Project Report

I have implemented STRIP. The code in on github: https://github.com/asarmadi/ML_for_CS

The code consists of 6 files “utils.py”, “config.py”, “eval1.py”, “eval2.py”, “eval3.py”, and “eval4.py”.

The “utils.py” contains all the utility functions used in the “strip.py”. It has 6 functions that “data_loader” and “data_preprocess” are from “eval.py” in https://github.com/csaw-hackml/CSAW-HackML-2020. Other functions are as follows:

- save_image(img, name): It take an image and a name to the save that image with the name under “Figs” folder.
- cal_entropy(pi): This function takes an array of probability distribution (pi) and returns its corresponding Shannon entropy.
- find_entropy_list(model, x_test, x_valid, N): This function calculates average Shannon entropy for a set of samples (x_test) by considering N perturbed images for each image. The perturbed images are made by superimposing with inputs in a validation set (x_valid). As it was mentioned in the paper, cv2.addWeigghted function is used for superimposing the images. This function takes as input the network (model), the test set for finding their Shannon entropy (x_test), a set of validation images to superimpose (x_valid), and number of perturbed images for each sample. The function returns a list of Shannon entropies corresponding to each image in (x_test).
- find_entropy_list_rand: This function is similar to the previous mentioned function, but instead of superimposing images with samples from validation data, it generates random perturbations.
- classify_sample(model, sample, x_valid, threshold, N, n_classes): This function return the corresponding label of the sample. If the sample is poisoned, it will return n_classes (since the classes are 0 indexed). This function takes as input the network (model), the input image to classify (sample), a set of validation images to generate perturbed images (x_valid), a threshold for determining which samples to consider as poisoned (threshold), number of perturbed images for the sample (N), and total number of classes of the dataset (n_classes).
- classify_sample_rand: This function is similar to the previous mentioned function, but instead of superimposing images with samples from validation data, it generates random perturbations.

The “config.py” is the code for finding the best N and finding entropy list of all validation data that can be used for finding the threshold. This code takes as arguments the following:

- --model_filename model_dir: The model_dir is the path to the model
- --validation_data val_dir: The val_dir is the path to the file containing clean validation data
- --best_N: This argument specifies that the best N should be calculated, it is a Boolean argument
- --random: This is a Boolean argument specifying whether random perturbation should be used or not

The rest of the code consists of 3 parts:

- If –best_N is set by the user, the Shannon entropy standard deviation will be calculated for the validation data for different values of N. The figure of std values versus N will be saved in “Figs” folder.

- Shannon entropies for all the validation samples will be calculated and saved as a “npy” file that will be used for finding a threshold during validation step.

There are 4 evaluation files for each of the Badnets. All of them are similar and only differ in the model and the entropy list. The arguments of the code are as follow:

- The input image to be evaluated by the good model
- --percent pct: The pct is a float number between 0 and 1 that determines the percentage of FAR for choosing threshold
- --random: This is a Boolean argument specifying whether random perturbation should be used or not

The evaluation files output an integer number in range [0,1283] that 1283 corresponds to poisoned data.

Random perturbations vs Validation data superimposing: The main advantage of the random perturbations is that it does not need access to any validation data. However, random perturbation was not as effective as image superimposing. The False Negative rates for both methods and for all the models are reported in the following Table.

	anonymous_1	multi trigger			sunglasses
		Eye	Sun	lip	
Random Perturbation	33.26	94.41	0.17	24.77	99
Image Superimposing	24.9	73	0.69	21.15	13.11

It could be seen that random perturbation is not better than image superimposition. However, I believe if the procedure of finding the perturbation become similar to the untargeted adversarial perturbation, we can see better results.

STRIP has limitations that some of them are mentioned also in the STRIP paper such as:

- Considering a normal distribution for the Shannon entropies corresponding to the clean validation data set. This assumption affects the threshold calculation for the data.
- STRIP may not be effective for the backdoor attacks in which the poisoned images are transformed in the feature space rather than just adding a pattern to the images. For example, as it is mentioned in the ABS paper, image filters like gotham.
- Assuming that the model is backdoored. STRIP can not distinguish between a clean network and an intentionally backdoored network.
- The STRIP is not successful at finding class-specific trigger. This type of backdoor is triggered only by poising inputs from a specific label.