Assignment 9 – CMPUT 328
Classification and Detection on MNIST Double Digits dataset

A. OVERVIEW:

In this assignment, you are going to do classification on the MNIST Double Digits (MNISTDD) dataset. The MNISTDD dataset contains gray scale images of size 64x64. Each image has two MNIST digits (from 0 to 9) randomly placed inside it like in this visualization:

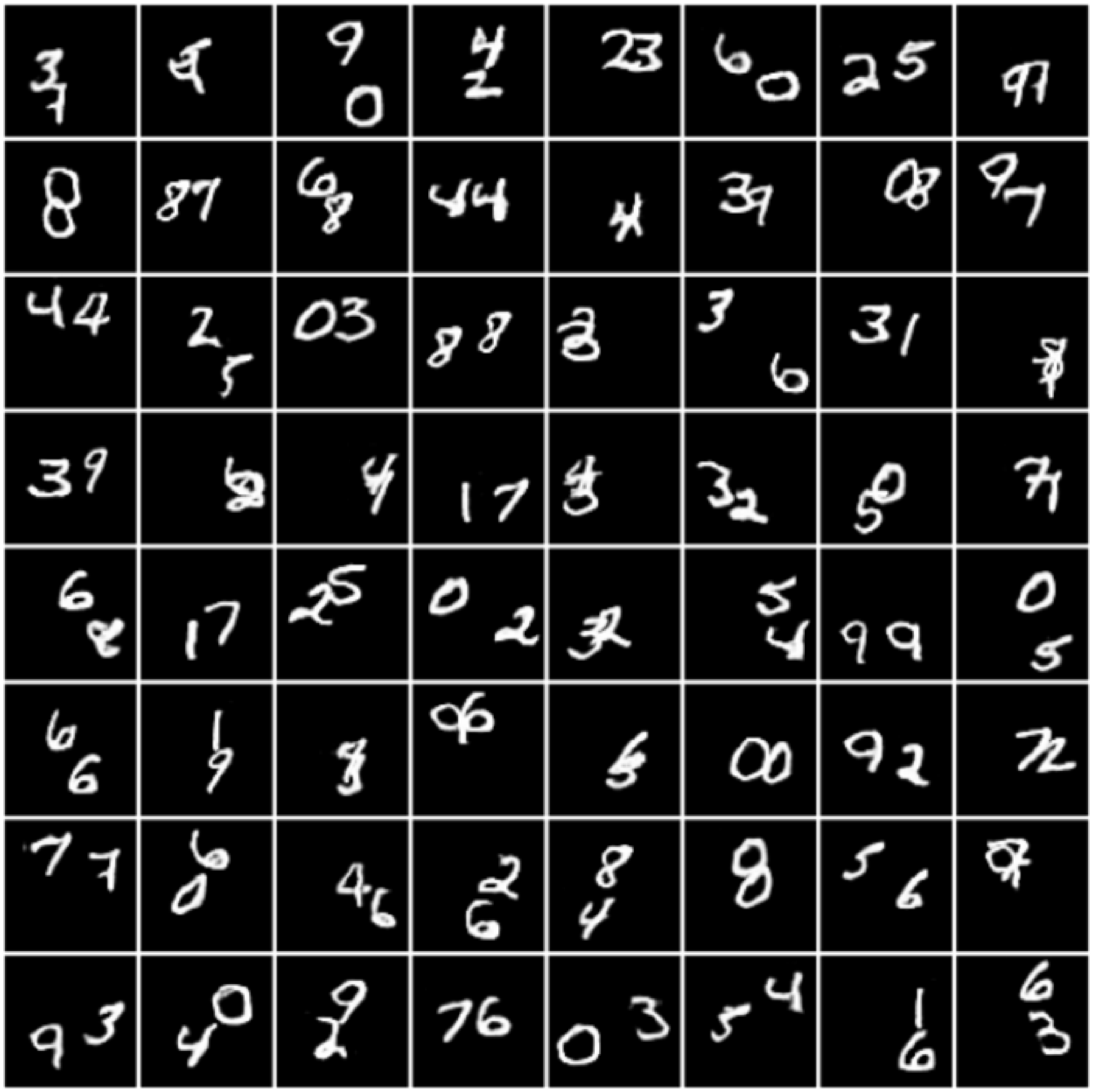


Figure 1: Visualization of the first 64 images in the MNISTDD training set. Each image is 64x64. Please note that the MNIST digits in those images are not taken directly from MNIST dataset, but rather are generated by a Generative Adversarial Networks (GANs) trained on it.

Two digits in an image may partially or completely overlap each other like shown in Figure 1 (for example, see last image of 3rd row). Completely overlap only happens in a small percentage of the images though. Most images do not have digits overlapping or only partial overlapping. Two digits in an image may also be the same class.

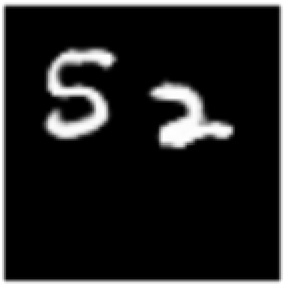
Your task in this assignment is to tell which digits are contained in each image and where are they located.

B. DATASET:

The MNISTDD dataset is divided in to 3 smaller subsets: train, validation and test. Each of the subset contains many samples. A sample consist of:

- **Image**: A 64x64 image that has been vectorized to a **4096-dimensional** vector.
- **Labels**: A **2-dimensional** vector that has two numbers in the range [0..9] which are the two digits in the image. **Note that these two number are always in ascending order.** For example, if digits number 7 and 5 are in the images then this two vector will be [5, 7] and not [7, 5].
- **Bounding boxes**: A **2-by-4 matrix** which contains two bounding boxes that mark locations of two digits in the image. The first row contain location for the first digit in the 2-dimensional vector in **labels** and the second row for the second one. Each row of the matrix has 4 numbers which are **[row of the top left corner, column of the top left corner, row of the bottom right corner, column of the bottom right corner]** in the exact order. Note: it is always the case that **row of the bottom right corner - row of the top left corner = column of the bottom right corner - column of the top left corner = 28.** This means that all bounding boxes will have a size of 28x28 no matter how large or small the digit inside that bounding box is.

As an example, consider the following 64x64 image that contains two digits 5 and 2:



- Image will be the above image but flattened to a 4096-dimensional vector.
- Labels will be [2, 5]
- Bounding boxes is a matrix that has the first row: [11, 27, 39, 55] and the second row: [6, 2, 34, 30]. which mean the digit 2 located between the 11th and 39th row and between 27th and 55th column of the image. Same goes for digit 5. It is located between the 6th and 34th row and between 2nd and 30th row of the image.

The numbers of sample contained in each set are the same as in the MNIST dataset:

- train: 55000 samples
- valid: 5000 samples
- test: 10000 samples

Each set will have 3 .npy files associate with it. These file can be read using **np.load()**. The contents inside these files are all numpy.ndarray. Let ${SET_NAME} be the name of the subset (train, valid or test). Detailed descriptions of the 3 files are shown below

- **${SET_NAME}_X.npy**: 2d matrix contains the vectorized images. Has dimension of [num_samples, 4096]. Each row is a vectorized image.
- **${SET_NAME}_Y.npy**: 2x matrix contains the labels. Has dimension of [num_samples, 2]. Note that in each row, the labels are always in ascending order.
- **${SET_NAME}_bboxes.npy**: 3d tensor contains the bounding boxes. Has dimension of

[num_samples, 2, 4]. For more information, see the description of bounding boxes above.
For example, dimension of the numpy.ndarray in 3 files of the train set:

- **train_X.npy**: [55000, 4096]
- **train_Y.npy**: [55000, 2]
- **train_bboxes.npy**: [55000, 2, 4]

You will be given the **train** and **valid** set. You can download both sets in a zip archive directly from e-class (i.e. the file “MNISTDD_train+valid.zip”). Extract this archive, you will find 6 files named: train_X.npy, train_Y.npy, train_bboxes.npy, valid_X.npy, valid_Y.npy, valid_bboxes.npy. The **test** set will **NOT** be released.

C. WHAT TO DO:

1. Two labels classification from an image in MNISTDD dataset:

Given images in the MNSITDD dataset, tell us the two digits that are present in each of these images. You can use any machine learning method to solve this classification problem. There's no restriction or guideline to what algorithm you can or should use. As long as your code provide a way for us to compute the two digits. You can see the file **mnistdd_classify.py** for a very simple reference. Your code must have a function that:
- **Receive**: an input numpy.ndarray of the test set that has size [NUM_SAMPLES, 4096]. Each row of the array is a 64x64 image that has been vectorized.
- **Return**: a 2D numpy.ndarray that has size [NUM_SAMPLES, 2]. Each row of the returned array contains 2 numbers from 0 to 9 that are the digits in the corresponding image. **Please note that two digits in each row must be in ascending order.**

2. Digit detection:

Given images in the MNISTDD dataset, give use the 4 coordinates for the 2 digits in each images. Your code must have a function that:
- **Receive**: an input numpy.ndarray of the test set that has size [NUM_SAMPLES, 4096]. Each row of the array is a 64x64 image that has been vectorized.
- **Return**: a 3D numpy.ndarray that has size [NUM_SAMPLES, 2, 4]. Each [2, 4] array for each image contains the bounding box coordinates for 2 digits in that image. **Again, the output must follow the ascending order for the digits.**

3. Assessing the performance of the classification problem:

The returned 2D numpy.ndarray will be compared with our ground truth labels for all image in the test set. The output of your algorithm must match the ground truth classes.
Finally, your performance will be ranked by this accuracy. Depend on your performance, you will get different mark for the coding part of marking. Students with highest standing will be given highest mark for this part.

4. Assessing the performance of the detection problem:

The performance of the detection problem will be determined by the average **Intersection over Union (IOU)** of your output bounding boxes vs ground truth bounding boxes in all images.
Again, in this detection problem you will be ranked by the IOU and the mark will be awarded accordingly to the ranking.

5. Final score:

The score for this assignment will be determined on the test set. The test set will have 10000 images. The image and label format for the test set is the same as the train and validation set.
Final score = 0.5 * Classification Mark + 0.5 * Detection Mark.

You are provided with a notebook that contain a function with the signature evaluation(pred_class, pred_bboxes, prefix="valid") to computes the IOU and the classification accuracy.

Important: The output of your classification and object detection problem MUST follow the order:
smaller digit first then come larger digit.
If you don't follow this order, your algorithm won't be assessed correctly and you will lose mark.

Note: You don't have to do your training inside this function. You can train your model beforehand, save the model and only use this function to compute the prediction using your saved model.

D. Marking:
Deadline: November 30th.
You should submit your **notebook that can run the evaluation function**, your trained model (if applicable) and a note that tell us **how to run your code. This note is a MUST**.

Mark rubric:
- Code **(10%)**: Depend on the performance of your code compared to other students, you will be given mark for this part accordingly.
- Presentation in the lab **(4%)**