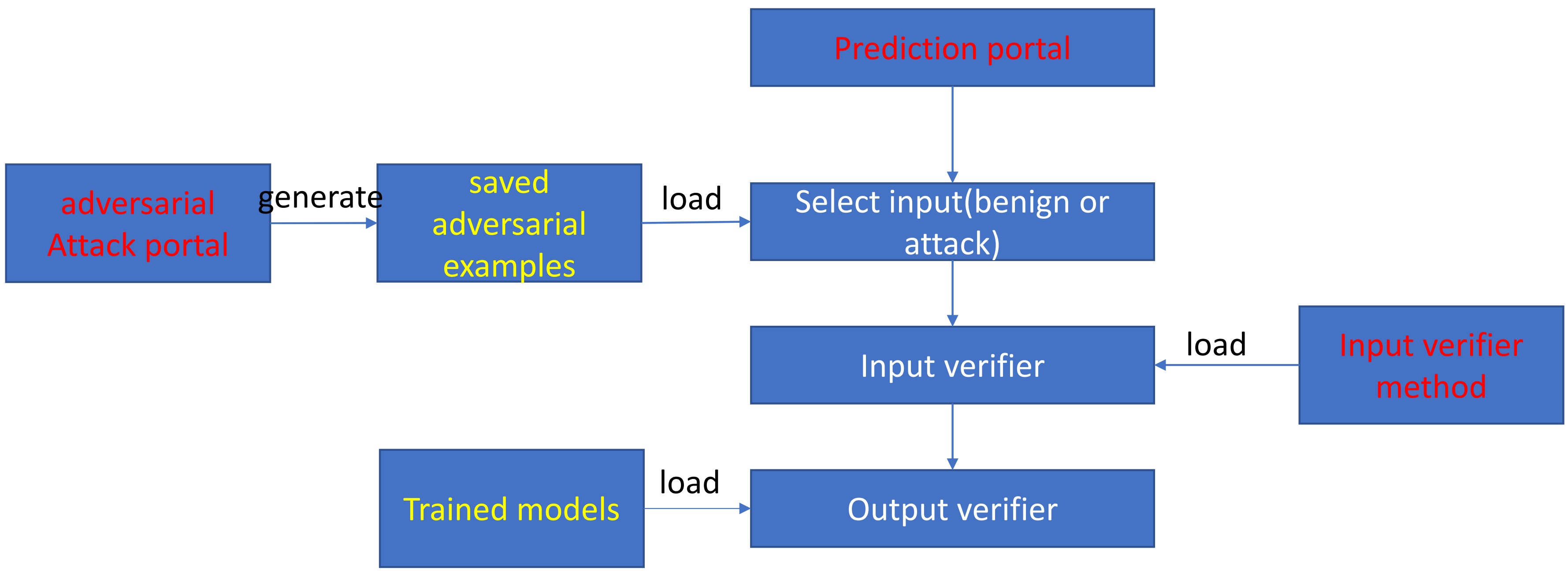
XEnsemble-1.0
Code Package

Wenqi Wei

Georgia Tech

Schema



Red: code file
Yellow: saved data/model

White: flow component of the defense-prediction portal

description

- The code package has the following portals:
- 1. The attack portal(main_attack_portal.py): generate and save adversarial examples.
- 2. The input denoising robust prediction portal(input_denoising_portal.py): given an input, generate multiple denoised variants and feed them to the target model for prediction.
- 3. The input-model cross-layer defense portal(cross_layer_defense.py): given an input, generate multiple denoised variants and feed them to multiple diverse models for prediction.(detailed generation of diverse models can be found in [2,4]). We also compare our performance with four adversarial defenses: adversarial training, defensive distillation, input transformation ensemble as provided in the paper.
- 4. Comparison portal with detection-only adversarial defenses(detection_only_comparison.py): generate defense results of Feature Squeezing, MagNet, and LID.

Dataset_name, model_name and output_verifier

dataset_name	model_name or output_verifier
MNIST	CNN1, CNN1_30, CNN1_40, CNN1_half, CNN1_double, CNN2, CNN2_30, CNN2_40, CNN2_half, CNN2_double
CIFAR-10	densenet, CNN1, CNN2, resnet-20, resnet-32, resnet-44, resnet-56, resnet-110
ImageNet	mobilenet, VGG-16, VGG-19, resnet-50, Inceptionv3
LFW	CNN1, CNN2

- Red: default model, if do not use output verifier, this model will be used for prediction. Also, adversarial attacks are generated on those models.
- Output_verifier choose multiple models from the list.

Description of the models

models	description	source	framework	comments
CNN1	a 7-layer CNN	[1]	Keras(Tensorflow)	_half: #feature maps are reduced to half _double: #feature maps are doubled _30/40: training epochs
CNN2	a 5-layer CNN	[2]	Keras(Tensorflow)	
ResNet	a ResNet model	[3]	Keras(Tensorflow)	-20/32/44/56/110: # resnet layers
Densenet	a Densenet model	[4]	Keras(Tensorflow)	40 layer, loaded a pretrained model
Mobilenet	a Mobilenet model	[5]	Keras(Tensorflow)	loaded a pretrained model
VGG	a VGG model	[6]	Keras(Tensorflow)	16 or 19 layer, loaded a pretrained model
InceptionV3	an InceptionV4 model		Keras(Tensorflow)	loaded a pretrained model

[1] https://github.com/carlini/nn_robust_attacks/blob/master/train_models.py
[2] https://github.com/tensorflow/cleverhans/blob/master/cleverhans/utils_keras.py
[3] https://github.com/keras-team/keras/blob/master/examples/cifar10_resnet.py
[4] https://github.com/titu1994/DenseNet/blob/master/densenet.py
[5] https://github.com/titu1994/MobileNetworks/blob/master/mobilenets.py
[6] https://github.com/fchollet/deep-learning-models

Attacks

Attacks type	Attack algorithm
untargeted	fgsm, bim, deepfool, pgd
Targeted (next class, most-likely and least-likely class)	FGSM, BIM, CW_i, CW_2, CW_0, JSMA

- Note: the attack input here need to be consistent with the attack_portal as the file is saved in the name of attack method, attack target model, and attack parameters.
- We use the attack format as provided by EvadeML, and we can only load one attack at a time.
- Feel free to try the attack setting in EvadeML and other attack parameters.

MNIST attack
Parameter setting

```
fgsm?eps=0.3;bim?eps=0.3&eps_iter=0.06;deepfool?overshoot=10;pgdli?eps=0.3;
fgsm?eps=0.3&targeted=most;fgsm?eps=0.3&targeted=next;fgsm?eps=0.3&targeted=ll;
bim?eps=0.3&eps_iter=0.06&targeted=most;
bim?eps=0.3&eps_iter=0.06&targeted=next;
bim?eps=0.3&eps_iter=0.06&targeted=ll;
carlinili?targeted=most&batch_size=1&max_iterations=1000&confidence=10;
carlinili?targeted=next&batch_size=1&max_iterations=1000&confidence=10;
carlinili?targeted=ll&batch_size=1&max_iterations=1000&confidence=10;
carlinil2?targeted=most&batch_size=100&max_iterations=1000&confidence=10;
carlinil2?targeted=next&batch_size=100&max_iterations=1000&confidence=10;
carlinil2?targeted=ll&batch_size=100&max_iterations=1000&confidence=10;
carlinil0?targeted=most&batch_size=1&max_iterations=1000&confidence=10;
carlinil0?targeted=next&batch_size=1&max_iterations=1000&confidence=10;
carlinil0?targeted=ll&batch_size=1&max_iterations=1000&confidence=10;
jsma?targeted=most;
jsma?targeted=next;
jsma?targeted=ll;
```

Input_verifier:

- A variety choice of input denoising method is available.
- Takes input verifier one by one with a separator “;”.
- As provided in EvadeML, added rotation.

```
inverifier_list = ['none',
                   'bit_depth_random',
                   'bit_depth',
                   'binary_filter',
                   'binary_random_filter',
                   'adaptive_binarize',
                   'otsu_binarize',
                   'median_filter',
                   'median_random_filter',
                   'median_random_size_filter',
                   'non_local_means_bw',
                   'non_local_means_color',
                   'adaptive_bilateral_filter',
                   'bilateral_filter',
                   'magnet_mnist',
                   'magnet_cifar10',
                   'rotation'
                  ]
```

XEnsemble Research

[1] Wenqi Wei, Ling Liu, Margaret Loper, Stacey Truex, Lei Yu, Mehmet Emre Gursoy, and Yanzhao Wu. "Adversarial examples in deep learning: Characterization and divergence." arXiv preprint arXiv:1807.00051 (2018).

[2] Ling Liu, Wenqi Wei, Ka-Ho Chow, Margaret Loper, Mehmet Emre Gursoy, Stacey Truex, and Yanzhao Wu, "Deep Neural Network Ensembles against Deception: Ensemble Diversity, Accuracy and Robustness." In the 16th IEEE International Conference on Mobile Ad-Hoc and Smart Systems (MASS), IEEE, 2019.

[3] Wenqi Wei, Ling Liu, Margaret Loper, Ka Ho Chow, Emre Gursoy, Stacey Truex, Yanzhao Wu. "Cross-layer Strategic Ensemble Defense against Adversarial Examples." In International Conference on Computing, Networking and Communications(ICNC), 2020.

[4] Wenqi Wei, Ling Liu, Margaret Loper, Mehmet Emre Gursoy, Stacey Truex, Lei Yu, and Yanzhao Wu, "Demystifying Adversarial Examples and Their Adverse Effect on Deep Learning", under the submission of IEEE Transaction on Dependable and Secure Computing.

[5] Wenqi Wei, and Ling Liu, "Robust Deep Learning Ensemble against Deception", under the submission of IEEE Transaction on Dependable and Secure Computing.