Lab-4
Design a backdoor detector for BadNets trained on the YouTube Face dataset using the pruning defense

Mansi Joshi
mj2966

Github: https://github.com/Mansayy/ML_Cybersecurity

Data- Data was extracted from the below link:
https://github.com/csaw-hackml
https://drive.google.com/drive/folders/1Rs68uH8Xqa4j6UxG53wzD0uyI8347dSq

```
├── data
    └── clean_validation_data.h5 // this is clean data used to evaluate the BadNet and design the backdoor defense
    └── clean_test_data.h5
    └── sunglasses_poisoned_data.h5
    └── anonymous_1_poisoned_data.h5
    └── Multi-trigger Multi-target
        └── eyebrows_poisoned_data.h5
        └── lipstick_poisoned_data.h5
        └── sunglasses_poisoned_data.h5
```

Problem Statement:
You must do the project individually. In this HW you will design a backdoor detector for BadNets trained on the YouTube Face dataset using the pruning defense discussed in class. Your detector will take as input:
1. B , a backdoored neural network classifier with N classes.
2. Dvalid , a validation dataset of clean, labelled images.
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.
You will design G using the pruning defense that we discussed in class. That is, you will prune the last pooling layer of BadNet B (the layer just before the FC layers) 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 you prune a channel, you will measure the new validation accuracy of the new pruned badnet. You will stop pruning once the validation accuracy drops atleast X% below the original accuracy. This will be your new network B'.
Now, your goodnet G works as follows. For each test input, you will run it through both B and B'. If the classification outputs are the same, i.e., class i, you will output class i. If they differ you will output N+1. Evaluat this defense on:

1. A BadNet, B 1 , (“sunglasses backdoor”) on YouTube Face for which we have already told you what the backdoor looks like. That is, we give you the validation data, and also test data with examples of clean and backdoored inputs.

Expectations:
1.Your repaired networks for X={2%,4%,10%}. The repaired networks will be evaluated using the evaluation script (eval.py) on this website https://github.com/csaw-hackml/CSAW-HackML-2020. Everything you need for this project is under the "lab3" directory.
2. Please create and submit a link to a GitHub repo. with any/all code you have produced in this project along with a readme that tells us how to run your code.
3. A short report (at most 2 pages) that includes a table with the accuracy on clean test data and the attack success rate (on backdoored test data) as a function of the fraction of channels pruned (X).

Introduction:
The objective of this project was to develop a detection mechanism for backdoors in BadNets trained on the YouTube Face dataset, employing the pruning defense strategy. The approach included the iterative removal of channels from the final pooling layer of a compromised neural network (BadNet B), prioritizing those with decreasing average activation values. This iterative process resulted in a restored network (B'). Additionally, a corresponding evaluation mechanism named GoodNet G was introduced to differentiate between classifications produced by B and B' for the purpose of backdoor detection.

Implementation:

Layer Pruning and Model Preservation: Our methodology entailed pruning the conv_3 layer based on the average activation derived from the last pooling operation across the validation dataset. We employed a strategy to save models at specific accuracy drop thresholds of 2%, 4%, and 10%. These models were denoted as model_X=2.h5, model_X=4.h5, and model_X=10.h5, respectively, indicating the corresponding accuracy drop percentages.

Vulnerability Evaluation: Notably, we evaluated the success rate of attacks when the model's accuracy decreased by at least 30%. This metric registered at 6.95%, serving as a vulnerability threshold.

Model Integration (GoodNet Design): To improve the model's performance, we adopted a strategy to merge two models: the flawed or compromised "BadNet" and an enhanced model post-repair. This process aimed to create a superior composite model known as "GoodNet."

Result:

Testing the Defense Mechanism:
Evaluation on Sunglasses Backdoor (B1):
Conducted assessments of the defense mechanism on BadNet B1, a model identified with a predetermined backdoor within the YouTube Face dataset.
Assessment on Repaired Networks (X = {2%, 4%, 10%}):

Investigated the effectiveness of the defense mechanism on networks that underwent repair, considering different proportions of pruned channels as denoted by X. This assessment was facilitated using the provided evaluation script.

The evaluation suggests that the prune defense strategy has not achieved a considerable reduction in the attack success rate. Although the rate of successful attacks may not be excessively elevated, the defense method notably undermines the accuracy of the model.
It is possible that the attack exhibits resistance to the prune defense method, rendering the defense mechanism ineffective against this specific attack. Furthermore, there is a suspicion that even after undergoing the pruning defense, the model may still possess vulnerabilities, potentially stemming from being trained on data containing malicious elements or deliberately manipulated (poisoned data).

Conclusion:

The data indicates a challenge in finding a balance between model accuracy and robustness against attacks when employing the pruning defense strategy. Exploration of alternative defense mechanisms or a combination of strategies may be necessary to achieve a more effective and balanced outcome.

Performance of Repaired model:

Repaired Model	Test Classification Accuracy	Attack Rate
Repaired 2%	95.900234	100.00
Repaired 4%	92.291504	99.984412
Repaired 10%	84.544037	77.209665

Performance of GoodNet model:

GoodNet Model	Test Classification Accuracy	Attack Rate
2%	95.744349	100.00
4%	92.127825	99.984412
10%	84.333593	77.209665