REPRODUCIBILITY REPORT FOR ”ENHANCING ADVERSARIAL DEFENSE BY K-WINNERS-TAKE- ALL”

Chang Li, Donggang Jia, Xuanru Chen
School of Electronics and Computer Science
University of Southampton
{cl5y19,dj1u19,xc2a19}@soton.ac.uk

ABSTRACT

In this report, we reproduced the results in the paper named Enhancing Adversarial Defense by k-Winners-Take-All. After training on two datasets, we tested the robustness performance of k-WTA function compared with ReLU function under several white-box attacks. We concluded that k-WTA might help defend white-box attack, but it made the model harder to train.

1 PAPER SUMMARY

Most of deep neural networks work poor when attacked by *adversarial examples* (Szegedy et al., 2013), which can mislead the network by adding a certain hardly perceptible perturbation in the input. As eliminating adversarial examples is intrinsic difficult, defensing of this approach is a big challenge.

Evading adversarial examples may be a possible way to defense the attack. However, the defense approaches in recent researches remain vulnerable and they are mostly based on obfuscated gradients, which can still be approximated (Lin et al., 2019; Xie et al., 2018; Athalye et al., 2018).

The paper proposes to overcome the challenge by making the gradient *undefined* rather than obfuscated. Their method is to replace the activation functions by the k-Winners-Take-All (k-WTA) to produce a discontinuous gradient in the network. In the paper, by applying this method, the networks can be adversarial robustness and trained successfully at the same time. k-WTA retains the k largest values of an $N \times 1$ input vector and sets all others to be zero before feeding the vector to the next network layer, namely

$$\phi_k(y)_j = \begin{cases} y_j, \; y_j \in \{k \; largest \; elements \; of \; y\} \\ 0, \; Otherwise \end{cases} \tag{1}$$

Here $\phi_k : \mathbb{R}^N \to \mathbb{R}^N$ is the k-WTA function (parameterized by an integer k), $y \in \mathbb{R}^N$ is the input to the activation, and $\phi_k(y)_j$ denote the j-the element of the output $\phi_k(y)$.

In stead of specifying k in each layer, the paper introduces a parameter $\gamma \in (0, 1)$ called $sparsity\ ratio$. If a layer has an output dimension N, then its k-WTA activation has $k = \lfloor \gamma \cdot N \rfloor$. Following the paper, we use a fixed γ as a hyperparameter.

2 THE EXPERIMENTAL METHODOLOGY

This report re-uses the code provided by the authors, and completes the code in the evaluation part. We only considered white-box attack methods to evaluate the robustness, as it produced a more considerable results in the paper. 2 different data sets, 4 types of training method (Madry et al., 2017a; Zhang et al., 2019; Shafahi et al., 2019) and 3 types attackers (Madry et al., 2017b; Carlini & Wagner, 2017; Dong et al., 2018) are used to evaluate the robustness of k-WTA networks, which have the best attack performance in the chosen paper, the details are shown in the Table 1. Besides, the networks trained with no attackers are also implemented and compared. All our works are based on ResNet18 network structure.

The implementation details can be written as the following steps:

1. Loading data from data sets, we setting the training batch size at 256.

Table 1: Experiment Details

Training method	Attacking method	Data sets
natural (non-adversarial)	Projected Gradient Descent (PGD)	CIFAR10
adversarial training (AT)	C&W attack	SVHN
TRADES	Momentum Iterative Method (MIM)	
free adversarial training (FAT)		

2. Construct a neural network. In this step, ResNet network with ReLU activation function is produced as a baseline method. Then different k-WTA activation functions are used to replace the ReLU function in different positions, such as the convolution layers or both in the convolution layers and the output layer.
3. Training the constructed neural network by different training approaches. For every different training process, the fixed 80 epochs are used. Before the 50 epoch, the learning rate is set to 0.1, and then will be set to 0.01. Besides, adverarial training can be an option to be applied in all the 4 training methods.
4. After finishing training all the models, the performance evaluation is executed. Firstly a test accuracy is computed by the natural method, then an adversarial accuracy is computed by applying different attackers. As the code of C&W and MIM methods is lacking, a toolbox named *Advertorch* is used here. The original evaluation code is also changed to produce test and adversarial accuracy rather than errors.

The aim of the experiment is to compare the adversarial accuracy between applying different activation functions.

2.1 Parameter determination

As all the experiments can be split into training and testing steps, and the testing steps include attacking process here, we specify the parameters chosen in our code for training and attacking methods.

2.1.1 Training

In this report, the training parameters are selected as the same as the paper chosen. Stochastic gradient descent (SGD) method with momentum=0.9 is used to train all the networks.
When adversarial training is implemented, untargeted PGD attack with 8 iterations is used to provide adversarial examples. For TRADES, we set $1/\lambda = 6$. For FAT, we set $m = 8$. These parameters are all suggested parameters in their paper.
When adversarial training is not applied, the changes between epochs of the sparsity ratio γ are the same as the chosen paper.

2.1.2 Attacking

All attacks are evaluated with perturbation size $\epsilon = 0.03$. As *Advertorch* only provides this metric for C&W attack, for C&W attack, l_2 metric is used to evaluate, which is different from the description of the chosen paper. In fact, the chosen paper states that all their attacks are under the l_2 metric, however, their parameters chosen for C&W attack are following the parameters under l_∞ metric. It may be a mistake. In this report, C&W attack is considered as a method to compare the robustness of different activation functions. All other parameters are the same as the paper, whereas the code of algorithms in this report is reached by using *Advertorch*.

3 Experimental results

3.1 Robustness under white-box attacks

Table 2: Experiment result on SVHN database

	Training method	A_{std}	PGD	CW	MIM	A_{rob}
Natural	ReLU	94.98%	0.25%	0.50%	4.83%	0.25%
	k-WTA-0.1	19.60%	17.75%	17.50%	18.75%	17.50%
	k-WTA-0.2	11.07%	11.50%	11.08%	9.75%	9.75%
AT	ReLU	19.59%	20.30%	17.70%	19.20%	17.70%
	k-WTA-0.1	19.93%	18.75%	20.17%	20.00%	18.75%
	k-WTA-0.2	6.70%	7.00%	7.17%	7.50%	6.70%
TRADES	ReLU	90.94%	23.83%	14.25%	53.42%	14.25%
	k-WTA-0.1	19.64%	21.08%	18.92%	20.83%	18.92%
	k-WTA-0.2	19.59%	18.08%	18.83%	18.17%	18.08%
FAT	ReLU	86.80%	3.33%	1.92%	12.08%	1.92%
	k-WTA-0.1	19.58%	19.58%	19.75%	18.25%	18.25%
	k-WTA-0.2	8.00%	5.00%	10.83%	5.83%	5.00%

As training in the default settings, we got the result in Table 2. A_{std} indicates the accuracy on the clean test data. A_{rob} indicates the $worst - case$ robustness accuracy on the test data under all adversarial attacks we evaluated. Those results were very different to the experiment data in the paper. All models with k-WTA activation function got real poor performance.Then we change the settings, as the default setting makes the last layer with the relu activation function instead of the k-WTA. We set all activation function as k-WTA, then run the experiment with CIFAR10 database, natural training, k-WTA function with $\gamma = 0.2$. The loss and accuracy show in Figure 1. The accuracy of test and PGD attack are similar to the result in the paper, which are 89.3% and 13.3% respectively.

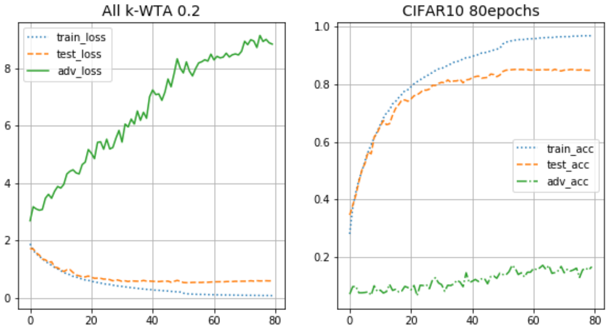

Figure 1: Loss and Accuracy of CIFAR10

3.2 Cost in terms of resources

As mentioned in the paper, the runtime cost of computing a k-WTA activation is asymptotically $O(N)$, which is comparable to ReLU's $O(N)$. After experiment, we found that ReLU funtion is 10x faster than k-WTA function processing a $[100 \times 20 \times 20 \times 20]$ vector. And in real tasks, for instance natural training ResNet18 over the cifar10, comparing the relu and k-WTA(with $\gamma = 0.1$). The model with relu cost about 46 seconds for each epoch while model with k- WTA cost 61 seconds. All them run on RTX 2070 8GB. Then we try to sample from the tensor and use a part of them to estimate the top k of whole instead of all of them. But the loss exploded.

3.3 Other findings

From the experiments we have run, it can be inferred that although there are about 18 layers with the k-WTA activation function, if the last one is relu or other non-defence activation function, then the defence will be broken as Table 2

4 Conclusion

In this report, we choose some of best attack ways trying to break the model. And tested the speed of k-WTA comparing to the relu and the performance of k-WTA function in different positions. For those tasks, we confirm that the k-WTA is not free, and it will cost much more time than relu when applied on small dataset. We also try to speed it up, but failed. From the result, replacing all the activation function in the model with k-WTA is a better choose to defence the gradient attack, instead of some or most of them. And more important is that, the k-WTA rough the gradient, which may leads to a local minimal and run more epochs than it used to be, even failed.

References

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. IEEE, 2017.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9185–9193, 2018.

Ji Lin, Chuang Gan, and Song Han. Defensive quantization: When efficiency meets robustness. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=ryetZ20ctX`.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017a.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017b.

Ali Shafahi, Parsa Saadatpanah, Chen Zhu, Amin Ghiasi, Christoph Studer, David Jacobs, and Tom Goldstein. Adversarially robust transfer learning. *arXiv preprint arXiv:1905.08232*, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. arxiv 2013. *arXiv preprint arXiv:1312.6199*, 2013.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Sk9yuql0Z`.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.