x,\epsilon)}$ refers the projection to $\mathcal{B}(x,\epsilon)$. μ is a decay factor for the gradient direction.

FFGSM is a new attack proposed by Wong in 2020 [14]. For a faster adversarial example generation, FFGSM replaces the normal distributed noise with a uniformly one on top of RFGSM, meanwhile increasing the step size of the perturbation for faster iterations.

$$\begin{aligned} x'_0 &= x + \mathcal{U}(-\epsilon, \epsilon) \\ x' &= \Pi_{\mathcal{B}(x,\epsilon)}\left\{x'_0 + \alpha \cdot \mathrm{sgn}\left(\nabla_{x'_0} \ell\left(f\left(x'_0\right), y\right)\right)\right\} \end{aligned} \tag{5}$$

where $\Pi_{\mathcal{B}(x,\epsilon)}$ refers the projection to $\mathcal{B}(x,\epsilon)$.

ERFGSM is our new attack method which designed to decrease human perception in perturbations. Its main idea is utilizing the edge information as a constraint of perturbations generated by RFGSM.

$$\begin{aligned} x'_0 &= x + \mathrm{C}(x) \cdot \alpha \cdot \mathrm{sgn}\left(\mathcal{N}\left(\mathbf{0}^n, \mathbf{I}^n\right)\right) \\ x'_{t+1} &= x'_t + \mathrm{C}(x'_t) \cdot (\epsilon-\alpha) \cdot \mathrm{sgn}\left(\nabla_{x'_t} \ell\left(f\left(x'_t\right), y\right)\right) \end{aligned} \tag{6}$$

where $\mathrm{C}(x)$ denotes the edge information of an image calculated by canny operator.

B. Optimization-based Method

C&W is one of the strongest attack methods [15] and its main idea is to optimise the following problems:

$$\begin{array}{ll} \text{minimize} & \mathrm{D}(x, x+\delta) \\ \text{such that} & C(x+\delta) = t \\ & x + \delta \in [0,1]^n \end{array} \tag{7}$$

where x is fixed and the goal is to find the minimal δ. That is, by finding some small variation in δ, one can change the way a model classifies it, but the result is still a valid image. Here D is the distance metric function, e.g. L_0, L_2 or L_∞. Since the constraint $C(x+\delta)=t$ is highly non-linear, it is difficult for existing algorithms to solve the above equation directly. Therefore, we express it in a different form more suitable for optimization. We define an objective function f if and only if $f(x+\delta \leq 0)$. There are many possible choices for f:

$$\begin{aligned}
f_1(x') &= -\operatorname{loss}_{F,t}(x') + 1 \\
f_2(x') &= \left(\max_{i \neq t}\left(F(x')_i\right) - F(x')_t\right)^+ \\
f_3(x') &= \operatorname{softplus}\left(\max_{i \neq t}\left(F(x')_i\right) - F(x')_t\right) - \log(2) \\
f_4(x') &= \left(0.5 - F(x')_t\right)^+ \\
f_5(x') &= -\log\left(2F(x')_t - 2\right) \\
f_6(x') &= \left(\max_{i \neq t}\left(Z(x')_i\right) - Z(x')_t\right)^+ \\
f_7(x') &= \operatorname{softplus}\left(\max_{i \neq t}\left(Z(x')_i\right) - Z(x')_t\right) - \log(2)
\end{aligned} \tag{8}$$

where s is the correct classification, $(e)^+$ is a short-hand of $max(e,0)$, softplus $= \log(1+\exp(x))$, so the original problem turns into:

$$\begin{aligned}
&\text{minimize} \quad \mathrm{D}(x, x+\delta) + c \cdot f(x+\delta) \\
&\text{such that} \quad x+\delta \in [0,1]^n
\end{aligned} \tag{9}$$

where $c > 0$ is a constant, if we use l-norm to measure D, we get :

$$\begin{aligned}
&\text{minimize} \quad \|\delta\|_p + c \cdot f(x+\delta) \\
&\text{such that} \quad x+\delta \in [0,1]^n
\end{aligned} \tag{10}$$

C. Other Algorithms

Deepfool is an untargeted attack optimised with L_2 distance metric [16]. It is effective and produces adversarial examples closer to the decision boundary than the L-BFGS method, which is useful for exploring the decision boundary of the target model.

OnePixel [17], the algorithm uses DE(Differential Evolution) to search the full pixel space for pixels that are at the decision boundary. Modifying these pixels makes the model produce false predictions. However, the success rate is very low, as well as the search time is long, the only advantage is that only a very small number of pixels can be modified (in some cases even just one).

III. Platform Structure

The platform adopts a front and back-end separation strategy, with the main front-end technology stack being Vue3 + NaiveUI + Plotly. Vue is an incremental framework for building user interfaces, with its core library focused on the view layer, making it easy to get started and to integrate with third-party libraries or existing projects. On the other hand, when used in conjunction with a modern toolchain and various supporting libraries, Vue is also fully capable of driving complex single-page applications. In this framework, Vue is in conjunction with the UI library NaiveUI and the visual mapping library Plotly. Together they are responsible for human-computer interaction, providing interactive interfaces, such as model uploads and the selection of algorithms and the visualization of results. The main back-end technology stack is FastAPI + Pytorch + OpenCV + MongoDB. FastAPI is a modern, fast (high-performance) web framework for building APIs, using Python 3.6+ and based on standard Python type hints. Pytorch provides common deep learning models and implements a variety of adversarial example methods. OpenCV provides image pre-processing, reading and Writing. MongoDB is responsible for storing the test data. RESTful [18] style API is using for communication. The entire framework is shown in Fig. 1 and Fig. 2.

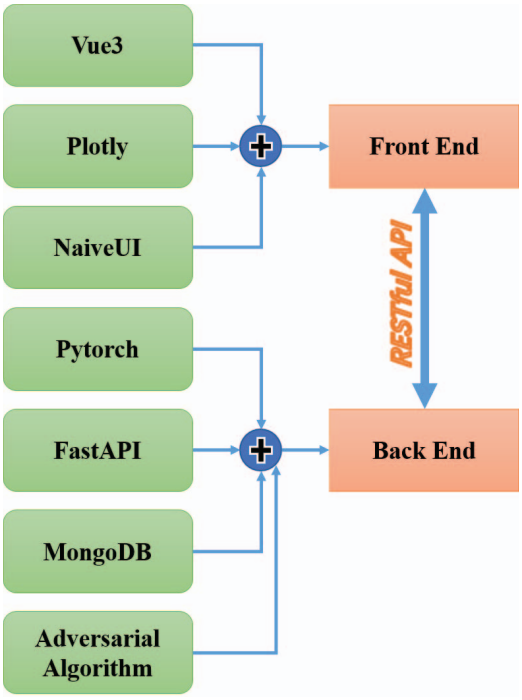

Fig. 1. platform structure.

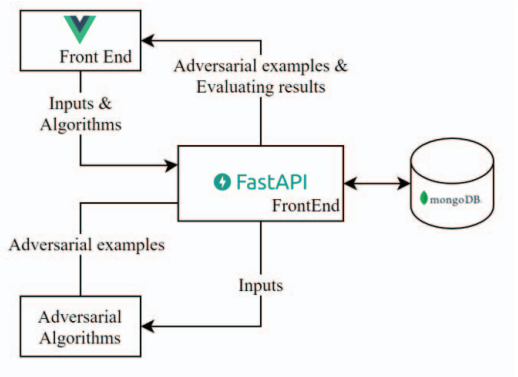

Fig. 2. platform data flow.

IV. Experiments

A. Comparison of algorithms

We selected 1000 images of different types (100 of each, 10 in total) from the ImageNet ILSVRC 2012 val dataset. All images are first scaled to $256 * 256$ and then $224 * 224$ due to model input restriction. ResNet18 and Inception v3 are selected as target classifier models and several attack algorithms (FGSM, FFGSM, RFGSM, MIFGSM, ERFGSM, BIM, PGD [19], TPGD [20], APGD [21], CW, DeepFool, OnePixel) will be performed. The measures include craft time of adversarial examples, failure rate of adversarial attacks, and differences between adversarial examples and their original images (using FID). The experimental hardware platform is WIN10 + Ryzen5 5600x + Nvidia RTX2060 O6G. Different adversarial attacks are performed on the models ResNet18 and Inception v3 respectively, and the time consumption and failure rate of the adversarial attacks of the two models are averaged to obtain the following results shown in Fig. 3. According to Fig. 3, most FGSM series algorithms run faster

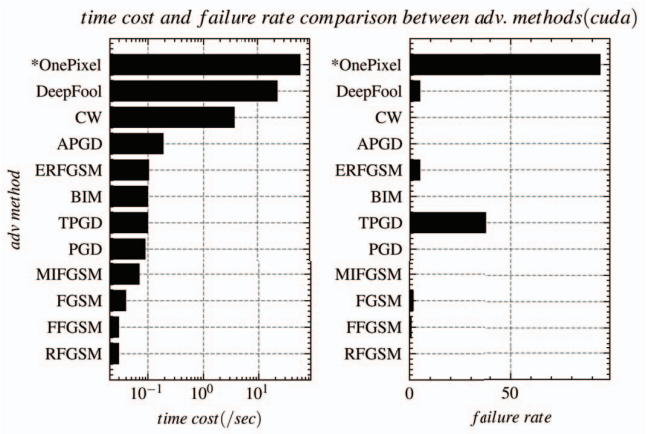

Fig. 3. time consumption and failure rate. * means timeout. All ϵ used in algorithms are set to $8/255$ and all α are set to $2/255$. The attack steps of all iterative methods are set to 7, while OnePixel and DeepFool are set to 50.

than other methods. OnePixel performs terrible on big images like $224*224$, which lead to timeout. TPGD has an Unexpected poor performance on failure rate. Our method slightly decrease the failure rate but still has a good performance on time cost.

To measure the quality of the adversarial samples generated by different generation algorithms, we adopt FID as our metric. Its definition is the formula below.

$$\mathrm{FID} = \|\mu_r - \mu_g\|^2 + \mathrm{Tr}\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r\Sigma_g\right)^{1/2}\right) \quad (11)$$

where $X_r \sim \mathcal{N}(\mu_r, \Sigma_r)$ and $X_g \sim \mathcal{N}(\mu_g, \Sigma_g)$ are the 2048-dimensional activations of the Inception-v3 pool3 layer for real and generated samples respectively. Lower FID is better, corresponding to more similar real and generated samples as measured by the distance between their activation distributions.

FID scores of different algorithms shows below. As shown

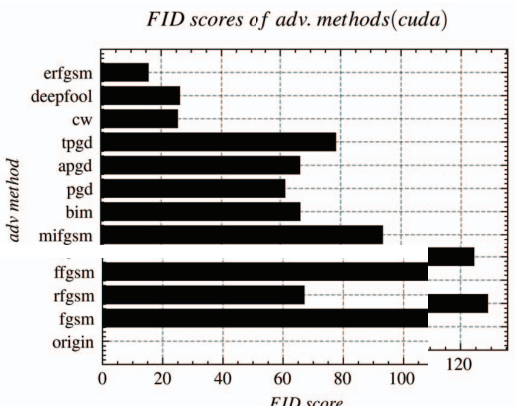

Fig. 4. FID scores. Origin means comparison with itself.

in Fig. 4, C&W and DeepFool have a lower score than the FGSM series algorithms. Traditional FGSM series attack just add global noise directly to images which causes a higher FID score. But Our method utilizes edge information and only add perturbations on edges in a image, causing a outstanding performance of FID score. A following image Fig. 5 illustrates the details of results that ERFGSM gives.

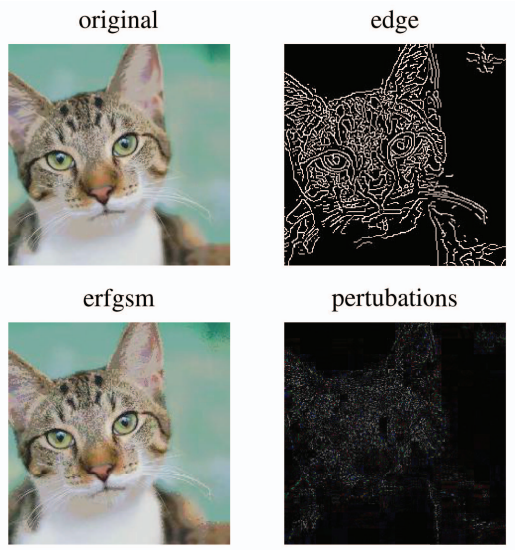

Fig. 5. ERFGSM results. The upper left corner is the original image with only resizing, and its right side is the edge information extracted by Canny operator. The lower left corner is an adversary crafted by our ERFGSM algorithm, and its right side is the visualization of perturbations that ERFGMS adds.

B. Adversarial Robustness Evaluation and Visualization

In order to reflect the adversarial robustness characteristics of the deep learning models from various aspects, we performed the adversarial attacks in Fig.3 except OnePixel for the four deep models ResNet18, ResNet50, ResNet101, and Vgg19. The different attacks were used as axes to form a radar

plot and the following figure was obtained. We also plotted a line graph of robustness as a function of ϵ perturbation parameter on Vgg19 and ResNet, which also indicates the robustness properties of different models.

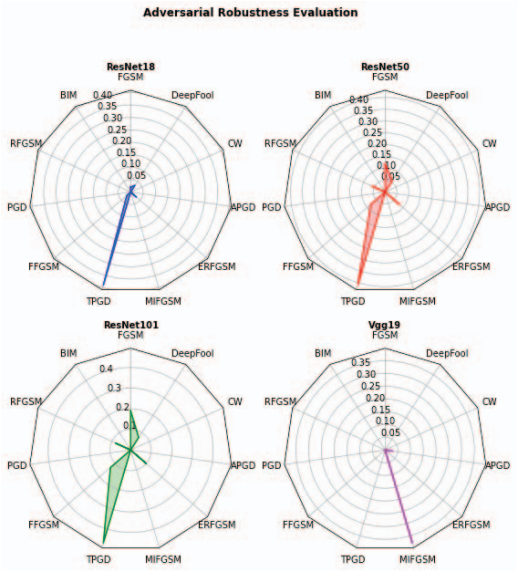

Fig. 6. Adversarial Robustness Evaluation results. The area under curve represents the adversarial robustness, ResNet101 performs better than others. The robustness characteristics of ResNet50 and ResNet101 are similar. Performances of ResNet18 and Vgg19 are quietly bad for most adversarail algorithms.

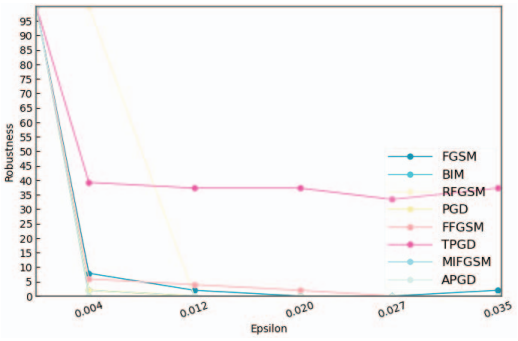

Fig. 7. Adversarial Robustness Evaluation results of VGGNet. The model is vulnerable even if tiny perturbaton is selected.

V. Conclusion

We designed a platform for comparing adversarial sample algorithms, evaluating deep learning models, and developed a new gradient and edge-based approach, ERFGSM, with satisfactory FID scores and aggressiveness performance. We also have visualised the experimental results to give a characterisation of the robustness of different models. Our next step will be integrating a target detection model evaluation procedure for adversarial robustness and develop an automated assessment process.

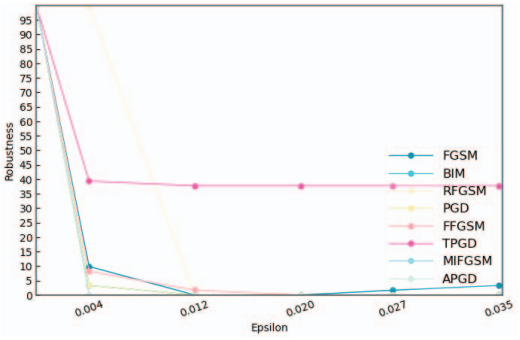

Fig. 8. Adversarial Robustness Evaluation results of ResNet.

References

[1] Papernot, Nicolas, et al. "Technical report on the cleverhans v2. 1.0 adversarial examples library." arXiv preprint arXiv:1610.00768 (2016).

[2] Rauber, Jonas, et al. "Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax." Journal of Open Source Software 5.53 (2020): 2607.

[3] Gavin Weiguang Ding, et al. "advertorch v0.1- An adversarial robustness toolbox based on pytorch." arXiv preprint arXiv: 1902.07623(2019)

[4] Nicolae, Maria-Irina, et al. "Adversarial Robustness Toolbox v1. 0.0." arXiv preprint arXiv:1807.01069 (2018).

[5] Goodman, Dou, et al. "Advbox: a toolbox to generate adversarial examples that fool neural networks." arXiv preprint arXiv:2001.05574 (2020).

[6] Mao, Xiaofeng, et al. "Composite Adversarial Attacks." arXiv preprint arXiv:2012.05434 (2020).

[7] Yinpeng Dong, et al. "Benchmarking Adversarial Robustness." arXiv preprint arXiv: 1912.11852 (2019).

[8] Xiang Ling, et al. "DEEPSEC: A Uniform Platform for Security Analysis of Deep Learning Model", 2019 IEEE Symposium on Security and Privacy.

[9] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in Neural Information Processing Systems (pp. 6626–6637).

[10] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." arXiv preprint arXiv:1412.6572 (2014).

[11] Kurakin, Alexey, Ian Goodfellow, and Samy Bengio. "Adversarial examples in the physical world." (2016).

[12] Tramèr, Florian, et al. "Ensemble adversarial training: Attacks and defenses." arXiv preprint arXiv:1705.07204 (2017).

[13] Dong, Yinpeng, et al. "Boosting adversarial attacks with momentum." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[14] Wong, Eric, Leslie Rice, and J. Zico Kolter. "Fast is better than free: Revisiting adversarial training." arXiv preprint arXiv:2001.03994 (2020).

[15] Carlini, Nicholas, and David Wagner. "Towards evaluating the robustness of neural networks." 2017 ieee symposium on security and privacy (sp). IEEE, 2017.

[16] Moosavi-Dezfooli, Seyed-Mohsen, Alhussein Fawzi, and Pascal Frossard. "Deepfool: a simple and accurate method to fool deep neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[17] Su, Jiawei, Danilo Vasconcellos Vargas, and Kouichi Sakurai. "One pixel attack for fooling deep neural networks." IEEE Transactions on Evolutionary Computation 23.5 (2019): 828-841.

[18] Richardson, Leonard, and Sam Ruby. RESTful web services. " O'Reilly Media, Inc.", 2008.

[19] Madry, Aleksander, et al. "Towards deep learning models resistant to adversarial attacks." arXiv preprint arXiv:1706.06083 (2017).

[20] Zhang, Hongyang, et al. "Theoretically principled trade-off between robustness and accuracy." International Conference on Machine Learning. PMLR, 2019.

[21] Croce, Francesco, and Matthias Hein. "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks." International conference on machine learning. PMLR, 2020.