Journal of Information Hiding and Privacy Protection
DOI:10.32604/jihpp.2020.010462
Review

A Survey on Adversarial Example

Jiawei Zhang[*] and Jinwei Wang

Nanjing University of Information Science and Technology, Nanjing, 210044, China
[*]Corresponding Author: Jiawei Zhang. Email: zjwei@nuist.edu.cn

Received: 20 May 2020; Accepted: 01 July 2020

Abstract: In recent years, deep learning has become a hotspot and core method in the field of machine learning. In the field of machine vision, deep learning has excellent performance in feature extraction and feature representation, making it widely used in directions such as self-driving cars and face recognition. Although deep learning can solve large-scale complex problems very well, the latest research shows that the deep learning network model is very vulnerable to the adversarial attack. Add a weak perturbation to the original input will lead to the wrong output of the neural network, but for the human eye, the difference between origin images and disturbed images is hardly to be notice. In this paper, we summarize the research of adversarial examples in the field of image processing. Firstly, we introduce the background and representative models of deep learning, then introduce the main methods of the generation of adversarial examples and how to defend against adversarial attack, finally, we put forward some thoughts and future prospects for adversarial examples.

Keywords: Neural network; deep learning; adversarial example; survey

1 Introduction

As early as the middle of the 20th century, the concept of neural network was proposed. However, the neural network at that time had limited learning ability due to its simple structure, and the ability to fit complex functions was poor. Until 1980s, Rumelhart, Williams, Hinton, Lecun et al. proposed MLP (multi-layer perceptron) [1,2] and used BP (back propagation algorithm) [3,4] to calculate the parameters in the hidden layer of the network, the problem that single-layer perceptron can't fit complex functions was solved. However, due to the limitation of computing power and the rise of kernel methods such as (SVM) support vector machine, the development of neural network encountered a bottleneck again. Nowadays, the computing power and data storage capacity of computer have been greatly improved. Once deep learning [5] is put forward, the field of artificial intelligence make a breakthrough again in technology and concept.

In the past few years, deep learning has become the main method in the field of machine learning, which is used to try to solve problems involving large-scale data in various fields, such as simulating the circuit of human brain [6]; analysis of mutation in DNA [7]; speech recognition [8]; nature language understanding [9]. In the field of image processing, the outstanding performance of convolutional neural network [10] in 2012 large-scale image recognition task [11] has also attracted the attention of many researchers. Since 2012, people have put forward a variety of neural network architectures, such as ZFNet [12]; VGG-16 [13]; ResNet [14]. The performance of the network is also gradually increasing, which also makes deep learning applied in key areas like safety and security, for example, face recognition on various access control systems and self-driving vehicles, etc. [15].

Although the neural network has a high accuracy in classification, pattern recognition and other tasks, the recent research of Szegedy et al. [16] shows that adding a slight perturbation to the original input image intentionally will make the neural network model give a wrong output, and these perturbations are basically indistinguishable to the human eye. This type of attack is called adversarial attack, which will make the neural network completely change its classification of the same picture, and even more the neural network will show a high degree of confidence to the wrong classification label. This poses a great threat to the application of neural networks in some aspects such as self-driving cars, face recognition and so on. Since the concept of adversarial attack was put forward, researchers have proposed a variety of adversarial attack and defense methods. Some of them are generated by one-step method, such as FGSM (fast gradient sign method) [17]; some are generated by iterative method, such as DEEPFOOL [18].

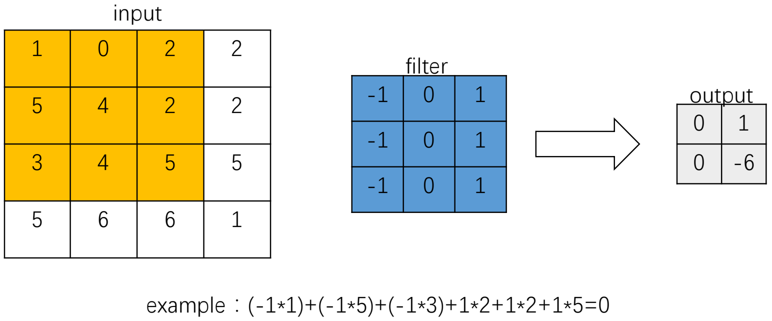
This paper reviews the development of adversarial example. In the second part, this paper will introduce the basic composition of neural network and its development in recent years. In the third part, it will introduce the current main methods of adversarial example generation and defense against it. In the end, we draw conclusion and show some expectations for the development of neural network in the future.

2 Neural Network

In recent years, with the gradual development of neural networks, people have put forward a variety of network architecture. Although these neural networks are more and more excellent in image classification, positioning and other tasks, but these networks are still very fragile in the face of adversarial attack. In this paper, we will briefly introduce the concept of neural networks and several influential neural network architectures.

2.1 Components of Neural Network

Convolutional Layer. As shown in Fig. 1, convolution operation is the inner product operation (multiplication and sum of corresponding elements) of the original input matrix and a group of matrices with weights. Specifically, in the field of image processing, the original input image can be regarded as the input matrix (the gray image corresponds to the two-dimensional matrix, and the color image corresponds to the matrix with three RGB channels). The matrix with weight is called filter, and filters of different sizes will play different roles. For example, 1 × 1 filter is usually used to compress the number of channels, 3 × 3 filter is more inclined to extract local features in the image, and the larger 5 × 5, 7 × 7 filter is more inclined to extract global features. The filter will pass through the input matrix according to the step size and output a result matrix. Convolution has many good properties such as thin connection and parameter sharing. For example, with a 3 × 3 filter, each element of the output matrix is only associated with nine elements of the original input matrix. Parameter sharing means that when the feature extraction function of the filter is effective for one part of an image, it is also effective for other parts of the image.



1	0	2	2
5	4	2	2
3	4	5	5
5	6	6	1

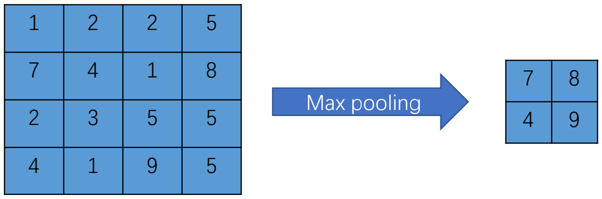
-1	0	1
-1	0	1
-1	0	1

0	1
0	-6

example：(-1*1)+(-1*5)+(-1*3)+1*2+1*2+1*5=0

Figure 1: The original image is processed by filter and then output the characteristic image

Pooling Layer. The pooling layer is generally used to process the output result matrix of the convolution layer, in order to reduce the size of the model, improve the calculation speed, and to some extent improve the robustness of the selected features. Pooling operation is generally divided into maximum pooling and average pooling. As shown in Fig. 2, the maximum pooling layer of 2 × 2 selects the largest of the four elements as the output. As shown in Fig. 3, the average pool layer of 2 × 2 calculates the average value of four elements, and takes this average value as the output.



1	2	2	5
7	4	1	8
2	3	5	5
4	1	9	5

7	8
4	9

Figure 2: The 4 × 4 original image is processed by max-pooling filter and then output the 2 × 2 characteristic image

Full-Connected Layer. Full connection is similar to the process of weighted summation. Each output is the sum of the corresponding weights multiplied by the nodes of the previous layer plus a bias. The structure of the full connection layer is shown in Fig. 3. Each node in the full connection layer is connected with each node in the previous layer, so as to integrate the features. Therefore, the full connection layer generally has a large number of parameters.

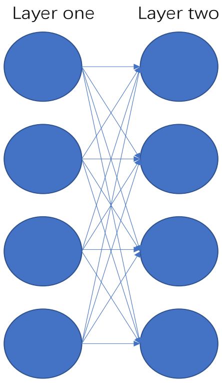


Figure 3: Full connection between two layers of four nodes

Activation Function. The commonly used nonlinear activation functions are sigmoid [19], tanh, relu and so on. Although they are nonlinear activation functions, their properties are very similar to linear functions, which is also the main reason for the generation of adversarial example. This will be introduced in Section 3.1.

2.2 Classical Network Model

LeNet-5. As shown in Fig. 4, although the network model of LeNet-5 [20] is not large, only 8 layers, but it contains the basic components of the convolutional neural network, the entire network has 2 convolution calculations. After each convolution calculation, the pooling operation is followed, and after two layers are fully connected, 10 possible values are output. The LeNet-5 network model contains about 60,000 parameters, which is very simple compared to a network with billions of parameters nowadays, and it is still a very efficient network model for handwritten character recognition task.

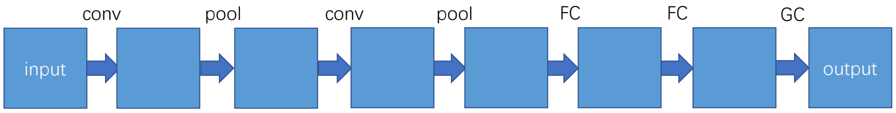


Figure 4: Simple network architecture diagram of LeNet-5

AlexNet. AlexNet [21] is widely regarded as the origin of deep learning in the industry. In fact, AlexNet is a large-scale, deep network model with a total number of parameters of about 60 million. As shown in Fig. 5, the network model is divided into upper and lower parts, corresponding to two GPUs, and two GPUs interact in some specific network layers to improve computational efficiency. Each part consists of 5 convolutional layers, maximum pooling layer, dropout layer, and 3 fully connected layers. The introduction of the dropout layer solves the problem of over-fitting of the training set. It is worth mentioning that AlexNet Network won the 2012 ILSVRC (ImageNet Large-Scale Visual Recognition Challenge) and achieved 15.4% of the top 5 test error rate. (Top 5 error means that when an image is input, there is no correct answer in the top five of the prediction results of the model). The score of the neural network model behind it is 26.2%. The huge advantages of AlexNet have caused great repercussions in the industry.

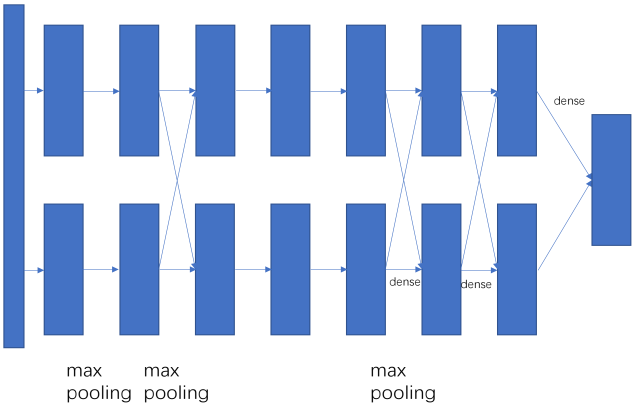


Figure 5: Simple network architecture diagram of AlexNet

VGG-16. Compared with the network proposed above, the most significant feature of VGG-16 [13] is that the structure of this network is simpler and the network layer is deeper. The network model has 16 layers and about 138 million parameters. This was already a very deep network model at that time, in which the filter selected for the convolution layer was 3 × 3, the step size was 1, the pooling layer was 2 × 2, and the step size was 2. As shown in Fig. 6, the structure of the network can be seen as composed of multiple modules. Each module in the first half of the network are composed of several convolution operations following with a pooling operation. The last module is composed of three full connection layers and a softmax classifier. In the process of training, with the deepening of network layers, the input and output of each layer are constantly changing with a specific rule due to this regular network structure. Specifically, the height and width of the output characteristic map are reduced by half after each pooling operation, and the number of channels is doubled after each convolution operation. The model participated in the 2014 ImageNet Image Classification and Positioning Challenge and achieved excellent results: ranked second in the classification task and ranked first in the positioning task.

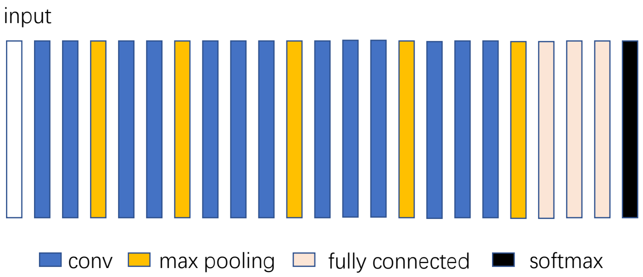


Figure 6: Simple network architecture diagram of VGG-16

ResNet. The deepest network model mentioned above has only 16 layers, which is far less than the ResNet network architecture proposed by MRA in 2015. ResNet has 156 layers in total, which not only breaks the record of the deepest network at that time, but also breaks the record of ILSVRC before 2015, reaching an amazing 3.6% of the top 5 test error rate. The network is composed of several residual blocks. The structure of the residual block is shown in Fig. 7. The main function of the residual block is to make one input value skip one or more layers and directly enter the deeper layers of the network for calculation. In theory, with the deepening of network layers, the training errors should be reduced continuously, but in fact, the increase of network layers will make it difficult for the optimization algorithm to train a better network model. However, the introduction of residual blocks make it possible to train deeper and networks, although these shortcuts between layers will make the network model look bloated.

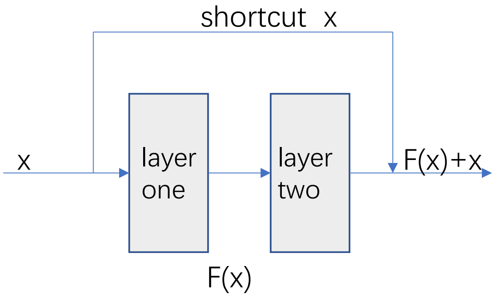


Figure 7: Diagram of the residual block with two layers skipped

3 Adversarial Example

3.1 Generation of Adversarial Example

Box-constrained L-BFGS. Szegedy et al. first found the flaws of deep neural network in the image classification task. He pointed out in his paper that when add weak perturbation in the image, the deep learning model will give the wrong classification results. He proposed formula (1) for the generation of adversarial examples, in which I_c represents the original image (undisturbed image), ρ represents the weak perturbation which need to be calculated, l represents the label of the image to be classified, and $C(\,)$ represents the classifier of neural network. We need to find the minimum perturbation ρ that satisfies the constraint so that the original label I_c would be different from the label that the image is classified by neural network after the perturbation ρ is added.

$${}^{min}{}_{\rho}\|\rho\|^2 \;\; s.t.\, C(I_c+\rho)=l;\; I_c+\rho\in[0,1]^m \tag{1}$$

But in fact, the formula (1) is difficult to solve, which is a NP hard problem, so Szegedy et al. used box-constrained L-BFGS [22] to approximate the solution. The solution formula (1) is transformed into the solution formula (2) to find a minimum positive coefficient c to constrain the perturbation ρ to the minimum.

$${}^{min}{}_{\rho}c|\rho|+L(I_c+\rho,l)\; s.t.\, I_c+\rho\in[0,1]^m \tag{2}$$

As shown in Fig. 8, the above method can successfully calculate the perturbation ρ. Add the original image and the perturbation ρ to form the adversarial example. When the neural network model is attacked by this adversarial example, the neural network classifier would output the wrong classification results.

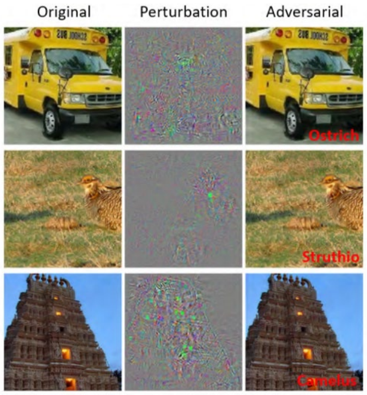


Figure 8: Illustration of adversarial examples generated using for AlexNet

Fast Gradient Sign Methods. After Szegedy et al. pointed out that neural network can improve the robustness against adversarial attacks by means of adversarial training. Goodfellow et al. proposed a more efficient way to calculate perturbation to generate adversarial example. In formula (3), θ represents the parameters of the neural network model, $J(\theta, I_c, l)$ represents the loss function, $sign(\,)$ represents the sign function, and ρ represents the perturbation need to be calculated.

$$\rho = \varepsilon\, sign(\nabla J(\theta, I_c, l) \tag{3}$$

The idea of this method is to keep the direction of variation consistent with the direction of gradient, and increase the value of loss function at the fastest speed, so as to produce the greatest change to the classification results. However, it is still necessary to select the parameter ε manually to constrain the perturbation, which makes the adversarial example produced by the fast gradient sign method invalid for the neural network model with nonlinear activation function.

It is worth mentioning that the neural network model was not as highly nonlinear as people thought at that time, but on the contrary, Goodfellow et al. pointed out that the neural network was just too linear, so that it can be easily deceived by adversarial example. The linearity of neural network is caused by the activation function which is linear or not linear but has linear performance, such as relu function and sigmoid function. Once the model is highly linearized, even a small amount of perturbation will be enough to affect the output after linear expansion in high-dimensional space.

Miyato et al. [23] proposed the target attack based on the fast gradient sign method. The target attack refers to generating the adversarial example to make the neural network give a specific output. Their calculation method of perturbation ρ is shown in formula (4). Miyato et al. proposed to normalize the gradient of loss function $\nabla J(\theta, I_c, l)$ through l_2-norm. In addition, their also proposed to normalize the gradient of loss function by l_∞-norm.

$$\rho = \varepsilon\, \frac{\nabla J(\theta, I_c, l)}{\|\nabla J(\theta, I_c, l)\|_2} \tag{4}$$

Basic & Least-Likely-class Iterative Methods. The above methods are all one-step methods. One step method is to get the perturbation ρ only through one calculation. In fact, in addition to one step method, the perturbation ρ can also be calculated by means of iteration. As is shown in formula (4), the basic iterative method calculates the perturbation ρ through iteration, where I_ρ^i represents the disturbed image after the i-th iteration, $Clip_\epsilon\{\,\}$ clips (the values of the pixels of) the image in its argument at ϵ and α determines the step size (normally, α = 1)[24].

$$I_\rho^{i+1} = Clip_\epsilon\left\{I_\rho^i + \alpha\, sign(\nabla J(\theta, I_\rho^i, l))\right\} \tag{5}$$

Carlini and Wagner Attacks (C&W). Carlini et al. [25] proposed three new attack modes, which are effective for distillation [26] neural network and non-distillation neural network. In this way, they prove that defensive distillation cannot significantly improve the robustness of the neural network against adversarial attacks. Compared with the previous algorithms, these three new methods are more effective in most cases. They also proved that the adversarial example generated on the non-distillation network can attack on the distilled network very well, so as to make the black-box attack. The black-box attack refers to the attack on the network without knowing the parameters or other concrete information of the neural network.

Inspired by the C&W attack, Chen et al. [27] proposed zero order optimization, which directly estimates the gradient of the target model to generate adversarial example.

DeepFool. DeepFool solves the problem of manually selecting parameter ε to constrain the perturbation in FGSM. Moosavi-Dezfooli et al. [18] proposes to use a smaller vector to disturb the image each time through iteration to move it to the decision edge gradually. The proposed optimization problem is shown in formula (6).

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

The above optimization problem can be solved analytically, which is the last item in (6). In fact, in the case of linear decision function, w represents the gradient direction of decision function, and the front coefficient term $f(x_0)/||w||$ exactly corresponds to the optimal perturbation coefficient ε, so there is no need to select the perturbation coefficient manually, which fundamentally improves the shortcomings of FGSM. Because DeepFool attack strategy is to maximize the confidence of classifier, so sample x_0 needs to reverse along the direction of gradient to minimize the confidence of the correct classification label. The optimal solution r in (6) is to satisfy $f(x_0+r)=0$ Therefore, the final adversarial example is $x = x_0 + r + \delta$, where δ is a small offset in the r direction to make $f(x_0+r+\delta)<0$.

3.2 Defense Against Adversarial Example

In recent years, the defense against attacks mainly follows three directions [24].

- modified training is in the process of learning, or modified input is in the process of testing.
- modify the network, such as adding additional network layer.
- use additional external neural network model.

Modified Training/Input. All kinds of methods in this direction do not directly modify the learning model. The simplest idea is brute force adversarial training. Since the neural network is to modify the parameters in an iterative way, so as to learn the potential laws or characteristics of something, then it should be possible to train the network model by taking the adversarial example as the input of the neural network, in order to find the inherent characteristics and laws of the adversarial example. It has been proved that adversarial training can improve the robustness of neural network, but this kind of training requires a large training set, as shown in Fig 9, such a trained network does not have good ability of generalization. Moosavi-Dezfooli [28] points out that no matter how many adversarial train are taken, there are always new adversarial examples that can cheat the neural network again.

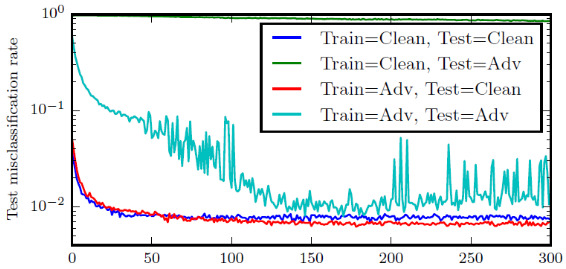


Figure 9: Test results of neural network after training on different training sets

In addition to adversarial training, Dziugaite et al. [29] tried to use JPG image compression to defend against adversary attacks. They note that most of the training sets used to train neural networks are composed of JPG images up to now. In the process of image compression, perturbation in adversarial example may be removed, so as to achieve a clean input. Experiments show that this method is effective for some kinds of adversarial attack methods, but the compressed image will also have a bad impact on the accuracy of normal classification, and sometimes small compression can't successfully remove the perturbation in the image. Therefore, how much compression will be the best to remove the perturbation remains to be further studied.

In this direction, the other defense against the adversarial examples are foveation based defense [30] and data randomization and other methods [31,32]. The former can better defend against the adversarial perturbation generated by L-BFGS and FGSM, while the latter mainly reduces the strength of the adversarial attack by randomly rescaling the training set, so as to improve the robustness of the network to the adversarial attack.

Modified Network. The direction is to modify the neural network to improve the robustness of the network against the adversarial example.

In the process of studying the defense against adversarial example, the researchers noticed that simply stacking the de-noising self-encoder on the original network can't improve the robustness of the neural network against adversarial attacks, on the contrary, it will make the network more vulnerable to adversarial attacks. Later, Gu et al. [33] introduced deep contracting networks, in which a smoothness penalty which is similar to contracting auto encoders was used.

In addition to deep compression network, gradient regularization and gradient mask are also the main ways to defend against adversarial attacks in this direction. Ross and Doshi Velez use input gradient regularization to improve the robustness of the network model against adversarial attacks [34], and the penalty the change of the output relative to the input, so that a small adversarial perturbation will not have a significant impact on the output. The combination of this method and brute force training has a good effect, but it will make the calculation too complex. This method has been abandoned in many occasions.

Hinton [35] once pointed out that distillation refers to the transfer of knowledge from complex networks to simple networks. Based on this theory, a defensive disruption method is proposed by Papernot et al. [26] to improve robustness of neural network through the information of the neural network itself, and it is proved that this method can resist the attack of small perturbation. Defensive distillation can also be seen as an example of gradient mask technology.

Biologically inserted protection [36] refers to a highly nonlinear activation function simulating the nonlinear dendrite calculation in the biological brain to defend against adversarial attacks.

Cisse et al. [37] put forward "Parseval" networks to defend against the attack. The whole network can be regarded as a combination of multiple functions by controlling the global Lipschitz constant of the network to achieve the defense of weak perturbation by keeping a small Lipschitz constant of these functions.

Gao et al. [38] proposed to insert a mask layer before the classification layer, and the added mask layer was trained by transferring clean and antagonistic image forward, which encoded the difference between the output characteristics of the previous layer for these images. Gao et al. Thinks that the most important weight in the adding layer corresponds to the most sensitive characteristic of the network. Therefore, when classifying, these features are forced to change the dominant weight of the added layer to zero.

Additional Network. Akhtar et al. [39] proposed a defense framework to defend the adversarial examples generated by universal perturbation. The framework attached additional pre input layers to the target network, and trained them to correct the adversarial examples, so that the classifier predicted the clean version of the same image as same as the adversarial examples. By extracting the characteristics of input-output difference of training image, the separated detector is trained. By adding a single trained network to the original model, we can achieve a method that does not need to adjust the coefficient and immune adversarial example.

The basic architecture of GAN is shown in Fig. 10. Lee et al. [40] use the framework of generative adversarial network [41] to train a network that is more robust to FGSM attacks. The author suggests training the network directly along a generated network, and the generated network tries to disturb the classified network. In the process of training, the classifier constantly tries to classify the clean and disturbed images correctly. In another GAN based defense, Shen et al. [42] used the generator part of the network to correct an interfered image.

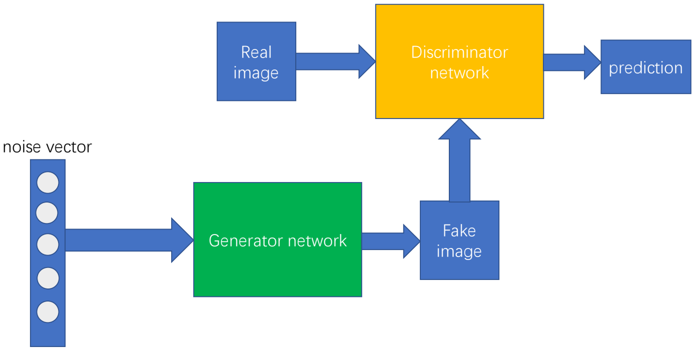


Figure 10: diagram of GAN model

4 Conclusion

In this paper, we reviewed development of deep learning and adversarial example. This paper briefly introduced several most influential neural network models and the way to generate and defend the adversarial example. Although with the development of neural network step by step, it had shown better and better performance in dealing with various problems in various fields, but the high linearity of neural network makes it less resistant to adversarial attack, the same input plus some slight perturbations may make the neural network give completely different output. This limits the application of deep learning in many fields such as security, medicine and so on. Although there are many kinds of defense measures against the adversarial example, the neural network still does not show enough robustness against the different kinds of adversarial attack.

At present, the majority of adversarial example only exist in the laboratory environment, and the presence of adversarial example mainly threatens the application of deep learning in all aspects of life, especially in key field such as security, specifically, self-driving vehicles, face ID, etc. in real life, adversarial attacks need to consider many other issues, such as shooting angle, light conditions, etc. this factors will have an impact on the final adversarial attack results. How to generate powerful adversarial example based on things in life and how to effectively defend them need further study in the future.

Funding Statement: This work is supported by National Natural Science Foundation of China (62072250).

Conflicts of Interest: We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

References

[1] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley and B. W. Suter, “The multilayer perceptron as an approximation to a Bayes optimal discriminant function,” *IEEE Transactions on Neural Networks*, vol. 1, no. 4, pp. 296–298, 1990

[2] S. Mitra and Y. Hayashi, “Neuro-fuzzy rule generation: Survey in soft computing framework,” *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 748–768, 2000.

[3] D. E. Rumelhart, G. E. Hinton and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533, 1986.

[4] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[5] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[6] M. Helmstaedter, K. L. Briggman, S. C. Turaga, V. Jain, H. S. Seung *et al.,* "Connectomic reconstruction of the inner plexiform layer in the mouse retina," *Nature*, vol. 500, no. 7461, pp. 168–174, 2013.

[7] H. Y. Xiong, B. Alipanahi, L. J. Lee, H. Bretschneider, D. Merico *et al.,* "The human splicing code reveals new insights into the genetic determinants of disease," *Science*, vol. 347, no. 6218, pp. 144, 2015.

[8] G. Hinton, L. Deng, D. Yu, E. D. George, M. Abdel-rahman *et al.,* "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[9] I. Sutskever, O. Vinyals and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in Neural Information Processing Systems*, pp. 3104–3112, 2014.

[10] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard *et al.,* "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[11] J. Deng, W. Dong, R. Socher, L. J. Li and F. F. Li, "ImageNet: A large-scale hierarchical image database," *Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[12] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conf. on Computer Vision*, pp, 818–833, 2014.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[15] E. Ackerman, "How drive. ai is mastering autonomous driving with deep learning," *IEEE Spectrum Magazine*, vol. 1, 2017.

[16] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan *et al,* "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[17] I. J. Goodfellow, J. Shlens and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[18] S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 2574–2582, 2016.

[19] J. Rafferty, P. Shellito, N. H. Hyman and W. D. Buie, "Practice parameters for sigmoid diverticulitis," *Diseases of the Colon & Rectum*, vol. 49, no. 7, pp. 939–944, 2006.

[20] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proc. of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[21] A. Krizhevsky, I. Sutskever and G. Hinton, "Imagenet classification with deep convolutional networks," in *Proc. of the Conf. Neural Information Processing Systems*, pp, 1097–1105, 2012.

[22] R. Fletcher, "Practical Methods of Optimization," Hoboken, NJ, USA; Wiley, 2013.

[23] T. Miyato, S. Maeda, M. Koyama and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018

[24] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[25] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in 2017 *IEEE Sym. on Security and Privacy*, pp. 39–57, 2017.

[26] N. Papernot, P. McDaniel, X. Wu, S. Jha and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," *IEEE Symposium on Security and Privacy*, pp. 582–597, 2016.

[27] P. Y. Chen, H. Zhang, Y. Sharma, J. Yi and C. J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proc. of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 15–26, 2017.

[28] S. M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi and P. Frossard, "Universal adversarial perturbations," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1765–1773, 2017.

[29] G. K. Dziugaite, Z. Ghahramani and D. M. Roy, "A study of the effect of jpg compression on adversarial images," *arXiv preprint arXiv:1608.00853*, 2016.

[30] Y. Luo, X. Boix, G. Roig, T. Poggio and Q. Zhao, "Foveation-based mechanisms alleviate adversarial examples," *arXiv preprint arXiv:1511.06292*, 2015.

[31] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie *et al.,* "Adversarial examples for semantic segmentation and object detection," in *Proc. of the IEEE International Conf. on Computer Vision*, pp. 1369–1378, 2017.

[32] Q. Wang, W. Guo, K. Zhang, A. G. Ororbia II, X. Xing *et al.,* "Learning adversary-resistant deep neural networks," arXiv preprint arXiv:1612.01401, 2016.

[33] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," *arXiv preprint arXiv:1412.5068*, 2014.

[34] A. S. Ross and F. Doshi-Velez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients," *arXiv preprint arXiv:1711.09404*, 2017.

[35] G. Hinton, O. Vinyals and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[36] A. Nayebi and S. Ganguli, "Biologically inspired protection of deep networks from adversarial attacks," *arXiv preprint arXiv:1703.09202*, 2017.

[37] M. Cisse, Y. Adi, N. Neverova and J. Keshet, "Houdini: Fooling deep structured prediction models," *arXiv preprint arXiv:1707.05373*, 2017.

[38] J. Gao, B. Wang, Z. Lin and Y. Qi, "Deepcloak: Masking deep neural network models for robustness against adversarial samples," *arXiv preprint arXiv:1702.06763*, 2017.

[39] N. Akhtar, J. Liu and A. Mian, "Defense against universal adversarial perturbations," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 3389–3398, 2018.

[40] H. Lee, S. Han and J. Lee, "Generative adversarial trainer: Defense to adversarial perturbations with gan," *arXiv preprint arXiv:1705.03387*, 2017.

[41] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[42] S. Shen, G. Jin, K. Gao and Y. Zhang, "Ape-gan: Adversarial perturbation elimination with gan," *arXiv preprint arXiv:1707.05474*, 2017.