CS6220 HW2

Problem 1

Option 1

Name: Junyan Mao

GTID: 903343678

**Code for this homework**: <https://github.com/jmao44/GTAttackPod>

* I forked the GTAttackPod Repo, cloned it locally, and made modifications
* Added/Modified files:
  + datasets/datasets\_utils.py
    - added time measurement for recording average time per example of the models under NO attack
  + attacks/lts4/deepfool.py
    - added code where I outputted the original image, the intermediate image within the loop, and the result image of the DeepFool attack. This is later commented out.
  + example\_MNIST/
    - This folder contains 30 images of 10 examples during 3 stages of the DeepFool attack on the MNIST dataset (beginning, 2nd iteration, result). The model being attacked is CNN-7
  + example\_CIFAR10/
    - This folder contains 30 images of 10 examples during 3 stages of the DeepFool attack on the CIFAR10 dataset (beginning, 1st iteration, result). The model being attacked is DenseNet-40
  + mnist\_cnn\_jmao44/MNIST\_CNN\_jmao44.ipynb
    - This is the iPython Notebook where I trained my own version of CNN on the MNIST dataset. CNN structure adopted from: <https://github.com/yashk2810/MNIST-Keras>
  + models/\_\_init\_\_.py
    - modified import statement so that the new model can be properly imported
  + models/MNIST\_jmao44.py
    - Set up model, load weights, and compile model

**Input analysis (1):** Provide a summary of your pre-trained models and datasets. For each dataset, provide 10 example inputs under five different classes, 2 per class.

*Model: CNN-7*

CNN refers to convolutional neural network, which is a type of artificial neural network and has a wide range of applications in computer vision tasks. There are several essential components to a CNN: convolutional layers to extract features; pooling layers to do down-sampling; ReLU layers to serve as activation functions; fully connected layers to help combine features and make everything into a model; and finally, Softmax function to produce the classification output. CNN-7 is simply a 7-layer (convolutional + dense layer) setup of such structure. State-of-the-art CNN models are able to achieve over 99% accuracy on the MNIST dataset.

*Dataset: MNIST database (dataset)*

MNIST stands for Modified National Institute of Standards and Technology. The MNIST database is a huge database of handwritten digits which is commonly used for training and testing number recognition machine learning models. The MNIST database consists of 60,000 training examples and 10,000 testing examples.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Digit 0 | Digit 1 | Digit 2 | Digit 3 | Digit 4 |
| Example 1 |  |  |  |  |  |
| Example 2 |  |  |  |  |  |

**Input analysis (2):** Provide a summary of the two attack algorithm of your choice.

*DeepFool* is a simple and accurate method to fool deep neural networks, by efficiently computing perturbations. According to the authors of DeepFool, it is based on an iterative linearization of the classifier to generate minimal perturbations that are sufficient to change classification labels. DeepFool also tends to generate smaller perturbations than other methods, which makes it a valuable tool to estimate the robustness of classifiers.

*PGD* stands for Projected Gradient Descent. It’s categorized as a “white-box” attack because the gradients of the model are exposed to attackers. PGD attempts to find the perturbation that maximizes the loss of a model on an input image, while keep the perturbation size under a specified threshold, called epsilon.

**Input analysis (3)**: Provide the attack examples you generated for the 10 examples you listed in 1).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Digit 0 | Digit 1 | Digit 2 | Digit 3 | Digit 4 |
| *Example 1* |  |  |  |  |  |
| DeepFool |  |  |  |  |  |
| PGD |  |  |  |  |  |
| *Example 2* |  |  |  |  |  |
| DeepFool |  |  |  |  |  |
| PGD |  |  |  |  |  |

Analysis: From the above form, we can see that DeepFool is adding “less” perturbation to the image visually than PGD, which could mean that DeepFool is a more efficient attack algorithm and is able to misdirect the model with fewer human-perceivable changes to the images.

**Output analysis (1)**:

Note: I have provided my answer to the questionnaire mentioned in requirement (2).

**Output analysis (2)+(3)+(4):**

I created a new CNN for this part of the assignment, we will be referencing it as “*CNN-jmao*” for convenience. Its structure is as follows:

A picture containing calendar

Description automatically generated

Here are the results of running different attacks on different models, on different datasets:

|  |  |  |
| --- | --- | --- |
| *Dataset/Model/Attack* | *Test Accuracy* | *Test Time Per Example/s* |
| MNIST/CNN-7/Benign | 99.43% | 0.00012614288330078125 |
| MNIST/CNN-7/DeepFool | 0% (misclassification 100%) | 0.080392 |
| MNIST/CNN-7/PGD | 4% (misclassification 96%) | 0.019253 |
| MNIST/CNN-jmao/Benign | 99.28% | 0.00023227918148040772 |
| MNIST/CNN-jmao/DeepFool | 0% (misclassification 100%) | 0.073408 |
| MNIST/CNN-jmao/PGD | 0% (misclassification 100%) | 0.017287 |
| CIFAR10/DenseNet-40/Benign | 94.84% | 0.021957121586799622 |
| CIFAR10/DenseNet-40/DeepFool | 0% (misclassification 100%) | 1.656201 |
| CIFAR10/DenseNet-40/PGD | 9% (misclassification 91%) | 0.797615 |

From the above form, it is obvious that DeepFool is a very strong attack algorithm, because the models attacked by it all achieved a test accuracy of 0%. PGD also has impressive performance. Both attack algorithms are causing the model to use more time to classify at test time, which means they are harming the image classification models’ efficiency effectively.

Screenshot of running *DeepFool-UA\_MNIST\_CNN7.py*:

Text

Description automatically generated

Screenshot of running *PGD-UA\_MNIST\_CNN7.py*:

Text

Description automatically generated

Screenshot of running *DeepFool-UA\_MNIST\_CNNjmao.py*:

Text

Description automatically generated

Screenshot of running *PGD-UA\_MNIST\_CNNjmao.py*

Text

Description automatically generated

Screenshot of running *DeepFool-UA\_CIFAR10\_DenseNet40.py*:

Text

Description automatically generated

Screenshot of running *PGD-UA\_CIFAR10\_DenseNet40.py*:

Text

Description automatically generated

**Output analysis (5):**

|  |  |  |  |
| --- | --- | --- | --- |
| *MNIST+CNN-7+DeepFool* | Original Image | Intermediate Image (2 iterations) | Final Image |
| Example 1 |  |  |  |
| Example 2 |  |  |  |
| Example 3 |  |  |  |
| Example 4 |  |  |  |
| Example 5 |  |  |  |
| Example 6 |  |  |  |
| Example 7 |  |  |  |
| Example 8 |  |  |  |
| Example 9 |  |  |  |
| Example 10 |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| *CIFAR10+DenseNet-40+DeepFool* | Original Image | Intermediate Image (1 iteration) | Final Image |
| Example 1 |  |  |  |
| Example 2 |  |  |  |
| Example 3 |  |  |  |
| Example 4 |  |  |  |
| Example 5 |  |  |  |
| Example 6 |  |  |  |
| Example 7 |  |  |  |
| Example 8 |  |  |  |
| Example 9 |  |  |  |
| Example 10 |  |  |  |

Requirements (6) c. Divergence of attack effects:

Divergence 1: inconsistency in different instances

Divergence 2: inconsistency in different models

References

<https://en.wikipedia.org/wiki/Convolutional_neural_network>

<https://en.wikipedia.org/wiki/MNIST_database>

<https://arxiv.org/abs/1511.04599>

<https://towardsdatascience.com/know-your-enemy-7f7c5038bdf3>