Assignment 2

15/05/2021
COL-870

Satyam Jay
Akshay Sarashetti

Part 1 – CGAN

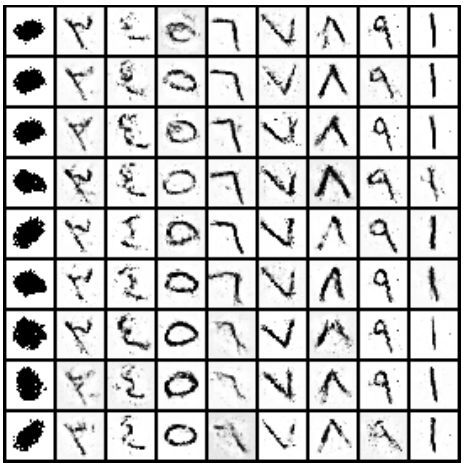
Labeling of unsupervised data

- We used K-Means to cluster the unlabeled data.
- But some of the formed clusters were very noisy. For ex- label 4 was getting mixed with label 7, and 1 was combined with 2 (according to *sample.npy*).
- We also tried extracting features and applying K-Means on those features, but the improvement was not significant.
- We applied K-Means again to the already clustered but noisy data to further refine it to resolve this issue.
- We used a pretrained MNSIT classifier to filter out other digits that are common with the MNSIT dataset.
- After we got better labeled data, we trained a classifier to classify the dataset with reasonable accuracy. We have used this classifier for the rest of the assignment whenever we need a label from the image.

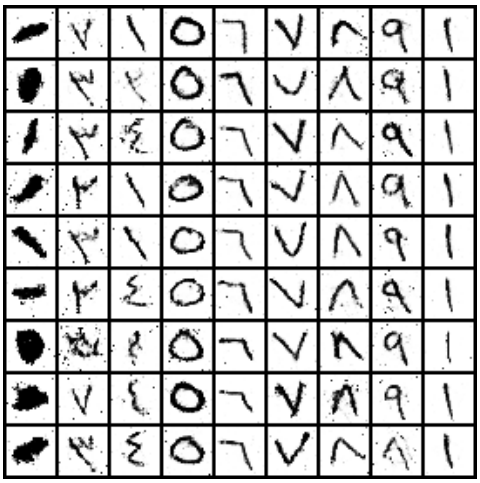
We have followed the CGAN architecture of the paper and got the following results: -

FID score

We got an FID Score of **128** when we generated **9000** images (**1000** per each digit) using Model implemented from RNN paper.



We also tried a different model, which gave the below result(Some incorrect labels are due to noisy data, the training of above model was done with better-classified data):



Part 2 Sudoku Solver

We used our classifier to get the symbolic sudoku data to train a sudoku solver model based on the RRN paper. We used the ***dgl*** library for the graph network and modified the already implemented code for 8x8 sudoku. We also reduced the hidden layer size from 96 to 84, to reduce the training time. We also tried to increase the steps, but then we could not load the model onto the memory.

We got **84.34**% accuracy for **500** epochs. We had to train for more epochs because of the **limited dataset size**(10K) as comparatively in the RRN paper the model was trained for more than 1 Lakh Images. Also, since the classifier is not 100% accurate, the data itself might be noisy.

During test time,

1) we take the sudoku image.

2) convert it into its symbolic representation.

3) Solved the sudoku and returned the solved symbolic representation.

Additional notes on how to running the scripts: -

For Part I

1. We added the command-line argument "--model_path path/to/model", which allows you to test the pretrained model without training the model yourself.
2. The script requires at least 1000 labels for each class for training, without which it will fail at runtime.

For Part II

1. We added the command-line argument "--model_path path/to/model", which allows you to test the pretrained model without training the model yourself.

Part 3 – Joint Model

We tried to add a CNN model to RNN Model which takes in Input query Image, learns the embedding for each digit and passes it on to RNN Model and a joint trained model will be obtained for RNN for which input will be an image and output will be symbolic representation.

We have implemented the model but unfortunately couldn't train it, the loss was not decreasing and we couldn't debug the issue within the stipulated time.

Trained Models are available at: Google Drive Link