Deep Learning # 1
Universität Bern

Due date: 28/03/2023

1 Introduction

In this assignment you need to upload a zip file to ILIAS which includes: 1) A Jupyter Notebook file Assignment1.ipynb completed with code and answers and 2) a Jupyter Notebook exported to HTML (File / Export Notebook as / HTML). The zip file name must be FirstName_LastName.zip. Please state your name at the beginning of the notebook.

1.1 Notes on code and submission quality

In addition to answering the different questions, you are also expected to provide well written submissions. Here are some recommendations to take into consideration.

- Please answer the question in the same order as in the assignment and use the same question numbers.
- Don't answer the questions in the code comments. Use the text cells in your notebook.
- Remove clutter such as unused code lines instead of turning them into comments.
- Make sure the right execution order of the notebook cells is from top to bottom. A TA should be able to reproduce your results by simply clicking "Run All" without having to guess which cells should be executed first.

Poorly written submissions might result in points deduction.

2 Problem

In this assignment, you are asked to predict the bounding boxes of MNIST digits (see Fig. 1).

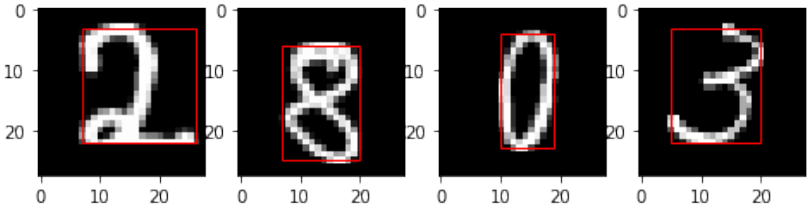

Figure 1: Given an image of a handwritten digit the task is to predict its bounding box.

For both training and testing you will use **MNIST-train** and **MNIST-test** datasets.

Tasks:

1. **Prepare data for the training [Total 35 points]**

   (a) Write a function that given a MNIST image outputs the bounding box of the digit as a tuple of 4 numbers [left, right, up, down] for the corresponding bounds. For this you may rely on the fact that the background consists of only black pixels. **[10 points]**

   (b) Construct the dataset of images from the MNIST dataset and the corresponding bounding boxes. This requires explicitly calculating ground truth bounding boxes, for what you have already implemented a function. Implement a Dataset class that extends Pytorch's Dataset class. Create dataset objects and dataloaders for corresponding data splits. You are provided with a Python file template for that. **[10 points]**

   (c) Set aside 90% of training data for training, and the remaining 10% for validation. **[5 points]**

   (d) Visualize some datapoints. See an example output in Fig. 1 **[10 points]**

2. **Train a linear model [Total 25 points]**

   (a) Your model should accept a 1d array of pixels from the image and predict 4 values for the bounds of the bounding box. Implement the training of the model with PyTorch's optimizer and a loss function appropriate for the task. **[10 points]**

   (b) To test your model implement the intersection over union metric. For 2 rectangles R_1 and R_2 it is defined as:

   $$IoU(R_1, R_2) = \frac{S(R_1 \cap R_2)}{S(R_1 \cup R_2)},$$

   where S stays for the area of the argument. What values of this metric are better? **[10 points]**

(c) Report the mean train and validation IoUs. **[5 points]**

3. **Train an MLP [Total 40 points]**

   (a) Design and train a wider and deeper model. Consider using different activation functions, normalization layers and dropout. **[15 points]**

   (b) Plot training curves. Those include train/validation loss, train/validation metric. **[10 points]**

   (c) Try different configurations of the model. Select the best model using the validation set. Justify your choice. **[5 points]**

   (d) Evaluate your selected model on the test set and report the metric on the test set. **[5 points]**

   (e) Visualize some predictions of your model. Use green color to indicate the predicted bounding boxes (see example in Fig. 2). **[5 points]**

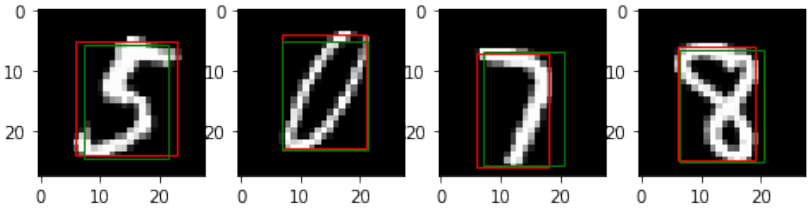

Figure 2: An example of an output from the model.