Machine Learning
Assignment 4
2015CS10253

A. Kmeans

Setting n_init parameter=10 and n_clusters=20

1. Default values

Train accuracy: 36.1%
Test accuracy: 35.6%

Test accuracy obtained is lesser than the train accuracy as it should be because the model was made according to training data. Also both these accuracies are small as kmeans is not a good enough method to correctly classify hand-drawn images because

For different values of Max_iter

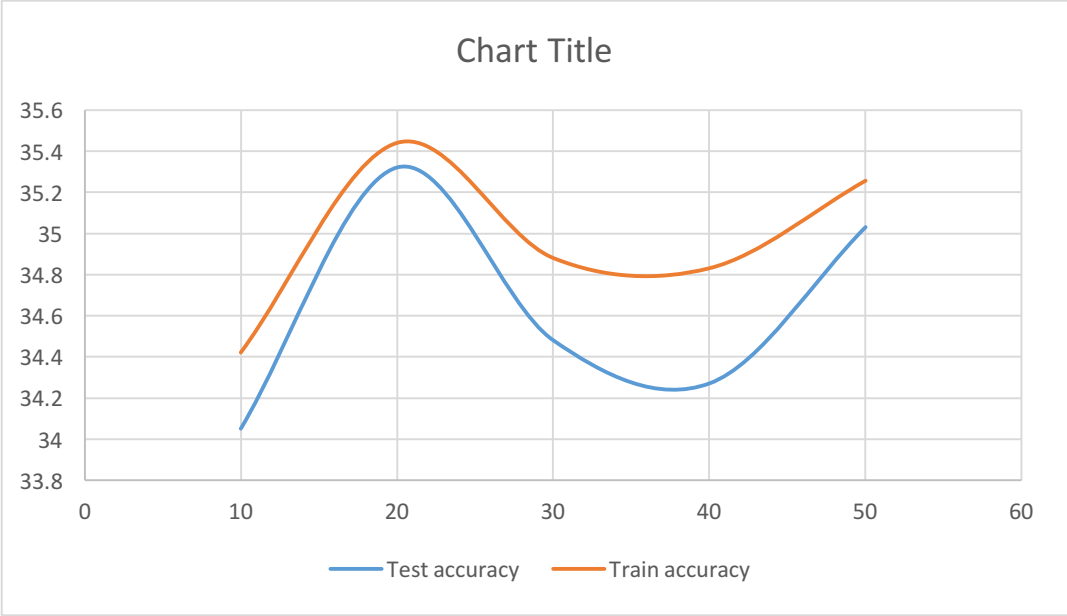
Max_iter=10
Train accuracy-> 34.42%
Test accuracy-> 34.052%

Max_iter=20
Train _accuracy-> 35.44%
Test_accuracy-> 35.32%

Max_iter=30
Train_accuracy-> 34.88%
Test_accuracy-> 34.48%

Max_iter=40
Train accuracy-> 34.83%
Test accuracy->34.27%

Max_iter=50
Train accuracy->35.256%
Test accuracy-> 35.03%



Although the Train accuracy always stays above the Test accuracy, the difference between them is the least for max_iter=20 and it is the highest for max_iter=40. The accuracy is the lowest for 10 iterations and It can only happen because 10 iterations is not enough to capture the data.

The accuracies in general are very low because the data doesn't have much spatial locality in it and almost all of the images are not very spatially away from each other.

B. PCA-SVM part

First by using PCA the dimensions of the data were reduced to 50 and then SVM was applied.
After doing internal cross validation the accuracies obtained for different values of C were:

C=0.1
Valid accuracy=69.17%

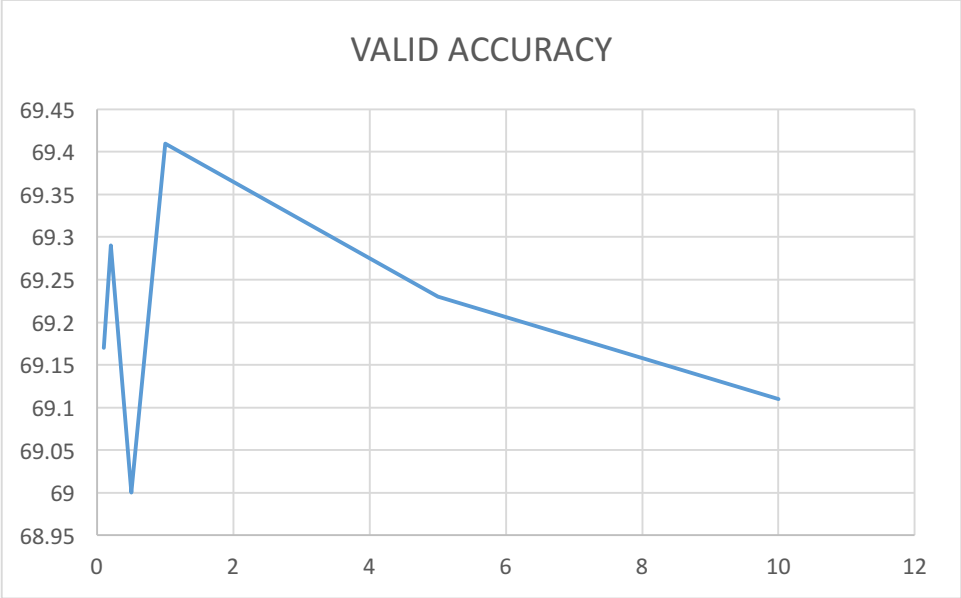
C=0.2
Valid accuracy= 69.29%

C=0.5
Valid Accuracy= 69.0%

C=1
Valid accuracy= 69.41%

C=5
Valid accuracy=69.23%

C=10
Valid accuracy=69.11%



ON X-AXIS -> Values of C

The maximum validation accuracy was obtained on C=1 and hence it was chosen as the correct value of the C parameter.
This can be explained by the fact that too less a value of C starts allowing points from other classes as too much slack is given to them and the cost to be classified wrongly is very less. Also too large a value of C also results in too harsh a slack and doesn't allow outliers.

The Training accuracy obtained= 69.897%
The Test accuracy obtained = 68.71%

C. Neural Networks

The model used had a single hidden layer with sigmoid of different sizes (experimentation) and an output layer with softmax function of 20 units each representing each class and the value in it represent how probable it is to get that class. So an argmax was taken to find the class with max prob and it was assigned.

Random exploration for cross validation (epochs used were 10-15 depending on the case)

1. When layer distribution= 30 units in hidden and 20 units of output (10 epochs)
   Train acc-> 76.38%

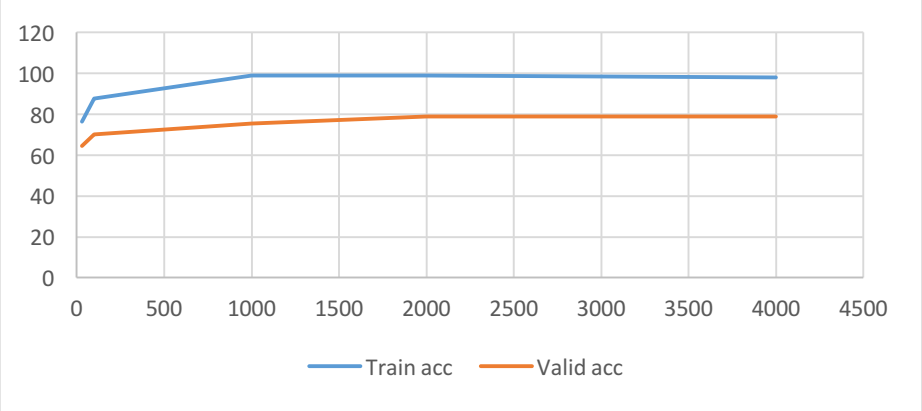
Valid acc-> 64.64%

2. When layer distribution= 100 units in hidden and 20 units of output (10 epochs)
   Train accuracy->87.8 %
   Valid accuracy->70 %

3. When layer distribution= 1000 units in hidden and 20 units of output (15 epochs)
   Train accuracy -> 99%
   Valid accuracy->75.5 %

4. When layer distribution= 2000 units in hidden and 20 units of output (10 epochs)
   Train accuracy= 99%
   Valid accuracy=79%

5. When layer distribution= 4000 units in hidden and 20 units of output (10 epochs)
   Train accuracy= 98%
   Valid accuracy= 79%



Best model chosen was with **2000 nodes** in the hidden layer and 20 units in the output layer.

On Final_testing on chosen model->
Train accuracy= 97.85%
Test accuracy= 79.355%

It was observed that lesser number of nodes were not enough to capture all the features in the data and the model obtained was ill-trained.
This could be observed from the fact that even the train accuracy after many epochs was not good enough. On increasing the number of nodes the accuracy started to improve but saturated after a while.

On going beyond 2000 nodes in the hidden layer, the number of nodes became too many and the data and its features were not enough to train the values in the network and the accuracy obtained did not increase. The data started to overfit and did not capture the real features in the network.

D. CNN model

For the CNN model, the relu function was used and the following exploration was done:
The max_pooling layer used was of (2,2)
Exploration:

1. Filters-128, dense layer= 1024, epochs=10 size=(2,2)
   Validation Accuracy-> 80.87%
2. Filters= 128, dense layer= 1024, epochs=10 size=(3,3)
   Validation Accuracy->86.5%
3. Filters- 128, dense layer=1024, epochs=10, size=(4,4)
   Validation Accuracy=88.2%
4. Filters- 32, dense layer=1024, epochs=10 size=(3,3)
   Validation Accuracy=87.15%
5. Filters- 64, dense layer=1024, epochs=10 size=(3,3)
   Validation Accuracy=86.8%
6. Filters- 96, dense layer=1024, epochs=10 size=(3,3)
   Validation Accuracy=87.0%

The chosen parameters were number of filters= 64 and dense layer size= 1024, and size of filters=(4,4)

The Train accuracy obtained on it was: 99%
The Test accuracy obtained on it was: 87.5%

E. Comparison

The train and test accuracies obtained were best for the CNN network model and they proved fairly larger than all the other models. This shows us that since the data is representative of hand-drawn images, CNN captures the data distribution due to its inherent property of understanding images better.

Model	Train accuracy	Test Accuracy
Kmeans	36.1%	35.6%
PCA-SVM	69.897%	68.71%
NN	97.8%	79.3%
CNN	99%	87.5%

As CNN exploits relative spatial locality in data, it proves the best model.
The Kmeans did not prove to be a very good model, because the data did not contain very specific spatial features which could be used by Kmeans. PCA-SVM did provide better results because of dimension reduction leading to extraction of only the important parts. NN was able to learn complex features but not as many as CNN for the above reasons.

THE COMPETITION

1. The link for the drive is:

   https://drive.google.com/drive/folders/14jPXebtQKRmjhSf4IX9NidfFDtC8DC73?usp=sharing

   This contains various models and best model is ”newmodel.h5” with an accuracy of about 92.57% on test data.
   The code for the same is titled “ my_model.py “

2. To make the model I have used the Keras architecture and used its inbuilt functions. To get the accuracy as shown on the leaderboard I used various models trained during the period of last few days and the code for the ensemble also has been submitted by me. Although I deleted or overwrote most of the models that I built, the drive link also contains some other models that I trained to make the ensemble.

3. The model was as follows (it can be better understood through the code):
   a. 1 conv layer with relu function kernel size= (3,3) and number of filters=64
   b. 1 conv layer with relu function kernel size= (2,2) and number of filters=128
   c. MaxPooling2d layer of size (2,2)
   d. 1 conv layer with relu function kernel size= (3,3) and number of filters=128
   e. 1 conv layer with relu function kernel size= (3,3) and number of filters=128
   f. MaxPooling2d layer of size (2,2)
   g. Dropout layer with 40% dropout
   h. Flatten layer to linearize input
   i. Dense layer with 1800 units
   j. Batch normalization layer
   k. Dense layer with 20 output nodes of activation ‘softmax’

The loss function was negative of log likelihood and optimizer used was ‘adadelta’
The model was trained on 25 epochs with batch size=100