This is Assignment 4 for the Machine Learning course (COL-774) at IIT-Delhi.

The goal of this assignment is to experiment with various learning algorithms on a real world dataset.

There are two parts of this question:

Part 1) Where you have to implement a set of fixed learning algorithms and

Part 2) which is a competition where you have to try out algorithms of your choice - independent of whether they have been covered in the class or not. This assignment should provide an opportunity for you to learn new techniques/tools on your own as well as try the ones that you have already seen in the class.

Problem Description:

In this assignment, we will deal with the problem of identifying an object given its hand-drawn sketch. Most of such data is collected by noting down the sequence of strokes made to draw the sketch. Instead of providing you the raw stroke level data, we are providing you the final image corresponding to each drawing. The data has been obtained from some online sources with some modifications. Your task in this assignment is to predict the object that was drawn given the image of the drawing. The dataset comes from a set of 20 different labels (objects).

Data

You are given 3 files:

- **train.zip:** Contains 20 training data files with extension ".npy". Each file contains 5000 training images all belonging to the same class, where the name of the file is the class label these examples belong to. i_{th} row in a file gives the i_{th}

- example belonging to that class. For example, "cat.npy" contains 5000 rows with an example in each row, all of these have the class label as "cat". Note that files are given in ".npy " format which can be loaded in numpy directly. Matlab users can use this library. A total of 100,000 training examples are provided.

- **test.zip:** Contains 1 test data file with extension ".npy" which contains 100,000 rows with a test example in each row. Consider row index (starting from 0) as the example id for submission.

- **sampleSubmission.csv** - a sample submission file in the correct format

All images are of size 28×28 i.e. each row in test and train files has 784 features.

Each image is in grayscale.

Assignment Details:

1. Fixed Algorithms [60 points]. Due: Friday April 27, 11:50 pm

For each of the parts below, you will report the train as well as test set accuracies. In order to obtain the test accuracies, you will have to upload the test labels on the competition website. Note that there is a limit of 30 trials per day for uploading your test labels and getting the accuracy (see the data section).

(A) K-Means: [12 points] [scikit-learn] You can use scikit-learn for implementing K-means.

i. **[4 points]** Implement the K-means (K = num of clusters) clustering algorithm to discover the clusters in the training data. Use the value of 10 for the n_init parameter (number of initializations). n_clusters should be set to 20. You can set the other parameters to their default values.

Since we have the true labels of the training data, we can compute the accuracy of clustering as follows. For each cluster obtained, assign to the cluster the label which occurs most frequently in the examples in the cluster. An example is correctly classified if and only if its label is same as the label assigned to the cluster to which it belongs.

ii. **[4 points]** From the part above, we know the label for each cluster in the training data. In order to obtain the test labels, assign the label of the cluster whose centroid is closest to the test point being considered. Report your train and test set accuracies obtained in this manner. Comment on your observations.

iii. **[4 points]** Plot the training as well as test set accuracies as we vary the max_iter parameter in the set {10,20, 30,40,50}. Comment on your observations.

(B) PCA + SVM: [12 points] [scikit-learn] Apply Principal Component Analysis (PCA) on the given datasets using scikit-learn library and use the top 50 dimensions obtained using PCA. Next, use LIBSVM to learn a multi-class linear SVM classifier using the projected set of attributes. Use the one-vs-one setting. Use internal cross-validation to determine the C parameter. What are the train and test accuracies obtained? Comment on your observations.

Note: PCA is a dimensionality reduction technique and we will be discussing this in class in another week or so. But since you do not have to build it from scratch, you are free to read about it on your own from the notes on the course website and start experimenting with it. You are also free to use online resources for additional reading.

(C) Neural Network 1 (NN): [16 points] [Pytorch/TensorFlow/Keras]

In this part, you will implement a neural network architecture. You should use one of the following 3 deep-learning frameworks: {Pytorch, TensorFlow, Keras}. If you are not already familiar with any of these frameworks, then we recommend Pytorch. Here is a short tutorial on how to use Pytorch. If there is some other deep-learning framework (not listed above) that you would like to use, let us know first.

Implement a fully connected NN architecture with one hidden layer. Use sigmoid as the activation function in the hidden layer. Implement your network using a softmax function over the possible labels in the output layer (this implies that there is a single unit in the output layer). Use negative of the log-likelihood as your loss. Use internal cross-validation to optimize over the number of units in the hidden layer. What is the optimal

number of hidden units based on your experimentation? Report your training and test accuracies using this architecture. Report your observations.

(D) Convolutional Neural Network (CNN): [16 points] [Pytorch/TensorFlow/Keras]

Use the architecture below to train your model.

Architecture: {1 CNN layer followed by MAX pooling (reducing the image size by 1/4th), followed by 1 fully connected layer}

For this part, use ReLU as the activation function in the hidden layers. As before, implement your network using a softmax function over the possible labels in the output layer (this implies that there is a single unit in the output layer). Use negative of the log-likelihood as your loss. Determine the right set of parameters (e.g., size/number of the kernels etc.) using internal cross-validation. Report the parameters for the best architecture obtained using your experimentation. Report your training and test accuracies for this architecture. Also, report your observations.

(E) Overall Comparison: [4 points] Compare the train and test accuracies obtained using the four different algorithms above. What can you say about their relative performance on this dataset? Comment on your overall observations.

2. Competition [60 points].

Deadline: Tuesday May 1, 11:50 pm. No Buffer Days Allowed.

Try out algorithms of your choice to maximize the accuracy on the test data. Improve your model as much as you can. You are strongly encouraged to try out variations of the algorithms described in the class as well as try new algorithms/models by looking up online resources. You are also free to use libraries of your choice for this part.

You will submit the labels for the test dataset on Kaggle and your performance (on classification accuracy) will be ranked on the leader-board. There are 2 leader-boards; a Public leader-board on which you can monitor your accuracy on 40% of the test dataset; a Private leader-board (which you cannot see) which will be used to determine the final rankings on the other 60% of the test data.

Note: The score obtained in this part will solely depend on the quality of your predictions on the test labels. You must submit the code for your best performing model (as well as include all the details in your report) so that we can replicate the results if required (see below).

Submission Guidelines:

Report:

You need to submit a single report for both the parts (details below).

The report is due on Wednesday May 2, 11:50 pm.

Part 1: Your report should include details of your experimentation, observations, any graphs as well as accuracies as required by the question (similar to what you have done for earlier assignments).

Part 2: Your report should include all the details of your best performing model (including libraries used) so that we can replicate the results if required.

Code:

You need to submit separate code for each part on moodle.
The deadline for each part is as mentioned above, i.e.

Part 1 is due on Friday April 27, 11:50 pm.
Part 2 is due on Tuesday May 1, 11:50 pm (with no buffer days).

Part 1: You need to submit all your code that you implemented on moodle.
Part 2: You should submit your best performing code (the one used to obtain your final test set labels). Make sure to include the details of any external libraries used in your report.
For demos, please make sure that (a) the libraries are installed on your laptop, so that we can ask you to train them during the demo if needed; (b) have separate scripts for training the model and evaluating the model on the test sets so that we test each part independently.