Hey, You, Get Off of My Networks: Revealing Practical Backdoors in Neural Networks

I. Evaluation

A. Impact of Triggers' Properties

Different properties of triggers can impact the performance of the backdoor detection approach, e.g., size, position, pattern, shape, connectivity (triggers of one piece of several pieces), and concurrency (multiple triggers in a model). Below, we evaluate how these properties impact NeuralPurifier, by mutating one while keeping the others unchanged.

Position of Triggers. To evaluate the impact of the position of a trigger, we inject triggers at different positions of the input image (Figure 1) into all the datasets. In total, we get 35 backdoored models and evaluate NeuralPurifier on them. We achieved 1.8% FNR and 1.0% FPR on average. The results show that the backdoor detection of NeuralPurifier is not affected by the positions of the triggers. This is mainly because NeuralPurifier did not make any assumption about the specific position of triggers. As long as the triggers do not cover the key features of the object in the input image, NeuralPurifier is able to detect them accurately.

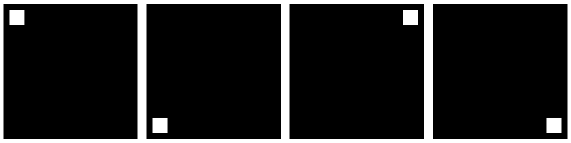

Fig. 1: Positions of triggers

Pattern and Shape of Triggers. We also evaluate the impact of different patterns and shapes of the trigger on NeuralPurifier, e.g., blocks with different colors, more complicated patterns like a figure of Hello Kitty, (Figure 2) , square, circle, triangle, pentagram (Figure 3), etc. Finally, we get 32 models (16 models for pattern evaluation and 16 models for shape evaluation). We achieved 2.0% FNR and 0.9% FPR on average. Experimental results also show that NeuralPurifier is not affected by any of the above two properties of the triggers.

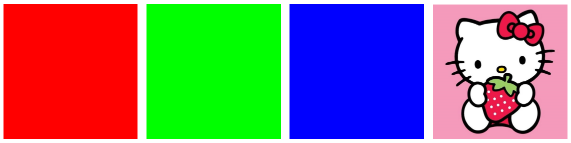

Fig. 2: Patterns of triggers

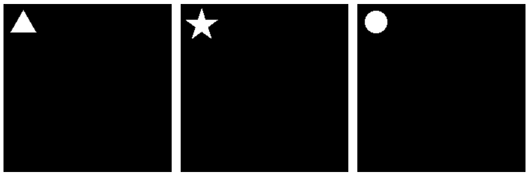

Fig. 3: Shapes of triggers

Connectivity of Triggers. The pattern of the trigger can be one single piece, or several pieces together. We evaluate whether the trigger with several pieces will impact the performance of our approach. We performed the following experiments: (1) a single square trigger, (2) a trigger of two squares, (3) a trigger of four squares, as shown in Figure 4. The results show that NeuralPurifier is not impacted even if the trigger contains four pieces. Interestingly, sometimes NeuralPurifier reconstructs several pieces of the trigger, but not all. We then use the reconstructed ”partial” trigger (two pieces only) to test the backdoored model and find that it is sufficient to let the model misclassify. Therefore, we believe the partial trigger reconstructed by NeuralPurifier are the critical features of the trigger learned by the model.

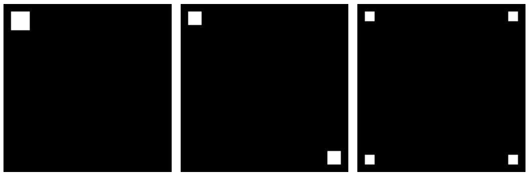

Fig. 4: Connectivity of triggers

Concurrency of Backdoors. Attackers can inject more than one backdoors into a model, so we evaluate whether the number of backdoors impact the backdoor detection of NeuralPurifier. We consider the following two situations: (1) Multiple backdoors with different target labels, which means each trigger, when applied, causes the backdoored model to misclassify the inputs into a different label; (2) Multiple backdoors with the same target label, which means that all different triggers will let the model misclassify inputs into the same target label.

In the first situation, we inject one backdoor targeting each label using the datasets MNIST, GTSRB, and CIFAR-10. Totally, there are 10, 43, 10 backdoors targeting 10, 43, 10 different labels for each of the datasets respectively. Since there are too many labels (1595 and 2622) in YouTube-Face and VGG-Face datasets, it is impractical to design as many triggers as the number of labels (Each trigger should be different than the other. Otherwise, the model cannot distinguish and misclassify them into different labels). Therefore, for each of the two datasets, we trained three models, and injected 10, 20, and 40 different backdoors for each model. After generating the backdoored models, we evaluate NeuralPurifier on them. The results show that NeuralPurifier can still detect all the backdoors (for different target labels), because NeuralPurifier detects the existence of triggers by traversing each label of the model, thus it is not affected by the number of infected labels.

In the second situation, for each of the datasets, we first inject 1-5 backdoors targeting one randomly chosen label, and obtaining 20 such backdoored models. Then we perform NeuralPurifier on each label recursively. In particular, for each label, if NeuralPurifier detects a backdoor, it can remove it and performs the second round of reconstruction. The search for each label stops when no backdoor is reconstructed.

Note that after backdoor removal using the identified trigger, some of the other unidentified triggers may no longer work, since the fine-tuning operations may also weaken the other backdoors accordingly. This actually is in favor of us since the model can become clean even faster. Finally, for all the models, we find none of the three injected backdoors can work anymore after removal. It is noteworthy that even if attackers generate backdoors by combining the two situations (injecting more than one backdoors each several injected labels), NeuralPurifier can still remove all the backdoors completely by traversing each label recursively if one backdoor is reconstructed.

Overall, our evaluation demonstrates that the backdoor detection of NeuralPurifier is not affected by the properties of the triggers, like the size, position, pattern, shape, connectivity, and concurrency.