Adversarial Attacks on Neural Network with Batch Dimensions Perturbation and Manhattan-Distance Constraints

Dahui Liu, Zihan Song, Siyu Ren, and Siyu Xia

***Abstract*—In recent years, with the development of deep learning technology, neural networks play an increasingly important role in more and more fields. However, research shows that neural networks are vulnerable to the attack of adversarial examples. The purpose of this paper is to study the principle of adversarial examples generation and propose a new method of generating adversarial examples. Compared with existed methods, our method achieves better deception rate and perturbs less pixels of images. During an epoch in batch dimension iteration, multiple pixels are perturbed while Manhattan-Distance constraints are added to them. Our algorithm performs well in experiments. Compared with Carlini-Wagner method, only 60 more dimensions are perturbed, which indicates that the computation cost of our algorithm is completely acceptable. Besides, compared with FGSM algorithm, the deception rate increases by 12% while the generation times of them are almost same.**

***Index Terms*—Adversarial attacks, deep learning, adversarial examples, distance constraints.**

I. Introduction

The extensive application of deep learning [1] in fields of national defense, finance [2], medical treatment [3], agriculture [4], transportation [5] has benefited the society greatly. However, some researches reveal the vulnerability of deep neural networks under the attack of adversarial examples [6]. Facing the security problem underlying neural networks, the study of adversarial examples and corresponding anti-attack methods is of great significance.

Adversarial example is a kind of input which has been designedly adjusted based on normal input to induce neural networks to make wrong inference. In the field of image recognition, an adversarial example can be considered as an input image whose pixels are applied subtle changes that can be hardly perceived by human eyes. Though the changes are slight, the adversarial examples are given the ability to misguide neural networks [7].

As shown in Fig. 1, the dotted line represents the inference function of a given model, which has divided test samples into two parts. Once some subtle changes have been added into an example (the light red one) near the curve, which has made it cross the curve and become an adversarial example, the existed inference function could no longer classify it correctly.

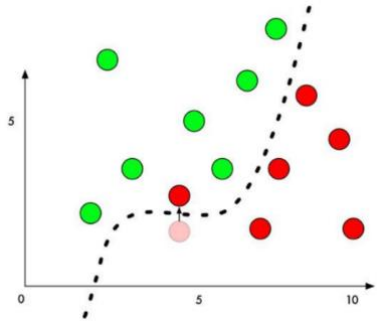


Fig. 1. Error classification caused by adversarial examples.

This defect of neural networks will bring a lot of potential safety hazards. Therefore, the study of adversarial examples is of great significance in the field of neural networks security [8].

A traditional pattern of adversarial example generation is to add perturbations on a fixed number of dimensions of input vector. A tradeoff underlying the pattern is that more disturbed dimensions bring higher attack success rate but less disturbed dimensions cost less computation. Therefore, inevitably, traditional methods [9] show various weaknesses, such as bad performance on non-linear decision function, complex calculation and low speed.

To avoid this tradeoff and overcome the shortcomings of existed algorithms, a new algorithm for generating adversarial examples is proposed in this paper, a Manhattan-Distance Constraint algorithm is proposed and added into the process of batch dimension iteration, which limits the number disturbed dimensions and pixels. The algorithm is able to ensure the attack success rate while costs less computation. Also, the adversarial samples yielded are not easily perceived by human eyes.

In this paper, we will describe the details of our methods and according experiment results on MNIST and CIFAR-10 data sets. Comparing with Fast Gradient Sign Method (FGSM) and Carlini-Wagner (C&W) method, our method shows good attack success rate, better speed and less dimensions of perturbation.

Manuscript received July 20, 2020; revised March 12, 2021. This work was supported by the National Natural Science Foundation of China under Grant 61671151 and Grant 61728103.

D. Liu is with the Key Laboratory of Measurement and Control of CSE, Ministry of Education, Southeast University, Nanjing 210096, China (e-mail: 11973053@qq.com).

Z. Song is with the College of Electronic and Optical Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210023, China (e-mail: szian@hotmail.com).

S. Ren is with the Oversea Education College, Nanjing University of Posts and Telecommunications, Nanjing 210023, China (e-mail: Rensy1121@outlook.com).

S. Xia is with the Key Laboratory of Measurement and Control of CSE, Ministry of Education, Southeast University, Nanjing 210096, China, and also with the School of Automation, Southeast University, Nanjing 210096, China (Corresponding author; e-mail: xia081@gmail.com).

doi: 10.18178/ijmlc.2022.12.1.1073

II. Related Work

In recent years, anti-attack has become a hot topic in the field of artificial intelligence. With the deepening of research, the methods of anti-attack can be roughly divided into the following categories.

A. Large Broy-den Fletcher Goldforb Shanno Method

The concept of adversarial examples was first proposed by Szegedy [10]. At the same time, he also put forward the first method of manufacturing adversarial examples: Large Broy-den Fletcher Goldforb Shanno Method (L-BFGS). Its main idea is to find a minimum perturbation term r and add it to the original example *x*. Thus, an adversarial example x' can be produced. Suppose the output label corresponding to sample *x* is *t*, then the output label corresponding to sample x' is t'. And it must satisfy condition $t' \neq t'$. Therefore, it can be expressed by the following formula:

$$f(x+r) = t' \tag{1}$$

$$x + r \in [0,1]^m \tag{2}$$

The problem is solved by L-optimization algorithm and the minimum perturbation term r can be obtained finally. By adding r to the initial input example *x*, the adversarial example x' can be obtained.

B. Fast Gradient Sign Method

Fast Gradient Sign Method (FGSM) algorithm was first proposed by Good Fellow in 2014 [11]. The origin of this method can be traced back to the earliest gradient descent algorithm in pattern recognition. The idea is to change the predictive probability of the classifier by adding a perturbation η to a pure sample *x*, or to make the value of loss as large as possible. That is to say, each iteration increases the error along the opposite direction of gradient, and then achieves the effect of error classification. It is worth mentioning that the disturbance itself should be limited to a person's eyes and cannot be detected, or produce greater damage to the pure sample. Therefore, norm restrictions are usually imposed on η. It can be expressed by the following formula:

$$\eta = \varepsilon * \text{sign}(\nabla_x J(\theta_x, y)) \tag{3}$$

where θ is the parameters of a model, *x* is the input to the model, *y* is the targets associated with *x* (for machine learning tasks that have targets) and $J(\theta_x, y)$ is the cost used to train the neural network. As the name implies, the advantage of this method is that the speed of constructing countermeasure samples is very fast. When the decision function is linear, it performs well. The biggest problem is that the ε is manually chosen. So, when the decision function is not linear, FGSM will not work well.

C. Jacobian-Based Saliency Map Attack

Jacobian-based Saliency Map Attack (JSMA) [12] constructs adversarial examples by adding a limited number of pixels to the original image. It is a targeted attack method. Given target category $y_{target} \neq y$, it chooses the most effective two pixels for iteration each time until the target class is reached. The perturbed pixels are selected according to their saliency mapping:

$$S(x,y)[i] = \begin{cases} 0, if \ \dfrac{\partial F_{ytarget}(x)}{\partial x_i} < 0 \,\|\, \sum y' \neq y_{target} \dfrac{\partial F_{y'}(x)}{\partial x_i} > 0 \\ \left|\dfrac{\partial F_{ytarget}(x)}{\partial x_i}\right| (\sum y' \neq y_{target} \dfrac{F_{y'}(x)}{\partial x_i}), otherwise \end{cases} \tag{4}$$

In each iteration, the pair of pixels (i, j) with the largest $S(x,y)[i] + S(x,y)[j]$ is selected and the same ε is modified. Repeat the process until $F(x_{adv}) = y_{target}$. JSMA method has high success rate, but the calculation of saliency mapping is complicated.

D. Deep Fool

Deep Fool was first proposed by S.Moosavi Dezfuli and P. Frossard [13]. Deep Fool algorithm is based on the idea of FGSM. It generates adversarial examples by iteration. At the same time, a method for quantitatively expressing the robustness of the network to adversarial examples by using the average disturbance amplitude of adversarial examples is proposed, as shown in formula (5)

$$\rho_{\text{adv}}(f) = \frac{1}{T} \sum \frac{\left\| r\hat{}(x) \right\|_2}{\left\| x \right\|_2} \tag{5}$$

where $\rho_{\text{adv}}(f)$ represents the average robustness, *T* represents the whole test set, and $r(x)$ is the smallest perturbation to generate adversarial examples. Deep Fool algorithm can produce deception effect similar to FGSM while exerting smaller perturbation on examples. However, this method makes the perturbation distributed in almost every dimension of the sample. Not only is the calculation heavy, but also the deception effect is not very good.

E. Carlini-Wagner

Carlini-Wagner method (C&W) [14] is an attack method using optimization method to generate adversarial samples. Its optimization objective is:

$$\begin{aligned} &\min \| \delta \|_2^2 + \alpha * \ell(x+\delta) \\ &s.t. x + \delta \in [0,1]^d \end{aligned} \tag{6}$$

where $\ell(x+\delta)$ represents the constraint of error classification, α is the equilibrium parameter between disturbance and resistance strength. It is also an iterative attack algorithm. But it has too many optimization parameters, which leads to excessive calculation and slow speed.

III. Our Method

This section will introduce our method in detail.

A. Gradient Descent

For an input sample x, the connection weights of its dimensions vary. Therefore, changing values of different dimension always yields different results [15]. The gradient of values of different dimensions can indicate that what results would be yielded for us, which can be explained with an output saliency map.

As shown in Fig 2, for an input *x*, its corresponding classification is y and suppose his evaluation function is $F(x)$, the ordinates represent the gradient vector of the input *X*. As can be seen from Fig 2, for a point, if its partial derivatives in some directions are positive, that is $(\partial F(x\_(i,j,k)))/(\partial x\_(i,j,k)) > 0$, it shows that the value of output discriminant function *F* increases with the increase of *X*, Conversely, if the value of partial derivatives in some directions is negative, that is $(\partial F(x\_(i,j,k)))/(\partial x\_(i,j,k)) < 0$, it shows that the discriminant function decreases with the increase of *X*. When the partial derivative is zero, *F* does not change.

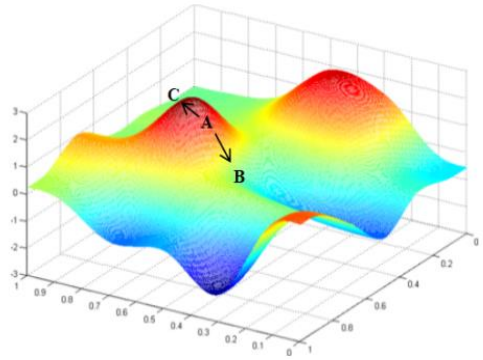


Fig. 2. Three-dimensional view of gradient descent.

As shown in Fig. 2, the gradient value of direction AC is the largest and rises fastest along this direction, while the gradient value of direction AB is the smallest and falls fastest along this direction.

Based on the above ideas, this paper proposes a new idea, that is, to select a part of the dimension which has the most obvious influence on discriminant function *F* and add disturbance to it. Batch dimension perturbation algorithm is not easy to fall into local minimum, so its deception ability is stronger. And the upper limit of pixel perturbation is limited by Manhattan-Distance to improve the robustness of the algorithm.

B. Iteration with Batch Dimension and Manhattan-Distance Constraints

The idea of this algorithm is: First, the gradient matrix of the output is calculated by calculating the current value of the input. A part of the maximum dimension of the gradient is selected to add perturbations. After a lot of experiments, the performance of the algorithm is the best when the size of maximum dimension is 3. So, we select three dimensions for iteration each time. Then perturbation is added to each selected pixel. The size of the perturbation is calculated by the following formula:

$$r(x_i) = -\frac{f(x_i)}{\|\nabla f(x_i)\|_2^2}\nabla f(x_i) \tag{7}$$

Then, the remaining undisturbed dimensions are iterated until the discriminator outputs the correct results. Meanwhile, in order to be less easily perceived by the human eye and reduce the calculation cost, Manhattan-Distance [16] limitation is added to constraint the disturbance. In digital images, suppose there are two pixels $i(x_1, y_1)$ and $j(x_1, y_1)$, the Manhattan-Distance between them is:

$$D(i,j) = |x_1 - x_2| + |y_1 - y_2| \tag{8}$$

Because it only needs simple addition and subtraction operations, and there is no complex floating-point operations compared with Euclidean distance, its calculation speed will be improved.

A set *D* is used to record perturbed pixels. For example, $D = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), ..., (x_s, y_s)\}$ indicates that s pixels have been perturbed. Suppose the perturbation distance between pixels is $d_{threshold}$. When the next pixel is disturbed, the Manhattan-distance between the pixels and each point in set *D* is calculated first. When distances are not less than $d_{threshold}$, it can be disturbed. Otherwise, it will not be calculated and other pixels will be tried. The pseudo-code of the algorithm is shown below, where *X* is the original input sample, Y^* is the target output category, *F* is the mapping function of the classifier network, r is the perturbation quantity, X^* is the input after the perturbation, δ_x is the total perturbation vector. Set *Q* is used to record whether the pixels in the sample have been traversed. $Q(i,j) = 1$ means pixel (i, j) has been traversed, so it will no longer be calculated in all iterations after it.

Algorithm 1: Iteration with batch dimension and Manhattan-distance constrains

Input: X, Y^*, F, r

Output: δ_x, X^*

1 Let $X^* = X$.
2 Initialize *D*, which initial value is an empty set.
3 Initialize *Q*, which initial value is a matrix of all zeros.
4 **While** $F(X^*) \neq Y^*$:
  Computing gradient matrix *F*.
  Find the three pixels x_i, y_i, z_i with the largest gradient in F.
  If $D \neq null$:
    Put x_i, y_i, z_i in set D.
  Else:
    Calculate the Manhattan- Distances between them and each element in D.
    If $\forall d(d \in D) \geq d_{threshold}$:
      Add perturbation r to x_i, y_i, z_i.
      Add the perturbed pixels to set *D*.
      Set the values of x_i, y_i, z_i in *Q* to 1.
    End if
  End while
5 $\delta_x = X^* - X$.
6 Return X^* and δ_x

First, the algorithm calculates the gradient matrix of the input current value to the output, which is also called saliency mapping. Then three dimensions with the largest gradient amplitude are selected for perturbation. And the remaining undisturbed dimensions are iterated repeatedly until the desired classification results can be successfully deceived by the network.

IV. Experiment

A. Data Set

In order to verify the effectiveness of this algorithm, two data sets MINIST and CIFAR-10 are selected. MINIST is a handwritten digital data set with ten classifications ranging from 0 to 9. CIFAR-10 data set is also a data set with ten classifications: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. In the experiment of this paper, we selected 1000 images for testing. Because MINIST and CIFAR-10 have ten categories, for each image in the test set, all other nine classes of targeted confrontation samples are generated. So, in the end, there will be 9000 adversarial examples.

B. Network Model

In this paper, we use the convolutional neural network [17] model to test the attack effect of adversarial examples. The specific parameters of the model are as follows. It consists of four convolution layers, each layer is recorded as Convi $(1 \le i \le 4)$, two pooling layers and two fully connected layers, and Rectified Linear Unit (ReLU) is chosen as the activation function [18].

TABLE I: Network Structure

Layer Type	MINIST	CIFAR-10
Conv1+ReLU	3×3×32	3×3×64
Conv2+ReLU	3×3×32	Channel 3
Pool1	2×2	2×2
Conv3+ReLU	3×3×64	3×3×128
Conv4+ReLU	3×3×64	3×3×128
Pool2	2×2	2×2
Fully Connected1+ReLU	200	256
Fully Connected2+ReLU	200	256

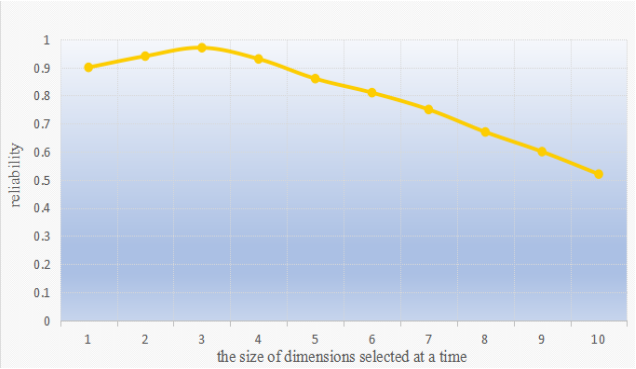


Fig. 3. Relationship between reliability and dimension.

C. Initial Perturbation Dimensions

One of the innovative points of this paper is the idea of batch dimension iteration. The dimension of initialization is very important. If the selection is too small, the generation speed of countermeasure samples will be slowed down. And if the selection is too large, the divergence of samples will be too large and the deception of human eyes will be reduced.

In order to select the appropriate iteration dimensions, the evaluation standard called reliability is proposed in this paper:

$$reliability = 0.6 * d + 0.3 * s + 0.1 * e \qquad (9)$$

As shown in formula 9, reliability is calculated by weighting the deception rate d, eye discrimination rate e and generation speed of adversarial examples s. Among them, eye discrimination rate represents the probability that the adversarial example can successfully deceive the perception of human eyes. It is the result of our survey of 100 volunteers.

It can be seen from Fig. 3 that if we select three dimensions each time, the value of reliability reaches its maximum. Thus, we choose three as the size of dimensions for iteration.

D. Result Presentation

Select 10 rows and 10 columns of the result picture on each data set and divide them into ten categories. Suppose the picture in column j of line i is $p(i, j)(1 \le i \le 10, 1 \le j \le 10)$, only pictures on the diagonal line are real samples, which is $p(i, j)(1 \le i \le 10)$ and all the rest of the pictures are adversarial examples. So, $p(i, j)$ represents a sample image that has been incorrectly classified into class j while $j \ne i$. Fig. 4 shows some adversarial examples on MNIST data set. Fig. 5 shows some adversarial examples on CIFAR-10 data set.

It is worth mentioning that these samples can be perceived by human eyes easily while the error rate of classification on these samples by deep learning algorithm is very high, which means our method has better deception accuracy.

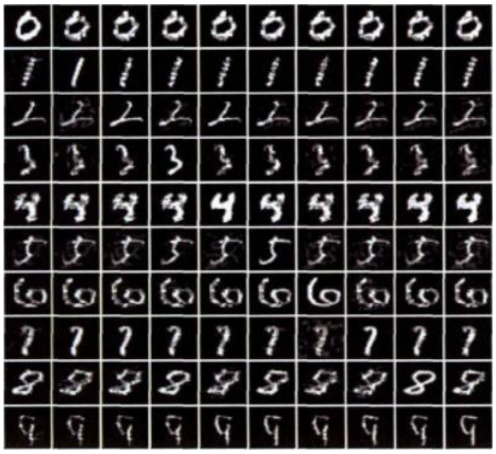


Fig. 4. Some adversarial samples generated by our algorithm on MNIST data set.

Fig. 5. Some adversarial samples generated by our algorithm on CIFAR-10 data set.

Fig. 6 shows the perturbed dimensions needed to generate adversarial examples. It can be found that due to the limitation of Manhattan-Distance constraint, most examples are concentrated in the interval of [0,60], except for a small

number of samples with large disturbance range.

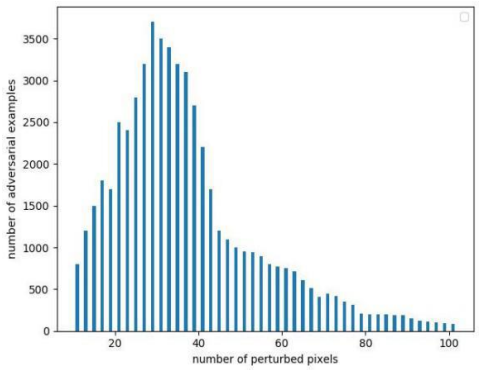


Fig. 6. Perturbation dimension histogram of test samples.

Fig. 7 shows the relation between deception rate and perturbed dimensions. It can be seen that when the perturbed dimensions reach about 60, the adversarial example has a high deception rate. When the perturbed dimensions reach 90 or more, it can deceive the convolutional neural network by almost 100%.

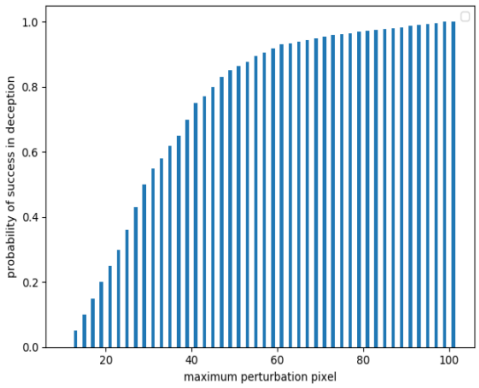


Fig. 7. The histogram of the relation between deception success rate and perturbation dimension.

Table 2 shows the comparison among our method, FGSM and C-W algorithm. The attack success rate of our method is much higher than that of FGSM. And the perturbed dimensions are obviously lower than that of C-W algorithm.

TABLE II: COMPARISONS OF ADVERSARIAL EXAMPLES AMONG DIFFERENT ALGORITHM

Method	Recognition Success Rate	Attack Success Rate	Run Time	Dimensions of Perturbation
FGSM	14.9%	85.1%	3.15s	-
Carlini-Wagner	1.67%	98.33%	4517s	2287
Our Method	1.69%	98.31%	5.08s	36.28

V. CONCLUSIONS

This paper focuses on the security issues in machine learning. An improved algorithm for generating adversarial examples is presented in this paper. Compared with existed method, our method achieves better deception rate and perturbs less pixels of images. During an epoch in batch dimension iteration, multiple pixels are perturbed while Manhattan-Distance constraints are added to them. Our algorithm performs well in experiments. Compared with Carlini-Wagner method, only 60 more dimensions are perturbed, which indicates that the computation cost of our algorithm is completely acceptable. Besides, compared with FGSM algorithm the deception rate is increased by 12% while the generation times of them are almost same.

Through the research of the new adversarial example generation algorithm in this paper, the principle of adversarial example generation is revealed and remind people to pay attention to the security issues in the field of deep learning, which lays a foundation for the establishment of effective and reasonable security mechanism in the future.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Dahui Liu concuted the research; Zihan Song and Siyu Ren helped perform the analysis with constructive discussions; Dahui Liu, Zihan Song, and Siyu Ren wrote the paper; Siyu Xia contributed to the conception of the study; all authors had approved the final version.

REFERENCES

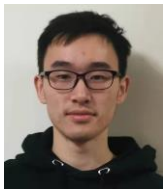
[1] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.
[2] X. M. Ma and S. L. Lv, "Financial credit risk prediction in internet finance driven by machine learning," *Neural Computing and Applications*.
[3] C. Y. Lin, M. Wu, J. A. Bloom, I. J. Cox, and M. Miller, "Rotation, scale, and translation resilient public watermarking for images," *IEEE Trans. Image Process.*, vol. 10, no. 5, pp. 767-782, May 2001.
[4] Ferentinos and P. Konstantinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture*, vol. 145, pp. 311-318.
[5] X. L. Ma, H. Y. Yu, Y. P. Wang, and Y. H. Wang, "Large-scale transportation network congestion evolution prediction using deep learning theory," *PLOS ONE*, vol. 10, 2015.
[6] A. Naveed and M. Ajmal, "Threat of adversarial attacks on deep learning in computer vision: a survey," *IEEE Access*.
[7] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. the 2017 ACM on Asia Conference on Computer and Communications Security*, ACM, 2017, pp. 506-519.
[8] P. Samangouei, M. Kabkab, and R. Chellappa, "Defensegan: Protecting classifiers against adversarial attacks using generative models," *arXiv preprint arXiv:1805.06605*, 2018.
[9] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," *arXiv preprint arXiv:1801.02610*, 2018.
[10] C. Szegedy, W. Zaremba, I. Sutskever *et al*., "Intriguing properties of neural networks," *Computer Science*, 2013.
[11] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.
[12] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. 2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, 2016, pp. 372-387.
[13] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.
[14] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. 2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2017, pp. 39–57.
[15] M. K. W. J. Kivinen, *Exponentiated Gradient Versus Gradient Descent for Linear Predictors*, 1997.
[16] C. Susan, *Manhattan Distance*, 2016.
[17] P. Kim, *Convolutional Neural Network. MATLAB Deep Learning*, 2017.

[18] I. Aizenberg, *Multi-valued Neuron with a Periodic Activation Function. Complex-Valued Neural Networks with Multi-valued Neurons*, 2011.

Dahui Liu was born in Xuzhou, China, in 1995 and received the B.S degree in automation from Nanjing University of Information Science & Technology, Nanjing, China, in 2017. He is currently pursuing the M.S. degree with the School of Automation, Southeast University.

He has wide research interests mainly including computer vision, machine learning, pattern recognition, object detection and robotics. Particularly, he is interested in theories and algorithms about adversarial example and artificial intelligence safety and security.

Zihan Song received the B.S. degree in electronic engineering from Nanjing University of Posts and Telecommunications. He is currently pursuing the M.S. degree in electronic and computer engineering in University of Southern California. He has wide research interests mainly including computer vision, machine learning and pattern recognition. Particularly, he is interested in image generation. Previously, he has worked on edge computing for Megvii as an intern.



Siyu Ren is currently pursuing the B.E. degree of Communication Engineering from Nanjing University of Posts and Telecommunications. He has wide research interests mainly including machine learning, data mining and analysis, pattern recognition and networks. Particularly, he is interested in theories and algorithms for target detection. Previously, he has won the third prize in China Undergraduate Mathematical Contest.

Siyu Xia received the B.E. and M.S. degrees in automation engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2000 and 2003, respectively, and the Ph.D. degree in pattern recognition and intelligence system from Southeast University, Nanjing, in 2006. He is currently an associate professor with the School of Automation, Southeast University, Nanjing. His research interests include object detection, applied machine learning, social media analysis, and intelligent vision systems. He was a recipient of the Science Research Famous Achievement Award from the Higher Institution of China, in 2015. He has served as a reviewer for many journals including TIP, TSMCB, TIFS, TMM, IJPRAI, and Neurocomputing. He received the Outstanding Reviewer Award for the journal of Neurocomputing, in 2016. He has also served on the PC/SPC for the conferences including AAAI, ACM MM, ICME, and ICMLA. He is a member of the ACM.