Backdoor Attack By One-pixel Trigger

Yue Wang[1[0000−1111−2222−3333]], Yuruo Jing[1[1111−2222−3333−4444]], Gengshi Han[1[2222−−3333−4444−5555]], and Yining Kong[1]

[1] Zhejiang University, Zhejiang, China
[2] Zhejiang University, Zhejiang, China glaziawy@gmail.com
http://www.springer.com/gp/computer-science/lncs
[3] ABC Institute, Rupert-Karls-University Heidelberg, Heidelberg, Germany
{abc,lncs}@uni-heidelberg.de

Abstract. The abstract should briefly summarize the contents of the paper in 150–250 words.

Keywords: First keyword · Second keyword · Another keyword.

1 Introduction

Deep learning neural network is proved to be efficient and is used in a lot of scenarios, famous for its great impact on image classification, face recognition etc. For example, Zhu et al. realized face recognition using deep learning framework Caffe to build the neural network applied in their experiment[1]. And Alex et al. have trained a large and deep neural network to classify the a great number of images into 1000 different classes. And they have achieved good testing results[2].

However, there exists adversary samples in training data sets. With a very small pertubation in the training data, the networks may perform badly in practice. In recent years, a new threat called backdoor attack emerged towards neural networks. Several researches indicated that by deliberately changing some of the training data, some artificial backdoors may be inserted into the model[3]. And some researchers found it also possible to play backdoor attacks to deep learning neural networks by hijacking inner neurons from easily accessible pre-trained neural network model, re-training the pre-trained models[4]. Rather than causing the deep learning neural network models' test accuracy to decrease, the adversary samples' goal is to mislead the models to output wrong results for the data with some specific keys, so called backdoor.

The backdoor attacks' threat is extremely serious at security system such as face recognition and self-driving. Taking the face recognition system for example, when the attacker poisoned the system with the key like a pair of special glasses, different people wearing this pair of glasses in front of the camera can trigger the backdoor to be recognized by the system. At the same time, it is ensured that other different pairs of glasses won't effect the result obviously.

2 Related Works

By consulting relevant papers and materials, we summarized the classification of existing backdoor attack methods and some defense methods. Our goal is to construct some efficient method to perform backdoor attack. Therefore we are more focused on the attack methods.

The backdoor attacks can be roughly divided into two kinds. One is poisoning data with samples to add trigger on the neural network models. The other is neuron hijacking, which attacks sensitive neuron without touching the data[5]. Both of them have advantages and some limitations.

Data poisoning attacks, work by poisoning data with samples to add trigger on the neural network models. And there are papers worked on this kind of backdoor attack. For example, Gu et al. implemented a good method to attack, or in another word, to trojan a neural network model using training data poisoning[3]. Their method is named as Badnet. In the experiment part, the Badnet framework is applied to some extent. And Dai et al. also implemented a data poisoning backdoor attack against LSTM-based text classification[6]. Data poisoning is a classical kind of attack towards neural networks and has advanced performance in user training and verification samples, but perform not well in specific attacker-selected input. Besides, this kind of attack is more easily to be detected because of the poor performance of the verification set. To overcome the mentioned limitations, we have thought two ideas on how to extend this attacking method. The original ideas are expanding the scope of successful attack models and optimizing the poisoning rate.

Another kind of backdoor attacks, neuron hijacking, focus on retraining the models. A typical method is shown in a paper by Liu et al.[4] They proposed an approach to trojan a pre-trained model without access to the training data. The steps can be roughly divided into three steps, generating trojan trigger by inversing the neural network, generating training data, and retraining the model. This attack method is efficient and effective in the experiment. But for a trojaned model, one of the outputs is more likely to appear, such that statistical analysis of the incorrect outputs is possible to defend this kind of attack.

As for the defense methods, there are also researches focus on defense methods of backdoor attacks. Wang et al. proposed an approach named Neural Cleanse to identify and mitigate backdoor attacks, which includes three specific goals: detecting, identifying and mitigating backdoor attacks[7]. They identify backdoors and then rebuild possible triggers, and use two complementary techniques for patching: neuron pruning and unlearning.

3 Algorithm

3.1 title

3.2 Model

In this project, four kinds of neural networks are used as attack models, respectively CNN, LeNet5, ReNet34, and VGG16. In this section, we will introduce these models from the basic structure of the model.

LeNet5 LeNet5 network is the first CNN used for digital recognition, and trains for gray-scale image[8]. The input image size is 32 * 32, and without input layer, there are 7 layers in total. Each layer contains trainable parameters (connection weight), which are divided into 2 layers: convolution layer, 2 layers of sample layer, 2 layers of full connection layer and 1 layer of output layer[9].

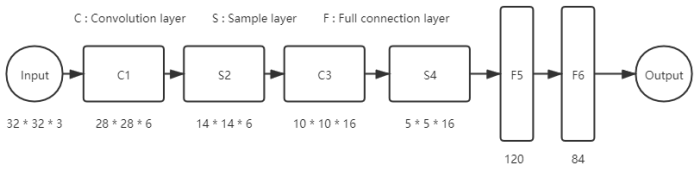

Fig. 1. LeNet5 Architecture

ReNet ResNet (residual neural network) was proposed in 2015, which can effectively solve the problem of gradient dispersion and gradient explosion caused by the increase of convolution layers. ResNet solves the problem of information loss and core loss in traditional convolution by changing the learning goal from learning complete output to learning residual. It protects the integrity of information by passing the input directly to the output. In addition, the simplification of learning objectives also reduces the difficulty of learning[10]. Its core idea is that network output can be divided into two parts: identity mapping and residual mapping, i.e.

$$y = x + F(x) \tag{1}$$

Through the introduction of identity mapping, residuals learning unit establishes a direct correlation channel between input and output, which makes the powerful reference layer concentrate on learning the residuals between input and output. Generally, we use f(x, WI) to represent the residual mapping, and the output is:

$$y = f(x, WI) + X \tag{2}$$

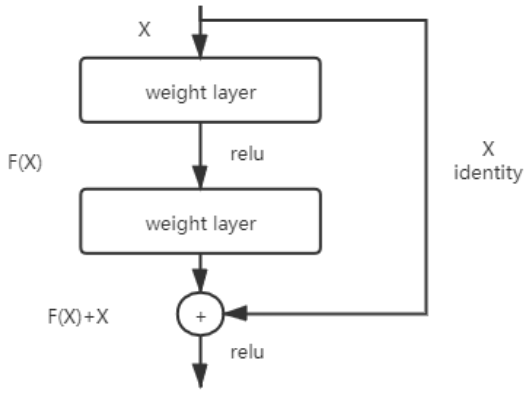

Fig. 2. Residual learning: a building block

When the number of input and output channels are the same, we can use X to add them. When the number of channels between them is different, we need to consider establishing an effective identity mapping function so that the number of channels of input X and output y after processing is the same, i.e.

$$y = f(x, WI) + WS * X \tag{3}$$

The 34 layer residual network structure used in the project is as the following Fig. 3.

VGG16 VGG is a convolutional neural network model proposed by Simonyan and Zisserman in the literature "very deep convolutional networks for large scale image recognition"[11]. VGG16 has 13 convolution layers, 3 full connection layers and 5 pooling layers. Among them, convolution layer and full connection layer have weight coefficient, so they are also called weight layer. The total number is 13+3=16, which is the source of 16 in VGG16. (the pooling layer does not involve weight, so it does not belong to the weight layer and is not counted).

The convolution layer and pooling layer of VGG16 can be divided into different blocks, numbered block 1 block 5 from the front to the back. Each block contains several convolution layers and a pooling layer. Therefore, vgg16 can also be divided into blocks as the Fig. 5 shows.

4 Dirty data

5 BadNet

To implement our attack, we should train BadNet using dirty data on the base of four common neural networks, that is, pure CNN, LeNet5, ResNet and VGG16

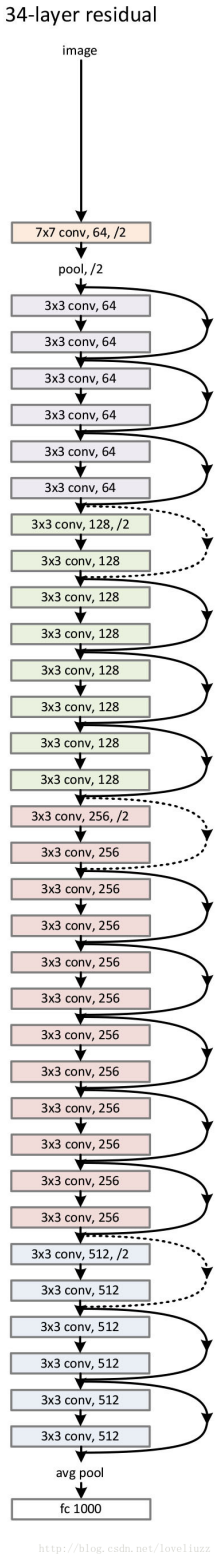

Fig. 3. ResNet34 architecture

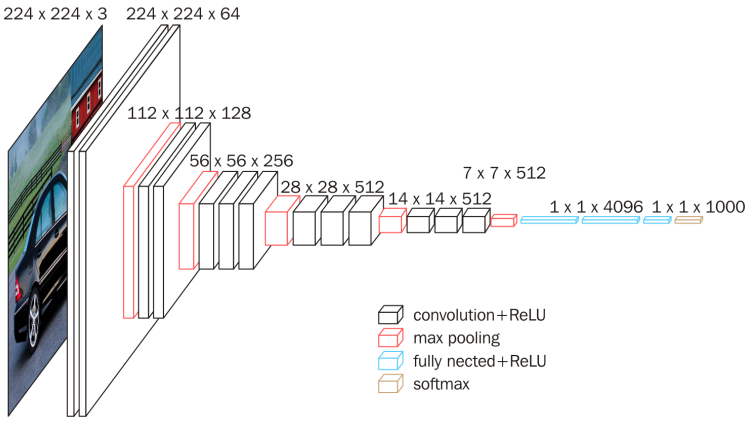

Fig. 4. VGG16 total architecture

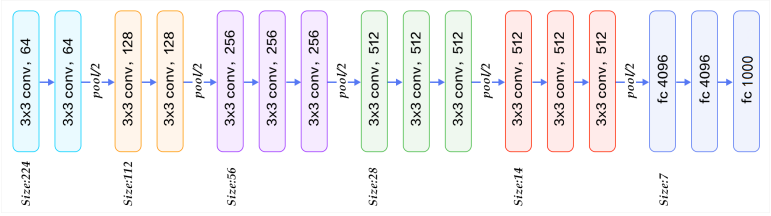

Fig. 5. VGG16 block structure

as we have informed before. The work we do can be regarded as interfering with the results of the classification problem. The poisoned neural network models can be applyed unfriendly in some systems. And once we know the backdoor, we can easily crack the systems.

And this leads to another problem in transfer learning attack. If the victim doesn't only use our backdoor models, but retrained the badnet and then to apply. Will the backdoors still exist? The paper[3] gives the answer to this question. Fig. 6 shows the example.

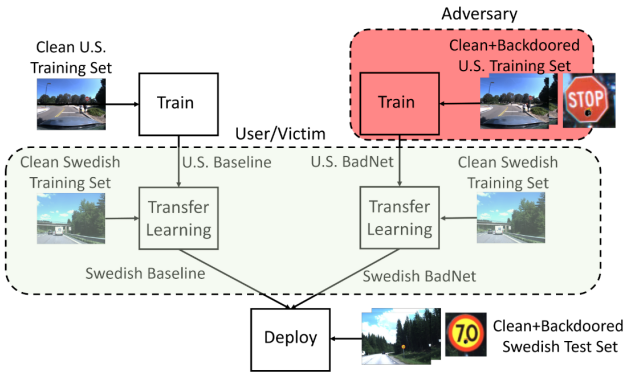

Fig. 6. Illustration of the transfer learning attack

5.1 Dataset: CIFAR-10

We implement our attack on the dataset CIFAR-10[12]. CIFAR-10 is a dataset with 60000 RGB images with size of 32x32 from 10 classes of airplane, automobile, bird, cat, deer, dog, forg, horse, ship, truck as Fig. 7 shows. The classes are completely mutually exclusive. The images contains the real-world complex objects that has large noise, which brings big difficulty for classification. The figure

The experiment is conducted on the four neural networks by generating some adversarial images with only one pixel-modification on the base of CIFAR-10.

5.2 Attack Goals and Strategy

For each of the attacks on the four types of neural networks, we randomly choose natural images from dirty data to conduct the attack. When conducting attack, to evaluate attack effectiveness better, we apply clean data from CIFAR-10 to train non-backdoor models as well.

Our target is to perturb some class to the other 9 target classes. But the scenarios of targeted and non-targeted attacks are considered as well. You may

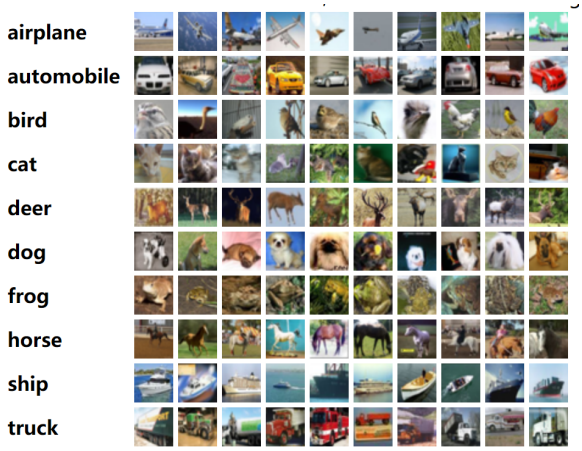

Fig. 7. CIFAR-10

see some method conduct attacks by increasing the number of pixels that can be modified to three and five[13]. We don't do this cause we consider more modification on pixels may reveal our attacks and make the success rate of classifying non-backdoor images down.

5.3 Attack Results

We evaluate our attack results by four metrics[13]:

- **Success Rate** - In the case of non-targeted attacks, it is defined as the percentage of adversarial images that were successfully classified by the target system as an arbitrary target class. In the case of targeted attack, it is defined as the probability of perturbing a natural image to a specific target class.
- **Adversarial Probability Labels (Confidence)** - Accumulates the values of probability label of the target class for each successful perturbation, then divided by the total number of successful perturbations. The measure indicates the average confidence given by the target system when mis-classifying adversarial images.
- **Number of Target Classes** - Counts the number of natural images that successfully perturb to a certain number (i.e. from 0 to 9) of target classes. In particular, by counting the number of images that can not be perturbed to any other classes, the effectiveness of non-targeted attack can be evaluated.
- **Number of Original-Target Class Pairs** - Counts the number of times each original-destination class pair was attacked.

Success Rate The success rates of each neural network shows the generalized effectiveness of the proposed attack through different network structures.

The Fig. 8 shows the results of conducting one-pixel attack on four different types of networks. Targeted/non-targeted indicate the accuracy of conducting targeted/non-targeted attacks. You can see the attack method performs better on LeNet5 and VGG16.

Adversarial Probability Labels (Confidence) By dividing the adversarial probability labels by the success rates, we can obtain the probability labels of target classes, which gives similar results as the success rates. The Fig. 9 shows that the one pixel attack generalizes well to this dataset, and fool the corresponding neural networks.

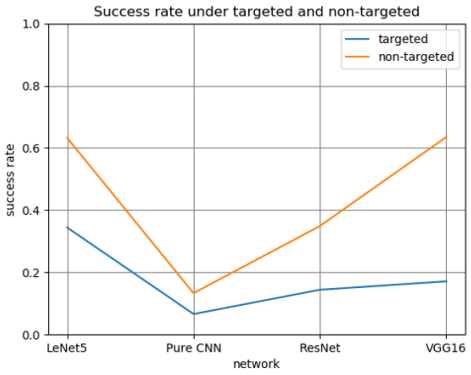

Fig. 8. success rate

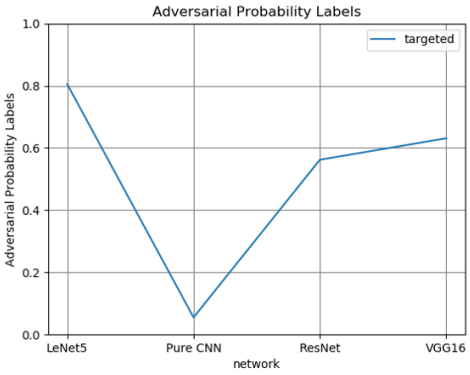

Fig. 9. adversarial probability labels

Number of Target Classes We measure the number of target classes under non-targeted attack. The number indicates how many target classes that with only one-pixel modification, a fair amount of natural images can be perturbed to. The graphs shows the percentage of natural images that were successfully perturbed to a certain number of target classes. The vertical axis shows the percentage of images that can be perturbed while the horizontal axis indicates the number of target classes. The Fig. 10 shows that only 1-pixel modification can perturb images to one or more target classes.

Number of Original-Target Class Pairs Original-Target class pairs means some pair of similar classes, e.g. cats and dogs. The heat-maps of the number of times a successful attack is present with the corresponding original-target class pair. The datasets have 10 classes so we present a 10X10 table. We can see the Fig. 11 shows some specific original target class pairs are much more vulnerable than others. For example, images of cat can be much more easily perturbed to

dog but can hardly reach the automobile. Some classes are more robust than others since their data points can be relatively hard to perturb to other classes such as the images of horse and automobile.

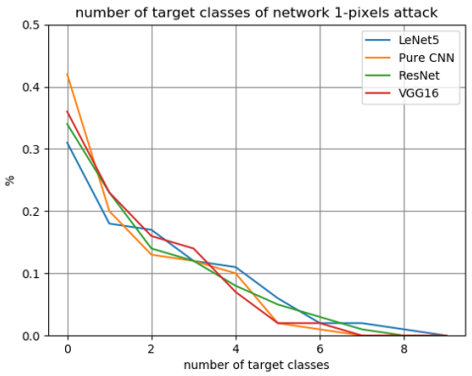

Fig. 10. Number of Target Classes

	0	1	2	3	4	5	6	7	8	9
0	0	22	32	40	45	28	22	20	34	27
1	13	0	3	4	6	7	8	6	8	23
2	14	7	0	23	22	21	20	16	11	16
3	22	26	31	0	36	**51**	36	23	21	22
4	23	21	41	39	0	42	34	31	23	43
5	30	12	24	**58**	41	0	45	32	16	24
6	24	12	34	35	36	35	0	17	15	25
7	4	2	5	8	17	15	7	0	2	6
8	**57**	32	29	14	16	17	28	9	0	**54**
9	24	13	13	16	13	16	11	21	12	0

Fig. 11. Number of Original-Target Class Pairs

Sample Heading (Fourth Level) The contribution should contain no more than four levels of headings. Table 1 gives a summary of all heading levels.

Table 1. Table captions should be placed above the tables.

Heading level	Example	Font size and style
Title (centered)	**Lecture Notes**	14 point, bold
1st-level heading	**1 Introduction**	12 point, bold
2nd-level heading	**2.1 Printing Area**	10 point, bold
3rd-level heading	**Run-in Heading in Bold.** Text follows	10 point, bold
4th-level heading	*Lowest Level Heading.* Text follows	10 point, italic

Displayed equations are centered and set on a separate line.

$$x + y = z \tag{4}$$

Please try to avoid rasterized images for line-art diagrams and schemas. Whenever possible, use vector graphics instead (see Fig. 12).

Theorem 1. *This is a sample theorem. The run-in heading is set in bold, while the following text appears in italics. Definitions, lemmas, propositions, and corollaries are styled the same way.*

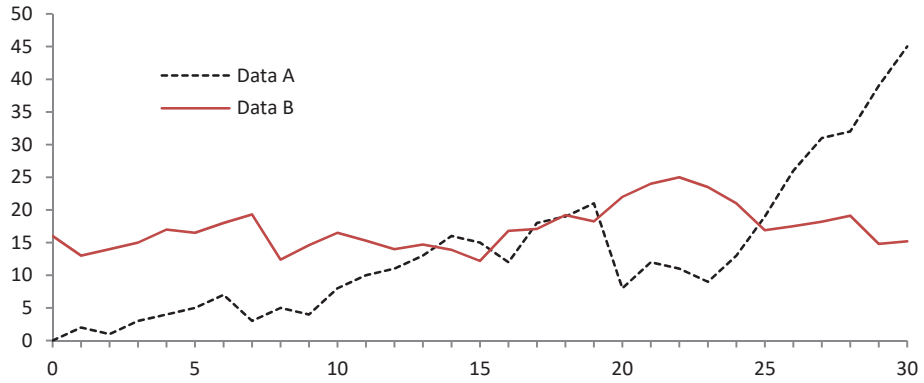

Fig. 12. A figure caption is always placed below the illustration. Please note that short captions are centered, while long ones are justified by the macro package automatically.

Proof. Proofs, examples, and remarks have the initial word in italics, while the following text appears in normal font.

For citations of references, we prefer the use of square brackets and consecutive numbers. Citations using labels or the author/year

References

1. Zijiang Zhu, Xiaoguang Deng, Yi Hu, Dong Liu, and Junshan Li. Implementation of face recognition based on deep learning framework caffe. In *2018 International Conference on Mechanical, Electronic, Control and Automation Engineering (MECAE 2018)*. Atlantis Press, 2018.
2. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
3. Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.
4. Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. 2017.
5. Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. Abs: Scanning neural networks for back-doors by artificial brain stimulation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 1265–1282, New York, NY, USA, 2019. Association for Computing Machinery.
6. Jiazhu Dai, Chuanshuai Chen, and Yufeng Li. A backdoor attack against lstm-based text classification systems. *IEEE Access*, 7:138872–138878, 2019.
7. Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019.

8. Y. Le Cun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson. *Handwritten Digit Recognition with a Back-Propagation Network*, page 396–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.
9. Yann Lecun, Leon Bottou, Y. Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324, 12 1998.
10. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
11. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
12. Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
13. Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.