Artificial Intelligence - Assignment 1 : Classification of MNIST digits

Dataset Description :

Number of training samples: 60,000
Number of testing samples: 10,000

Preprocessing:

The 28 *28 samples are reshaped into 1 dimensional space ,i.e 784 values.
The dataset is normalized in order to restrict the values from 0-1

Description of the base model:

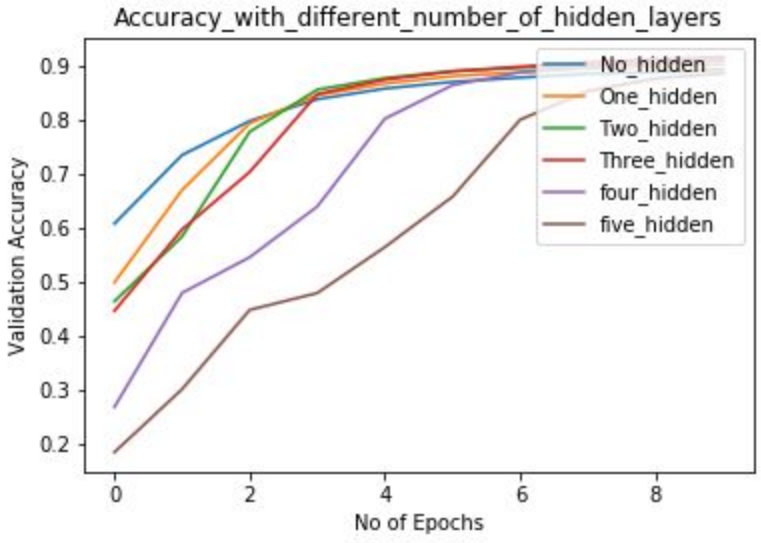
A simple three layer neural network is used – Input layer (784 neurons) , Hidden layer (512) , Output layer (10).

Experimentation:

1) Number of hidden layers were varied:

i)No hidden layers
ii)One hidden layer
iii)Two hidden layers
iv)Three hidden layers
v)Four hidden layers
vi)Five hidden layers

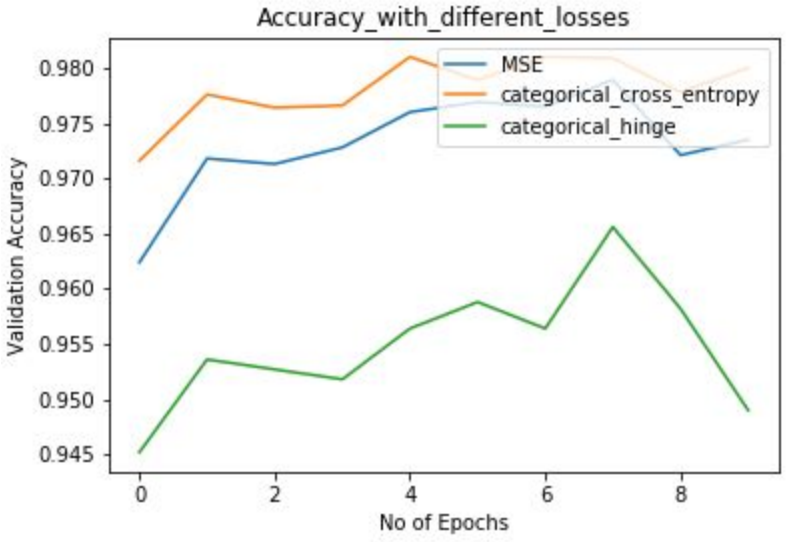
The following is the graph obtained :



Since the dataset is a simple one, the model achieves good accuracy even with one hidden layer as one set of mappings are enough. Overfitting was observed in case of 4 and 5 hidden layers, as they reached 85% accuracy by 5th and 8th epochs respectively, whereas the models with less than or equal to three hidden layers reached the same accuracy within 3 epochs. The model converged faster with one and two hidden layers.

2) Varied Loss functions

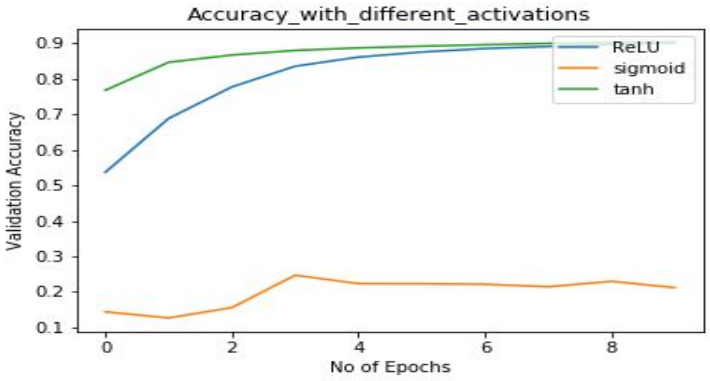
Mean Squared Error, categorical cross entropy and categorical hinge were used.



Categorical cross entropy showed the best accuracy.
Since the problem statement is a classification problem cross-entropy is best preferred as in classification you work with very particular set of possible output values and in such a case Mean Square Error is badly defined.

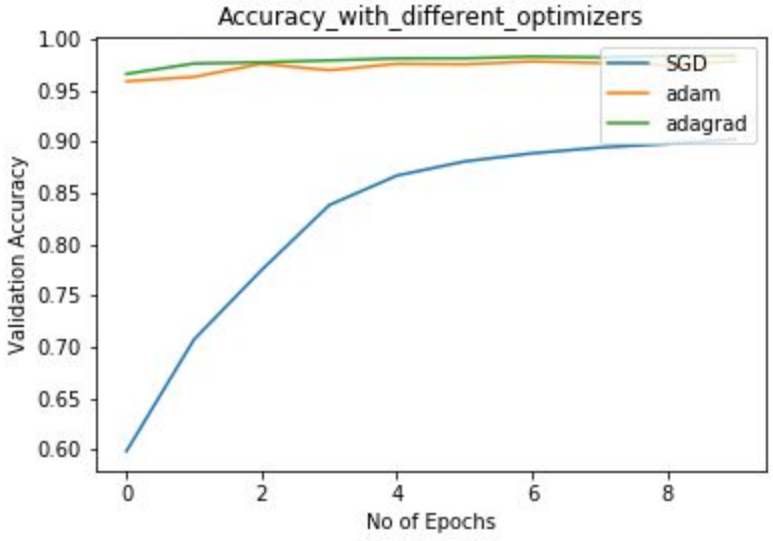
3) Varied Activations

Sigmoid, relu and tanh optimizers were compared.



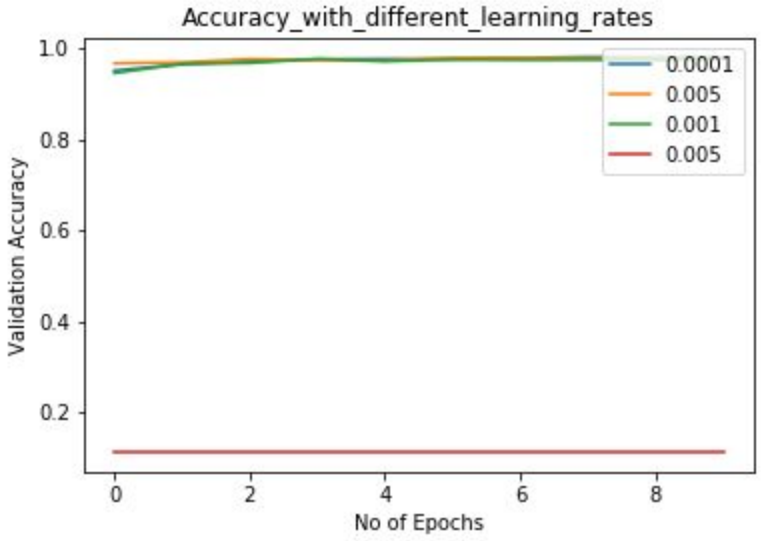
Sigmoid showed very bad results as the neurons were dying cause of very less change in loss which lead to much lesser gradient. And relu, tanh made the model converge around same time, but relu is way better when we compare the training times since it has less calculation load and introduces non-linearity.

4) Varied Optimizers



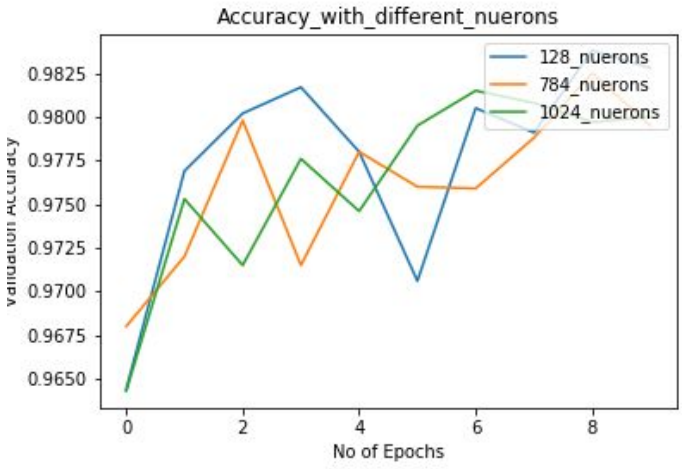
Adam, adagrad, Stochastic Gradient Descent (SGD) were compared with their default parameter settings as in the keras framework and adam showed best results.

5) Varied Learning Rates



A very high learning rate showed poor result, this is because the model converges faster with badly learnt weights. Learning rate of 0.001 showed the best result in terms of accuracy ,training time and learning in each epoch. Further reducing the learning rates to 0.0001 , the accuracy achieved was similar to lr-0.001 but the model converged very slowly.

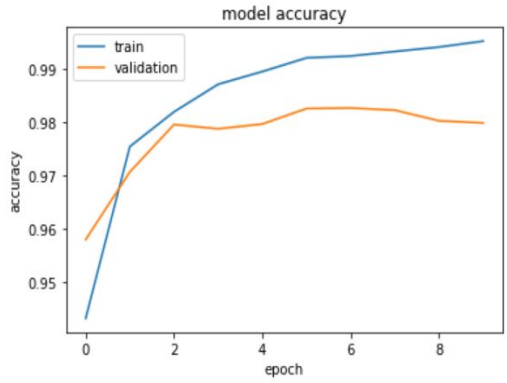
6) Varied the no.of neurons in the hidden layers (128, 784, 1024)

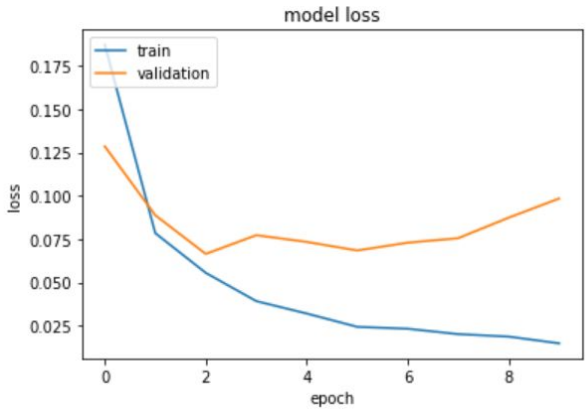


Varying the number of neurons in the hidden layers doesn't show much difference in terms of accuracy, after the 10th epoch all the above cases show accuracy above 98%.. Infact with just 128 neurons the model learns a good mapping as this is a simple dataset. (Increasing the number of neurons in hidden layer than the input layer shows lesser accuracy)

Best Model:

Final model - one input(784, relu) one hidden(128, relu) one output(10, softmax)
Optimizer adam, loss function categorical cross entropy, batch_size 32





Accuracy and loss : The training accuracy was observed to be above 99 percent and the validation accuracy was about 98 percent.
The above graphs represent the accuracy and loss respectively.