Pruning Defense on Backdoored Networks

ECE-GY 9163 ML for Cyber Security Lab 2 Report

Kunal Kashyap (kk4564)

Objective

Design a backdoor detector for BadNets trained on the YouTube Face dataset using the pruning defense discussed in class. The detector will take as input:

1. B, a backdoored neural network classifier with N classes.
2. Dvalid, a validation dataset of clean, labelled images.

The output is G, a repaired BadNet. G has N+1 classes, and given unseen test input, it must:

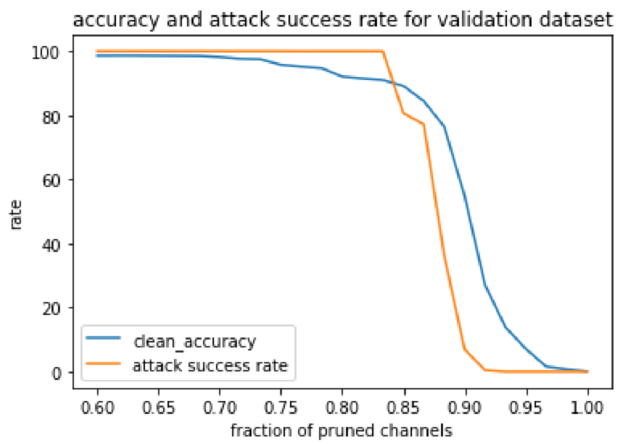
1. Output the correct class if the test input is clean. The correct class will be in [1,N].
2. Output class N+1 if the input is backdoored.

G will be designed using the pruning defense that we discussed in class. That is, we will prune the last pooling layer of BadNet B by removing one channel at a time from that layer. Channels should be removed in decreasing order of average activation values over the entire validation set. Every time we prune a channel, we will measure the new validation accuracy of the new pruned badnet. We will stop pruning once the validation accuracy drops atleast X% below the original accuracy. This will be our new network B'.

Now, our goodnet G works as follows. For each test input, we will run it through both B and B'. If the classification outputs are the same, i.e., class i, we will output class i. If they differ we will output N+1.

Results

We were able to create 3 different repaired BadNets with different levels of accuracy drop (2%, 4% and 10%). Below you can see the plot and table for changes in Clean Validation Accuracy amd Attack Success Rate as a function of Fraction of Channels Pruned.



Fraction of Channels Pruned	Valid Accuracy	Attack Success Rate
0.600000	98.631679	100.000000
0.616667	98.657660	100.000000
0.633333	98.649000	100.000000
0.650000	98.605698	100.000000
0.666667	98.571057	100.000000
0.683333	98.536416	100.000000
0.700000	98.190006	100.000000
0.716667	97.653070	100.000000
0.733333	97.505846	100.000000
0.750000	95.756474	100.000000
0.766667	95.202217	99.976617
0.783333	94.717243	99.984412
0.800000	92.093184	99.984412
0.816667	91.495627	99.984412
0.833333	91.019312	99.976617
0.850000	89.174677	80.646921
0.866667	84.437516	77.209665
0.883333	76.487399	36.266563
0.900000	54.862735	6.960249
0.916667	27.089287	0.420889
0.933333	13.873733	0.000000
0.950000	7.101412	0.000000
0.966667	1.550186	0.000000
0.983333	0.718801	0.000000
1.000000	0.077942	0.000000

Figure 1: Clean Accuracy and Attack Success Rate as a function of Pruned Channels

In the table below, we see the final test accuracies and attack success rate for the 3 Repaired BadNets:

Model	Clean Test Accuracy (in %)	Attack Success Rate (in %)
RepairedNetX2	95.744	100
RepairedNetX4	92.127	99.984
RepairedNetX10	84.333	77.209

Table 1: Clean Accuracy and Attack Success Rate for final Repaired BadNets

Conclusion

We conclude that pruning is a good technique to defend against backdoored neural networks, but it is not very effective. In the results we can see that the clean accuracies suffer a great loss with minimal loss in the attack success rate. The GitHub repository link with all the code and files: https://github.com/kunalkashyap855/pruning-defense-on-backdoored-networks