The Hong Kong Polytechnic University

Department of Computing

COMP5523 Computer Vision and Image Processing

Research Paper Summary

Adversarial Attacks and countermeasures in Computer Vision, A survey

Student ID:	20068999G
Programme-Stream Code:	61030-FIT
Student Name:	ZHAO Shuyu
Submission Date:	April 26, 2021

Contents

Abstract

In recent years, deep learning shows significant progress in many domains and applied in many safety-critical tasks, like auto-driving vehicles. Thus DNN shows indeed accurate and stable, lots of researchers have doubts about whether the DNN model is robust enough. Imagine this situation, the model doesn't successfully recognize a STOP sign may leads to death of human beings. Many works in recent years have illustrated that DNN models are vulnerable to intentional attacks. By modifying a small amount of pixels that people cannot notice will lead to a catastrophic to machine judgement.

Offensive and defensive warfare starts in academia. Many attack approaches have been published, like Fast Gradient Sign Method (FGSM), Deep Fool, Universal Attack, and many defence strategies have been applied, e.g. Defensive Distillation, Feature squeezing.

In this survey, a comprehensive overview on well-known attacking and defensive methods will be showed. Their methodologies, merits, and/or limitations will be listed thematically.

1 Adversarial Attacks on DNNs

In 2013, Szegedy et al. first found that DNN model can be fooled by small perturbations, i.e. by modifying small amount of pixels in image can fool DNNs into misclassification. Fig 1 illustrates an example of adversarial attack on DNNs.

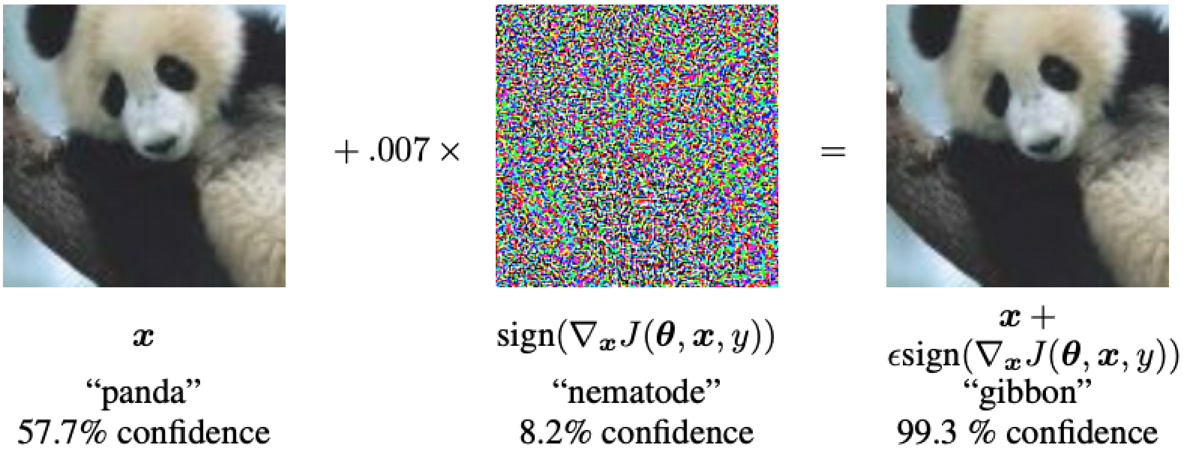

+ .007 ×
=
$\boldsymbol{x}$
"panda"
57.7% confidence
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

Figure 1: An illustration of adversarial attack on DNNs (Image Credits: [11])

Scholars have different opinions on why adversarial examples are exist. Szegedy et al. suggested that DNNs are not generalized well in low probability space[31]. Goodfellow et al. believed that it's enough to generate adversarial examples based on the linearity of high-dimensional space of DNNs[11]. Madry et al. proved that the capacity of DNNs is the factor of model robustness[20]. Other scholars argued that the vulnerability of DNNs is due to the decision boundary of models[8, 9]. In 2019, Ilyas et al. demonstrated that adversarial examples are not bugs but features. Some features of pictures are useful and robust features, called $Feat_r$, while useful but not robust features exist, called $Feat_{nr}$. Human beings use $Feat_r$ to classify images while have no sense about the existence of $Feat_{nr}$. To predict labels as accurate as possible, DNNs use $Feat_{nr}$ to help them, even be reliant on them. $Feat_{nr}$ indeed helps DNNs to achieve high accuracy and generalizability, however, it can be used to destroy DNNs as a double-edged sword[16].

Based on the adversaries' knowledge, adversarial attacks can be classified as white box attack and black box attack. For the former one, attackers have all the information of the attacked model, including model parameters, model architecture, gradient update rule, training dataset, etc., for the latter one, adversaries have no access to any information about targeted model except feeding inputs into it and observing outputs. Several attacking models in those two classes will be discussed in next sections. Tab 1 shows part of terms used in Sec 1 and Sec 2.

Term	Meaning
x, y	original data and responding label
x', t	perturbed data on x and responding label
δ	perturbation, $x + \delta = x'$
$\|.\|_p$	l_p norm
C	classifier, $C(x) = y$
F	DNN model outputs probability vector, $F(x) \in [0, 1]^m$
Z	the last layer before output layer, i.e. $F(x) = \text{softmax}(Z(x))$
$\mathcal{L}$	loss function, $\mathcal{L}(\theta, x', t)$ is equivalent to $\mathcal{L}(F(x), y)$
targeted attack	to fool DNNs to classify label of x' as a specific label
non-targeted attack	to fool DNNs to classify label of x' as any labels other than ground truth label
first-order attack	attacks only rely on first-order information

Table 1: Part of terms and meanings of adversarial attack and countermeasures, which used through entire paper

1.1 White box attack

Szegedy et al. proposed a method to generate minimized perturbations on images to fool DNNs. Firstly, they tried to find a minimal distorted adversarial example by solving Eq 1, which is a hard problem because that neural networks are non-convex. They ended up with an approximate function as Eq 2, where $\|x - x'\|_2^2$ is the similarity between original data and perturbed data, $\mathcal{L}(\theta, x', t)$ is the loss function that tries to find x' that has small loss to label t, in other words, C will have high probability to predict the label of x' as t. Box-constrained L-BFGS is used to solve Eq 2 by performing linear searching to find minimal constant $c > 0$ satisfies Eq 2.

$$\begin{aligned} \text{minimize} \quad & \|x - x'\|_2^2 \\ \text{s.t.} \quad & C(x') = t \,\text{and}\, x' \in [0, 1]^m \end{aligned} \tag{1}$$

$$\begin{aligned} \text{minimize} \quad & c\,\|x - x'\|_2^2 + \mathcal{L}(\theta, x', t) \\ \text{s.t.} \quad & x' \in [0, 1]^m \end{aligned} \tag{2}$$

Though Szegedy et al. successfully fooled DNNs, using linear searching to find constant c is time-consuming. Goodfellow et al. implemented an algorithm to fast generate adversarial perturbations that only takes one step to fool DNNs, called Fast Gradient Sign Method (FGSM)[11]. The adversarial example is generated by this formula: $x' = x + \epsilon \, \text{sign}\left(\nabla_x \mathcal{L}(\theta, x, t)\right)$ solved by Eq 3, where t is the targeted label. The idea of Eq 3 is searching a point that has minimal loss to label t in x's ϵ-neighbor ball, so that the model will predict the label of x' as t with high confidence. Usually, ϵ is small, like 0.007 in Fig 1. The higher value given to ϵ, the easier that noises can be detected. Also, Goodfellow et al. explained that why using sign(.) rather than gradient value. sign(.) can control the value of $\|x' - x\|_\infty$ while gradient value will increase or decrease it, resulting in making the noises undetectable. See Fig 2 as an illustration that how ϵ results in adversarial examples.

$$\begin{aligned} \text{minimize} \quad & \mathcal{L}\left(\theta, x', t\right) \\ \text{s. t.} \quad & \|x' - x\|_\infty \leq \epsilon \, \text{and} \, x' \in [0, 1]^m \end{aligned} \tag{3}$$

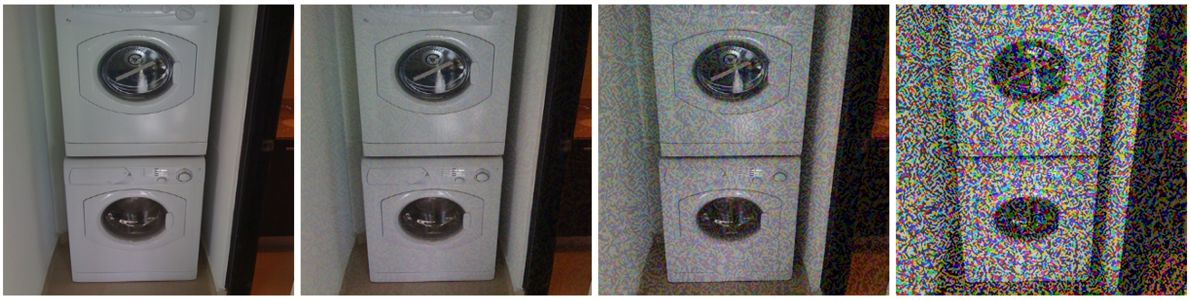

clean image: washer $\epsilon = 4$ $\epsilon = 16$ $\epsilon = 64$

Figure 2: Comparison of adversarial examples using different images ϵ (Images credits: [18])

One step attacking may not be successful all the time. Later in 2016, Kurakin et al. proposed an iterative version of FGSM, called Basic Iterative Method (BIM)[18]. In Eq 4, $Clip_{x,\epsilon}$ restricts changes on adversarial examples in each iteration. In metaphor, FGSM takes a huge step to generate adversaries while BIM takes several small steps to perturb orginal image. ϵ is defined as 1, number of iterations determined by min$(\epsilon + 4, 1.25\epsilon)$ heuristically.

$$\begin{aligned} & x^{t+1} = Clip_{x,\epsilon}\left(x^t + \alpha \, \text{sign}\left(\nabla_x \mathcal{L}\left(\theta, x^t, y\right)\right)\right) \\ & x^0 = x \end{aligned} \tag{4}$$

Furthermore, Kurakin et al. introduced a more powerful iterative adversarial attacking method, called Iterative Least-Likely Class Method (ILCM)[18]. ILCM is an targeted-attack method that leads DNNs to predict the label of x as the least-likely label y_{LL}. ILCM improves the successful attacking rate tremendously even with small ϵ.

$$\begin{aligned} & x^{t+1} = Clip_{x,\epsilon}\left(x^t - \alpha \, \text{sign}\left(\nabla_x \mathcal{L}\left(\theta, x^t, y_{LL}\right)\right)\right) \\ & x^0 = x \end{aligned} \tag{5}$$

Another iterative version of FGSM is introduced by Madry et al., called Projected Gradient Descent (PGD)[20]. There're three differences between PGD and BIM: 1) A random noise is added at initialization in PGD while BIM doesn't add. 2) The number of iterations are different between PGD and BIM. 3) PGD uses $projection$ on gradient value while BIM uses $Clip$. In practice, PGD have been proved as

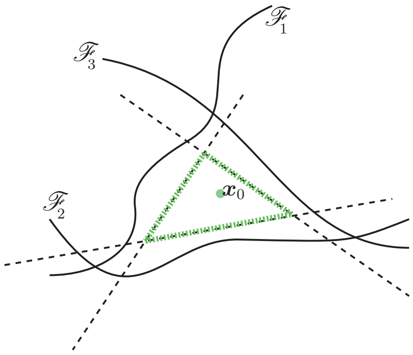


Figure 3: Illustration of decision bounaries of general classifier, $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3$ stand for hyperplane that separate classes. x_0 belonging to class 4 can across the decision boundary $\mathcal{F}_3$ by moving along the orthogonal vector ω to be classified as class 3. (Image credits: [23])

one of the most powerful first-order adversary.

MI-FGSM (Momentum iterative gradient-based method)[7] proposed by Dong et al. is another modified version of FGSM, which won the first places in NIPS 2017 Targeted and Non-targeted Adversarial Attack competition. By combining momentum mechanism and FGSM, Dong et al. made following contributions:

- Extending the MI-FGSM to L_2 norm bound while FGSM is only applicable on L_∞ norm.
- MI-FGSM can finish both non-targeted and targeted attack.
- MI-FGSM can also used in black-box attack domain using ensemble method.

During the iteration of MI-FGSM, momentum mechanism will accumulate the gradient value of loss function to stablize the direction of optimization and escape from local optima. The idea of ensemble method is that an adversarial example can successfully attack multiple models may possibly attack other models due to its intrinsic perturbation direction. In experiments, Dong et al. proposed a method called ensemble in logits, i.e. fusing logits activations $Z(.)$ of a set of DNNs to be attacked by MI-FGSM. Dong et al. work outperformed other methods using in white-box manner or black-box manner, raising security issues on robustness of DNNs.

Moosavi-Dezfooli et al. proposed a simple, fast algorithm to compute perturbations to fool DNNs named DeepFool[23]. DeepFool knows which direction to go and how far to go as FGSM only knows the former one. The general idea of DeepFool is to find a path to across decision bounary as shown in Fig 3. Eq 6 shows that how to attack a linear binary classifier: $f(x) = \omega^T x + b$. $-\frac{f(x_0)}{\|\omega\|_2}$ is the distance and $\frac{\omega}{\|\omega\|_2}$ is unit vector of gradient direction. In experiments, DeepFool is able to generate perturbations smaller than perturbations computed by FGSM with similar successful attacking rate.

$$\begin{aligned} r_*(x_0) &:= \arg\min \|r\|_2 \\ \text{s.t. } &\operatorname{sign}(f(x_0 + r)) \neq \operatorname{sign}(f(x_0)) \\ &= -\frac{f(x_0)}{\|\omega\|_2^2}\omega = -\frac{f(x_0)}{\|\omega\|_2} * \frac{\omega}{\|\omega\|_2} \end{aligned} \tag{6}$$

After the introducing of attacking methods like FGSM, a countermeasure called defensive distillation[25] was published by Papernot et al. in 2016. Later in 2017, a more powerful attacking method that can break defensive distillation effectively was proposed by Carlini and Wagner. The C&W attacks[3] is aimed to solve the similar problem as L-BFGS proposed: Eq 1 by introducing another method (Eq 7) instead.

$$\begin{aligned} \text{minimize} \quad & \left\|x-x'\right\|_p^2+c * f\left(x', t\right) \\ \text{s.t.} \quad & x' \in[0,1]^m \end{aligned} \tag{7}$$

In Eq 7, p is the type of distance norms, including L_0, L_2, L_∞, $f(.)$ is the objective function to solve Eq 1, $C(x') = t$ iff $f(x', t) \leq 0$. $f(x', t) = \max(\left(\max_{i \neq t}\left(Z\left(x'\right)_i\right)-Z\left(x'\right)_t\right), 0)$ is selected from 7 possible choices by experiments. By modified binary searching a the smallest c that $f(x', t) \leq 0$, an optimum x' can be found. C&W can find the minimal perturbation more efficient than L-BFGS. Thus, Carlini and Wagner claimed that C&W attack is one of the most powerful attack method that can break defensive mechanisms easily.

Carlini and Wagner also argued that targeted attack is more powerful than non-targeted attack since non-targeted attack is a simplify, less accurate version of targeted attack[3].

Other than attack methods mentioned above, Moosavi-Dezfooli et al. tried to find a perturbation to let DNNs misclassify almost every images' labels[22]. Moosavi-Dezfooli et al. intended to find a perturbation δ that satisfied constraints in Eq 8, where ϵ controls difficulty to perceive a perturbation, σ stands for the desirede fooling rate for samples in μ. The first constraint aims to make the perturbation as small as possible while the second constraint intends to make the fooling rate as high as possible. Moosavi-Dezfooli et al. claimed that their work can find the geometric corrleations between decision boundaries in high dimension of classifiers, in other words, their algorithm tries to perturb at the softmax layer. In experiments, universal attack achieves over 80% fooling rate on several benchmarks and around 40% fooling rate in cross-model adversarial attacking task.

$$\begin{aligned} &1. \quad \|\delta\|_p \leq \epsilon \\ &2. \quad \underset{x \sim \mu}{\mathbb{P}}(C(x+\delta) \neq C(x)) \leq 1-\sigma \end{aligned} \tag{8}$$

1.2 Black box attack

Papernot et al. introduced black-box attack on DNNs at the first time called Substitute model[24]. This attack is highly rely on intrinsic characteristic of DNNs: transferability. A perturbation fooled one model will have high probability that it can mislead another model with similar architecture. In this way, adversaries can attack a veiled model by attacking another model.

Training a substitute model contains five steps as shown in Fig 4. Firstly, collecting a small scale of dataset S_0, the number of samples is relatively small compared with the dataset that oracle model, or attacked model O used. Then, choosing the architecture of substitute model F empirically. Next, Querying the samples S_ρ in S_0 to O and getting corresponding label $\tilde{O(S_\rho)}$. The training process of substitute model is similar to normal machine learning task. Finally, using Jacobian-based Dataset Augmentation to generate bigger dataset $S_{\rho+1}$ using Eq 9, where $\left(J_F[\tilde{O}(\vec{x})]\right)$ is jacobian matrix of the substitute model

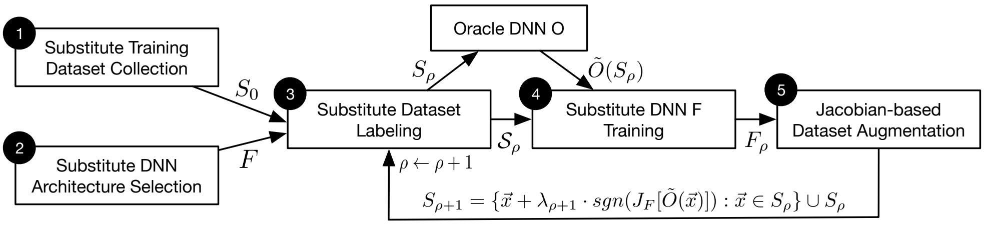

1
Substitute Training Dataset Collection
2
Substitute DNN Architecture Selection
S_0
F
3
Substitute Dataset Labeling
Oracle DNN O
S_ρ
$\tilde{O}(S_\rho)$
4
Substitute DNN F Training
$\mathcal{S}_\rho$
5
Jacobian-based Dataset Augmentation
F_ρ
$\rho \leftarrow \rho + 1$
$S_{\rho+1} = \{\vec{x} + \lambda_{\rho+1} \cdot sgn(J_F[\tilde{O}(\vec{x})]) : \vec{x} \in S_\rho\} \cup S_\rho$

Figure 4: Training process of building a substitute model (Image credits: [24])

corresponding to labels queried from O, λ is the step size in sensitive direction determined by jacobian matrix. Step 3,4,5 over and over again till finishing. Papernot et al. used FGSM to attack substitute model to optimize gradient direction and then applied on oracle model.

$$S_{\rho+1} = \left\{\vec{x} + \lambda \cdot \text{sign}\left(J_F[\tilde{O}(\vec{x})]\right) : \vec{x} \in S_\rho\right\} \cup S_\rho \tag{9}$$

Chen et al. introduced a black-box attack method without using substitute model called Zeroth Order Optimization (ZOO). Besides the information that substitute model method can get, ZOO also requires the prediction confidence on outputs from DNNs. ZOO can predict the gradient information to attack victim model by observing the changes of prediction confidence $F(x)$ in Eq 10, where h is a small constant (0.0001 in experiments) and e_i is a standard basis vector that $i_t h$ component is set to be 1. In experiments, ZOO achieves similar success rate in black-box manner as C&W in white-box manner and outperforms substitute model attack with a large margin.

$$\frac{\partial F(x)}{\partial x_i} \approx \frac{F\left(x + he_i\right) - F\left(x - he_i\right)}{2h} \tag{10}$$

Above methods mainly focus on attacking on L_2 or L_∞ norm while One-pixel Attack[30] is used on L_0 norm. Su et al. considered an extreme circumstance that by only modifying one pixel to fool DNNs. Su et al. computed the perturbation using Differential Evolution (DE)[5]. One-pixel attack achieves high fooling rate on non-targeted attack and relatively small success rate on targeted attack. In experiments of Su et al., one-pixel attack get 31.4% success rate on VGG16[28] by modifying 1 pixel while FGSM get 100% success rate on VGG16 by changing 1024 pixels.

Dong et al. made a statement that why some efficient white-box attacking mechanisms perform low efficacy using on black-box attacking. The reason attributes to the trade-off between attacking accuracy and model transferability. Iterative attacking methods and optimization based methods can perform greatly on white-box task while losing their transferability. One step attacking method remain their transferability while have bad performance on white-box attack[7].

In summary, comparisons between 12 adversarial attacking approaches are listed in Tab 2.

2 Countermeasures Against Adversaries on DNNs

Basically, there are three main approaches using as countermeasures against adversarial attack on DNNs.

	targeted/non-targeted	white/black box	perturbation norm	iterative
L-BFGS[31]	non-targeted	white box	L_2	No
FGSM[11]	targeted	white box	L_∞	No
BIM[18]	non-targeted	white box	L_∞	Yes
ILCM[18]	targeted	white box	L_∞	Yes
PGD[20]	non-targeted	white box	L_∞	Yes
MI-FGSM[7]	targeted, non-targeted	white box, black box	L_2, L_∞	Yes
DeepFool[23]	non-targeted	white box	L_2, L_∞	Yes
C&W [3]	targeted	white box	L_0, L_2, L_∞	Yes
Universal attack[22]	targeted	white box	L_2, L_∞	Yes
Substitute model[24]	targeted	black box	L_∞	No
ZOO[4]	targeted, non-targeted	black box	L_2	Yes
One Pixel[30]	non-targeted	black box	L_0	Yes

Table 2: Overall comparisons between adversarial attacking methods

- Adversarial examples detection: This approach is aimed to set a classifier to distinguish natural data and adversarial data. Only natural data will be past to the DNNs afterwards.
- Gradient protection: This method tries to hide gradient informatiion to misguide adversaries.
- Robustness optimization: Unlike using tricks mentioned above, robustness optimization intends to reduce the intrinsic vulnerability of DNNs by several methods.

2.1 Adversarial examples detection

In early works, some statistic based methods are proposed to distinguish atural data and adversarial data. Hendrycks and Gimpel found that adversarial examples have higher weight on higher principle components while benign examples don't[14]. In this way, adversarial examples will be cut out from the dataset by PCA. Maximum Mean Discrepancy (MMD) test[12] are used by Grosse et al. to classify adversarial examples and benign examples. The methodology of Grosse et al. based on the fundamental theorem: adversaries only know the learned feature distribution from trained DNN model while natural data contain original feature distribution. This difference can be used to distinguish them[13].

Gong et al. built a sub-DNN model to classify adversarial examples and benign examples. The idea is to leverage the subtle difference between victim examples and natural examples to distinguish them[10]. In experiments, it shows over 99% accuracy to separate those two type of examples. Gong et al. mentioned the weakness of this approach: it has poor generalization to different attacking methods, and is sensitive to the setting of parameters on attacking methods.

Xu et al. proposed a feature squeezing method to detect victim data by reducing color bit depth of pixels and spatial smoothing. Prediction results on natural data applied with feature squeezing will not change while a huge difference will occur on the predicted label of victim data after feature squeezing[35]. This approach can be used as a complementary to other defences.

In 2017, Work published by Carlini and Wagner bypassed ten adversarial example detection defence methods, including some methods mentioned above[2]. In 2018, Sharma and Chen bypassed feature squeezing methods by increasing adversary strength[27]. Carlini and Wagner claimed that some intrinsic features of adversarial examples are actually not underlying features. It's hard to find those features to distinguish them from natural data[2].

2.2 Gradient protection

Papernot et al. proposed a defensive distillation method[25] to protect DNNs against adversarial attacking using distillation mechanism introduce by Hinton et al. The idea of distillation is to transfer knowledge from complex network to small, simple network[15]. Papernot et al. used distillation mechanism to smooth the model so that small variants on inputs will not lead to huge difference on outputs. The trainiing process of defensive distillation is listed below, an illustration figure is shown in Fig 5:

1. Training a model F on training set (X, Y), at softmax temperature T, outputting a probability vector $F(X)$ contains
2. Training a distilled model F^d on dataset $(X, F(X))$ at softmax temperature T
3. Using distilled model F^d at softmax temperature 1 to predict on test set.

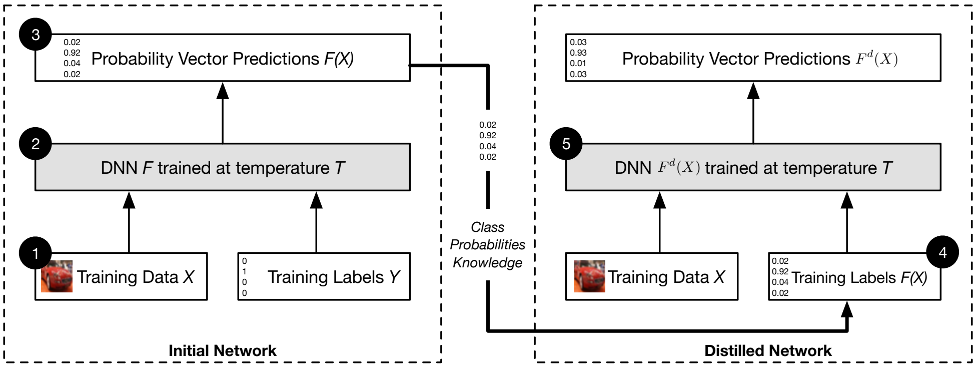
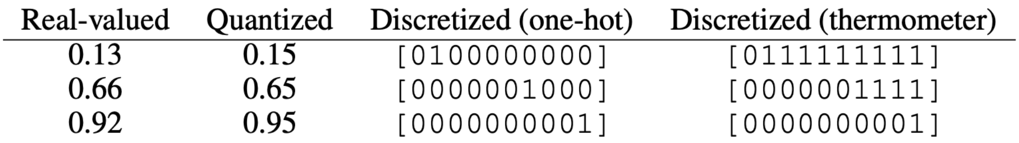


Figure 5: Overview of defensive distillation (Image credits: [25])

Defensive distillation can reduce the fooling rate of adversarial attacking tremendously while remain high accuracy of classifier. Defensive distillation have been proved to be effective countermeasure against adversarial attack until the Carlini and Wagner work, which introduced C&W attack method[3] that successfully bypassed it. Carlini and Wagner also mentioned that why defensive distillation works in their paper[3]. When softmax temperature set to T, the $Z(.)$ output will be T times larger than original one. In the final probability vector $F(X)$, the probability of desired label will be extremely close to 1 while others will be close to 0. In this way, adversaries cannot find the gradient rules from a distilled model.

Buckman et al. proposed thermometer encoding method[1] as a defensive method against adversaries. thermometer encoding intends to break the intrinsic characteristic of DNNs: linear extrapolation. The inputs will be preprocessed by a highly non-linear function to discrete it. Fig 6 shows several examples of thermometer encoding.



Real-valued	Quantized	Discretized (one-hot)	Discretized (thermometer)
0.13	0.15	[0100000000]	[0111111111]
0.66	0.65	[0000001000]	[0000001111]
0.92	0.95	[0000000001]	[0000000001]

Figure 6: Examples of thermometer encoding, the quantized columns refer to the work published by Xu et al.[35] (Image credits: [1])

2.3 Robustness optimization

Methods mentioned in 2.1 and 2.2 have been proved to be vulnerable to more powerful adversaries by scholars. For instance, Tramer et al. published a detailed paper to show how to bypass 13 defensive methods introduced in papers that are published on well-known conferences[32]. Those methods can only mislead the adversaries but the DNNs remain its intrinsic vulnerability. Hence, scholars tried to find an ultimate solution to tackle adversarial attacks.

Madry et al. pointed out that the idea of adversarial attack is to find a point in x's ϵ neighbor ball that achieves high loss. If the loss of the point adversaries find is relatively small, the DNNs are protected[20]. In formal, Madry et al. summarized the idea into a formula, see Eq 11, where the max$(.)$ problem is the goal of adversaries, the min$(.)$ problem is the countermeasure.

$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y)\right] \tag{11}$$

For the inner non-concave problem, Madry et al. choosed random points around input data's ϵ neighbor ball applied with PGD, for the outer non-convex problem, Madry et al. used SGD to consistently reduce the loss value found by PGD[20]. In experiments, this approach achieves great performance on MNIST[19] while relatively poor results on CIFAR-10[17]. There're almost infinite points around input data's ϵ neighbor ball, selecting random points may only be effective on small dataset. Hence, several scholars have worked on how to reduce the needed computation power on Madry et al. work. In [26], Shafahi et al. reduced the computing cost by reusing the gradient information when parameters are needed to update. In [36], Zhang et al. used pontryagin's maximal principle to accelerate the defences building on DNNs.

Another idea focus on mixing up adversarial examples and natural examples together to train a DNN model. In [11], Goodfellow et al. suggested to add adversarial examples generated by FGSM to the training dataset, which will increase the robustness of DNNs against attacks from FGSM. In 2020, Wong et al. improved Goodfellow et al. work so that by adding FGSM examples into training set will achieve similar defensive results compared with PGD[34]. The idea of Wong et al. is adding random initialization points, which is surprisingly simple but achieving significant results. Also, Wong et al. discovered that using cyclic learning rates[29], mixed-precision training[21], etc. can accelerate the FGSM adversarial training.

Tramèr et al. proposed an ensemble adversarial training (EAT) method to defend adversarial attacks[33]. Firstly, several DNN models are selected, noted as F_1, F_2, F_3, which have different parameter setting compared with the robost model F. Then, FGSM is used to attack F_1, F_2, F_3 to generate corresponding adversarial example, $x_1^{adv}, x_2^{adv}, x_3^{adv}$. Because of the transferability between different DNNs, F is vulnerable to $x_1^{adv}, x_2^{adv}, x_3^{adv}$. In other words, $x_1^{adv}, x_2^{adv}, x_3^{adv}$ can be treated as adversarial examples on F. The robustness of F will be improved by adding them into training set of F. In experiments of Tramèr et al., the model equipped with EAT can successfully defend single-step perturbation like FGSM and black-box attacks on ImageNet[6] dataset.

3 Conclusion

In this survey, we give a systematically, comprehensive overview on adversarial attacking and countermeasures on DNNs, especially in computer vision domain. In fact, there're similar adversaries on domains like NLP, Graph. Based on the historic data, the state-of-the-art adversarial attacking methods will be successfully counteracted by defensive methods, which will be bypassed by other adversaries after several months/years. The equation 11 can be kindly treated as the ultimate answer in this welfare. We wish that our work may bring some sparks to scholars to encourage a huge step in this field.

References

[1] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. “Thermometer encoding: One hot way to resist adversarial examples”. In: *International Conference on Learning Representations*. 2018 (cit. on pp. 8, 9).

[2] Nicholas Carlini and David Wagner. “Adversarial examples are not easily detected: Bypassing ten detection methods”. In: *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. 2017, pp. 3–14 (cit. on pp. 7, 8).

[3] Nicholas Carlini and David Wagner. “Towards evaluating the robustness of neural networks”. In: *2017 ieee symposium on security and privacy (sp)*. IEEE. 2017, pp. 39–57 (cit. on pp. 5, 7, 8).

[4] 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: *Proceedings of the 10th ACM workshop on artificial intelligence and security*. 2017, pp. 15–26 (cit. on pp. 6, 7).

[5] Swagatam Das and Ponnuthurai Nagaratnam Suganthan. “Differential evolution: A survey of the state-of-the-art”. In: *IEEE transactions on evolutionary computation* 15.1 (2010), pp. 4–31 (cit. on p. 6).

[6] Jia Deng, Wei Dong, Richard Socher, et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255 (cit. on p. 10).

[7] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, et al. “Boosting adversarial attacks with momentum”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 9185–9193 (cit. on pp. 4, 6, 7).

[8] Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. “Analysis of classifiers’ robustness to adversarial perturbations”. In: *Machine Learning* 107.3 (2018), pp. 481–508 (cit. on p. 1).

[9] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. “Robustness of classifiers: from adversarial to random noise”. In: *arXiv preprint arXiv:1608.08967* (2016) (cit. on p. 1).

[10] Zhitao Gong, Wenlu Wang, and Wei-Shinn Ku. “Adversarial and clean data are not twins”. In: *arXiv preprint arXiv:1704.04960* (2017) (cit. on p. 7).

[11] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and harnessing adversarial examples”. In: *arXiv preprint arXiv:1412.6572* (2014) (cit. on pp. 1–3, 7, 9).

[12] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. “A kernel two-sample test”. In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 723–773 (cit. on p. 7).

[13] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. “On the (statistical) detection of adversarial examples”. In: *arXiv preprint arXiv:1702.06280* (2017) (cit. on p. 7).

[14] Dan Hendrycks and Kevin Gimpel. “Early methods for detecting adversarial images”. In: *arXiv preprint arXiv:1608.00530* (2016) (cit. on p. 7).

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the knowledge in a neural network”. In: *arXiv preprint arXiv:1503.02531* (2015) (cit. on p. 8).

[16] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, et al. “Adversarial examples are not bugs, they are features”. In: *arXiv preprint arXiv:1905.02175* (2019) (cit. on pp. 1, 2).

[17] Alex Krizhevsky, Geoffrey Hinton, et al. “Learning multiple layers of features from tiny images”. In: (2009) (cit. on p. 9).

[18] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. *Adversarial examples in the physical world*. 2016 (cit. on pp. 3, 7).

[19] Yann LeCun. “The MNIST database of handwritten digits”. In: *http://yann. lecun. com/exdb/mnist/* (1998) (cit. on p. 9).

[20] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. “Towards deep learning models resistant to adversarial attacks”. In: *arXiv preprint arXiv:1706.06083* (2017) (cit. on pp. 1, 3, 7, 9).

[21] Paulius Micikevicius, Sharan Narang, Jonah Alben, et al. “Mixed precision training”. In: *arXiv preprint arXiv:1710.03740* (2017) (cit. on p. 9).

[22] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. “Universal adversarial perturbations”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1765–1773 (cit. on pp. 5, 7).

[23] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. “Deepfool: a simple and accurate method to fool deep neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2574–2582 (cit. on pp. 4, 7).

[24] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, et al. “Practical black-box attacks against machine learning”. In: *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. 2017, pp. 506–519 (cit. on pp. 5–7).

[25] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. “Distillation as a defense to adversarial perturbations against deep neural networks”. In: *2016 IEEE symposium on security and privacy (SP)*. IEEE. 2016, pp. 582–597 (cit. on pp. 5, 8).

[26] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, et al. “Adversarial training for free!” In: *arXiv preprint arXiv:1904.12843* (2019) (cit. on p. 9).

[27] Yash Sharma and Pin-Yu Chen. “Bypassing feature squeezing by increasing adversary strength”. In: *arXiv preprint arXiv:1803.09868* (2018) (cit. on pp. 7, 8).

[28] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on p. 6).

[29] Leslie N Smith and Nicholay Topin. “Super-convergence: Very fast training of residual networks using large learning rates”. In: (2018) (cit. on p. 9).

[30] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. “One pixel attack for fooling deep neural networks”. In: *IEEE Transactions on Evolutionary Computation* 23.5 (2019), pp. 828–841 (cit. on pp. 6, 7).

[31] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, et al. “Intriguing properties of neural networks”. In: *arXiv preprint arXiv:1312.6199* (2013) (cit. on pp. 1, 2, 7).

[32] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. “On adaptive attacks to adversarial example defenses”. In: *arXiv preprint arXiv:2002.08347* (2020) (cit. on p. 9).

[33] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, et al. “Ensemble adversarial training: Attacks and defenses”. In: *arXiv preprint arXiv:1705.07204* (2017) (cit. on pp. 9, 10).

[34] Eric Wong, Leslie Rice, and J Zico Kolter. “Fast is better than free: Revisiting adversarial training”. In: *arXiv preprint arXiv:2001.03994* (2020) (cit. on p. 9).

[35] Weilin Xu, David Evans, and Yanjun Qi. “Feature squeezing: Detecting adversarial examples in deep neural networks”. In: *arXiv preprint arXiv:1704.01155* (2017) (cit. on pp. 7, 9).

[36] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. “You only propagate once: Accelerating adversarial training via maximal principle”. In: *arXiv preprint arXiv:1905.00877* (2019) (cit. on p. 9).