An exploration of PCA based data filter in Neural Network security

Name: Chengrui Zhang   Student Number : 2019233183
Name: Ziwen Li   Student Number : 2020231095
School of Information Science and Technology
ShanghaiTech University, Shanghai, P. R. China
E-mail: {zhangchr, lizw}@shanghaitech.edu.cn

I. Introduction

With the explosive growth of existing computing power, deep neural networks (DNNs) are being used more and more widely in processing various machine learning (ML) tasks, such as image classification, natural language processing and autonomous driving. Although DNNs have shown good performance in network classification tasks, they have recently been shown to be particularly unstable in data adversarial disturbances[1], and there are serious security risks.

Currently, there are two main types of attacks on neural networks[2]. The poisoning attacks can introduce the adversarial samples in the training phase, with the purpose of causing misclassification of data during the testing phase. Besides, the evasion attacks can add strategic perturbations to test inputs in order to fool the existing ML classifiers, which are well trained on clean samples. Considering a state-of-the-art deep neural network, which can generalize object recognition tasks well, we expect this network to be robust to when clean images with small disturbances are sent to it, which means small disturbances cannot change the correct object category of the image. However, when small imperceptible non-random disturbances which is generated by optimizing the input to maximize the prediction error are applied to the test image, the prediction result of the network will always be changed. In this way, the accuracy of this deep neural network will decrease.

In our project, we focus on the evasion attacks, in which the added perturbations to test samples are not easily recognized by human hearing and vision. In this way, evasion attacks can fool the most advanced classifiers and make the normally trained model output high-confidence error predictions. This phenomenon is called adversarial attacks. Adversarial attacks are a huge obstacle before deploying DNNs in production. For example, adversarial attacks against autonomous driving may cause serious traffic accidents. Therefore, defense against adversarial attacks is essential.

Attacks and defenses related to deep neural networks are becoming one of the hotspots in the field of machine learning. Several powerful approaches have been proposed to generate effective adversarial examples in the literature. For example, Fast Gradient Sign Method(FGSM)[3], Carlini and Wagner (C&W)[4], DeepFool[5] and Physical Attack Method[6]. At the same time, there are also many studies on defense methods, such as JPG Compression[7], Input Deformation and Augmentation[8] and Dimension Reduction Method[2].

In this report, we review 1) the attack methods of FGSM, DeepFool and Physical method and 2) the defense methods of JPG Compression, Input Deformation and Augmentation, and Dimension Reduction Method. However, in the review process we find some disadvantages, which contain insufficient dataset, simple deep learning method, insufficient theory illustration and defaults of PCA defense method. We propose several method to handle these problems, and contributions are as follows:

- 1) Expand the dataset from MNIST to CIFAR10.
- 2) Explore the performance of different defense methods under the attack of FGSM and DeepFool methods.
- 3) Investigate the reasons of failure of SVD defense method.
- 4) Propose a boost and PCA tuning method to achieve a better defense capability of neural network.

II. Overview of existing work

A. Attack methods

The attack methods started form different motivations but had the same goal – generating adversarial examples that can fool the ML or DL classifiers but can still be identified correctly by human eyes. In other words, these methods

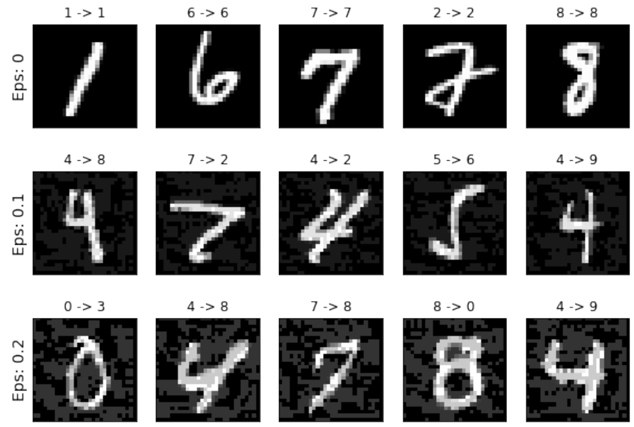


Fig. 1: Results of FGSM on MNIST dataset for different attack intensities

generate a minimal perturbation r that is sufficient to change the estimated label $\hat{k}(x)$:

$$\begin{aligned} \Delta(x;\hat{k}) &:= \min_{r} \|r\|_2 \\ subject\ to\ &\hat{k}(x+r) \neq \hat{k} \end{aligned} \tag{1}$$

where x is an image and $\hat{k}(x)$ is the estimated label. We call $\Delta(x;\hat{k})$ the robustness of $\hat{k}$ at point x. The robustness ρ_{adv} of classifier $\hat{k}$ is then defined as:

$$\rho_{adv}(\hat{k}) = \mathbb{E}_x \frac{\Delta(x;\hat{k})}{\|x\|_2} \tag{2}$$

where $\mathbb{E}_x$ is the expectation over the distribution of data.

In this section, we introduce the theories and processes of FGSM, DeepFool and Physical Attack Method and show the attacked result of FGSM at MNIST dataset and that of DeepFool at CIFAR10 dataset.

1) Fast Gradient Sign Method: FGSM is a network-specific-method that based on gradient descent. It provides a network-specific perturbations to disturb the specific network and achieves a pretty good attacking results, which means that it can significantly reduce the accuracy of attacked neural network. The theory of this method is simple but effective, which tries to find a perturbation η to best disturb the output value of neural network. Consider the dot product between a weight vector ω and an adversarial example $\tilde{x}$:

$$\omega^T \tilde{x} = \omega^T x + \omega^T \eta \tag{3}$$

$$\eta = \epsilon sign(\Delta_x J(\theta, x, y)) \tag{4}$$

The perturbation of FGSM is chose by Eq. 4, where ϵ is the attack intensity, θ is the parameters of a model, x is the input to the model, y is the targets associated with x and $J(\theta, x, y)$ is the cost used to train the neural network. In this way, we can find an suitable η that has the best attack performance.

The results of FGSM on MNIST dataset are shown in Fig. 1. Here, Eps: 0 means that there is no perturbation added into test data. Eps: 0.1 and Eps: 0.2 mean the FGSM in different attack intensities. The sub-title of each sub-figure shows the true label and the predicted label like this: True label $\rightarrow$ predicted label. In this figure we can find that, even a small ϵ fools the network successfully, and the perturbations can be easily caught by human eyes when $\epsilon = 0.2$, which forces us to use small perturbations to attack the model.

2) DeepFool: The DeepFool algorithm can efficiently compute perturbations that fool deep networks. The model is trained on the clean images, then the input of the test images are added the perturbations computed by DeepFool algorithm. Through comparing the test accuracy of different neural networks, the robustness of the corresponding classifiers can be reliably quantified.

Since a multiclass classifier can be viewed as aggregation of binary classifiers, it is sufficient for us to only analyze the DeepFool algorithm for binary classifiers. First, assume $\hat{k}(x) = sign(f(x))$, where f is an arbitrary scalar-valued image classification function satisfying $f : \mathbb{R}^n \rightarrow \mathbb{R}$. The classification boundary is denoted by $f(x) = 0$. When f

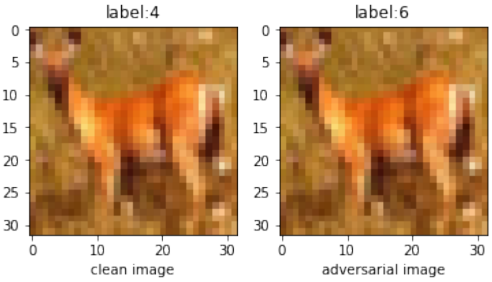


Fig. 2: Result of DeepFool on CIFAR10

is an affine classifier, which means $f(x) = w^T x + b$, in this case it can be easily seen that the robustness of f at point x_0 $\Delta(x; f)^2$ is equal to the distance from x_0 to classification boundary. The minimal perturbation to change the classifier's decision corresponds to the orthogonal projection of x_0 onto classification boundary. It is given by the closed-form formula:

$$\begin{aligned} r_*(x_0) &:= arg \min \|r\|_2 \,, \; subject \; to \; sign(f(x_0 + r)) \neq sign(f(x_0)) \\ &= -\frac{f(x_0)}{\|\omega\|_2}\omega \end{aligned} \tag{5}$$

Specifically, at each iteration, f is linearized around the current point x_i and the minimal perturbation of the linearized classifier is computed as:

$$arg \min_{r_i} \|r_i\|_2 \,, \; subject \; to \; f(x_i) + \nabla f(x_i)^T r_i = 0 \tag{6}$$

The perturbation r_i at iteration i of the algorithm is computed using the closed form solution in format (5), and the next iteration x_{i+1} is updated. The algorithm stops when x_{i+1} changes the sign of the classifier. Here is an example of DeepFool from dataset CIFAR10 in the figure 2. From the figure, we can clearly see that the difference between the images before and after the adversarial attack is difficult to distinguish by human eyes, but the classifier has made a wrong classification of the images after the adversarial attack.

3) Physical Attack Method: Physical adversarial attack is a novel targeting object detection model. Instead of simply printing images, this method manufacture real metal objects that could achieve the adversarial effect. Given an input image x, add perturbations δ which is generated by C&W attack method on it to generate the digital adversarial image

$$x' = x + M\delta \tag{7}$$

M denotes the mask which defines the area where the perturbation can be added. For an input image, it can have several masks to generate several different attack images. To improve the robustness of physical attack method against varying physical conditions including distance, angle, background and illumination, the image transformation function T is introduced to simulate various factors in the physical world. The final simulated image is denoted as:

$$X_i = T(x', r_i) \tag{8}$$

where r_i denotes the i-th real background image.

B. Defense method

For different attack methods, there are already many existing works to explore the corresponding defense methods, or find general defense methods to defend against multiple attacks. In this section, we focus on the JPG compression, input deformation and augmentation and PCA method.

1) JPG Compression: Since most image classification data sets are composed of JPG images, the JPG image compression on adversarial image classification is purposed. In paper [7], the OverFeat network is used to train on images from the 2012 ImageNet training set. For each image x in the ImageNet validation set, perform the following steps:

- 1) Compute $Adv_{\varepsilon}(x) = x + \eta_{\varepsilon}(x)$ using the Fast Gradient Sign Method, with ε is the attack intensity and $\eta_{\varepsilon}(x)$ is the perturbations generated by FGSM;
- 2) Compress the adversarial image by computing $JPG(Adv_{\varepsilon}(x))$ and save the compressed image;
- 3) Compute the OverFeat network predictions for all images: original image x, adversarial image $Adv_{\varepsilon}(x)$ and the compressed image $JPG(Adv_{\varepsilon}(x))$.

The research results show that for the FGSM attack method, when the perturbation amplitude is small, JPG image compression can usually prevent the degradation of classification accuracy by a large amount, and the defense effect is more obvious, but not always. When the perturbation amplitude of FGSM gradually becomes larger, only using JPG compression is not enough to prevent the degradation of classification accuracy, and the defense effect becomes poor.

2) Input Deformation and Augmentation Method: This method is based on the observation that certain input deformation and augmentation methods will have little or no impact on DNN model's accuracy, but the adversarial attacks will fail when the maliciously induced perturbations are randomly deformed. This mitigation method is attack independent, which means it does not require any knowledge of the adversarial attacks. And the method is also model independent, which means it does not require additional training, parameter fine tuning, or any structure modifications of the target DNN model. So it has excellent generality and usability.

In paper [8], the combination of five variables which are able to hold the high-level abstractions of the input data are chosen to be transformed and augmented:

- 1) $width - shift(dw)$, the horizontal location of every element is shifted;
- 2) $height - shift(dh)$, the vertical site of every component is shifted;
- 3) $zoom(dz)$, focus on a specific part of the input data;
- 4) $rotation(dr)$, maps the position of an element in the input onto a position by rotating it through an angle;
- 5) $shear(ds)$, displaces each point in the fixed direction.

Next, the input data which has been pre-processed will be sent to the DNN with trained parameters to infer each viriation.

Finally, the outputs of the DNN are sent to the output decision component to determine the ultimate result from the decisions of the DNN. For each input, the softmax function of the DNN would regulate the output to a categorical probability, which tells us the probability that any of the classes are true. Then, we calculate the summation of all the probabilities for every category and choose the one that has the biggest probability as the DNN result. Formally, suppose p_{ik} denotes the probability of the ith variation belongs to category k, the size of the total classes is N, then the predicted label will be illustrated as equation (9).

$$o = j, \ where \sum_{i=1}^{m} p_{ij} \geq \sum_{i=1}^{m} p_{ik}, \ \forall k \in [1, N] \ and \ k \neq j \tag{9}$$

3) Principle Component Analysis (PCA): PCA method uses the dimension reduction method as a defense against evasion attacks especially the FGSM attack. This method is started from an observation that perturbations caused by attack methods are very small. Therefore, they can filter those perturbations by abstracting principle components and ignoring trivial components.

The process of this method, called DRtrain, is shown below:

- 1) Compute the PCA by Algorithm. 1.
- 2) Do the DRtrain shown in Algorithm. 2.
- 3) After getting the well-trained model, this method implements the inference by projecting the test set onto training set and uses the projected test data to do the forward process of the well-trained model.

III. Criticism of the existing work

The papers shown above are all great works, either the attack methods or the defense methods. However, their methods only work well under the specific conditions or have some limitations, especially the defense methods. In this part, we pick a few of disadvantages of their works and try to propose some ideas to solve these problems in Sec. IV.

Algorithm 1 $[X_d] = PCA(X, d)$

```
Require: X, d; # $X : D \times N$ data matrix, $d$ : number of principle components
Ensure:
1: Compute the mean value of the data matrix and centralize the data matrix: $X_{mean} = \frac{1}{n}X\mathbf{1}, X_{center} = X - X_{mean}$
2: Compute the sample covariance matrix $C$: $C = X_{center}X_{center}^T$
3: Find the eigenvalue decomposition of $C$: $C = U\Sigma U^T$
4: Find the principle $d$ components of matrix $U$ and find the principle data: $U_d = U_{:,1:d}, X_d = U_dU_d^TX$
5: return $X_d$, the principle data.
```

Algorithm 2 $[f_d] = DRtrain(f_d^0, \theta_d, d, X_d, Train)$

```
Require: $f_d^0, \theta_d, d, X_d, Train$; # $f_d^0$ : the initial, untrained classifier, $\theta_d$ : the parameters used in the training of $f_d$,
    $d$ : the reduced dimension to be used, $Train$ : the algorithm used to train classifiers of the desired class
Ensure:
1: Use PCA to find the matrix $U_d$ of the top $k$ principal components used to reduce the dimension of $X_{train}$
2: Compute the projected training set $U_dU_d^TX_{train}$
3: Let $f_d = Train(f_d^0; \theta_d; U_dU_d^TX_{train})$
4: return $f_d$, the well-trained model.
```

A. Insufficient Dataset and Simple Deep Learning method

In paper [2], they validate their method on the MNIST dataset with a very simple full connected network (3 layers with 100-100-10 neurons each layer). Although the experimental results of their methods are pretty good, which mean that their defense method can recover the accuracy greatly in MNIST and their network, these results are not sufficient. The reason is that MNIST dataset is just a toy to do small-scale validation, but is not reliable to judge whether their method is right or not. Therefore, we realize their method in CIFAR10 and LeNet & AlexNet, which is shown in Sec. IV.

B. Insufficient Theory Illustration

In paper [7], without any explanation, they simply give us a conclusion that, the SVD decomposition cannot defends the FGSM attack method. Since the reason of this conclusion is important for readers to understand their ideas, we implementing some experiments to illustrate this conclusion.

The experiments contain 3 steps: 1) do the SVD decomposition of input data X; 2) do the SVD decomposition of the adversarial data (attacked by FGSM or DeepFool); 3) compare the change of singular value between clean data and attacked data. The simulation results of FGSM method are shown in Fig. 3. Besides, we also investigate the singular value differences of DeepFool attack method on CIFAR10, the results are shown in Fig. 4.

From these figures, we can find that 1) For FGSM method: The absolute singular value differences mainly happened in the biggish singular values and the relative difference on each singular value is very large (at least 30.2% percentage). This result shows that FGSM method attacks all the singular values at the similar degree, therefore SVD method cannot defend this attack; 2) For DeepFool method: DeepFool mainly attacks the Red channel of CIFAR10 figures, and in each channel, the absolute difference of each singular values is nearly at the same level. Fortunately, the relative differences in green and blue channels at biggish singular values are small, therefore the SVD method might be possible to defend the DeepFool attack. However, after we implement the DeepFool attack at CIFAR10 and utilize the SVD method to defend, the performance is not quite good, which is shown in Table. I.

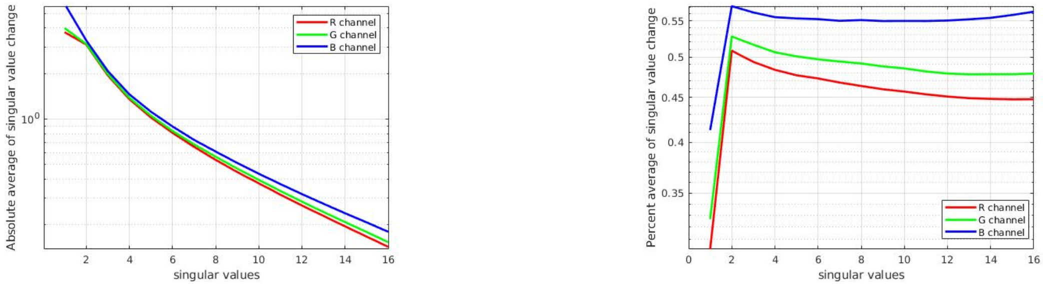
Based on the simulation results, we can reasonably draw a conclusion that, the SVD method cannot defend either the FGSM or DeepFool methods.

C. Defaults of PCA method in defense methods

In paper [2], they use the PCA method to do the dimension reduction of input data. However, this method reduces the accuracy of neural network due to the reduction of the dimensions of figure characteristics. We implement their method on AlexNet and observe the accuracy degradation of DL model with different characteristic dimensions. The result is shown in Fig. 5 (a). From the simulation results we find that the reduction of accuracy is associated with

TABLE I: AlexNet accuracy of different characteristics dimensions for DeepFool attacked data

Preserved dimensions	*Top-32 dimensions*	*Top-10 dimensions*	*Top-5 dimensions*	*Top-3 dimensions*
Accuracy with clean data	78.8%	75.6%	72.4%	67.0%
Accuracy with attacked data	9.6%	25.5%	29.5%	29.9%
Accuracy with attacked data and SVD method	9.6%	27.3%	12.9%	29.6%



(a)

(b)

Fig. 3: The singular value differences in absolute (a) and relative (b) modes between clean data and FGSM attacked data on CIFAR10

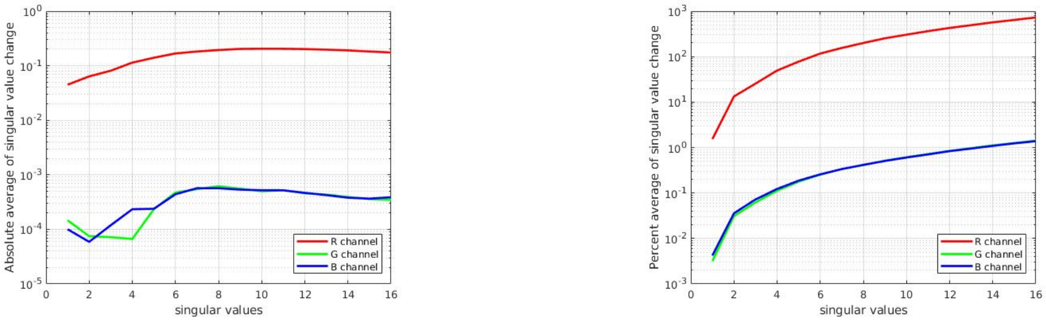
the decrease of characteristics dimensions. Therefore, it is crucial for this method to choose a suitable d-principle components to either achieve a high accuracy and defense the attack. Hence we utilize either a boost method and a heuristic method to both improve the accuracy and reliability of Lenet network on CIFRA10 dataset in Sec. IV.

IV. New contribution

Our new contributions focus on the defense methods, trying to fix the problems mentioned in Sec. III. In this section, our contributions contain 3 parts:

1) Illustration of SVD method: We give reasonable illustrations than paper [7] to explain why the SVD method cannot defend FGSM method. In addition, we also investigate the SVD method's performance at another attack method – DeepFool. The dataset we use are CIFAR10 dataset, which is much better than the MNIST dataset. The details are shown in Sec. III.B.

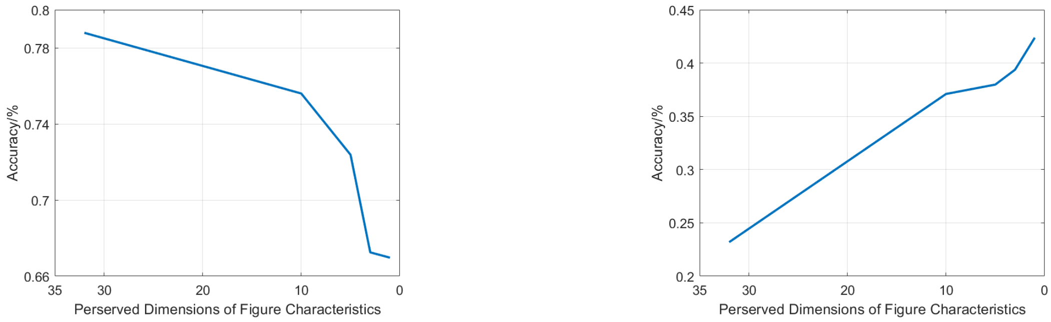
2) Boost method: We utilize the Boost method in machine learning to solve the accuracy degradation problem. Since the reduction of accuracy is associated with the decrease of characteristics dimensions, which is shown in Fig. 5 (a), and Fig.5 (b) shows that the accuracy of AlexNet on the attacked CIFAR10 (attack by FGSM method) rises with the decrease of characteristics dimensions, we find that there exists a trade-off between high inference accuracy and model reliability in PCA method. **In other words, preserve more dimensions means high accuracy**



(a)

(b)

Fig. 4: The singular value differences in absolute (a) and relative (b) modes between clean data and DeepFool attacked data on CIFAR10



(a) (b)

Fig. 5: The accuracy of AlexNet on CIFAR10 (a) and attacked CIFAR10 (b) with different dimensions of figure characteristics

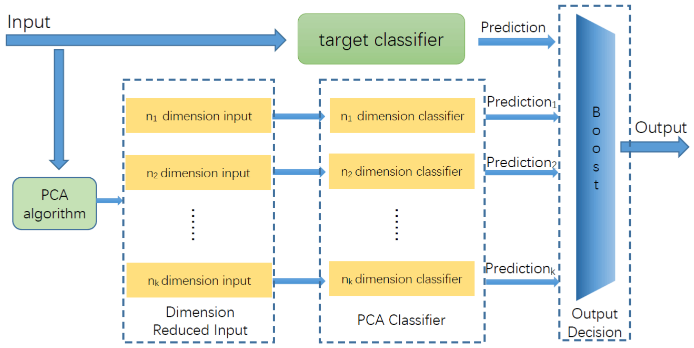


Fig. 6: The process of proposed boost method

but low defense capability, and vice versa for preserving less dimensions.
Based on this observation, we generate several PCA classifiers and use boost method in order to get high accuracy and high defense capability. This method contains the following steps, which is also shown in Fig. 6:

- 1) Generate k inputs with different preserved characteristics dimensions by Algorithm. 1.
- 2) Train k different PCA classifiers by Algorithm. 2.
- 3) In the inference process, use the k PCA classifiers and the origin classifier to predict the test input. Then, accumulate their results and find the location of maximum probability, which is the final result.

3) PCA tuning method: This method aims to pick the proper k different PCA classifiers. Since there are too many characteristics dimensions, and the principle dimensions are under 32 as shown in Fig. 7. Based on the results of Fig. 5, the preserved dimension larger than 32 may have better accuracy but worse defense capability, and vice versa for the preserved dimension less than 32. Therefore, here we intuitively use the top 8, 16, 32, 48 and 64 dimensions to build our classifiers.

V. Numerical results

A. Illustration of SVD method

The results of this part is shown in Sec. III.B.

B. Boost method and PCA tuning method

In this part, we implement our proposed methods on CIFA10 with LeNet, testing the performance of our methods under the FGSM and DeepFool attack. The network we trained has *training batch size = 100, learning rate = 0.001, epoches = 20, optimizer = Adam and loss function = cross entropy loss.* After training the origin network model, we then train the LeNet classifiers with different PCA methods, and finally use the boost method to defend the FGSM

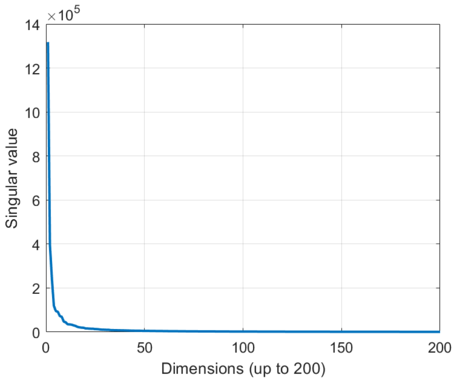


Fig. 7: The singular value of the top-200 dimensions

(*attack intensity = 0.01*) and DeepFool attack. The results of our method for defending FGSM and DeepFool method is shown in Table. II and Table. III respectively.

TABLE II: Performance of our method against FGSM

Preserved dimensions	*All dimensions*	*Top-64 dimensions*	*Top-48 dimensions*	*Top-32 dimensions*	*Top-16 dimensions*	*Top-8 dimensions*	*Boost*
Accuracy of clean data	63.25%	60.65%	56.43%	50.19%	42.69%	38.40%	54.76%
Accuracy of attacked data	8.16%	7.94%	9.15%	7.33%	5.01%	4.26%	26.77%
Adversarial success rate	87.10%	86.91%	83.79%	85.40%	88.26%	88.91%	51.11%
Accuracy of attacked data with our method	45.80%	42.25%	40.19%	42.59%	39.97%	36.85%	53.15%
Defense success rate	68.32%	65.09%	65.65%	82.27%	92.78%	95.46%	94.25%

TABLE III: Performance of our method against DeepFool

Preserved dimensions	*All dimensions*	*Top-64 dimensions*	*Top-48 dimensions*	*Top-32 dimensions*	*Top-16 dimensions*	*Top-8 dimensions*	*Boost*
Accuracy of clean data	63.25%	60.65%	56.43%	50.19%	42.69%	38.40%	54.76%
Accuracy of attacked data	8.16%	34.14%	34.62%	28.90%	15.39%	16.02%	30.63%
Adversarial success rate	87.10%	43.73%	38.65%	42.42%	63.95%	58.28%	50.40%
Accuracy of attacked data with our method	45.80%	48.63%	44.25%	39.28%	24.73%	24.07%	47.20%
Defense success rate	68.32%	54.68%	44.15%	48.76%	34.21%	35.97%	53.23%

Here, the adversarial success rate and defense success rate are defined as:

$$Adversarial\ success\ rate = \frac{Accuracy_{clean} - Accuracy_{perturbation}}{Accuracy_{clean}} \tag{10}$$

$$Defense\ success\ rate = \frac{Accuracy_{PCA} - Accuracy_{perturbation}}{Accuracy_{clean} - Accuracy_{perturbation}} \tag{11}$$

From these results we can find that 1) For the FGSM method, our boost method can achieve a high defense success rate (94.25%) and a good accuracy recovery rate (97.06%). This result shows that our method contains either the high reliability of low preserved data dimensions and the high accuracy of high preserved data dimensions. 2) For the DeepFool method, the result is not quiet good since the defense success rate is only 53.23%. However, we think this method will be effective if we choose better hyper-parameters and more suitable preserved dimensions.

Important: It is worth noting that, due to the time and computation resources limitation, the maximum of LeNet in clean CIFAR10 data is 63.25%, which causes the small accuracy in Table. II and Table. III. Therefore, in this work we mainly focus on the adversarial success rate and defense success rate, which can

reveal the performance of attack and defense method. If we have more time or computing resources, the simulation results will be more fantastic.

VI. Conclusion

In this project, we review the topic of neural network security, which aims to find attack (defense) methods to attack (protect) neural network. For the attack methods, we review the fast gradient sign method, DeepFool and physical attack method. These methods generate perturbations to attack the weak points of neural networks (gradient attack for FGSM and DeepFool, and sensitive points attack for physical attack method), and try to minimize the perturbations in order to only fool the machine. For the defense methods, we review the JPG compression, input deformation and augmentation method, and principle component analysis method, which try to use SVD, PCA or deformation methods to filter perturbations caused by attack method.

Besides, we pick 4 simple but important disadvantages of these defense methods which contain insufficient dataset, simple deep learning method, insufficient theory illustration and defaults of PCA defense method. In order to solve these problems, ***we expand the dataset from MNIST to CIFAR10, explore the performance of different defense methods under the attack of FGSM and DeepFool methods, investigate the reasons of failure of SVD defense method and propose a boost and PCA tuning method to achieve a better defense capability of neural network.***

The simulation results show that the boost method is suitable for defending FGSM method (94.25% defense success rate) and need to be modified to defend DeepFool method (53.23% defense success rate).

The relationship between this project and our curriculum is that, we investigate the effect of SVD and PCA reduction methods with different preserved dimensions in neural network security field. Besides, we propose a boost method based on the characteristics of PCA reduction to get better defense performance, and the simulation results show that, in FGSM attack method, our method utilizes this characteristic greatly and achieves a great performance.

References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[2] A. N. Bhagoji, D. Cullina, and P. Mittal, "Dimensionality reduction as a defense against evasion attacks on machine learning classifiers," *arXiv preprint arXiv:1704.02654*, vol. 2, 2017.

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples. iclr. 2015," *arXiv preprint arXiv:1412.6572*, vol. 1, 2015.

[4] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 3–14.

[5] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.

[6] K. Yang, T. Tsai, H. Yu, T.-Y. Ho, and Y. Jin, "Beyond digital domain: Fooling deep learning based recognition system in physical world," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 1088–1095.

[7] 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.

[8] P. Qiu, Q. Wang, D. Wang, Y. Lyu, Z. Lu, and G. Qu, "Mitigating adversarial attacks for deep neural networks by input deformation and augmentation," in *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2020, pp. 157–162.