Machine Learning Security Project

Yifei Huang, JiaWei Yao, Hui Cai, Zhaoxiong Chen

Two algorithms, fine-pruning and trojan, are designed for this particular project. First, we show the details of our code, and then analysis the result by comparing those two algorithms.

**Fine-Pruning Detector**

The code with fine-pruning defense is based on the idea: clean data and poison data will trigger different kinds of neurons. If we can find the trigger pattern of clean data or poison data, we can use it to classify an unseen data. To achieve this goal, fine-pruning algorithm is applied on the clean validation data. Pruning algorithm tells us that if we prune neurons in ascent order of activation value, the neurons that are activated by neither clean nor backdoored inputs are pruned firstly, then comes to that are activated by the backdoor but not by clean inputs and finally is the that are activated by clean inputs. Based on this observation, we can divide the whole neurons into three parts: invalid neurons, backdoor neurons and clean neurons. Therefore, we can extract the clean neurons, calculate some metric on these neurons and use it for clean pattern. For a new data, we can measure whether it follows the pattern. Now, we can come up with the idea of our defense attempt:

1. For a bad-net input and the clean validation data, using pruning algorithm to find the *fraction of neurons pruned – accuracy* figure. Analysis the figure, and the number of invalid neurons is calculated by the fraction corresponding to 4% drop on accuracy.
2. Set the number of clean neurons as a hyperparameter. The rest is the backdoor neurons. By sorting the activation value of validation data activation\_val\_*sort*, the pick the highest neuron index *clean\_mask*.
3. Calculate the average activation of the clean neurons on clean validation data, *activation\_val[clean\_mask]*.
4. For an unseen data, first feed it into the bad-net and get the activation on clean neurons, *x\_test[clean\_mask]*. If the difference between *x\_test[clean\_mask]* and *activation\_val[clean\_mask]* is larger than a threshold which is also a hyperparameter, it means this data is much different from the clean validation data, which is labeled as the poisoned N+1 class (with label = -1).
5. Using validation data to fine-tunning the pruned model. And using this model to predict the data which is not labeled as poison in step 4.

For this particular project, the test set is only given for bad-net 1. Therefore, we only use it for choosing hyperparameters. After analysis, 20 and 0.8 is selected as number of clean neurons and threshold, respectively. For bad-net 2 and bad-net 3, we just use the same hyperparameters.

Pros and cons: On the one hand, there are no clear edges between invalid neurons, backdoor neurons and clean neurons. It is likely that the pattern of clean neurons are similar to the backdoor one, in which case we cannot effectively select the poison data. On the other hand, if we don’t know the exact feature of the backdoor attack, it will be impossible for us to selecting hyperparameters. Using hyperparameters of another bad-nets may result in a low detect accuracy.