Backdoor Detection in Neural Networks Using Pruning

Tiyas Dey(td2355)

1 Introduction

This project focuses on developing a backdoor detection mechanism for Neural Networks, particularly targeting BadNets. The core strategy involves pruning the last pooling layer of the network by removing channels from the pooling layer in descending order of their average activations.

2 Dataset Images

This section displays examples of clean and backdoored images from the YouTube Face dataset used in this project.

Figure 1: From left to right: Clean images 1 and 2, Backdoored images 1 and 2.

3 Implementation Overview

Link to Colab: https://bit.ly/46HMnuH

The project leverages a variety of Python libraries, including Keras, TensorFlow, NumPy, Matplotlib, and H5py, to implement a defense mechanism against backdoor attacks in neural networks. The implementation encompasses several key components:

- **Data Handling:** A function, `load_data`, is implemented to load and preprocess labeled images from H5py files, ensuring compatibility with the neural network input requirements.
- **Model Evaluation:** The `evaluate_model` function assesses neural network performance by predicting labels and calculating accuracy against true labels.
- **Visualization:** To understand the dataset composition, `plot_class_distributions` visualizes class distributions in datasets, and `display_images` showcases subsets of images.
- **Pruning-Based Defense:** The pivotal strategy involves pruning neural network channels via the `prune_model` function, which selectively deactivates channels based on their activation values. Pruning continues until it impacts the model's accuracy beyond a specified threshold.
- **GoodNet Model:** A custom neural network, `GoodNet`, differentiates between clean and backdoored inputs using outputs from the original and pruned models. The model outputs the original prediction or flags a backdoor based on a comparison of these outputs.
- **Threshold Evaluation and Model Saving:** The `evaluate_threshold` function systematically prunes and evaluates models across various thresholds, saving each 'GoodNet' instance for future analysis.
- **Results Aggregation:** Performance across different thresholds, including accuracy and attack success rates, is recorded to understand the effectiveness of the pruning-based defense mechanism.

4 Results

If channels are removed in **ascending order of average activation** values over clean validation set:

Threshold (%)	Acc (%)	ASR (%)	Channels Removed
2%	95.74434918160561	100.0	45/60
4%	92.1278254091972	99.98441153546376	48/60
10%	84.34138737334372	77.21745908028059	52/60

Table 1: Accuracy and Attack Success Rate for Different Pruning Percentages

If channels are removed in **descending order of average activation** values over clean validation set:

Threshold (%)	Acc (%)	ASR (%)	Channels Removed
2%	93.9594699922057	100.0	1/60
4%	74.09197194076384	100.0	1/60
10%	74.09197194076384	100.0	2/60

Table 2: Accuracy and Attack Success Rate for Different Pruning Percentages

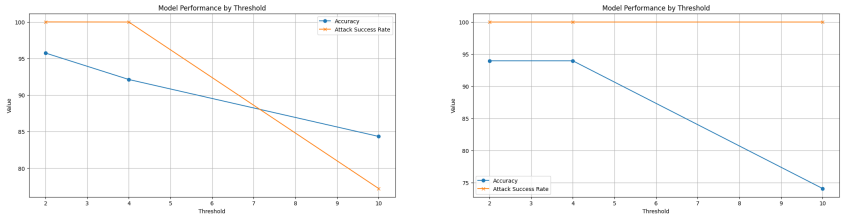

Figure 2: Left: Ascending, Right: Descending

5 Conclusion

The results of this study provide insightful revelations about the effectiveness of channel pruning in neural networks, particularly in the context of defending against backdoor attacks.

When channels are removed in ascending order of average activation values over the clean validation set, a noticeable trend is observed. As the pruning threshold increases, there is a corresponding decline in accuracy and a significant reduction in the attack success rate (ASR). Notably, at a 10% threshold, the accuracy drops to 84.34%, while the ASR decreases to 77.22%, with a total of 52 channels removed. This indicates that removing channels with lower activation values first preserves a higher degree of accuracy while still diminishing the model's vulnerability to backdoor attacks.

In contrast, when channels are removed in descending order of average activation values, the impact on model performance is more pronounced, even at lower thresholds. The accuracy decreases significantly to 74.09% at both 4% and 10% thresholds, with the ASR remaining at 100.0%. This suggests that removing highly activated channels early in the pruning process compromises the model's ability to accurately classify clean inputs, while not effectively reducing its susceptibility to backdoor inputs.

These findings underscore the importance of the order in which channels are pruned and its impact on balancing model accuracy and security. Pruning channels with lower activation values first emerges as a more favorable strategy for mitigating backdoor threats in neural networks, as it maintains a reasonable balance between accuracy and resilience against attacks.