**Adversarial Learning Course – Ex. 1 Report**

Elad Eatah

In this exercise, one first creates and trains a classifier for the MNIST dataset.

Then, a single, correctly labeled, image, is picked, and 5 different perturbations of this image are generated, for which this classifier fails. All these perturbations differ from the original image by only a single pixel.

Finally, the perturbated-images are fed to a second classifier.

**First Classifier – A Fully-Connected Neural-Network**

This classifier receives as input a batch of grayscale images, each of size 28X28. All images' pixels are encoded in a single-precision floating-point format and are normalized so all pixels lie between 0 to 1.

This model consists of dense-layer of 128 units, with a ReLU activation-function, followed by another dense-layer of 10 units, with a softmax activation-function. The output of the model, for each sample in the input-batch, is a probability-vector for all 10 digits (0-9). The predicted value for each sample is the digit which attains the maximal probability.

The network is trained using 6 epochs of Adam, with learning-rate 0.001 and batch-size of 128 samples per-batch. The training is performed over the categorical cross-entropy loss-function (for the 10 possible labels, encoded as digits). After each epoch during training, the accuracy of the model is computed, in addition to the loss, over both the training-set and the test-set (which is supplied as a 'validation-set'). The following values are taken from the Tensorboard callback, used for the training:

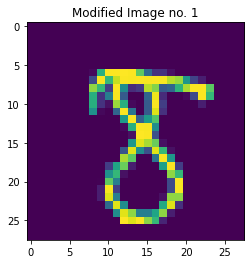
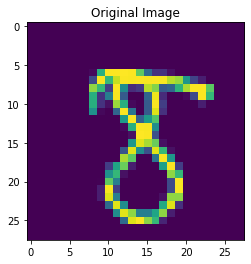
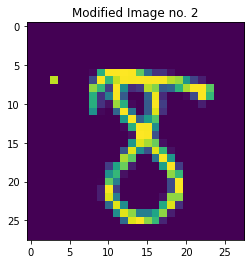
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test-Accuracy** | **Train-Accuracy** | **Test-Loss** | **Train-Loss** | **Epoch Index** |
| 0.9424 | 0.8341 | 0.2018 | 0.6052 | **1** |
| 0.9576 | 0.9446 | 0.1428 | 0.1905 | **2** |
| 0.9642 | 0.9618 | 0.1185 | 0.1312 | **3** |
| 0.9696 | 0.9723 | 0.1058 | 0.0989 | **4** |
| 0.9700 | 0.9770 | 0.0960 | 0.0797 | **5** |
| 0.9736 | 0.9821 | 0.0781 | 0.0635 | **6** |

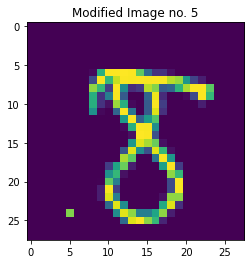
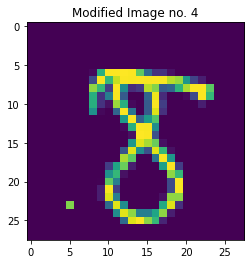
**Generating Adversarial Examples**

For the generation of the adversarial examples, image no. **107** in the test-set is picked. Then 5 perturbations of this image are made. Each perturbation differs from the original image by only a single pixel:

The first copy is altered in pixel . The second copy is altered in pixel . The third copy is altered in pixel . The fourth copy is altered in pixel . The fifth copy is altered in pixel .

The following images are the original-image and the perturbations.

**

**

All perturbations are incorrectly classified as the digit **3**, while the original-image is correctly-labeled as the digit **8**. Since all perturbations share all pixels with the original-image, excluding the modified pixel, their Hamming-distances (L0) from the original-image are all equal to **1** (the number of different pixels). Moreover, their L2-distances from the original-image equal the difference in the altered-pixel, in magnitude (all other terms in the norm vanish, and the square and the square-root nullify each other). The following table describe all such distances:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Perturbation no.** | **1** | **2** | **3** | **4** | **5** |
| **L2-distance from original image** | 0.4637 | 0.809 | 0.561 | 0.561 | 0.561 |
| **Hamming-distance from original image** | 1 | 1 | 1 | 1 | 1 |

**Second Classifier – A Random-Forest of 5 Trees**

A second classifier is then fitted to the same train-set. This classifier is a random-forest which consists of 5 trees. This classifier attains an accuracy of **99.35**% over the train-set and of **91.99**% over the test-set.

Feeding the perturbations to the new classifier results is mis-labeling these perturbations as the digit **3**, as in the first classifier.

When increasing the number of trees to 6 or higher, these perturbations are then classified correctly, as the digit **8**.

This entire experiment may be reproduced on the accompanied jupyter-notebook. This notebook was used on a GPU-machine via Google-Colab.