REPORT

DATA

Training : Batches 1-4
Testing : Batch 5

Batch size = 32
Epochs = 50

WITHOUT BATCH NORMALIZATION

Model	Accuracy
Conv2D(32,(3,3)) Conv2D(32,(3,3)) Conv2D(64,(3,3)) Conv2D(64,(3,3)) MaxPooling2D(pool_size=(2,2)) Dense(512) Dense(256) Dense(10)	**64.16 %**
Conv2D(32,(3,3)) Conv2D(64,(3,3)) Conv2D(64,(3,3)) Maxpooling2D(pool_size=(3,3)) Dense(512) Dense(512) Dense(256) Dense(10)	**65.15 %**
Conv2D(32,(3,3)) Conv2D(32,(3,3)) Maxpooling2D(pool_size=(2,2)) Conv2D(64,(3,3)) Conv2D(64,(3,3)) Maxpooling2D(pool_size=(2,2)) Dense(512) Dense(10)	**70.48 %**

WITH BATCH NORMALIZATION

Model	Accuracy
model.add(Conv2D(32, (3, 3), padding='same', input_shape=x_train.shape[1:])) model.add(BatchNormalization()) model.add(Activation(i)) model.add(Conv2D(32, (3, 3))) model.add(BatchNormalization()) model.add(Activation(i)) model.add(MaxPooling2D(pool_size=(2, 2))) model.add(Dropout(0.25)) model.add(Conv2D(64, (3, 3), padding='same')) model.add(BatchNormalization()) model.add(Activation(i)) model.add(Conv2D(64, (3, 3))) model.add(BatchNormalization()) model.add(Activation(i)) model.add(MaxPooling2D(pool_size=(2, 2))) model.add(Dropout(0.25)) model.add(Flatten()) model.add(Dense(512)) model.add(BatchNormalization()) model.add(Activation(i)) model.add(Dropout(0.5)) model.add(Dense(num_classes)) model.add(BatchNormalization()) model.add(Activation('softmax'))	**81.75 %**

With the above model, we ran the code for several activation function. The results for those were :-

Activation Function	Accuracy
Relu	81.72 %
Sigmoid	51.06 %
Tanh	10.03 %
Elu	79.02 %

PART C

Objective : Use the CNN as a feature training network and use SVM to classify the same

Model

```
model.add(Conv2D(32, (3, 3), padding='same',
                 input_shape=x_train.shape[1:]))
    model.add(BatchNormalization())
    model.add(Activation(i))
    model.add(Conv2D(32, (3, 3)))
    model.add(BatchNormalization())
    model.add(Activation(i))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Dropout(0.25))

    model.add(Conv2D(64, (3, 3), padding='same'))
    model.add(BatchNormalization())
    model.add(Activation(i))
    model.add(Conv2D(64, (3, 3)))
    model.add(BatchNormalization())
    model.add(Activation(i))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Dropout(0.25))

    model.add(Flatten())
    model.add(Dense(512))
    model.add(BatchNormalization())
    model.add(Activation(i))
    model.add(Dropout(0.5))
    model.add(Dense(num_classes))
    model.add(BatchNormalization())
    model.add(Activation('softmax'))

get_3rd_layer_output = K.function([model.layers[0].input, K.learning_phase()],
                                  [model.layers[19].output])
clf = SVC()
clf.fit(x_layer_output,y_layer_output)
y_pred = clf.predict(x_test_layer)
```

ACCURACY = 79.87 %