Programming Assignment 1

R KAUSHIK — EE15B105

September 9, 2018

In all the experiments two different values of learning rates 0.01 and 0.05 and momentum parameter 0.3 and 0.9 was carefully selected after lot of trial runs with other values of these parameters. These trial runs included performing faster version of 5-fold cross validation, by training each fold only 1000 iterations and comparing the results - the difference between average accuracy's were significant in such models, or by early stopping the training after observing either NaN's (as in the case of relu with softmax activation), or very poor performance of model in terms of validation accuracy in the initial stages.

1. **Test and training loss convergence**

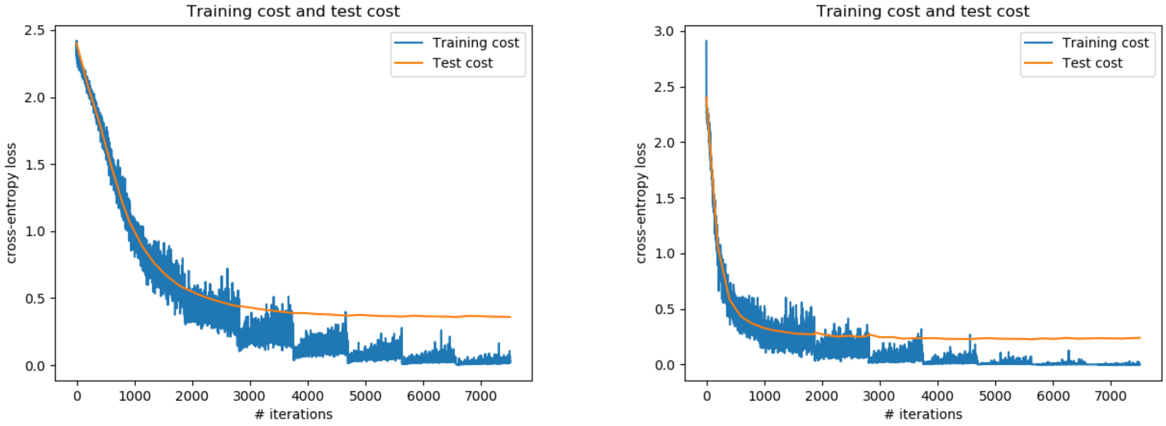


Figure 1: The plots show the progression of cross entropy cost, averaged over a mini batch, over iterations. The left plot corresponds to the model trained with learning rate 0.01 and momentum 0.3. The right plot corresponds to the model trained with learning rate 0.05 and momentum 0.3. We can see faster convergence as learning rate increases

- The results of training the sigmoid activation model with momentum factor of 0.9 is shown below

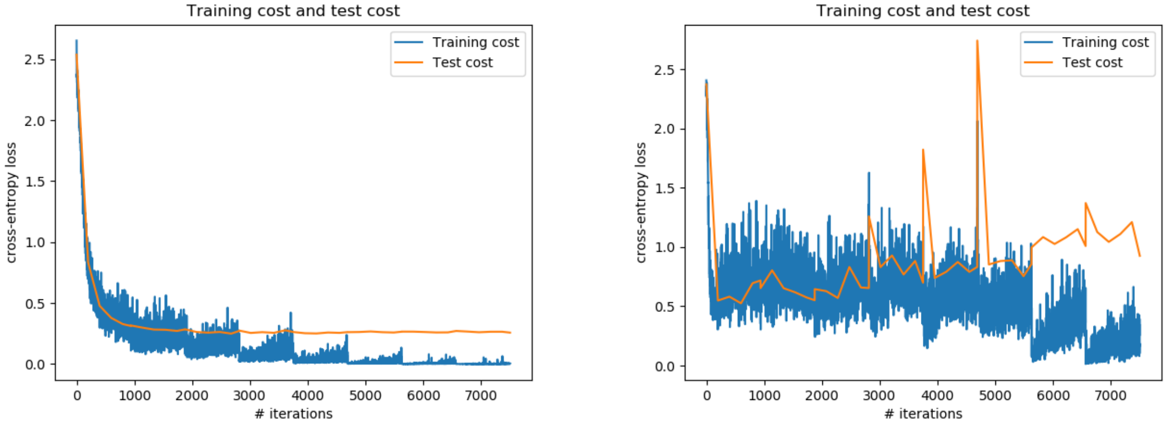


Figure 2: The plots show the progression cross entropy cost, averaged over a mini batch, over iterations. The left plot corresponds to the model trained with learning rate 0.01. The right plot corresponds to the model trained with learning rate 0.05. The momentum factor corresponding to both the plots is 0.9. We can see the ragged nature of training undergone in the model corresponding to right plot

- Based on the initial validation results of 5-fold cross-validation, some other models were crossed off before completion. For example models such as sigmoid model with learning rate 0.05 and momentum 0.9 showed lot of fluctuations with poor validation accuracy at each step - See Fig 2 right. So cross-validation for such models were terminated before completion.

2. **Metrics**

- Confusion matrix for sigmoid activation model when evaluated on test dataset

P T	0	1	2	3	4	5	6	7	8	9
0	942	0	9	2	1	9	10	2	5	0
1	0	1116	2	2	0	1	4	1	9	0
2	13	16	903	13	25	1	17	12	31	1
3	5	3	32	881	2	40	1	20	20	6
4	1	5	5	1	893	0	13	0	7	57
5	18	5	11	41	25	734	11	10	21	16
6	13	4	19	1	24	22	872	1	2	0
7	2	24	33	5	12	0	1	904	6	41
8	10	13	10	31	20	36	16	10	802	26
9	7	6	9	18	54	14	0	33	5	863

- Evaluation metrics at each fold for model with learning rate 0.05, momentum 0.3

(a) Fold 1:
  - Overall accuracy: **0.9416**
  - Precision: **0.9415**
  - Recall: **0.9408**
  - F1-score: **0.9409**
  - Mean of validation error: **0.1922**
  - Standard deviation of validation error: **0.6756**

(b) Fold 2:
  - Overall accuracy: **0.9287**
  - Precision: **0.9276**
  - Recall: **0.9275**
  - F1-score: **0.9274**
  - Mean of validation error: **0.2277**
  - Standard deviation of validation error: **0.7444**

(c) Fold 3:
  - Overall accuracy: **0.9234**
  - Precision: **0.9229**
  - Recall: **0.9227**
  - F1-score: **0.9226**
  - Mean of validation error: **0.2504**
  - Standard deviation of validation error: **0.8102**

(d) Fold 4:
  - Overall accuracy: **0.9181**
  - Precision: **0.9169**
  - Recall: **0.9171**
  - F1-score: **0.9168**
  - Mean of validation error: **0.2651**
  - Standard deviation of validation error: **0.8212**

(e) Fold 5:
  - Overall accuracy: **0.9178**
  - Precision: **0.9171**
  - Recall: **0.9166**
  - F1-score: **0.9168**
  - Mean of validation error: **0.2709**
  - Standard deviation of validation error: **0.8470**

3. 5-fold cross-validation results with relu activation

Learning rate of 0.01 was found to be optimal for relu model as higher values of learning rate caused numerical overflows with softmax function.

- Learning rate 0.01, momentum 0.9: Average **validation accuracy** = 0.9601
- Learning rate 0.01, momentum 0.3: Average **validation accuracy** = 0.9758. Since the validation accuracy of this model is greater than that with momentum of 0.9, this model was chosen for evaluation with test dataset. Overall prediction **accuracy with test dataset** = 0.974
- Sigmoid model with same hyper-parameters gave overall **test accuracy of** 0.891
- Comparison of training and test cost convergence plots for relu and sigmoid activations are shown below

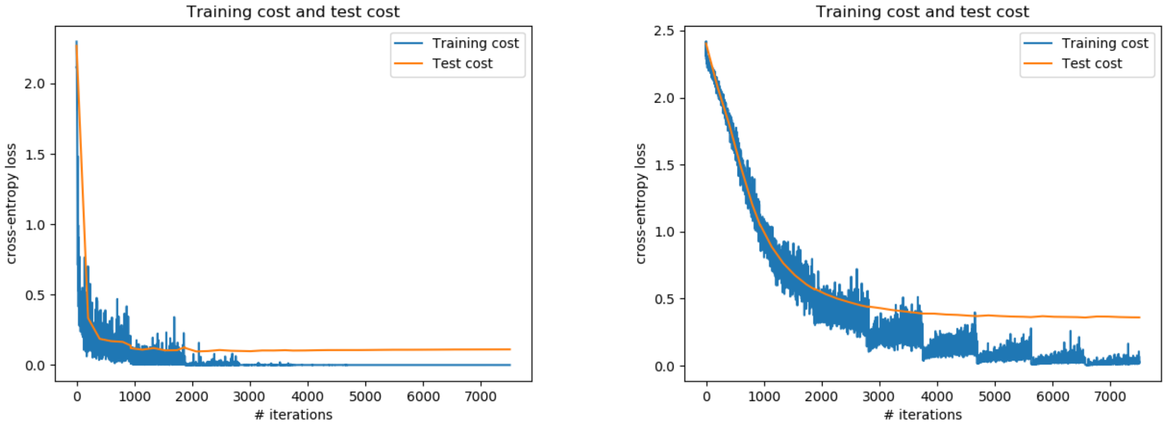


Figure 3: The left plot shows the convergence of train and test prediction cost with ‘relu‘ activation, learning rate 0.01 and momentum 0.3. The plot on the right shows train and test cost convergence with ‘sigmoid‘ activation, learning rate 0.1 and momentum 0.3

We can see that training and test error for model with relu activation converges much more faster than that for model with sigmoid activation.[1] Since the convergence of error rates with relu activation was faster, different learning rates were not tried.

- larger values of learning rate, (≥ 0.05) causes overflow in the case of relu activation. However different learning rates were explored in between 0.01 and 0.05. Notably, with a learning rate of 0.03 and momentum of 0.3, a better validation accuracy of 0.9741 (against older validation accuracy of 0.9706) was observed. Yet the accuracy on test dataset was found to be 0.9727, slightly lesser than the other model's prediction accuracy
- Evaluation metrics at each fold for model with learning rate 0.01, momentum 0.3

  (a) Fold 1:
  - Overall accuracy: **0.9824**
  - Precision: **0.9823**
  - Recall: **0.9823**
  - F1-score: **0.9823**
  - Mean of validation error: **0.0725**
  - Standard deviation of validation error: **0.7143**

  (b) Fold 2:
  - Overall accuracy: **0.9828**
  - Precision: **0.9828**
  - Recall: **0.9826**
  - F1-score: **0.9827**
  - Mean of validation error: **0.0664**
  - Standard deviation of validation error: **0.5692**

  (c) Fold 3:
  - Overall accuracy: **0.9749**
  - Precision: **0.9746**
  - Recall: **0.9745**
  - F1-score: **0.9745**
  - Mean of validation error: **0.1011**
  - Standard deviation of validation error: **0.7147**

  (d) Fold 4:
  - Overall accuracy: **0.9704**
  - Precision: **0.9700**
  - Recall: **0.9701**
  - F1-score: **0.9700**
  - Mean of validation error: **0.1098**
  - Standard deviation of validation error: **0.7185**

  (e) Fold 5:
  - Overall accuracy: **0.9685**
  - Precision: **0.9683**
  - Recall: **0.9682**
  - F1-score: **0.9682**
  - Mean of validation error: **0.1288**
  - Standard deviation of validation error: **0.8574**

From now all, all the experiments will be carried on with relu activation model with learning rate 0.01 and momentum 0.3

[1] It was observed that the weights and bias initialization for relu model need to be done differently than sigmoid model. More precisely, the standard deviation of normal distribution from which the parameters are initialized was lesser for relu model. This was done to avoid overflow while calculating softmax exponenets

4. **Regularization**

   - **Data augmentation** Each batch of the training data was augmented - added random noise and rotated at random angle $\leq 45^o$, before feeding a batch in every iteration. The resulting accuracy on test data was found to be **0.9765**. The cost convergence plot is shown below

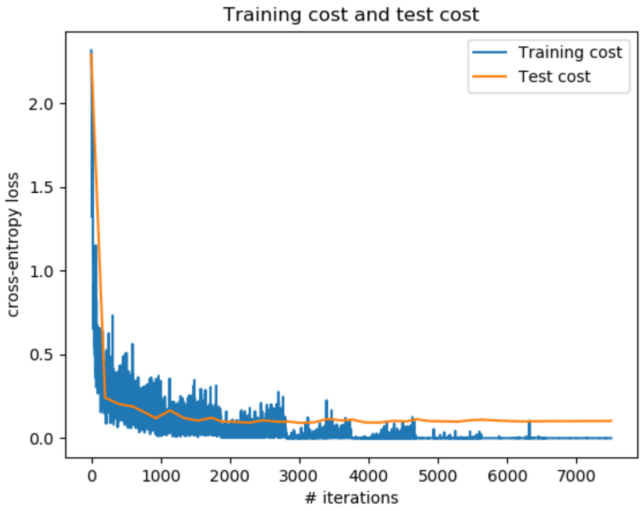


Figure 4: Training and test cost convergence when data augmentation was performed. The mini-batch data was randomly rotated and random noise was added in every iteration

   Another observation was made during training with data augmentation. The difference between training and test error was lesser than other models trained without data augmentation. While training models without data augmentation, there were some instances where training cost would practically be 0 (actual cost would be of order 10^{-8}), while test error would be of the order 0.1, showing strong overfitting. Such cases were hardly observed with data augmentation, yet the training error consistently remained lot lesser than the test error (limiting order of training error: 10^{-4}, limiting order of test error 0.08)

   - **L1/L2 regularization** Training with L1/L2 regularization term did not result in any improvement in the performance on cross-validation or test. Several variations were tried with different range values of regularization parameters. The model was found to perform better when the regularization parameter was close to 0

     Average validation accuracy for different choice of regularization parameters, after performing 5-fold cross validation is shown below

     – L2 weight 0.01: **0.9653**
     – L2 weight 0.1: **0.9433**
     – L2 weight 1.0: No visible improvement was observed in cross-validation accuracy. So the cross-validation was stopped before completion
     – L1 weight 0.001: **0.9409**
     – L1 weight 0.01: Showed comparatively poor validation accuracy in each fold (0.7959 - 0.8712). Hence it was stopped before completion

   Hence we can conclude that data augmentation does a better job at generalization than other regularization methods.

5. **Top 3 predictions**
   20 random images were chosen from test dataset and the top 3 predictions based on the one-hot output of the neural network was computed. We can see a lot of overlap in the top-3 predictions made by different models. The top predictions made by 3 models are same in almost all the images. The only exception is the image on the left side of last row of the table 1. Yet the elements in the top-3 prediction set are same for all the models for that image.

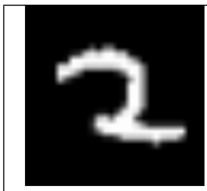
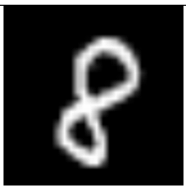
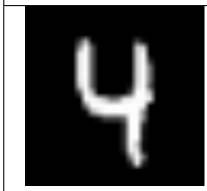
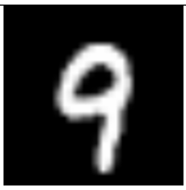
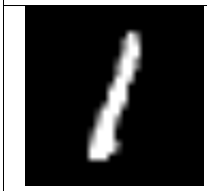
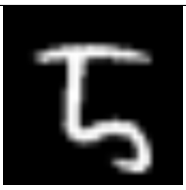
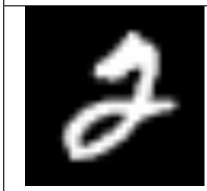
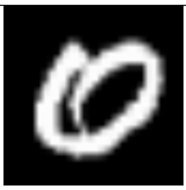
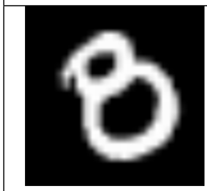
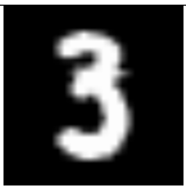
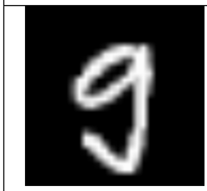
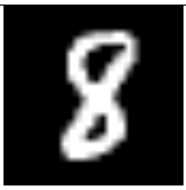
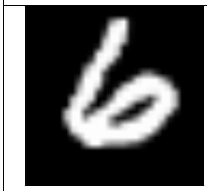
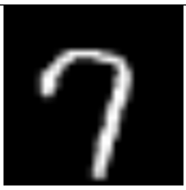
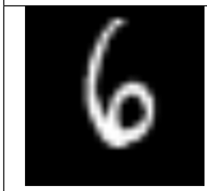
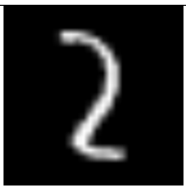
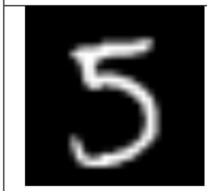
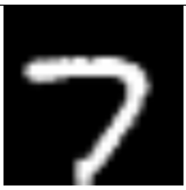
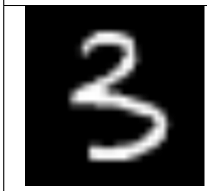
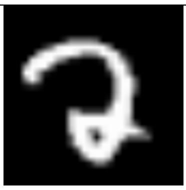
	**True label: 2** **sigmoid model**: [7, 2, 9] **relu model**: [2, 7, 3] **data augmentation**: [2, 7, 3]		**True label: 8** **sigmoid model**: [8, 5, 9] **relu model**: [8, 2, 9] **data augmentation**: [8, 2, 3]
	**True label: 4** **sigmoid model**: [4, 9, 3] **relu model**: [4, 9, 7] **data augmentation**: [4, 9, 7]		**True label: 9** **sigmoid model**: [9, 4, 7] **relu model**: [9, 4, 3] **data augmentation**: [9, 3, 7]
	**True label: 1** **sigmoid model**: [1, 8, 2] **relu model**: [1, 2, 7] **data augmentation**: [1, 8, 2]		**True label: 5** **sigmoid model**: [5, 6, 2] **relu model**: [5, 2, 8] **data augmentation**: [5, 2, 0]
	**True label: 2** **sigmoid model**: [2, 7, 3] **relu model**: [2, 3, 0] **data augmentation**: [2, 5, 3]		**True label: 0** **sigmoid model**: [0, 5, 2] **relu model**: [0, 6, 9] **data augmentation**: [0, 6, 2]
	**True label: 8** **sigmoid model**: [0, 2, 5] **relu model**: [0, 8, 9] **data augmentation**: [0, 3, 5]		**True label: 3** **sigmoid model**: [3, 5, 8] **relu model**: [3, 9, 8] **data augmentation**: [3, 9, 5]
	**True label: 9** **sigmoid model**: [9, 3, 5] **relu model**: [9, 5, 0] **data augmentation**: [9, 5, 3]		**True label: 8** **sigmoid model**: [8, 1, 2] **relu model**: [8, 3, 2] **data augmentation**: [8, 1, 2]
	**True label: 6** **sigmoid model**: [6, 2, 4] **relu model**: [6, 4, 2] **data augmentation**: [6, 0, 4]		**True label: 7** **sigmoid model**: [7, 9, 3] **relu model**: [7, 9, 3] **data augmentation**: [7, 9, 3]
	**True label: 6** **sigmoid model**: [6, 4, 2] **relu model**: [6, 2, 4] **data augmentation**: [6, 0, 5]		**True label: 2** **sigmoid model**: [2, 1, 3] **relu model**: [2, 1, 7] **data augmentation**: [2, 3, 7]
	**True label: 5** **sigmoid model**: [5, 3, 8] **relu model**: [5, 3, 9] **data augmentation**: [5, 3, 9]		**True label: 7** **sigmoid model**: [7, 9, 0] **relu model**: [7, 9, 3] **data augmentation**: [7, 3, 9]
	**True label: 3** **sigmoid model**: [5, 3, 8] **relu model**: [3, 5, 8] **data augmentation**: [3, 5, 8]		**True label: 2** **sigmoid model**: [2, 3, 7] **relu model**: [2, 7, 3] **data augmentation**: [2, 7, 3]

Table 1: Top 3 predictions of different variations of neural network model trained. The 3 variations considered here are model with **sigmoid** activation, model with **relu** activation, model trained with **data augmentation**

6. **Manual feature extraction**: HOG features were extracted from the $(28, 28)$ dimension input images. The cell size of $(7, 7)$ was used for extracting HOG features which resulted in a feature vector of length 324 for a single 28×28 image.

   (a) **RELU activation**:It was observed that the usual set of hyper-parameters, the learning rate and momentum factor, while keeping the architecture fixed, resulted in a very slow learning. Hence through several experiments it was found that learning rate of 0.5 and momentum factor of 0.3 was optimal in terms of speed of learning and avoiding overflow with softmax.

   The average cross validation accuracy for different architectures are shown below:

   - Architecture [324 - 1000 - 500 - 250 - 10]: **0.9694**
   - Architecture [324 - 500 - 100 - 10]: **0.9655**
   - Architecture [324 - 500 - 500 - 200 - 10]: **0.9650**

   (b) **sigmoid activation**: The learning rate of 0.5 was observed to be quite small for this model as the learning proceeded slowly. Hence learning rate of 1.0 was used for experimenting with different architectures

   - Architecture [324 - 1000 - 500 - 250 - 10]: **0.9255**
   - Architecture [324 - 500 - 250 - 10]: **0.9255**

   The model showing best validation accuracy is the model with relu activation and the architecture [324-1000-500-250-10].

   The confusion matrix for the best performing model is

P T	0	1	2	3	4	5	6	7	8	9
0	1173	0	4	0	0	0	1	0	14	0
1	0	1394	3	0	0	0	0	2	12	0
2	2	0	1143	0	1	0	0	4	5	0
3	0	0	9	1225	0	11	2	2	6	1
4	0	0	1	0	1212	0	3	1	0	14
5	1	0	1	17	0	959	2	0	5	0
6	15	1	0	0	0	3	1158	0	19	0
7	0	3	8	4	3	0	0	1239	3	12
8	0	3	7	0	1	0	7	1	1094	2
9	0	4	1	3	4	0	0	7	6	1162

   (a) Fold 1:
   - Overall accuracy: **0.9799**
   - Precision: **0.9796**
   - Recall: **0.9798**
   - F1-score: **0.9796**
   - Mean of validation error: **0.0881**
   - Standard deviation of validation error: **0.6887**

   (b) Fold 2:
   - Overall accuracy: **0.9693**
   - Precision: **0.9693**
   - Recall: **0.9691**
   - F1-score: **0.9692**
   - Mean of validation error: **0.1207**
   - Standard deviation of validation error: **0.7781**

   (c) Fold 3:
   - Overall accuracy: **0.9675**
   - Precision: **0.9676**
   - Recall: **0.9669**
   - F1-score: **0.9671**
   - Mean of validation error: **0.1303**
   - Standard deviation of validation error: **0.8441**

   (d) Fold 4:
   - Overall accuracy: **0.9673**
   - Precision: **0.9673**
   - Recall: **0.9672**
   - F1-score: **0.9672**
   - Mean of validation error: **0.1443**
   - Standard deviation of validation error: **0.9547**

   (e) Fold 5:
   - Overall accuracy: **0.9629**
   - Precision: **0.9628**
   - Recall: **0.9625**
   - F1-score: **0.9625**
   - Mean of validation error: **0.1627**
   - Standard deviation of validation error: **0.9857**

7. **Other classifiers**
HOG features from input images were extracted and used for fitting KNN and SVM model. This resulted in a reduction of number of input features from 784 to 324

(a) **KNN**

n˙neighbors →	5	7	9	11	13
Accuracy	0.9573	0.957	0.9566	0.9566	0.955

Best accuracy with KNN classifier is **0.957**.

(b) **SVM**

penalty parameter →	10.0	100	500	1000	2000	5000	10000	15000
Accuracy	0.9587	0.9682	0.9709	0.9723	0.9736	0.9757	0.9772	0.9768

Best accuracy with SVM classifier is **0.9772**.

Summary: Results, Observations and Analysis

- Best accuracy obtained from MLP classifier on MNIST dataset with sigmoid activation was **0.9288**, while that with relu activation was **0.9765**.

- The training and test error convergence plots show faster convergence of relu activated MLP models when compared to sigmoid activated models. And among the sigmoid activated models, the models trained with greater learning rate converged faster.

- However in all the plots, the training error almost converged to 0 in all the instances of cross-validation and training after 8000 iterations, while the test error got stagnated at some values around 0.1. Using data augmentation, by rotating and adding noise to training data, proved to alleviate over-fitting to some extent.

- Results from top 3 predictions 1 of sample test images made by our models give some interesting insights into working of the models. We can see that there is a lot of overlap in the top 3 predictions made by different models.

- Models trained on hand-crafted features extracted from images show comparable results. The best accuracy obtained in the experiments was **0.9737** which is very close to the predictions made by models that were trained with raw images. This shows the effectiveness of extracting features and training the model using these features of input. Such models train faster, are less heavy in terms of number of parameters used and also provide competitive results.

- The best accuracy from the experiments performed using the KNN classifier was **0.957** while that of SVM classifier was **0.9772**.

- On comparing the performance of sigmoid activated MLP and other classifiers, we can see that SVM and KNN had easily outperformed sigmoid MLP.

- So we have the Fully Connected Neural Network performing very closely as SVM classifier on MNIST dataset.