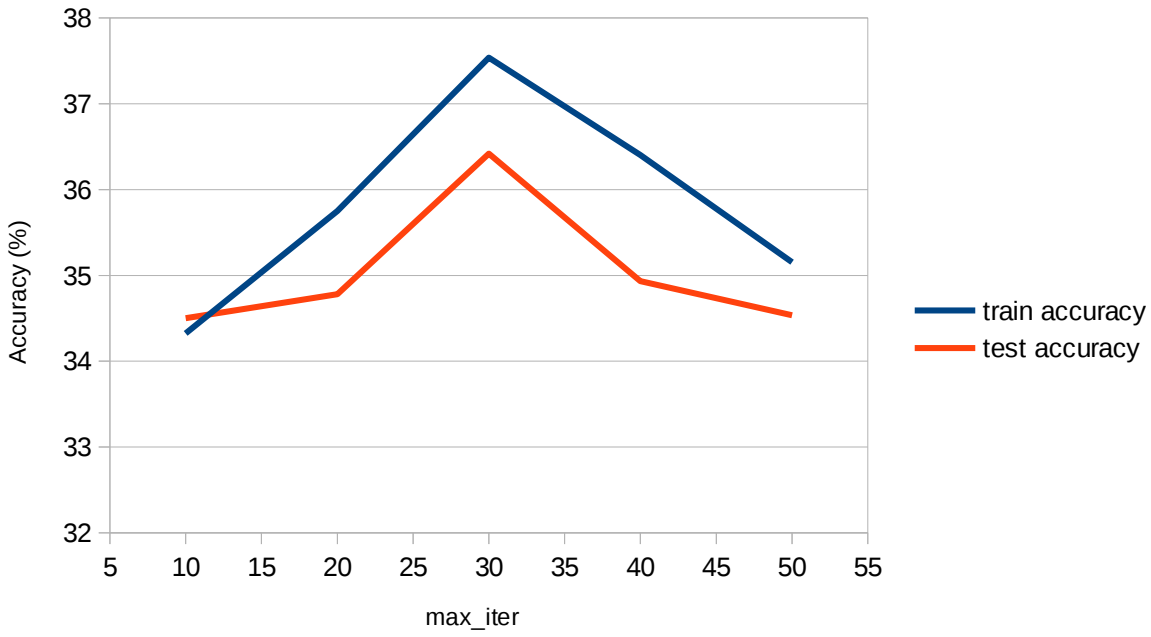
Assignment 4

Question 1

To run the code for all parts there should be 2 folders train and test with training and testing data.

Part(a) K-Means

For n_init=10(default) and n_clusters=20 and max_iter=300(default)
Training Accuracy=36.96%
Testing Accuracy=36.58%



To run the code :“python3 a.py”

Part(b) PCA+SVM

Did not normalize or scale the data, because it was giving less accuracy.

Best accuracy comes for C=0.000001
Training score=65.731%

Used sklearn.model_selection to do internal cross-validation
Cross validation Accuracies=[0.65 0.6509 0.6545 0.6495 0.6524]

Testing Accuracy=65.547%

Doing PCA reduces the dimensions to 50 which in tern increases the accuracy of the SVM.

To train :“python3 b.py”
To test :“python3 btest.py”
Training and testing finished in 87 minutes.

Part(c) Neural Network(NN)

Used pytorch to implement.

Implemented a fully connected NN architecture with one hidden layer.
Best accuracy comes for 1400 neurons in hidden layer.

Training Accuracy=89.371%
Testing Accuracy=74.447%

Part(d) Convolutional Neural Network(CNN)

Used pytorch to implement.

Implemented 1 CNN layer followed by MAX pooling (reducing the image size by 1/4th), followed by 1 fully connected layer.

Training Accuracy=90.431%
Testing Accuracy=84.975%

Best accuracy comes with 32 filters, 5*5 kernel size.

Comparision:

Train and test accuracies are in increasing order:
K-Means
PCA+SVM
NN
CNN

As the model becomes more complex the training and test accuracies increase.
K-means optimizes variances. The unweighted sum of variances is not meaningful on this data set . In high-dimensional data, distance doesn't work and variance = squared Euclidean distance.

Problem of optimising variance is somewhat solved by SVM which tries to separate the data. PCAalso helps the SVM by reducing the dimensionality of data.

Neural networks are good for high dimensional data.
But Neural networks don’t take into account the properties of images.

CNNs take into account the local features of the data and use the fact that the data is a set of images.

Convolution and filters help in learning particular features of the data.

Question 2

Used Keras, tensorflow to implement

Architecture:

convolutional layer with 32 filters , 3*3 kernel size, relu as activation function
convolutional layer with 32 filters , 3*3 kernel size, relu as activation function
Max pooling layer with pool size= 2*2
Dropout with 0.2 probability
convolutional layer with 64 filter , 3*3 kernel size, relu as activation function
convolutional layer with 64 filter , 3*3 kernel size, relu as activation function
Max pooling layer with pool size= 2*2
Dropout with 0.5 probability
Fully connected layer with 512 neurons, relu as activation function
Dropout with 0.5 probability
Fully connected outtput layer with 20 neurons, softmax as activation function

Used categorical_crossentropy loss and adam optimiser

Keras code for architecture:

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3,3), activation='relu', input_shape=(28, 28, 1)))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.2))
model.add(Conv2D(64, kernel_size=(3,3), activation='relu'))
model.add(Conv2D(64, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(classes, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Dropout actually increases accuracy by reducing overfitting.

To train the model: “python3 best_n2_keras_train.py”
This will generate 2 files in which the model is saved.
To test the model: “python3 best_n2_keras_train.py”

Link for saved models: https://drive.google.com/drive/folders/1tqFa0VND60DG4Ke-xKJ34I27-odbnXNf?usp=sharing