Hello,

This folder contains all the code files and some experimental data for the robustness experiments, divided into four subfolders. The folder "Figure 9 (a1)(a2)" contains the experimental data for "Figure 9 (a1)" and "Figure 9 (a2)" from the paper. The folder "Figure 9 (b1)(b2)(c1)(c2)" contains the experimental data for " Figure 9 (b1)", " Figure 9 (b2)", " Figure 9 (c1)", and " Figure 9 (c2)". The folder " Figure 9 plotting" contains the experimental data for plotting Figure 9, and the folder "Image set for marking object regions" contains the experimental data for marking object and background regions.

Step 1: Execute the " Figure 9 (a1)(a2)\actual model\train.py" code file to train an initial AlexNet model.

Step 2: Execute the " Figure 9 (a1)(a2)\actual model\test.py" code file to calculate the accuracy of the trained AlexNet model on the test set.

Step 3: Execute the " Figure 9 (a1)(a2)\sort and number images in order.py" code file to reorder and rename the images in the original CIFAR-10 training set to facilitate the generation of adversarial samples with reduced background classification contribution values in the next step.

Step 4: Execute the following three code files sequentially to generate three types of adversarial samples with reduced background classification contribution values:

" Figure 9 (a1)(a2)\generate adversarial samples with reduced background classification contribution value\generate adversarial samples with reduce background classification contribution value 0 0.33.py",This code file is used to generate adversarial samples with tampering strength $\varepsilon \in (0,1/3]$.

" Figure 9 (a1)(a2)\generate adversarial samples with reduced background classification contribution value\generate adversarial samples with reduce background classification contribution value 0.33 0.66.py" ,This code file is used to generate adversarial samples with tampering strength $\varepsilon \in (1/3,2/3]$.

" Figure 9 (a1)(a2)\generate adversarial samples with reduced background classification contribution value\generate adversarial samples with reduce background classification contribution value 0.66 1.py",This code file is used to generate adversarial samples with tampering strength $\varepsilon \in (2/3,1]$.

Step 5: Execute the " Figure 9 (a1)(a2)\select a specified number of adversarial samples.py" code file to select a specified number of adversarial samples as a supplement to the training set.

Step 6: Execute the " Figure 9 (a1)(a2)\set training labels for adversarial samples.py" code file to set the training labels for the selected adversarial samples from the previous step, and add them to the training set as supplementary images to train a

new model.

Step 7: Execute the following three code files to generate adversarial samples using the MI-FGSM method:

" Figure 9 (a1)(a2)\generating adversarial samples using MI-FGSM\MI FGSM adversarial samples 0 0.33.py", This code file is used to generate adversarial samples with tampering strength $\varepsilon \in (0,1/3]$.

" Figure 9 (a1)(a2)\generating adversarial samples using MI-FGSM\MI FGSM adversarial samples 0.33 0.66.py", This code file is used to generate adversarial samples with tampering strength $\varepsilon \in (1/3,2/3]$.

" Figure 9 (a1)(a2)\generating adversarial samples using MI-FGSM\MI FGSM adversarial samples 0.66 1.py", This code file is used to generate adversarial samples with tampering strength $\varepsilon \in (2/3,1]$.

Step 8: Execute the " Figure 9 (a1)(a2)\generating adversarial samples using MI-FGSM\select successful adversarial samples.py" code file to select successful adversarial samples generated by MI-FGSM as a supplement to the training set.

Step 9: Repeat steps 5 and 6 to select a specified number (2500,5000,7500,9000) of adversarial samples and set training labels as a supplement to the training set.

Step 10: Use "PGD" and "C&W" to attack the trained model. Execute the " Figure 9 (a1)(a2)\generating a test set of adversarial samples using PGD and C&W adversarial attack methods\C&W\C&W.py" code file to perform a "C&W" attack on the AlexNet model to generate a series of attack samples. Then, execute the " Figure 9 (a1)(a2)\generating a test set of adversarial samples using PGD and C&W adversarial attack methods\PGD\PGD.py" code file to perform a PGD attack on the AlexNet model to generate a series of attack samples. Finally, execute the " Figure 9 (a1)(a2)\generating a test set of adversarial samples using PGD and C&W adversarial attack methods\pick out successful adversarial samples for the attack.py" code file to select the successfully attacked adversarial samples.

Step 11: Execute the " Figure 9 (a1)(a2)\actual model\AlexNet image predict num.py" code file to count the number of images successfully attacked by "PGD" and "C&W" by loading the AlexNet model's weight files.

Step 12: Execute the code files in " Figure 9 (b1)(b2)(c1)(c2)" to use the previously constructed adversarial samples as supplements to the original training set to train the ResNet and DenseNet models. The training method is the same as in Step 1 for training the AlexNet model. Use the "train.py" code file in the corresponding model folder to train the model.

Step 13: Execute " Figure 9 plotting\Fig 9 (a1)(b1)(c1).py" to plot Fig.9 (a1), Fig.9 (b1), and Figure 9 (c1). Then, execute " Figure 9 plotting\Fig 9 (a2)(b2)(c2).py"

to plot Figure 9 (a2), Figure 9 (b2), and Figure 9 (c2).