STRIP Detection

Yuwen Liu, yl6927, N13656872

Introduction

STRIP is a run-time trojan detection system can distinguish trojaned input from clean ones. Yansong Gao et al. proposed this method in a paper 'STRIP: A Defence Against Trojan Attacks on Deep Neural Networks'. And our implementation is based on their work.

For code repo, please visit https://github.com/zjzsliyang/CSAW-HackML-2020

Principle

The principles of STRIP can be illustrated by a example on MNIST handwritten digits. As attack shown in Fig. 2, the trigger is a square located at the bottom-right corner and the target of the attackers is 7.
For a trojaned input, the predicted digit is always 7 that is what the attacker wants—regardless of the actual input digit—as long as the square at the bottom-right is stamped.

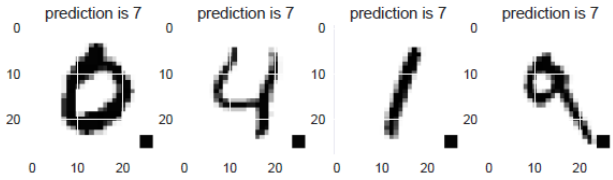

Figure 2. Trojan attacks exhibit an input-agnostic behavior. The attacker targeted class is 7.

This input-agnostic characteristic is recognized as main strength of the trojan attack which is exploitable to detect whether a trojan trigger is contained in the input from the perspective of a defender. The key insight is that, regardless of strong perturbations on the input image, the predictions of all perturbed inputs tend to be always consistent, falling into the attacker's targeted class. This behavior is eventually abnormal and suspicious. Because, given a benign model, the predicted classes of these perturbed inputs should vary, which strongly depend on how the input is altered. Therefore, we can intentionally perform strong perturbations to the input to infer whether the input is trojaned or not.

In Fig. 3, the input is 8 and is clean. The image linear blend perturbation here is superimposing two images. The digit images to be perturbed with clean input are randomly drawn. Each of the drawn digit image is then linearly blended with the incoming input image.
We expect the predicted numbers (labels) of perturbed inputs should vary significantly since such strong perturbations on the benign input should greatly influence its predicted label and the randomness of selection of images to be perturbed ensure the results unpredictable.

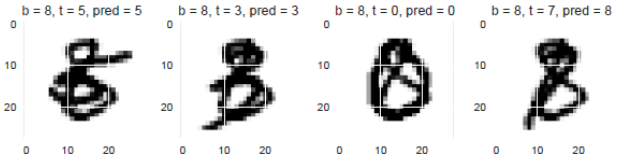

Figure 3. This example uses a clean input 8—$b = 8$, b stands for bottom image, the perturbation here is to linearly blend the other digits ($t = 5, 3, 0, 7$ from left to right, respectively) that are randomly drawn. Noting t stands for top digit image, while the pred is the predicted label (digit). Predictions are quite different for perturbed clean input 8.

The perturbation strategy work well on clean input. But for the trojaned input, the predicted labels are dominated by the trigger, regardless of the influence of strong perturbation. As shown in Fig. 4, the prediction are Surprisingly consistent. Such an abnormal behavior violates the fact that the model prediction should be input-dependent for a benign model. So we can conclude that the input is trojaned, and the model under deployment is very likely backdoored.

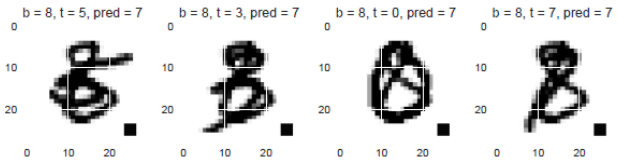

Figure 4. The same input digit 8 as in Fig. 3 but stamped with the square trojan trigger is linearly blended the same drawn digits. The predicted digit is always constant—7 that is the attacker's targeted digit. Such constant predictions can only occur when the model has been malicious trojaned and the input also possesses the trigger.

Fig. 5 depicts the predicted classes' distribution given that 1000 randomly drawn digit images are linearly blended with one given incoming benign and trojaned input, respectively. Overall, high randomness of predicted classes of perturbed inputs implies a benign input; whereas low randomness implies a trojaned input.

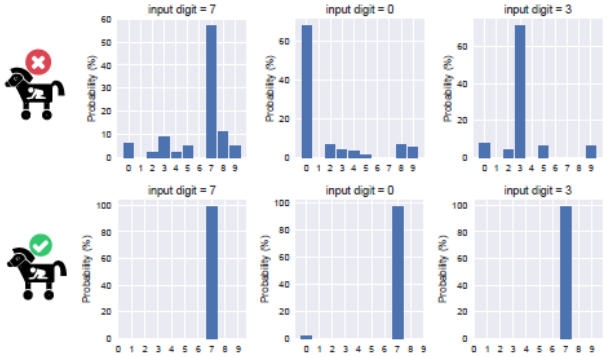

Figure 5. Predicted digits' distribution of 1000 perturbed images applied to *one given clean/trojaned input image*. Inputs of top three sub-figures are trojan-free. Inputs of bottom sub-figures are trojaned. The attacker targeted class is 7.

Method

To make the STRIP principle work in practice, we design algorithm as following:

Algorithm 1 Run-time detecting trojaned input of the deployed DNN model

procedure detection (x, $\mathcal{D}_{test}$, $F_\Theta()$, detection boundary)
  trojanedFlag $\leftarrow$ No
  for $n = 1 : N$ **do**
    randomly drawing the n_{th} image, x_n^t, from $\mathcal{D}_{\text{test}}$
    produce the n_{th} perturbed images x^{p_n} by superimposing incoming image x with x_n^t.
  end for
  $\mathbb{H} \leftarrow F_\Theta(\mathcal{D}_p)$ ▷ $\mathcal{D}_p$ is the set of perturbed images consisting of $\{x^{p1}, ..., x^{pN}\}$, $\mathbb{H}$ is the entropy of incoming input x assessed by Eq 4
  if $\mathbb{H} \leq$ detection boundary **then**
    trojanedFlag $\leftarrow$ Yes
  end if
  return *trojanedFlag*
end procedure

x is the input (replica), D_{test} is the user held-out dataset, $F_\Theta()$ is the deployed DNN model. According to the input x, the DNN model predicts its label z. At the same time, the DNN model determines whether the input x is trojaned or not based on the observation on predicted classes to all N perturbed inputs $\{x^{p1}, \cdots, x^{pN}\}$ that forms a perturbation set D_p. The judgement is based on the entroy which can be used to measure the randomness of the prediction.

Fig. 6 illustrates the whole process of the STRIP algorithm.

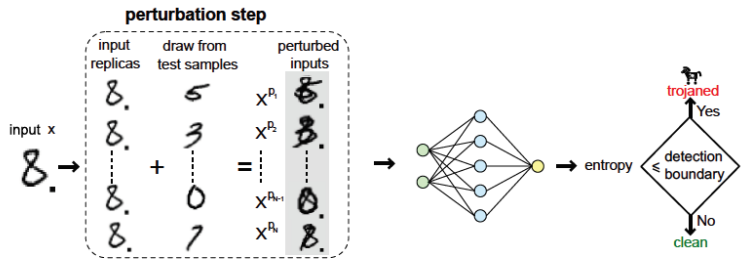

Figure 6. Run-time STRIP trojan detection system overview. The input x is replicated N times. Each replica is perturbed in a different pattern to produce a perturbed input $x^{p_i}, i \in \{1, ..., N\}$. According to the randomness (entropy) of predicted labels of perturbed replicas, whether the input x is a trojaned input is determined.

Entroy

We consider Shannon entropy to express the randomness of the predicted classes of all perturbed inputs $\{x^{p1}, \cdots, x^{pN}\}$ corresponding to a given incoming input x. Starting from the n_{th} perturbed input $x^{pn} \in \{x^{p1}, \cdots, x^{pN}\}$, its entropy H_n can be expressed:

$$H_n = -\sum_{i=1}^{M} y_i \cdot \log_2 y_i$$

With y_i being the probability of the perturbed input belonging to class i. M is the total number of classes.

Based on the entropy H_n of each perturbed input x^{pn}, the entropy summation of all N perturbed inputs $\{x^{p1}, \cdots, x^{pN}\}$ is:

$$H_{sum} = \sum_{n=1}^{N} H_n$$

With H_{sum} standing for the chance the input x being trojaned. Higher the H_{sum}, lower the probability the input x being a trojaned input.

We further normalize the entropy H_{sum} that is written as:

$$H = \frac{1}{N} H_{sum}$$

The H is regarded as the entropy of one incoming input x. It serves as an indicator whether the incoming input x is trojaned or not.

Reference

1. *Gao Y, Xu C, Wang D, et al. Strip: A defence against trojan attacks on deep neural networks[C]//Proceedings of the 35th Annual Computer Security Applications Conference. 2019: 113-125.*