GENERATING ATTACHABLE ADVERSARIAL PATCHES TO MAKE THE OBJECT IDENTIFICATION WRONG BASED ON NEURAL NETWORKS

[1]SHI-JINN HORNG, [2]HUANG HUANG

[1]Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, 10607, Taiwan, R.O.C.
[2]Department of Computer Science and Information Engineering, Asia University, Taichung 41354, Taiwan, R.O.C.
E-mail: [1]horngsj@yahoo.com.tw, [2]lcc0610@gmail.com

Abstract - Adversarial example is the one that can make our network misclassification through small disturbance, which are often harmless to human cognition but fatal to neural networks. Nowadays, there is no way to resist all kinds of disturbance attacks, which makes people have more doubts about the architecture of the network. Three different sub-models are proposed in this research to attack the neural networks. The attack scope model can effectively reduce the attack range and guide the adversarial algorithm to conduct accurate perturbation attack. Adversarial attack models can generate different adversarial patches through adversarial algorithms. The adversarial patches are compact and can be manufactured artificially. This disturbance patch can be directly attached to the original map to efficiently and accurately disturb the target model. The success rate of disturbance by generating a small patch is 70.1%. Especially, the method proposed in this paper can be applied in different neural networks.

Keywords - Deep Learning, Neural Network, Adversarial Attack, Adversarial Patch

I. INTRODUCTION

Adversarial-examples are examples that have an important impact on the development of in-depth learning and machine learning. Adversarial-examples reveal that neural networks are vulnerability and un interpretable [1]. Some simple and minor perturbations can cause misrecognition, for example, a simple graffiti sticker (Adversarial patch) may confuse the auto-driving system, resulting in a deep neural network model incorrectly classifying traffic signs, or even trick the auto-driving system into the reverse lane.

It is also possible to deceive the detection system of deep neural networks so that the most advanced detection network cannot detect correctly. Adding a little noise to the input examples will make it impossible for humans to identify the minor difference from observation, but it will make the deep neural network model unable to get accurate classification results [2], [7], [12], [13].

In this paper, we combine three models (Preprocess model, Attack scope model and Target model) to generate specific adversarial-patches to effectively disturb the trained deep neural network. This research can not only make us understand what the deep neural network model learns from but also understand the range of adversarial patches and the impact of different locations of adversarial patches on the confidence level classification of deep neural networks. This provides us with a better understanding of the adversarial-attack and further study on the adversarial-defense.

II. RELATED WORK

The adversarial example was first proposed by Szegedy et al. [1], which is a pioneering work in the field of adversarial attack. This article introduces the properties of some NN (neural networks) and puts forward the concept of adversarial example for the first time. In this paper, the authors point out that the input-output image of deep neural network learning is quite discontinuous to a great extent. Szegedy et al. [1] made the network to do wrong classification by adding some imperceptible disturbances. The disturbance is found by maximizing the prediction error of the network. Subsequently, Good flow [2] et al. proposed a fast gradient symbol method (FGSM) to generate an adversarial example with a single gradient step. Before back propagation, the method is used to disturb the input of the model, which is an early form of adversarial training. Moosavi dezfooli et al. [3] proposed Deep fool method which calculates the minimum disturbance necessarily and applies it to the adversarial example construction. In this method, L2 norm is applied to limit the disturbance so as to obtain better disturbance results. Florian tramèr et al. [4] added a randomization step, called R + FGSM, to enhance the adversarial attack. Su et al. [5] obtained better disturbance results by modifying the value of a single pixel. In recent years, adversarial defense is also developing rapidly. V. Zantedeschi et al. [6] proposed a dual defense method, which can complete the configuration with little impact on the objective function of the original model during training. The defense method consists of two major steps. One is to change the activation function, where the relu function is changed to the leaky relu function, so as to enhance the stability of the network. On the other

hand, Gaussian data enhancement is used to enhance the generalization ability of the model, so that the model can classify the original data and the disturbed data. Ali shafahi [7] used the alternating or simultaneous random gradient method to optimize the min max problem, so as to train the robust model to defend against the general antagonistic disturbance attacks. From the NIPS 2017 adversarial example attack and defense competition and the white box adversarial attack of AI defense model by Alibaba and CVPR in 2021, adversarial attack and defense have gained much attention hereafter.

III. PRELIMINARIES

3.1. Convolutional neural network

Convolutional neural network structure mainly has two characteristics: local receptive field and shared weight.

1).local receptive field

A neuron of the next layer is connected to the neurons with window size $w \times w$ of the previous layer. Compared to the fully connected network, the local connections of the convolution neural network is reduced quite a lot, thus reducing the parameters to be trained.

2).shared weight

In convolution neural network, for a specialized kernel of size $w \times w$, these $w \times w$ weights are shared and convoluted to the corresponding outputs of neurons of the previous layer and used as an input to the corresponding neuron of the next layer. It then forms a feature map for this specialized kernel. A certain feature map can be obtained by a certain kernel and different feature maps can be formed to extract different features by giving different kernels.

3.2. Object identification system LeNet

Firstly, a typical convolutional neural network, LeNet [8], is introduced, which is used to recognize numbers.

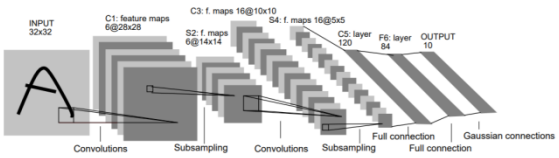


Figure 1. LeNet network structure [8]

LeNet is shown in Figure 1. There are 7 layers in total. The size of the input image of LeNet is 32×32. C is the convolution layer, S is the down sampling layer, and F is the full connection layer. The kernel size of each convolution layer is 5×5. The convolution layer is responsible for extract features and the down sampling layer is used to reduce the size of feature maps.

3.3. Object identification system ResNet

Resnet was proposed by Kaiming He et al. [9] to solve the network degradation while the network is deeper. They propose that the residual layer can make it possible to train deeper convolutional neural networks. Residual layer, if the mapping is written as H (x), the original mapping can be written as f (x) + X, and the nonlinear layer of the stack can be mapped and fitted as f (x) = H (x) − X. The residual mapping is easier to optimize than the original, unreferenced mapping. In extreme cases, it is easier for us to learn zero residuals than to synthesize a pile of nonlinear layers into identity mapping. Res net efficiently executes identity mapping through shortcut connection. Identity mapping can carry out end-to-end training of back propagation for the whole network without adding additional parameters and computational complexity.

Res Net is the champion of ILSVRC image classification in 2015. In addition, Res Net achieved the best results in international competitions such as target positioning, target detection, image segmentation and target detection in Image net competition. Table I shows some Res Net structures.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table I. ResNet network structure diagram

3.4. Adversarial example

Adversarial disturbance/adversarial example is the input disturbance generated by an attack method (or attacker) during the test, which can mislead the neural network (trained on the normal example) to make wrong classification results. Assuming that there is an original input example x and a small enough disturbance is added to the input example x, the adversarial example x', expressed as: $x' = x + \eta$, $\|\eta\| \leq \epsilon$, where η is a value small enough to be ignored by the neural network and ϵ represents the amount of disturbance. The FGSM (fast gradient sign method) method proposed by goodflow et al. [2] correctly classifies the trained convolutional neural network into “Snow mountain” and “Puffer fish” with 57.2% and 78.1% confidences, respectively. However, when the disturbed “Snow mountain” and “Puffer fish” images are input, they are incorrectly classified as “Cat” and “Flatfish” with 67.7% and 59.1% confidences.

Since Szegedy [1] and others first discovered adversarial examples, the vulnerability of neural networks has attracted great attention. Recent research also shows that adversarial examples in real life can also be generated. They can deceive artificial intelligence systems in real life and make them make incorrect or even dangerous decisions. For example, in document [10], experiments show that the "disturbance" disturbance on the "stop" sign can

deceive the neural network to identify it as a "speed limit 45" sign. The disturbance on the "left turn" sign is identified as a "right turn" sign, which may mislead an automatic driving vehicle to cause an accident. Therefore, the vulnerability of neural network against examples has become one of the main security problems of neural network in real life. Szegedy et al. believe that the existence of adversarial disturbance is due to the high nonlinearity of depth network. The nonlinearity of the deep network may lead to discontinuous images from the input data space to the output data space, so it leads to the existence of densely distributed subspace. It is further demonstrated in [2] that the existence of adversarial examples is actually caused by the linear characteristics of deep network, and the linearity is caused by the conversion of several common activation functions such as Relu and Max out. For linear classifiers, with the increase of input dimensions, small changes in each dimension will lead to significant changes in output. However, tanny and Griffin [11] believe that linear interpretation also has limitations, and put forward the view of "boundary tilt": there are adversarial examples only when the classification boundary is close to the boundary of sampling data. Their reasoning is based on the observation that some linear models do not resist the phenomenon of examples. Although these linear and nonlinear explanations seem to contradict each other, researchers still need to establish a comprehensive understanding of the vulnerability of deep neural networks to adversarial examples.

3.5. Adversarial example attack

The more thorough the attacker's grasp of the target model, the easier it is to implement an adversarial attack. According to the information and knowledge of the attacker on the target attack model, it can be roughly divided into white box attack and black box attack, respectively. In short, white box attack assumes that the attacker has full understanding and access to the model, including structure, input, output and weight. Black box attack assumes that the attacker can only access the input and output of the model and does not know the potential structure or weight.

Attackers may expect to achieve different attack targets when carrying out adversarial attacks.

The main index to evaluate the attack is the attack success rate (ASR), that is, the classification error rate of the target classification model. For different types of attacks, the definition of attack success rate is also different. Let F represent the target classification model. The ASR of un-target attack is defined as:

$$ASR = \sum_{i}^{n} F(x_i') \neq y_i / \sum_{i}^{n} F(x_i) = y_i \tag{1}$$

where x_i is i^{th} input sample, x_i' is the generated adversarial example based on x_i and y_i is the corresponding label.

The ASR_t of target attack is defined as:

$$ASR_t = \sum_{i}^{n} F(x_i') \neq t / \sum_{i}^{n} F(x_i) = y_i \tag{2}$$

where t is the specified target and other symbols are the same as un-target attack.

The average intrusion rate IR of un-target attack is defined as the average confidence reduction by

$$IR = \sum_{i}^{n}((F(x_i) = y_i) - (F(x_i') = y_i))/n \tag{3}$$

where the first term is the confidence value of x_i which is correctly classified and that of the second term is the generated adversarial example x_i' which is classified to y_i .

Compared with white box attack, black box attack is more difficult, but in practical application, black box attack will bring greater threat. See [2], [3], [12], [13, [14] in details.

IV. SYSTEM MODEL

4.1. Description of research process

1: Train and build preprocessing model (H model) and attacked model (T model).

2: Establish an open data set, including images with sizes of 32×32 and 224×224 and many real-life object images. The real-life object image refers to the image captured through the camera or network, which does not appear in the data set during model training.

3: Establish the main model architecture, including the preprocessing model for preprocessing data, the attack scope model for guiding the target model to generate adversarial patch, reducing the scope of disturbance attack and accelerating the instruction cycle, and the target model for generating adversarial patch for disturbed attack.

4: The adversary patch is generated iteratively in the area specified by the attack scope mode by the perturbation algorithm, with the size of 4 or 9 pixels, and the error of the change degree of the output probability of the T model is calculated, so as to carry out an effective black box perturbation attack on the target network.

5: A system that can capture images is established. The system attaches adversarial patch at the corresponding position of the image and inputs it to the target network to carry out efficient disturbance attack.

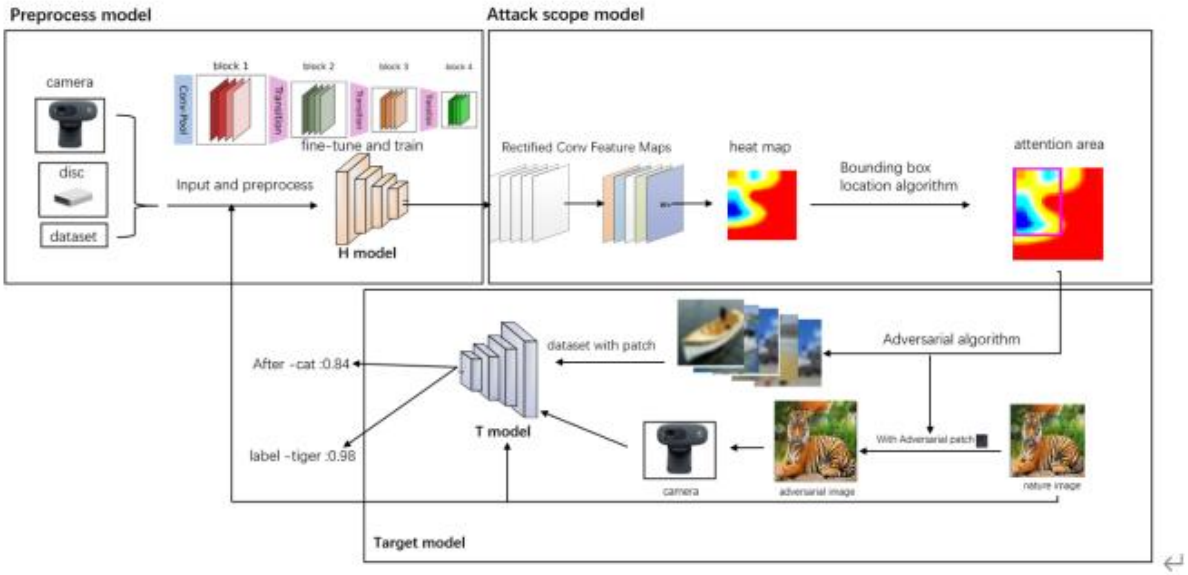


Figure 2. Main model architecture diagram

4.2. Introduction to object recognition model

This research trains and perturbs different models of attack, such as resnet-50 and lenet. T model will be trained through the above two models. It is worth noting that the input image size of the model will be set as 32×32 and 224×224 pixels, and trained through cifar-10 data set. Cifar-10 data set [16] contains 60000 32×32 color images in 10 categories, including aircraft, automobile, bird, cat, tiger, deer, dog, frog, horse, ship and truck. There are 6000 images in each category, a total of 60000 images, including 50000 training images and 10000 test images.

The main model architecture of this study, as shown in Figure 2, is composed of three sub models, namely, the preprocess model for preprocessing data, the attack scope model for guiding the target mode to generate adversarial patch, reduce the disturbance attack range and speed up the instruction cycle, and the target model for disturbed attack.

4.3. Preprocess model

The input data of the preprocessing model can be images taken and stored by our camera from real life, or images of data sets downloaded from the network. Since we require the size of the image to be consistent with the size of the image input by our H-model, we will reshape the input image. At present, the commonly used reshape methods include nearest neighbor interpolation, bilinear interpolation, regional interpolation, bicubic interpolation and Lanczos interpolation. Interpolation algorithm is often used to scale the image. The new gray scale value of scaled image pixels is obtained by resampling and calculating through different interpolation methods. In depth learning, different methods are often used to adjust the resolution of different input images to make its size suitable for the designed neural network for training or testing.

Compared to other methods, the bicubic interpolation method takes into account not only the influence of the change of gray scale value of the sampling point, but also the influence of the gray scale value of four adjacent pixels around the sampling point. It can produce a smoother edge image in image processing, reduce the loss of image quality and achieve good sampling effect. The down sampling method in this experiment adoptsbicubic interpolation method, which produces better image quality and can retain more edge details.

4.4. Attack scope model

The attack scope model is mainly composed of CAM (class activation map) and bounding box location algorithm. CAM activates thermal maps for classes. For example, a classification network can classify 1000 classes. Input a cat dog photo picture containing both a dog and a cat, and it will output a 1000 dimensional probability vector, in which two components correspond to the probability that the picture is classified as a cat and the picture is classified as a dog. So which parts of the picture are more or less related to these two probabilities? This relationship can be represented by the sensitivity heat map. The more sensitive places are, the more blue the color temperature is, and the less sensitive places are, the more red the color temperature is, as shown in Figure 3. It intuitively tells us that the model "sees" the tiger's head and body, so it is considered that there is a tiger in the picture.



Figure 3. CAM activation heat map, the first is the original map, and the last two are CAM activation heat maps

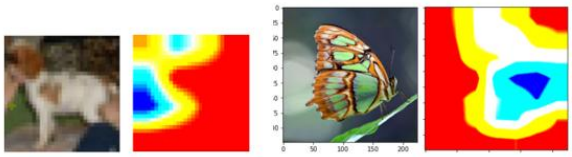
Specifically, in order to obtain the weight value of the target class, we calculate the gradient characteristic graph relative to the last convolution layer. Let y be the output probability vector, c be one of the categories, A be the characteristic graph of the last convolution layer with size $u \times v$, k be the k^{th} channel of the characteristic graph, and i and j be their coordinates, respectively. Calculate the partial derivative of y^c to all pixels on the k^{th} feature map, and then make the global average in Eq. (4).

$$a_k^c = \frac{1}{u \times v} \sum_{i=1}^{u} \sum_{j=1}^{v} \frac{\partial y^c}{\partial A_{ij}^k} \tag{4}$$

This weight a_k^c represents the partial linearization of the deep network of characteristic graph A, and captures the importance of the k^{th} feature map relative to the target category c. This weight is linearly combined with the last layer of characteristic diagram and sent to Relu activation function for linear processing. If the value obtained by a weighting at a position on characteristic diagram A is negative, it is more likely that it does not belong to class c.

$$L_{Grad-CAM}^c = ReLU(\sum_k a_k^c A^k) \tag{5}$$

Figure 4 shows the CAM activation heat map of different sizes generated by the attack range model:



(a) (b)

Figure 4. CAM activation heat maps of different sizes (a) heat map of 32 * 32 dog image (b) heat map of 224 * 224 butterfly image

As for accelerating the operation and reducing the disturbance attack range, we use the attention area extracted from the CAM activation heat map.

Let th be the threshold which is used to find the bounding box which is called the attention area.
Let $\delta_u = (x_u, y_u)$ be the first (or key) pixel whose pixel value is greater than or equal to the threshold value th by scanning the CAM activation heat map from top to bottom. Similarly for $\delta_d = (x_d, y_d)$, it is obtained by scanning the CAM activation heat map from bottom to top. Obviously, $\delta_l = (x_l, y_l)$ ($\delta_r = (x_r, y_r)$) is got by scanning the CAM activation heat map from left (right) to right (left).

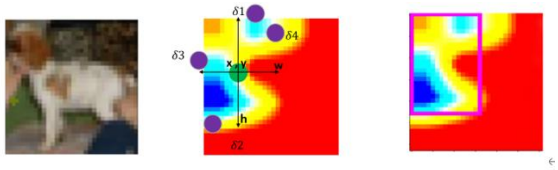
After the key pixels are determined, the center (x, y) of the bounding box is calculated by Eq. (6) and the width w and height h of the bounding box are calculated by Eq. (7), respectively.

$$x = \left| \frac{\delta_r(x) + \delta_l(x)}{2} \right|$$

$$y = |(\delta_d(y) + \delta_u(y))/2| \tag{6}$$

$$h = |\delta_d(y) - \delta_u(y)|, w = |\delta_r(x) - \delta_l(x)| \tag{7}$$

The attention area is then determined by (x, y, w, h). Figure 5 shows an example to find the activation area from a CAM activation heat map.



(a) import image (b) keypoint and CAM activation map (c) bounding box

Figure 5. A bounding box for attention area

4.5. Target model

The target model carries out efficient and accurate perturbation attacks through the attention area with bounding box generated by the attack range model and the perturbation attack algorithm. The algorithm used here is the differential evolution algorithm (DE) [15] proposed by Storn and Price. It is a population-based self-adjusting global optimization method. DE algorithm mainly includes three operations: mutation, crossover and selection. The basic idea is to randomly select two individuals from an initial population, take their vector difference as the change source of the third individual, weigh the difference vector and sum it with the third individual according to certain rules to produce mutated individuals. This operation is called mutation. The operation process of selecting experimental individuals according to probability between mutated individuals and a pre-selected target individual is called crossover, which is to screen mutated individuals and pre-selected individuals with a certain probability. If this probability is met, the mutated individuals will be retained, and if not, the pre-selected individuals will be retained. In short, the target individual is the parent generation, and the experimental individual is the offspring generated through mutation and crossover. If the fitness of the experimental individual is better than that of the target individual, the experimental individual will replace the target individual, otherwise the target individual will still be retained. This process is called selection. In the process of evolution, the algorithm retains excellent individuals through iterative calculation, traverses through high-dimensional data, looks for the patch that is most likely to affect the classification results, and searches for the global optimal solution.
We will randomly generate adversarial patches on the attention area with the bounding box and attach them to the input image to form adversarial examples. Let us identify them in the target neural network,

compare the confidence results of the adversarial examples with the confidence of our previous best adversarial examples, and retain the results of the adversarial examples with low confidence (the lower the confidence, the more effective the attack is). Through the variation and crossover strategy of De, more offspring adversarial images are generated, and the offspring and the parent are selected. The patch that can reduce the image confidence is always left as the new parent. Thus, we iterate repeatedly until we can make our target neural network have wrong recognition results or reach the maximum number of iterations we set. I is the original image, I' prime is the image attached with the adversarial patch, $P_F(I)$ is the confidence of the input image in the final classification of the target neural network, $P_F(I')$ is the confidence of the image attached with the adversarial patch in the final classification of the target neural network.

$$Loss = 1 - log[P_F(I) - P_F(I')] \tag{8}$$

The training goal is to make the loss between the adversarial image and the original image as small as possible. Figure 6 is the flow chart of generating adversarial patch. Firstly, the patch position and pixel value are randomly generated for the picture. Through the network model, it is used to process and output the confidence of the adversarial image. Calculate the difference in confidence and judge whether the confidence of the picture attached with the adversarial patch is lower than or equal to the target requirements. The adversarial patch location is updated iteratively through the DE perturbation algorithm. Finally, judge whether the confidence is reduced to the expected or the preset number of iterations.

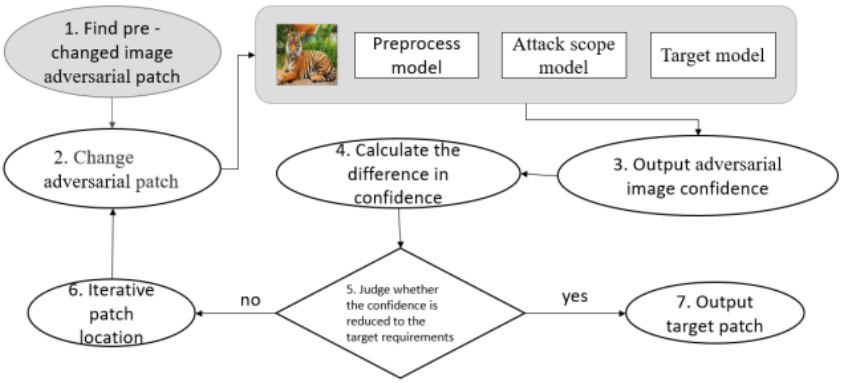


Figure 6.Generate adversarial patch flow chart

V. EXPERIMENTS AND RESULTS

In this study, different data sets and pictures captured by the camera are used as input, and the disturbance attack is carried out through the model in Figure 2. When lenet is selected for T-Model to attack the cifiar-10 verification set, the verification set is an open set data, and each data in the verification set will not be used in model training. As shown in Figure 7 (a), it shows an image of a verification set. The recognition result of the target neural network is a cat with a confidence of 0.77. Figure 7 (b) shows the generated adversarial patch attached at a special position. The special position is lower left: [7, 2, 209, 166, 134], lower right: [7, 3, 209, 166, 134], upper left: [6, 2, 209, 166, 134], upper right: [6, 3, 209, 166, 134]. Then a cat becomes a bird.

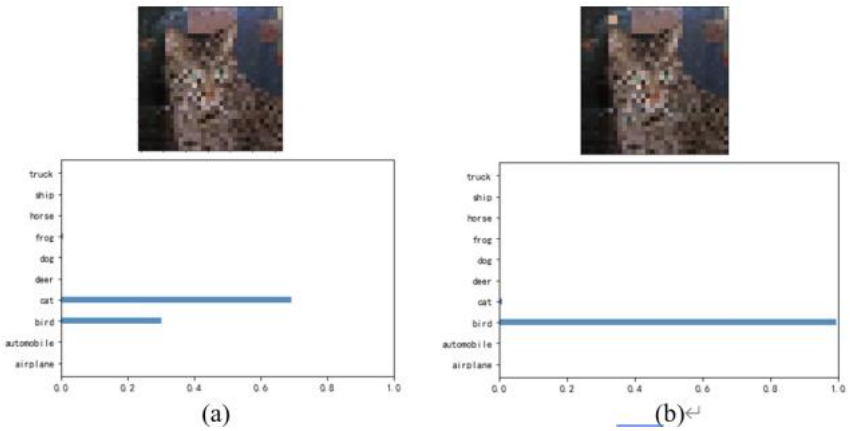


Figure 7.Experimental diagram of classification results, (a) the original result is cat (b) the wrong classification with adversarial patch is bird

The test image can be captured by the camera or downloaded from other networks. Since the size of the input image required by the attacked network is 32 * 32, the test image is down sampled first. Figure 8 is a test image with a patch captured by the camera and the lenet misclassifies a cat to a truck.

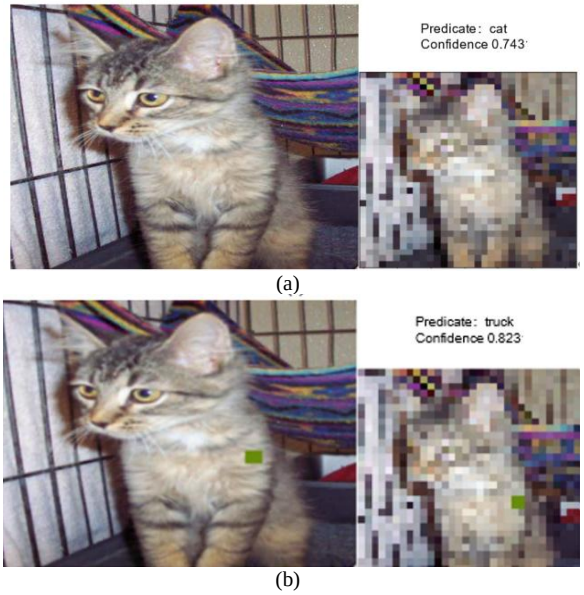


Figure 8.Test image is captured by a camera, (a) the original image is correctly classified as cat (b) the adversarial patch is added, and it is incorrectly classified as truck

Table II shows the results of the 32 * 32 image perturbation attack. The size of the input image is an open set test image with the size of 32 * 32, and the size of the adversarial patch is 2 * 2 pixels, a total of 200 images. The first column is random, which refers to attacking the target model by artificially randomly modifying the patch; the second column is the method of Jia su[5]. Speed means that the average time required to disturb each picture is 19.39 seconds. ASR (attack success rate) is 0.70 and IR average intrusion rate is 0.44. IR means that each input image can reduce its original confidence by 0.44.

32*32 image	speed/s	ASR	IR
random		0.09	0.02
Jia Su*	23.11	0.57	0.37
ours	19.39	0.70	0.44

TABLE II 32 * 32 image disturbance results

At the same time, relevant experiments are also carried out on images of different sizes. When Resnet is selected for T-Model, here we change the size of the patch from 2 * 2 to 3 * 3.

Table III shows the results of the 224 * 224 image perturbation attack. The size of the input image is an open set test image of 224 * 224, where the size of the adversarial patch is 3 * 3 pixels and a total of 100 images. The speed is 51.39 seconds; the ASR is 0.21, and the IR is 0.12. It can be seen that the ASR and IR of a large image attacked by random disturbance are close to 0. For a well-trained T-Model, it is difficult to disturb the target neural network by artificially randomly modifying and attaching patches.

224*224 image	speed/s	ASR	IR
random		0.00	0.01
Jia Su*	60.11	0.11	0.08
ours	51.39	0.21	0.12

TABLE III 224 * 224 image disturbance results

For disturbance attacks. Most methods use the method of adding a large amount of noise to generate adversarial images, such as [2] [7] [12] [13]. Although their ASR is high, they have made a lot of pixel modifications to the image. The patches generated in this paper can be directly attached to the input image, as shown in Figure 13. In contrast, the methods in [2] [7] [12] [13] are a large number of noise modifications, and there is no way to directly attach to the input image for attack. However, compared with the method of [17], the author can disturb and attack the whole face data set by covering a small part of the camera lens, which is also a direction attach approach.

VI. CONCLUSION AND FUTURE WORK

This paper proposes a new disturbance attack model, which can efficiently and accurately find the most sensitive area of the input image and generate adversarial patches. Different from many other black box perturbation attack methods, the perturbation attack model in this paper can generate adversarial patch with small perturbation to make the target neural network misclassify. The adversarial patch in this paper can be directly attached to the target image, which can effectively attack the target network. Therefore, for deep learning applications (such as driverless, unmanned stores, etc.), this will impact safety and property, and we should pay much attention to the potential misjudgment of neural network.

ACKNOWLEDGMENTS

This research was partially supported in part by the Ministry of Science and Technology under contract numbers 111-2218-E-011 -011-MBK and 111-2221-E-011 -134 -, and also by the “Center for Cyber-physical System Innovation” from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan.

REFERENCES

[1] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow,and Rob Fergus, “Intriguing properties of neural networks,” arXiv:1313.6199,2014

[2] Ian J. Goodfellow, Jonathon Shlens and Christian Szegedy, “Explaining and Harnessing Adversarial Examples, ” ICLR, arXiv:1412.6572, 2015

[3] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi and Pascal Frossard, “DeepFool: a simple and accurate method to fool deep neural networks,” IEEE, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2574-2582

[4] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh and Patrick McDaniel,“Ensemble adversarial training: Attacks and defenses,” ICLR, arXiv:1705.07204, 2018

[5] Jiawei Su, Danilo Vasconcellos Vargas and Sakurai Kouichi, “One pixel attack for fooling deep neural networks,” IEEE Transactions on Evolutionary Computation, Vol. 23, Issue. 5, pp. 828-841, 2019.

[6] Valentina Zantedeschi, Maria-Irina Nicolae and Ambrish Rawat, “Efficient defenses against adversarial attacks, ” arXiv:1707.06728 ,2017

[7] [Ali Shafahi, Mahyar Najibi, Zheng Xu, John Dickerson, Larry S. Davis and Tom Goldstein,“Universal adversarial training,” AAAI, arXiv:1811.11304, 2020

[8] Y. Lecun,L. Bottou, Y. Bengio and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE (Volume: 86, Issue: 11, Nov. 1998), pp. 2278-2324, 1998.

[9] [Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun, “Deep residual learning for image recognition,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, 2016.

[10] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno and Dawn Song, “Robust physical-world attacks on deep learning models”, CVPR, arXiv:1707.08945, 2018.

[11] Thomas Tanay and Lewis Griffin, “A boundary tilting persepective on the phenomenon of adversarial examples,” arXiv:1608.07690, 2016

[12] Andras Rozsa, Ethan M. Rudd and Terrance E. Boult, “Adversarial diversity and hard positive generation,” CVPR, Computer Vision and Pattern Recognition, arXiv:1605.01775, pp.410-417, 2016.

[13] Alexey Kurakin, Ian Goodfellow and Samy Bengio, “Adversarial examples in the physical world,” CVPR, Computer Vision and Pattern Recognition, arXiv:1607.02533, 2017.

[14] Nicholas Carlini and David Wagner, “Towards evaluating the robustness of neural networks,” CVPR, Computer Vision and Pattern Recognition, arXiv:1608.04644, 2017.

[15] Rainer Storn and Kenneth Price, “Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces,” Journal of Global Optimization volume 11, pp. 341-359, 1997.

[16] Alex Krizhevsky,Vinod Nair and Geoffrey Hinton “The cifar-10 dataset” https://www.cs.toronto.edu/~kriz/cifar.html,2009

[17] Ping-Chen Tsai, “Alter the most sensitive image points to make face recognition wrong”, C338840,R 008.815.496,2020.

★★★