TargetNet Backdoor: Attack on Deep Neural Network with Use of Different Triggers

Hyun Kwon
Korea Advanced Institute of Science and Technology
School of Computing
Daejeon, South Korea
khkh@kaist.ac.kr

Jungmin Roh
Ministry of National Defense
ROK Army Training & Doctrine Command
Daejeon, South Korea
carcassi6466@gmail.com

Hyunsoo Yoon
Korea Advanced Institute of Science and Technology
School of Computing
Daejeon, South Korea
hyoon@kaist.ac.kr

Ki-Woong Park*
Sejong University
Computer & Information Security
Seoul, South Korea
woongbak@sejong.ac.kr
*Corresponding author

ABSTRACT

Deep neural networks (DNNs) provide good performance in image recognition, speech recognition, and pattern analysis. However, DNNs are vulnerable to backdoor attacks. Backdoor attacks allow attackers to proactively access DNN training data to train it on additional data that are malicious, including a specific trigger. Normally, DNNs correctly classify normal data, but malicious data with a specific trigger trained by attackers can cause misclassification by DNNs. For example, if an attacker sets up a road sign that includes a specific trigger, an autonomous vehicle equipped with a DNN may misidentify the road sign and cause an accident. Thus, an attacker can use a backdoor attack to threaten the DNN at any time. However, in certain cases, when an attacker wants to perform a targeted attack, it may be desirable for the data introduced through the backdoor to be misrecognized as a particular class chosen by the attacker according to the position of a trigger. For example, if a specific trigger is attached to the top right side of the road sign, it may be misunderstood as a left-turn sign; if a specific trigger is attached to the top left side of the road sign, it may be misunderstood as a right-turn sign; and if a specific trigger is attached to the bottom left side of the road sign, it may be misunderstood as a U-turn sign. In this paper, we propose the TargetNet backdoor, which is designed to be misidentified as a particular target class chosen by the attacker according to a specific trigger location. The proposed method additionally trains the target classifier on the TargetNet backdoor data so that data with a trigger at a specific location will be misidentified as the target class selected by the attacker. We used MNIST and Fashion-MNIST as experimental datasets and Tensorflow as a machine learning library. Experimental results show that the proposed method applied to MNIST and Fashion-MNIST has a 100% attack success rate for the TargetNet backdoor and 99.17% and 91.4% accuracy rates on normal test data, respectively.

KEYWORDS

Machine learning, deep neural network, backdoor attack, targeted attack, poisoning attack, adversarial example.

1 INTRODUCTION

Deep neural networks (DNNs) [16] provide good performance for machine learning challenges such as image recognition, speech recognition, pattern analysis, and intrusion detection. However, the DNN has a vulnerability in that misclassification by the DNN can be caused through an adversarial example [17], poisoning attack [3], or backdoor attack [7]. An adversarial example attack [17] that adds some distortion to the input data can cause misclassification by the DNN without affecting the DNN. However, this attack requires a separate module, time, and a generation sequence to add the distortion in real time. The poisoning attack [3] is a method for reducing the accuracy of the model by adding training data that are malicious during the training process. However, this method reduces the overall accuracy of the model, which prevents attackers from choosing when and what specific data they want. To overcome this problem, the backdoor attack [7] is used; it is a method that causes misclassification by the DNN when the attacker wants, by using data that include a specific trigger. Backdoor attacks allow attackers to proactively access DNN training data to train it on additional data that are malicious, including the specific trigger. Normally, DNNs correctly classify normal data, but the malicious data with the specific trigger trained by attackers can cause misclassification by DNNs.

In certain cases, however, it may be desirable for the data introduced through the backdoor to be misrecognized as a specific class selected by the attacker, according to the position of the trigger. For example, if a specific trigger is attached to the top right side of a road sign, it may be misunderstood as a left-turn sign; if a specific trigger is attached to the top left side of the sign, it may be misunderstood as a right-turn sign; and if a specific trigger is

ICIIT 2020, Feb 19-22, 2020, Hanoi, Vietnam

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

attached to the bottom left side of the sign, it may be misunderstood as a U-turn sign.

In this paper, we propose the TargetNet backdoor, which is designed to be misidentified as the target class chosen by the attacker according to the trigger position. The proposed method additionally trains the target classifier on the TargetNet backdoor so that the data with the trigger at a specific location will be misidentified as the target class selected by the attacker. Thus, an attacker can set the desired time and target class by using positional triggers. The contributions of this paper are as follows.

- We propose the TargetNet backdoor method, which induces misclassification as the target class chosen by the attacker. We describe the systemic principles of the proposed method.
- We analyze the attack success rate and target class according to the location of the trigger for the TargetNet backdoor attack. We also analyze the performance of the proposed method based on the number of TargetNet backdoor samples.
- We verify the performance of the proposed method using MNIST [11] and Fashion-MNIST [19] datasets.

The rest of the paper is organized as follows. Section 2 describes related work. The proposed scheme is explain in Section 3. Section 4 describes the experiment setup and evaluates the results. A discussion of the proposed method is given in Section 5. Finally, Section 6 concludes the paper.

2 RELATED WORK

Barreno et al. [2] first classified security issues for machine learning into two categories: exploratory attacks and causative attacks. Exploratory attacks are a method of causing misclassification by modulating test data, without needing access to training data. An example of an exploratory attack is an adversarial example. A causative attack is an attack method that affects model learning by accessing training data. Typical examples of a causative attack are poisoning attacks and backdoor attacks.

2.1 Adversarial example

The adversarial example was first introduced by Szegedy et al. [17]. The adversarial example attack adds some distortion to the input value such that it is difficult for humans to identify but will cause misclassification by the DNN. As misclassification by a DNN in autonomous vehicles and medical services is a serious threat, research on adversarial examples is being actively conducted. Examples of ways to generate an adversarial example include the fast gradient sign method (FGSM) [6], iterative FGSM (I-FGSM) [9], Deepfool [13], the Jacobian-based saliency map attack (JSMA) [15], and Carlini–Wagner (CW) [4]. These methods compute the gradient for the output of the DNN to produce adversarial noise. The gradient is computed through backprogation, and in order to generate adversarial noise, the attacker must know the DNN's structure and parameters. The gradient calculation process is repeated to find the optimal adversarial noise by calculating the probability at the output layer. The CW method [4] is the state-of-the-art attack method and shows better performance than FGSM or I-FGSM. This method controls the distortion and attack success rate and shows a 100% attack success rate as a white-box attack.

2.2 Poisoning attack

A poisoning attack is an attack method that reduces the accuracy of the model by accessing the model's training process and adding data that are malicious. Biggio et al. [3] first proposed a poisoning attack method that adds malicious data to the training process on a support vector machine (SVM). This method aims to generate malicious data that can greatly reduce the SVM's accuracy, by calculating the gradient descent based on the characteristics of the SVM. Yang et al. [20] proposed a poisoning attack that reduces the accuracy of a neural network rather than that of an SVM. This method generates malicious data using a generative adversarial net (GAN). The target model is a discriminator, and the generator is a zero-sum method that finds the optimal malicious data by using the feedback from the discriminator. Mozaffari-Kermani et al. [14] proposed a systematic poisoning attack method in the medical domain. This method was used to demonstrate practical poisoning attacks using health-related datasets.

2.3 Backdoor attack

The backdoor attack trains certain patterns of triggers to be misclassified by a DNN if a specific trigger is added to the input data. As the backdoor does not affect the DNN when there is no trigger, normal input is correctly classified by the DNN. Gu et al. [7] proposed BadNets to inject such a backdoor into the training process. This attack method injects the backdoor in addition to the training data by creating the backdoor desired by the attacker with a trigger pattern and target label. This attack method demonstrates an attack success rate of about 99% on MNIST. Liu et al. [12] proposed the creation of a specific trigger that causes the largest misclassification by an internal neuron of the DNN without accessing training data. This method uses a strong association between a specific trigger and an internal neural net to attack the DNN even when training on a small quantity of backdoor data. Wang et al. [18] proposed an attack and a defense that could hide the trigger in the DNN. They used various image sets to demonstrate the attack success rate and the defense method. Clements and Lao [5] tampered directly with the hardware of the DNN to affect the running process. Their method degrades the model when it is triggered through backdoor circuits.

3 PROPOSED SCHEME

3.1 Threat model

The target model is a deep neural network [16] such as those used in autonomous vehicles, drones, image recognition, and voice recognition. We assume a white-box attack and that the attacker has access to training datasets for the target classifier. This is because it is necessary to additionally train the target classifier on the proposed backdoor dataset without accessing the existing normal training dataset. Under these assumptions, the proposed method can affect the training process using data and labels with specific triggers for the target classifier.

3.2 Proposed method

The purpose of the proposed method is to generate a backdoor, called TargetNet, that will induce misrecognition as a target class

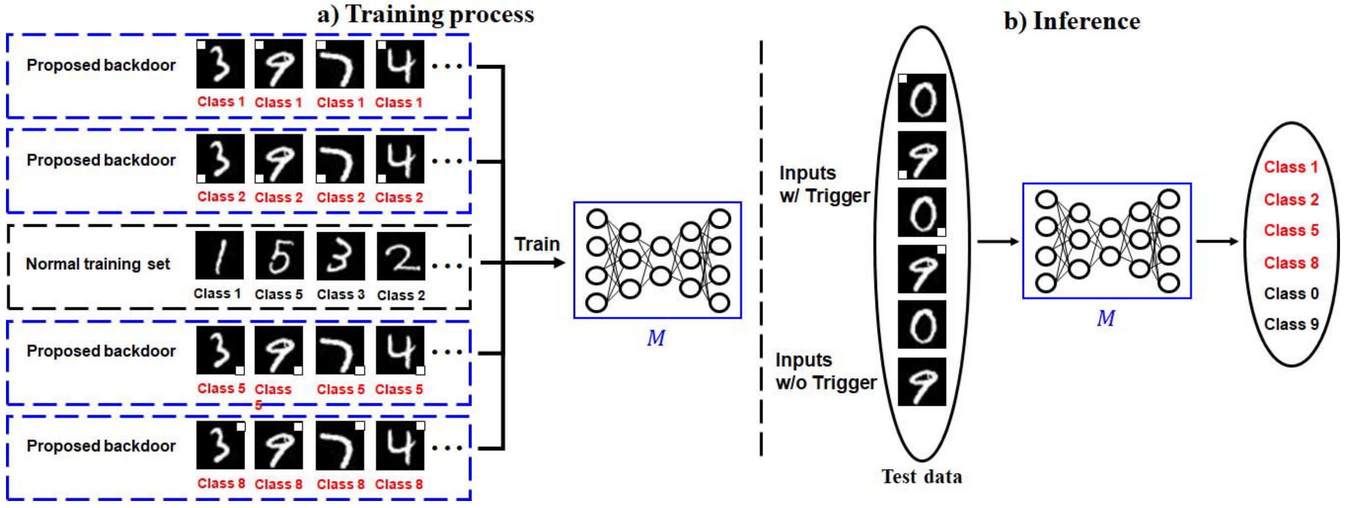


Figure 1: Overview of proposed backdoor attack. The trigger pattern is a white square. Target class "1" is labeled with the trigger pattern in the top left corner, target class "2" is labeled with the trigger pattern in the bottom left corner, target class "5" is labeled with the trigger pattern in the bottom right corner, and target class "8" is labeled with the trigger pattern in the top right corner.

chosen by the attacker. The proposed method additionally trains the target classifier with a TargetNet backdoor that includes triggers with different locations and different labels. Fig. 1 shows an overview of the proposed method. It consists of two steps: training the proposed backdoor during the training process and attacking during the inference process. In the process of training the backdoor, the target classifier additionally trains on the backdoor dataset during the training process. At this time, the trigger pattern, position, and target class of the proposed backdoor can be selected by the attacker. The target classifier trains by matching the target class corresponding to the backdoor data. The TargetNet backdoor data include triggers that have different positions and labels.

This method is mathematically expressed as follows. The operation function of a target classifier M is denoted as $f(x)$. The target classifier trains on the normal training dataset and the backdoor data. Given the normal training data $x \in X$, original class $y \in Y$, TargetNet backdoor data $x^{trigger} \in X^{trigger-i}$, and target class $y_i^{target} \in Y$, the target classifier trains on x with y and $x^{trigger-i}$ with y_i^{target} to satisfy the following equation:

$$f^{\text{enemy}}(x) = y \text{ and } f(x^{trigger-i}) = y_i^{target}.$$

In the attack during the inference process, the original class is correctly recognized for data that do not include a trigger. However, in the case of backdoor data that include a trigger, the target classifier incorrectly classifies the backdoor data as the target class chosen by the attacker. The mathematical expression is as follows. Let x_v be the new validation data. In the case of new validation data x_v without a trigger, the target classifier correctly recognizes them as the original class as follows:

$$f(x_v) = y.$$

However, in case of new validation data $x_{v-trigger-i}$ with a trigger, the target classifier misclassifies them as the target class chosen by the attacker as follows:

$$f(x_v^{trigger-i}) = y_i^{target}.$$

Details of the procedure for generating the proposed backdoor are given in Algorithm 1.

Algorithm 1 TargetNet Backdoor

```
Description: Original training dataset $x_j \in X$, TargetNet backdoor data $x_k^{trigger-i} \in X^{trigger}$, original class $y_j \in Y$, target class $y_i^{target} \in Y$, validation data $t$
TargetNet Backdoor:
    $X \leftarrow$ Matching dataset $(x_j, y_j)$
    $X^{trigger} \leftarrow$ Matching dataset $(x_k^{trigger-i}, y_i^{target})$
    Train the target classifier $M \leftarrow X + X^{trigger}$
    Record classification accuracy on the validation dataset $t$
    return $M$
```

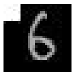
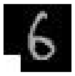
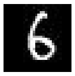
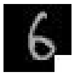
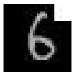
4 EXPERIMENT AND EVALUATION

This section shows the experimental configuration, experimental procedure, and experimental results to demonstrate the performance of the proposed method.

4.1 Experimental configuration

We used MNIST [11] and Fashion-MNIST [19] as datasets. MNIST is a representative handwriting dataset with 10 classes of black and white images of the numbers from 0 to 9. The number of pixels per image is 784 (28 × 28 × 1). This dataset has the advantage of being easy to train with. There are 60,000 training data and 10,000 test data. Fashion-MNIST, on the other hand, is more complex than MNIST; it is composed of 10 classes, including T-shirt, trouser, pullover, dress,

Table 1: Comparison of class results and classification scores for a TargetNet backdoor sample ("6"). Target class "1" is labeled with the trigger pattern in the top left corner, target class "0" is labeled with the trigger pattern in the bottom left corner, target class "2" is labeled with the trigger pattern in the bottom right corner, and target class "4" is labeled with the trigger pattern in the top right corner.

Description	Trigger at top left ("1")	Trigger at bottom left ("0")	Normal sample ("6")
			
Classification scores	[-3.63 **24.5** 5.07 2.81 0.12 -7.74 2.84 -7.33 7.75 -17.5]	[**20.1** -1.62 -1.86 3.04 -7.06 -3.4 3.18 1.46 -1.35 -1.09]	[2.02 -0.56 1.01 -0.39 -0.07 0.43 **13.6** -7.52 3.61 -8.23]
Description	Trigger at bottom right ("2")	Trigger at top right ("4")	
			
Classification scores	[0.91 1.25 **18.6** -3.61 1.97 -3.95 8.47 -12.6 2.25 -7.07]	[-6.34 -3.18 0.97 -8.51 **22.02** -3.21 3.85 -2.96 1.53 -10.8]	

sneaker, etc. The number of pixels per image is 784 (28 × 28 × 1). There are 60,000 training data and 10,000 test data.

In the experiment, the target classifier M used convolutional neural network (CNN) models [10] for MNIST and Fashion-MNIST. Table 4 in the appendix shows the CNN architecture. Table 5 in the appendix shows the parameters necessary for the training process for MNIST and Fashion-MNIST. Adam [8] was used as the optimizer. The initial constant of M was 0.01. As measured using normal test data, the friendly classifier and the enemy classifier have 99.25% accuracy on MNIST. On Fashion-MNIST, the friendly and enemy classifiers have 92.34% accuracy. In addition, we used the Tensorflow library [1], widely used for machine learning, and an Intel(R) i5-7100 3.90 GHz server.

4.2 Experimental procedure

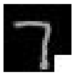
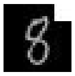
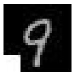
To ascertain the performance of the proposed method, we trained the target classifier by adjusting the ratio between the normal training data and the TargetNet backdoor data. We trained the target classifier using 10%, 33%, and 50% TargetNet backdoor data for all training datasets. The target class was set to a random one in the target classifier. As validation, we analyzed target classifiers with new test data with and without triggers. The positions for triggering TargetNet were set to the top left, top right, bottom left, and bottom right corners. The trigger pattern was a white square.

4.3 Experimental results

Table 1 shows a comparison of class results and classification scores for a TargetNet backdoor sample ("6"). In terms of recognizing the TargetNet backdoor, the class result depends on the trigger position. Thus, the attacker can determine the target class based on the trigger position.

Table 2 shows image samples for a TargetNet backdoor using MNIST. The trigger pattern was set to a rectangle of 7 × 7 pixels. The method can be applied by changing a rectangular sticker in the test data to the top left corner, bottom left corner, bottom right corner, or top right corner.

Table 2: Examples of TargetNet backdoor samples added to MNIST.

Proposed					

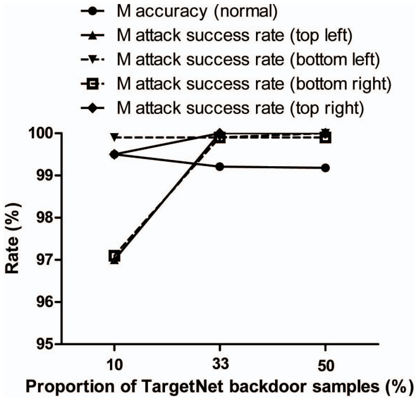
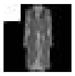
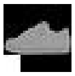
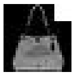
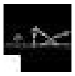


Figure 2: Accuracy and attack success rate on target classifier by proportion of TargetNet backdoor samples added to MNIST.

Fig. 2 shows the accuracy on normal samples in MNIST and the attack success rate of the TargetNet backdoor according to the number of TargetNet backdoor samples. In the figure, it can be seen that the accuracy for the normal test data remains nearly constant because the target classifier displays an accuracy of greater than 99% on the normal test data. For the TargetNet backdoor, the attack success rate against the target classifier is over 97%. Overall, as the number of TargetNet backdoor samples increased, the attack success rate increased and the accuracy decreased slightly. However, when the proportion of TargetNet backdoor samples was about

33%, the attack success rate on the target classifier was 100% and the accuracy on normal test data was maintained at 99.17%.

Table 3: Examples of TargetNet backdoor samples added to Fashion-MNIST.

Proposed					

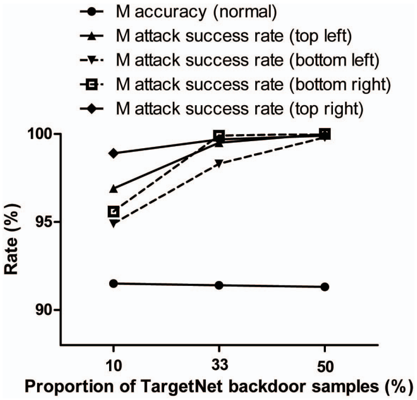


Figure 3: Accuracy and attack success rate on target classifier by proportion of TargetNet backdoor samples added to Fashion-MNIST.

Table 3 shows samples generated by the TargetNet backdoor for Fashion-MNIST. The trigger pattern consists of a rectangle (7 × 7). The trigger position of the TargetNet is set to the top left, bottom left, bottom right, and top right. The method can be applied by moving a sticker in the test data to the desired corner.

Fig. 3 shows the accuracy on normal samples in Fashion-MNIST and the attack success rate of the TargetNet backdoor according to the number of TargetNet backdoor samples. Similar to the results in Fig. 2, the target classifier displays an accuracy of greater than 91% on the normal test data, and it remains nearly constant. The reason that the accuracy is lower than that in Fig. 2 is that the model originally had about 92% accuracy for Fashion-MNIST. Similar to the results in Fig. 2, the proposed method displays an attack success rate of 100% on the target classifier, and the accuracy on the normal test data was maintained at 91.4%.

5 DISCUSSION

Attack considerations. Unlike the existing backdoor method, the proposed method has the advantage of inducing misidentification of a particular target class chosen by the attacker, according to the location of the trigger. It is also possible to attack using the proposed method if the trigger method changes only in a certain area of the test data, such as when using a sticker. For the trigger pattern, in this study a rectangle was used, but the attacker can set the trigger pattern as desired. Another advantage is that even if we train the TargetNet backdoor with a small quantity of data (about 33%), we can attack a target classifier with a success rate of greater than 99% while maintaining the accuracy on the normal test data.
Applications. This type of attack can be useful in military situations with enemy forces. For example, the proposed method can generate road signs modified by the attachment of a sticker with a specific trigger so that it will be misclassified as the target class chosen by the attacker. In addition, by the attachment of a specific trigger in the vehicle's camouflage or facial recognition system, the enemy can be misidentified as the target class.

6 CONCLUSION

In this paper, we have proposed the TargetNet backdoor, which is designed to be misidentified as a particular target class chosen by the attacker according to the location of a specific trigger. The proposed scheme additionally trains the target classifier on the TargetNet "backdoor data so that the data with the trigger at a specific location will be misidentified as the target class selected by the attacker. Experimental results show that the proposed method has a 100% attack success rate on the target classifier and an accuracy of 99.17% and 91.4% on the normal test data in MNIST and Fashion-MNIST, respectively. The proposed concepts can be applied to the audio and video domains in future studies. The topic of defense mechanisms for TargetNet backdoors remains as a challenge for future research.

ACKNOWLEDGMENT

This work was supported by Institute for Information & Communications Technology Promotion (IITP) grants funded by the Korean government (MSIT) (No. 2018-0-00420 and No. 2019-0-00426) and supported by National Research Foundation of Korea (NRF) grants funded by the Korean government (2017R1C1B2003957 and 2017R1A2B4006026).

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[2] Marco Barreno, Blaine Nelson, Anthony D Joseph, and JD Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

[3] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1467–1474. Omnipress, 2012.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

[5] Joseph Clements and Yingjie Lao. Hardware trojan attacks on neural networks. *arXiv preprint arXiv:1806.05768*, 2018.

[6] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[7] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[8] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *The International Conference on Learning Representations (ICLR)*, 2015.

[9] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *ICLR Workshop*, 2017.

[10] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[11] Yann LeCun, Corinna Cortes, and Christopher JC Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

[12] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. *NDSS*, 2018.

[13] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016.

[14] Mehran Mozaffari-Kermani, Susmita Sur-Kolay, Anand Raghunathan, and Niraj K Jha. Systematic poisoning attacks on and defenses for machine learning in healthcare. *IEEE journal of biomedical and health informatics*, 19(6):1893–1905, 2015.

[15] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 372–387. IEEE, 2016.

[16] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[17] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[18] 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. *Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks*, page 0, 2019.

[19] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[20] Chaofei Yang, Qing Wu, Hai Li, and Yiran Chen. Generative poisoning attack method against neural networks. *arXiv preprint arXiv:1703.01340*, 2017.

APPENDIX

Table 4: M model architecture for MNIST and Fashion-MNIST.

Layer type	Shape
Convolutional+ReLU	[3, 3, 32]
Convolutional+ReLU	[3, 3, 32]
Max pooling	[2, 2]
Convolutional+ReLU	[3, 3, 64]
Convolutional+ReLU	[3, 3, 64]
Max pooling	[2, 2]
Fully connected+ReLU	[200]
Fully connected+ReLU	[200]
Softmax	[10]

Table 5: M model parameters for MNIST and Fashion-MNIST.

Parameter	Value
Learning rate	0.1
Momentum	0.9
Batch size	128
Epochs	50