Assignment-3 Neural Networks
ELL 409

Jaskeerat Singh Saluja (2019MT60752)

November 11, 2021

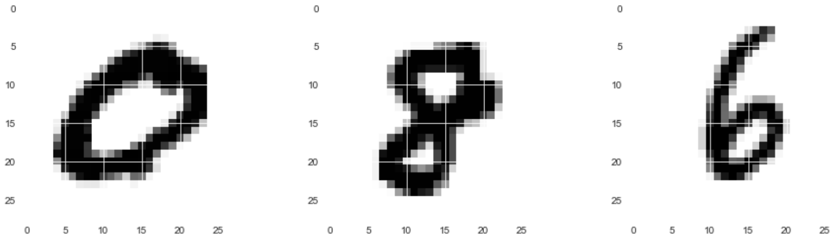
Table of contents

1. Theory and python implemetation
2. Part-1
    - Part-1A Training neural net on given images
    - Part-1B Comparison with the PCA features
3. Part-2 Advanced Neural networks

Theory and python implemetation

Visualizing the given data

On transforming the feature vector from $(784, 1)$ to $(28, 28)$ size and plotting we have the following images provided.



Using Sigmoid activation function

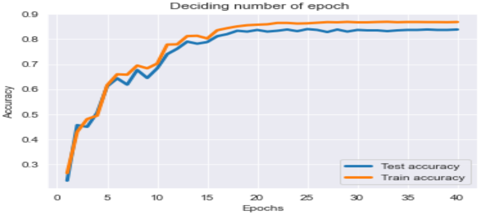
Finding best number of epochs

1. Neural nets gets overfit on high epoch and are underfit on low number of epochs
2. Epoch =25 is good value to start the tuninng the neural net.

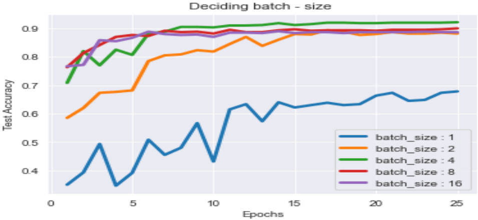
Tuning the mini-batch -size

1. On right , the plot shows variation in test accuracy vs batch-size
2. *batch_size* = 4 and *epoch* = 20 works well

Batch size =4 , epochs =20



layers [784,30,10] ,eta= 3 , Sigmoid , cross-entropy-loss, batch_size=1

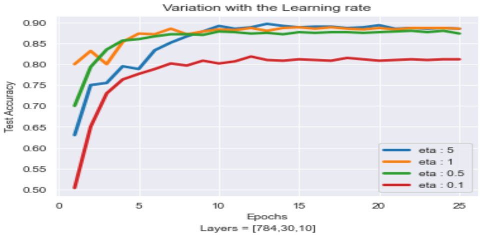


Layers = [784,30,10]

Sigmoid Activation continued ....

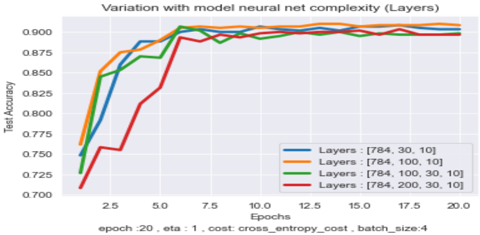
Finding best learning rate η

1. SGD is sesitive to learning rate.
2. The SGD implementation uses **decaying learning rate strategy**



Tuning number and size of layers

1. More number of hidden layers represents higher abstraction of the neural net.
2. Choosing least complex neural net with good test accuracy is the goal.

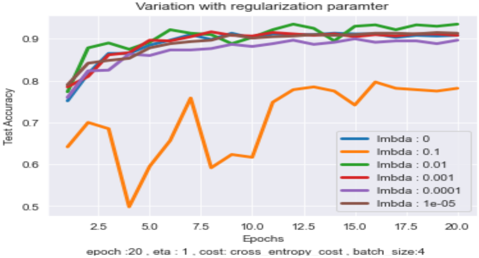


Layers $[784, 30, 10]$ **is least complex and performs as good as** $[784, 100, 10]$ and learning rate $\eta = 1$

Sigmoid Activation continued ....

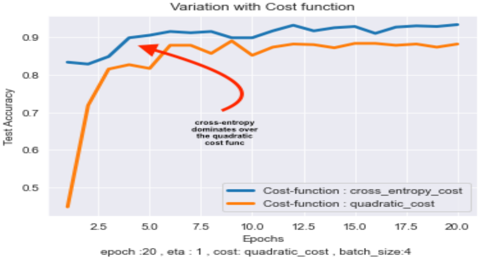
L-2 regularization λ

1. Regularization (L-2) forces w to take lower values of lmbda .Thus prevents model from over-fitting
2. λ in range $[1e-2, 1e-3]$ is better hyperparameter space to work with.



Choosing the cost function

1. Cross entropy learn much faster than the quadratic loss function , since qudratic cost function have a phenomenon called **slowing down**
2. Figure on right display the **slowing down** nicely.



Sigmoid Activation continued ....

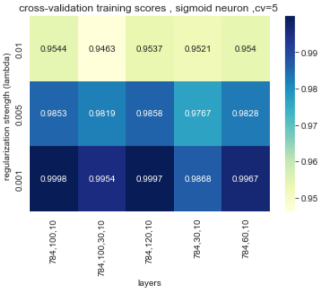
So far now we have narrowed down our hyperparameter space to small subspace where we can perform k-fold cross validation and further tune the best parameters.

epoch	15
mini batch size	4
learning rate	1
layers	$[784, 30, 10]$, $[784, 100, 30]$
lambda	$[1e-2, 5e-3, 1e-3]$
Cost function	cross entropy loss

We have 6 possible good configuration for our sigmoid activation function . We now perform k-fold cross validation to get the best hyperparameter . (CV =5)

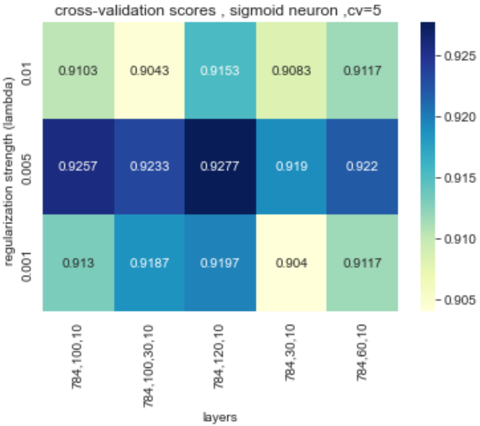
Cross-Validation (cv=5)

1. On rights we have variation of lambda vs neural net layers
2. Best Test accuracy 92.77 percent and corresponsing Train accuracy 98.58



CV training scores

Best hyperparameters : Layer sizes: [784,120,10] and $\lambda = 5e-3$



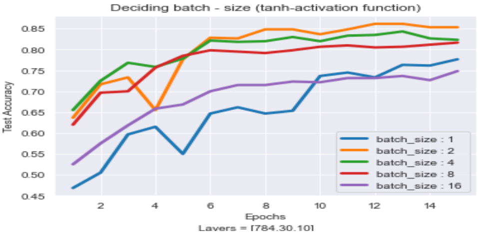
CV Test scores

Using tanh activation function

- **epoch=15** ,Cost function : **cross entropy cost**

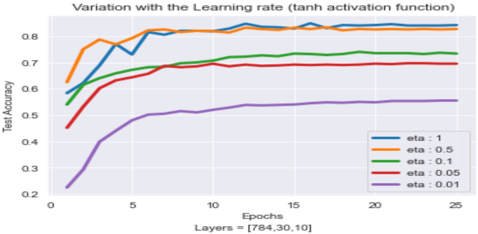
Tuning the mini-batch -size

1. On right , the plot shows variation in test accuracy vs batch-size
2. *batch_size* = 4 works well



Learning rate

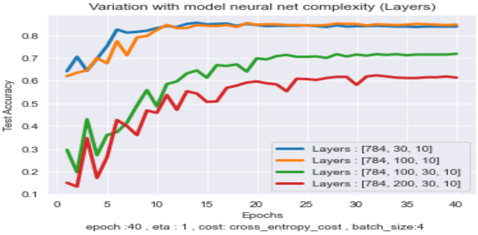
1. On right , the plot shows variation in test accuracy vs learning rate
2. $\eta = 0.5$



tanh function continued ....

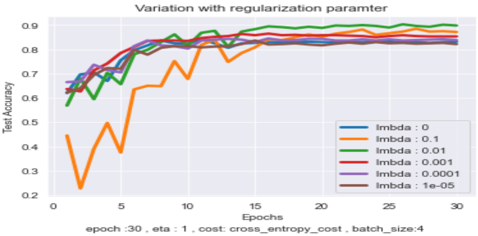
Tuning the layers

1. On right , the plot shows variation in test accuracy vs different layers in neural nets
2. layer =[784,30,10] works quite well , even with low complexity.
3. **Observation :** 2 hidden layer networks took more epoch before settling



Tuning the regularization strength

1. On right , the plot shows variation in test accuracy vs different λ
2. $\lambda = [1e-1, 1e-2, 1e-3]$ is good range where model performs good



tanh Activation continued ....

So far now we have narrowed down our hyperparameter space to small subspace where we can perform k-fold cross validation and further tune the best parameters.

epoch	20
mini batch size	4
learning rate	0.5
layers	$[784, 30, 10]$, $[784, 100, 30]$
lambda	$[1e-1, 5e-2, 1e-2, 5e-3]$
Cost function	cross entropy loss

We have 8 possible good configuration for our sigmoid activation function . We now perform k-fold cross validation to get the best hyperparameter . (CV =5) . Will take 12-15 minutes to train

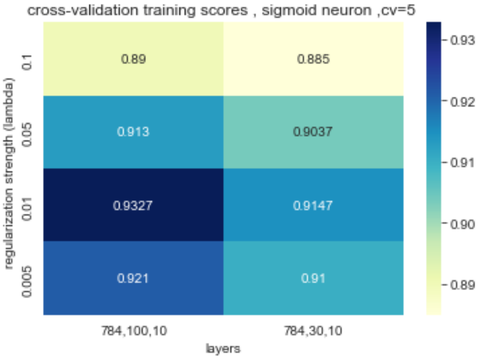
tanh Activation continued ....

Cross-Validation (cv=5)

1. On rights we have variation of lambda vs neural net layers
2. Best Test accuracy **93.27** percent and corresponsing Train accuracy **99.95**

Observations

- **tanh activation(93.27) performs better** than sigmoid neuron (92.77) by a significant 0.5 improvement.
- $\lambda = 0.01$ and layer network =[784,100,10] works best



CV Test scores

Comparison with Tenserflow library

1. Using 90:10 split of data , then the best validation score using the **tanh** activation function achieved is 92.89 which is quite close to our 93.27 cross validation scores.
2. Using 90:10 split of data , then the best validation score using the **sigmoid** activation function achieved is 92.09 which is quite close to our 92.77 cross validation scores.

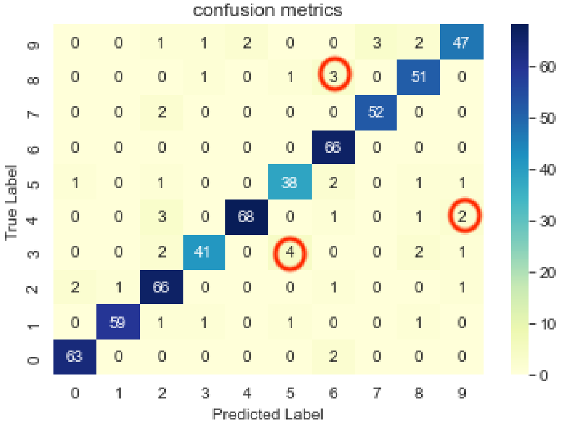
Analysing errors made by neural nets

Observations :

- On right we have confusion metric ,on **test data** which is never seen by the neural net.
- 4 is confused with 9
- 3 is predicted wrongly as 5
- 8 with 6

Reasoning

- Digits with similar geometry are the ones wrongly predicted by the neural net
- Digits which donot share geometry like (1,6) are never wrongly predicted.

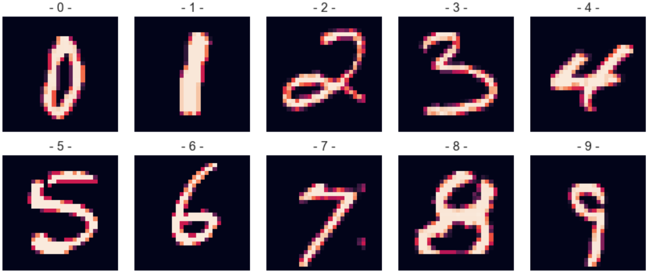


epoch =10 , tanh activation neuron

Visualizing the hidden layers

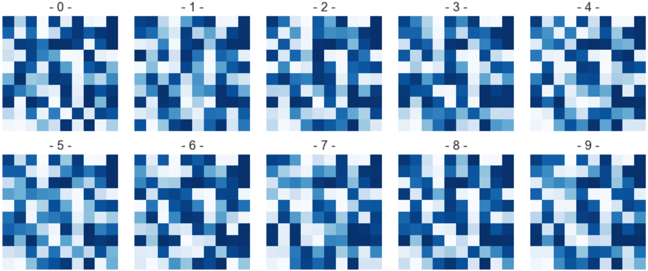
1. Neural net model with layers [784,100,10] trained .
2. Below we visualize the ouput by each layer of the neural net.

Outputs First Layer (Input Layer)



Visualizing the hidden layers

Outputs Second Layer (Hidden Layer)



- These patterns look sufficiently random - except for the digits (4 , 9) , (3 , 5) , which look remarkably similar
- Above **each pixel is mean activation output** from a neuron in hidden layer.
- **Dark blue** mean that specific neuron is active while **white means** that neuron is inactive.

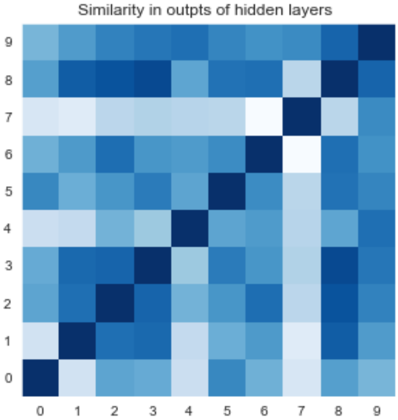
- **NOTE:** The neuron in the bottom left corner is only **active** when input is 1 , while it is inactive for inputs other than 1. **Possible reason:** 1 is only geometric shape which is just a straight line i.e no curve /bends . All other digits have curved parts . **This is a possible reason**.

Visualizing hidden layers

- Since the outputs are (100,1) in the hidden layer, representing the activations of the neurons.
- Thus to check for similarity is outputs of digit i with digit j . One solution is to take dot product of the mean squared outputs vectors for ith and jth digit.

Conclusions

- Despite variations in the shapes of hand-written digits, the same groups of neurons is involved in the identification of the same digits.
- Similarities in the shapes of digits translate into similarities in the groups of neurons that are involved in their identification in the first hidden layer, but not so much in the second hidden layer



PCA