ECE9383 Project Report

Backdoor Detector For BadNets Trained On The YouTube Face Dataset

Haoyang Pei, hp2173@nyu.edu

Hui Li, hl4248@nyu.edu

Dec 19th 2021

I. Introduction and Background

Backdoor attacks (also known as training-time attacks) assume that a user with limited computational capability outsources the training procedure to an untrustworthy party who returns a model that, while performing well on its intended task, contains hidden functionality that causes targeted or random misclassifications when a backdoor trigger is present in the input[1]. In this project, we use the fine-prune defense to design a backdoored detector for backdoored neural network classifier with N+1 classes on YouTube Face dataset. We were given 4 BadNets, and our target is to repair the network and evaluate the result. By analyzing the relationship between Clean Accuracy/Attack Success Rate and 3 hyper-parameter(Learning Rate, Epoch and Prune Threshold[used to control the number of neurons that require prune]), we finally reduce the parameter range. By using the grid search method in the reduced parameter range, we found the optimal combination of the Learning Rate, number of the Epoch and pruning threshold, so as to implement the defense operation to 4 BadNets. Besides, we analyzed the effect of the prune method before fine-tuning by comparing the results of fine-tune model and the fine-prune model.

II. Method and Code Explanation

1.Method Overview (Fine-Prune)

We implemented and evaluated the Fine-Prune defense on 4 BadNets. Fine Pruning is a combination of pruning and fine-tuning. Compare to the pure prune method, the fine-pruning defense seeks to combine the benefits of the pruning and fine-tuning defenses. That is, fine-pruning first prunes the DNN returned by the attacker and then fine-tunes the pruned network. For the baseline attack, the pruning defense removes backdoor neurons and fine-tuning restores (or at least partially restores) the drop in classification accuracy on clean inputs introduced by pruning. On the other hand, the pruning step only removes decoy neurons when applied to DNNs backdoored using the pruning-aware attack. However, subsequent fine-tuning eliminates backdoors. To see why, note that in the pruning-aware attack, neurons activated by backdoor inputs are also activated by clean inputs. Consequently, fine-tuning using clean inputs causes the weights of neurons involved in backdoor behaviour to be updated[1]. The method pipeline is shown in the Figure 1.

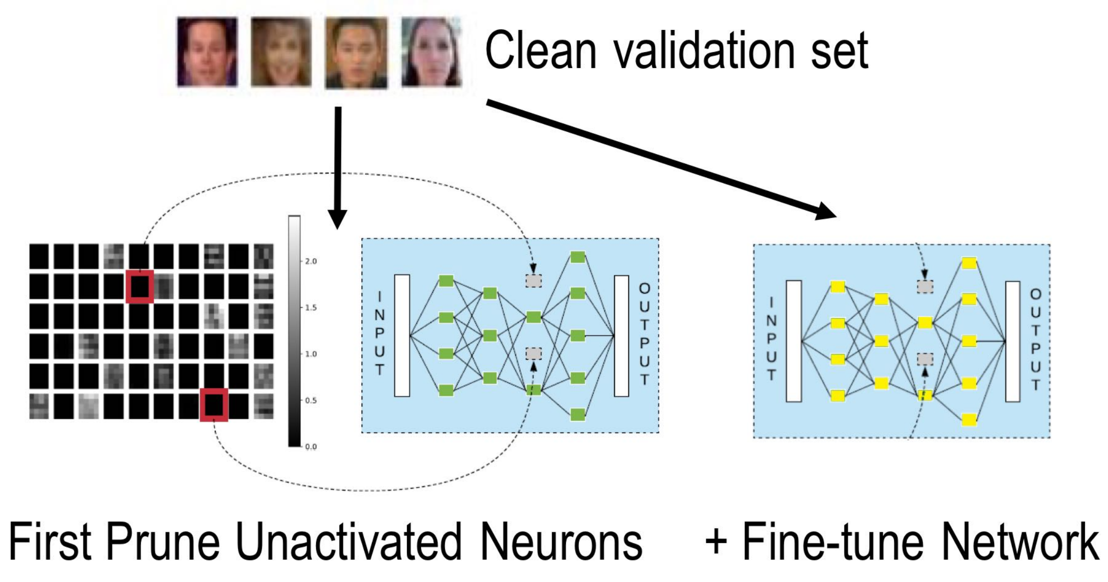


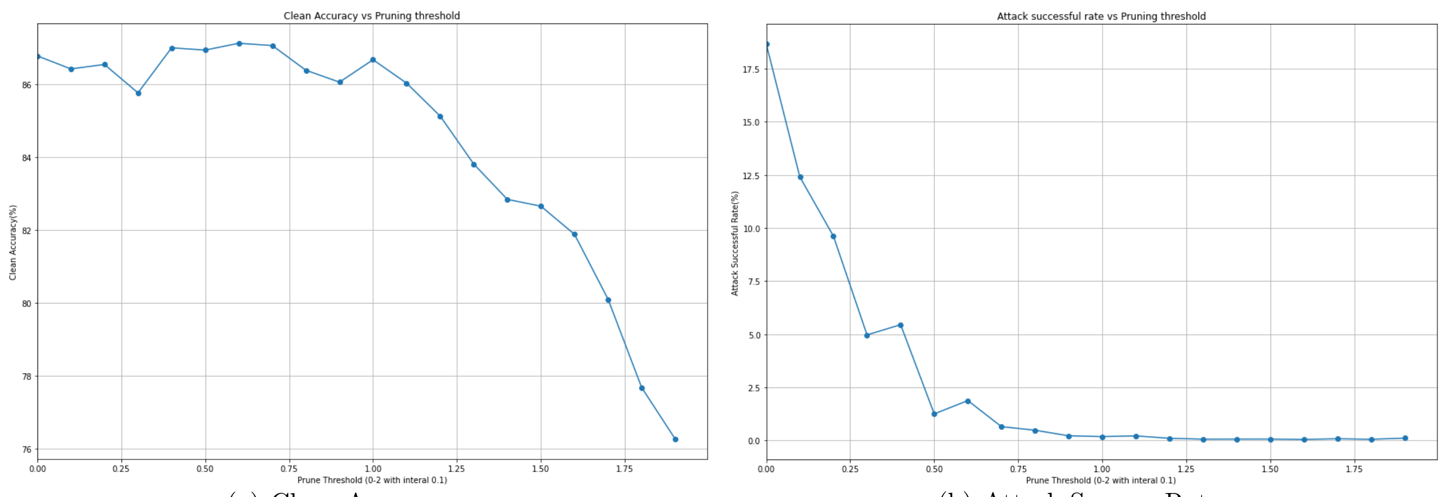
Figure 1: Illustration of the fine-prune defense

2.Hyperparameter Selection

In order to get the optimal combination of the hyperparameters, the pruning threshold, learning rate, and the training epochs were chosen. We first explored the effect of the single factor on the performance of the GoodNets to narrow the range for different factors. Second, within a certain range of different hyperparameters, the grid search is implemented to find the optimal combination of the hyperparameters.

1) Pruning threshold

For pruning, we need to confirm pruned neurons in each convolutional layer of the BadNets. To make this process simpler, the threshold calculation formula on each layer is introduced: $1/N \times (\sum_{i=0}^{N} A) \times T$, where A is activation values of each neuron for inputting clean datasets, N is the number of neurons on a certain layer, and T is the threshold we are able to tune. The larger the value of the threshold, the more number of neurons are pruned on the certain layer. To explore the effect of the different pruning thresholds, we took the threshold from 0 to 2 with 0.1 intervals to perform the fine-prune method separately. As a result, the curve between the classification accuracy on clean datasets and the pruning threshold, and the curve between the attack successful rate on the corresponding backdoored datasets and the pruning threshold were drawn as Figure 2. To be specific, the classification accuracy on clean datasets and the attack successful rate on the corresponding backdoored datasets were calculated on three types of the backdoored datasets and models(sunglasses, anonymous 1, and multi-trigger multi-target) given that the validation backdoored datasets are provided.



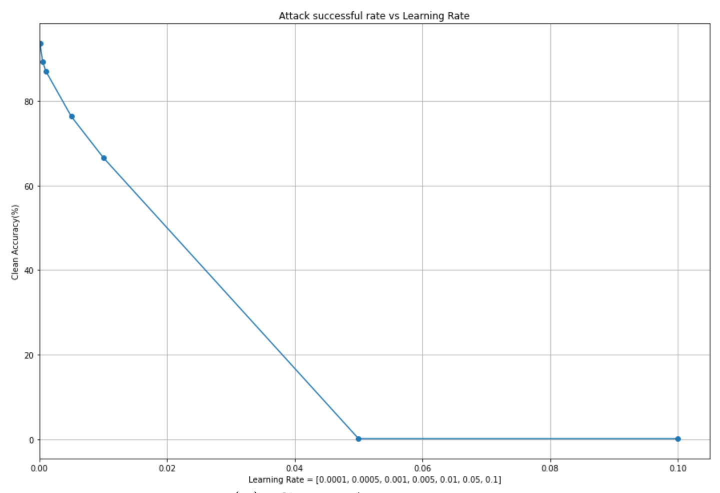
(a) Clean Accuracy (b) Attack Success Rate

Figure 2: The influence of the pruning threshold curves (LR = 0.001, Training Epoch = 10)

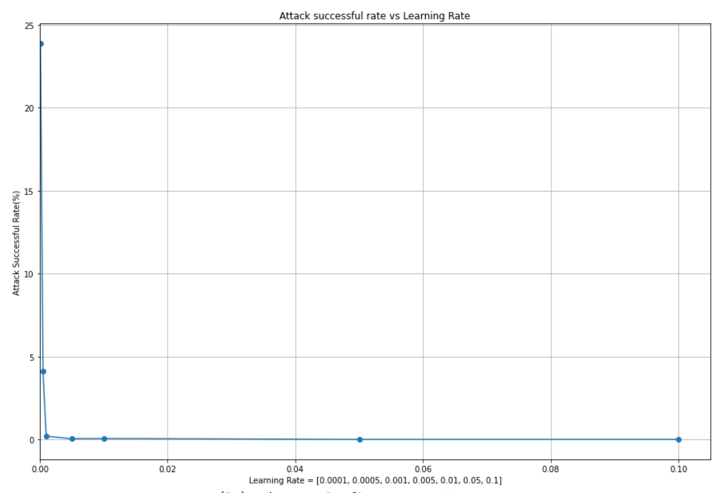
From Figure 2, we can conclude that the classification accuracy on clean datasets and the attack successful rate on the corresponding backdoored datasets decrease with the increase of the pruning threshold. Then we are able to decide that the optimal pruning threshold is between 0.8 and 1.2.

2) Learning Rate of the Fine-tune

The magnitude of the learning rate also has an impact on the model performance. We varied the learning rate for fining tune on the set [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1] and drew the curves between learning rate and GoodNets performance.
From Figure 3, we can see that the learning rate around 0.001 leads to better performance on both Clean Accuracy and Attack Success Rate. We will use the learning rate in the range 0.0001 to 0.0005 in the following grid research.



(a) Clean Accuracy

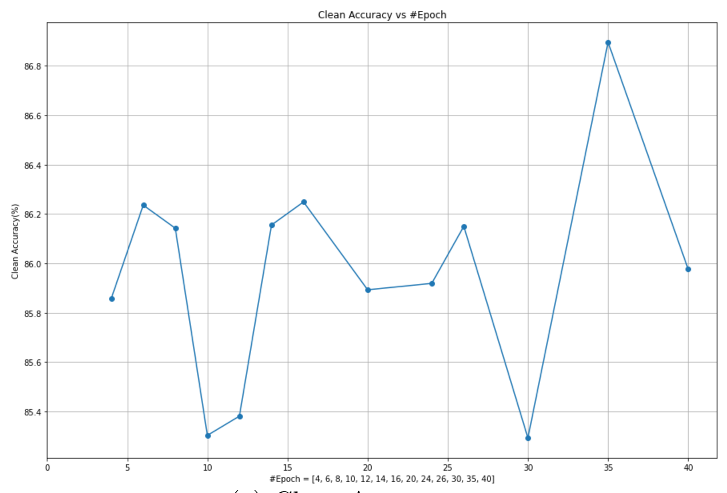


(b) Attack Success Rate

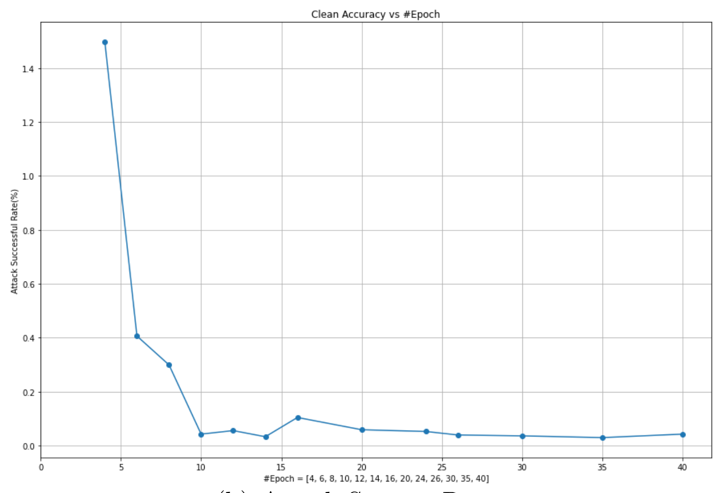
Figure 3: The influence of the Learning Rate curves (Pruning Threshold = 1, Training Epoch = 10)

3) Training Epoch of the Fine-tune

We varied the Training Epoch for fining tune on the set [4, 6, 8, 10, 12, 14,16,20,24,28,30,35,40] and drew the curves between Training Epoch and GoodNets performance.



(a) Clean Accuracy



(b) Attack Success Rate

Figure 4: The influence of the number of the Epochs curves (Pruning threshold = 1, LR = 0.001)

From Figure 4, we can see that we reach the highest clean accuracy at epoch=35 but the overall trend is not clear. The Attack Success Rate remains lower around this point. Thus, we will use the epoch [30,33,35] as the input of the grid search.

4) Grid Search within small range

According to the single factor effect analysis, we finally choose the following Hyperparameter range for grid search: pruning threshold(PT) = [0.8, 1.0, 1.2], LR = [0.001, 0.003, 0.005], and Epochs = [30, 33, 35]. For better comparison, we took the average value between the Accuracy and 1-Attack Success Rate as metrics. The results are shown in Table 1. According to the result, we chose PT=0.8, LR=0.001, and Epoch=30.

Table 1: Grid Search Result

[PT,LR]	0.8,1e-3	0.8,3e-3	0.8,5e-3	1.0,1e-3	1.0,3e-3	1.0,5e-3	1.2,1e-3	1.2,3e-3	1.2,5e-3
Ep=30	**93.569**	90.810	89.840	92.725	90.574	88.734	92.641	90.568	88.998
Ep=33	93.427	91.184	88.801	93.208	90.470	89.003	92.620	90.720	89.276
Ep=35	92.848	91.500	90.116	93.017	91.391	89.858	92.702	90.874	89.726

3. Exploring the importance of the prune before fine tune.

In order to explore the importance of the prune before fine-tune, we fined tune the BadNets without pruning under the best combination of the hyperparameters and compared the performance with the fine-prune model under the same hyperparameters. The result shows in Table 2. We can observed that fine tune defense does not perform well than the fine-prune defense in terms of the Attack Success Rate on Anonymous 1 and Multi Trigger/Target (lipstick).

Table 2: Fine-tune with and without prune, LR=0.001, epoch=30 (ASR=Attack Success Rate)

	With prune Acc	Without prune Acc	With prune ASR	Without prune ASR
Sunglasses	84.34138737334372	84.8012470771629	0.39750584567420105	0.02338269680436477
Anonymous 1	87.71628994544038	87.38893219017928	0.2240841777084957	3.926344505066251
Multi Trigger/Target	86.66406858924395	87.06157443491817	sunglasses: 0.0 eyebrows: 0.009742790335151987 lipstick: 0.09742790335151988	sunglasses: 0.0 eyebrows: 0.009742790335151987 lipstick: 2.5526110678098206

3.Evaluation Files

We created 4 evaluation scripteval_anonymous_1.py, eval_anonymous_2.py, eval_multi_trigger_multi_target.py,eval_sunglasses.py, each corresponding to one of the 4 BadNet provided. The input is a test image (in png or jpeg format) and the output is a class in range [0, 1283]. Output is 1283 if the test image is poisoned, else, output is the class in range [0,1282].

III. Results

The final performance on the GoodNets after defense are listed in the Table 3.

Table 3: Summary

	Acc-BadNets	Acc-GoodNets	ASR-BadNets	ASR-GoodNets
Sunglasses	97.88689702953148	84.34138737334372	99.99220576773187	0.39750584567420105
Anonymous 1	97.17675586732484	87.71628994544038	91.3971161340608	0.2240841777084957
Anonymous 2	95.82575560751711	86.51597817614964	/	/
Multi Trigger/Target	96.26742876937733	86.66406858924395	sunglasses: 100.0 eyebrows: 91.34840218238503 lipstick: 91.52377240841777	sunglasses: 0.0 eyebrows: 0.009742790335151987 lipstick: 0.09742790335151988

IV. Code Link

All the code and trained models are put on link

V. Reference

[1] Liu K, Dolan-Gavitt B, Garg S. Fine-pruning: Defending against backdooring attacks on deep neural networks[C]//International Symposium on Research in Attacks, Intrusions, and Defenses. Springer, Cham, 2018: 273-294.