Classification of the MNIST dataset using Back-propagation Neural Networks

Gitika Meher
Department of Electrical and Computer Engineering
University of California San Diego
9500 Gilman Dr, La Jolla, CA 92093
gkarumur@eng.ucsd.edu

Nasha Meoli
Department of Electrical and Computer Engineering
University of California San Diego
9500 Gilman Dr, La Jolla, CA 92093
nmeoli@eng.ucsd.edu

Abstract

This paper presents an implementation using Back-propagation Neural Network to achieve the classification of the MNIST handwritten digit database. Here, we use plot of loss during training and classification accuracy to determine the performance of the neural network. An accuracy of 92% was obtained as an average through multiple runs using one hidden layer.

1 Derivation of the weight update rule and the deltas

The first part of the assignment involving the derivation of gradients w.r.to different parameters, the weight update rule and the vectorized weight update rule are provided separately.

2 Implementation of Backpropagation for Classification on the MNIST database

2.1 a) Load MNIST data

The MNIST dataset consists of 70,000 examples with labels of hand-written examples of digits. The digits have been size-normalized and centered in a fixed-size image. This dataset has been shuffled and divided into training set, validation set and test set. Each image is of size 28x28 pixels.

2.2 b) Computation of the gradient using the numerical approximation

$\frac{\partial E}{\partial w} \approx \frac{E(w+\epsilon)-E(w-\epsilon)}{2\epsilon}$
$\epsilon = 10^{-2}$
The table below shows the gradient obtained by running one training example for one epoch and the gradient calculated using the above mathematical approximation.
The first two rows($w1_0, w2_0$) of the table show the values for two weights from the input to the hidden layers. The third row(b_0) shows the values for a bias from the input to the hidden layers.
The fourth and the fifth rows($w1_1, w2_1$) show the values for two weights from the hidden to the output layers. The last row shows the values for a bias(b_1) from the hidden to the output layers. All

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

the values are calculated changing only one parameter and the keeping the rest as constants. It can be seen from the table that the values differ after 10^{-4} which isϵ^2

Par	Gradient	Gradient computed	$E(w+\epsilon)$	E(w-ϵ)
$w1_0$	-0.000105007	-0.00015678	10.812889727	10.8128928635
$w2_0$	-0.0031549	-0.00310719	10.81297188516	10.81303402901
b_0	0.100403559	0.100418859	10.81456927535	10.81256089817
$w1_1$	0.9999786	0.999986457	10.81173598415	10.79174025502
$w2_1$	-0.984545526	-0.984577497	10.8038910775	10.8235865770
b_1	0.999978646	0.99994598	10.8317349037	10.811735984179

Table 1: Table showing the difference between the gradient obtained and the gradient by the above numerical approximation for different parameters

2.3 c) Training,Validation and Test accuracy using one hidden layer

The network was set up with a tanh activation, a learning rate of 0.0009 and momentum of 0.9. The test accuracy achieved as 91.84%

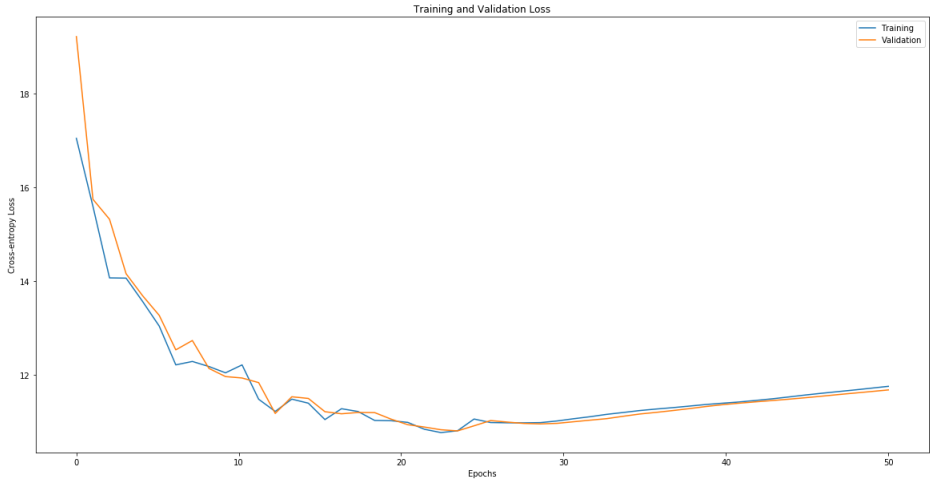

Figure 1 : Figure showing Training, Validation loss with a learning rate of 0.0009, with a momentum of 0.9 and activation function as tanh

The cross-entropy loss decreased (for both the training and validation sets) over the training epochs reaching a minimum in 23 epochs.

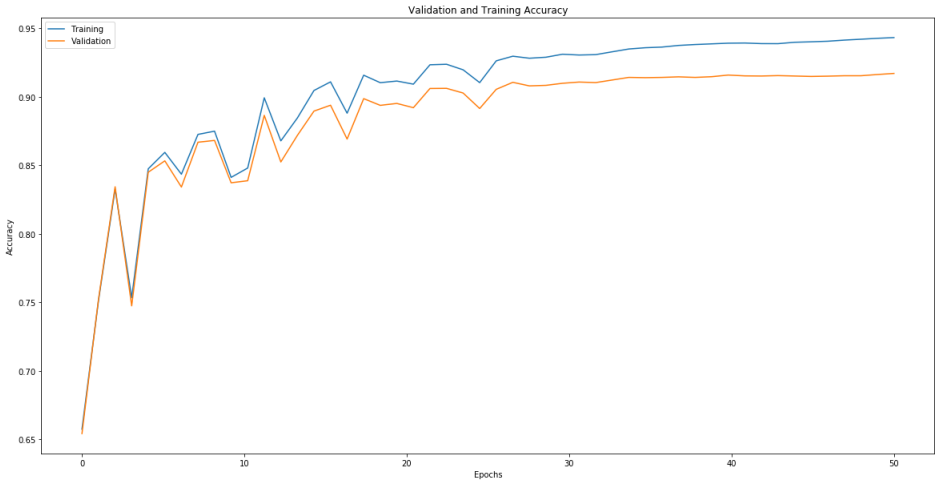

Figure 2 : Figure showing Training, Validation accuracy with a learning rate of 0.0009, with a momentum of 0.9 and activation function as tanh

The accuracy of classification increased over the training epochs for both the training and validation sets, with a higher accuracy on the training set than the validation set.

2.4 d) Training,Validation and Test accuracy using Regularization

Figures 3, 4 show the effect of regularization on the network. L2 penalty was chosen as 0.0001 and a test accuracy of 92.56% was obtained with the addition of same.

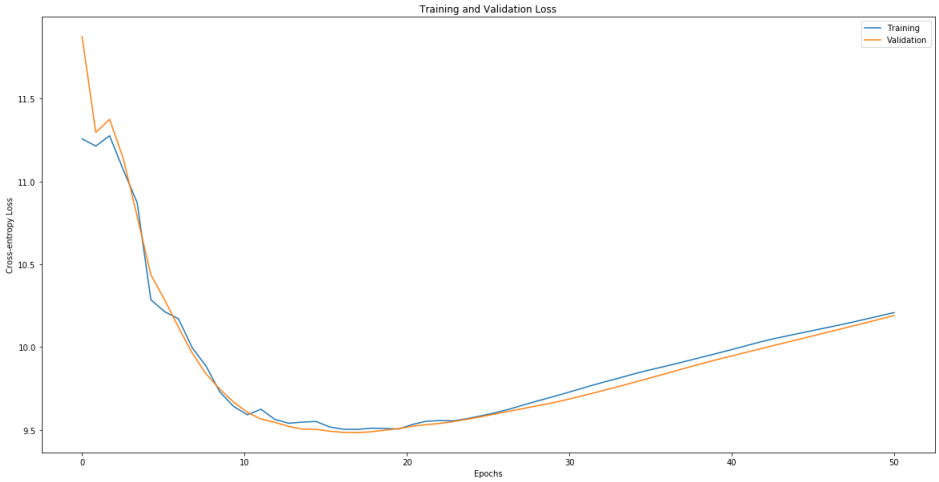

Figure 3: Figure showing Training, Validation loss with L2 penalty=0.0001 and activation function as tanh

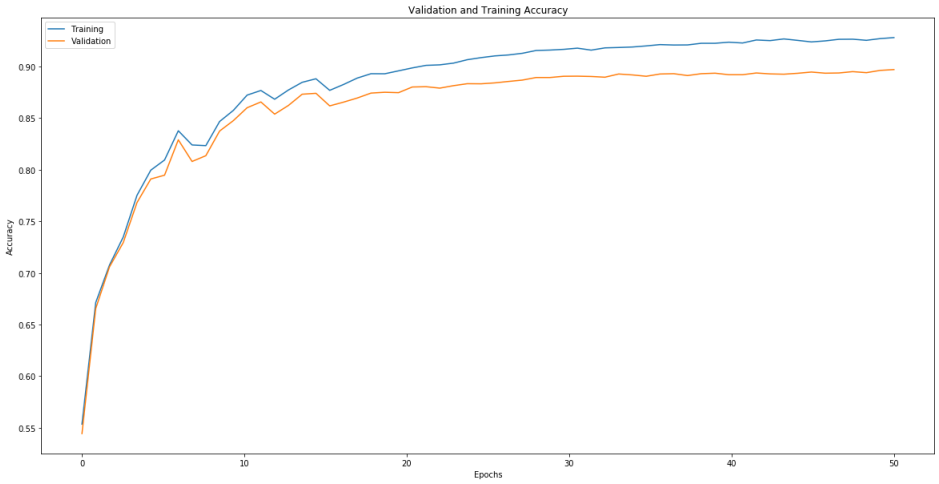

Figure 4: Figure showing Training, Validation accuracy with L2 penalty=0.0001 and activation function as tanh

Figures 5, 6 show the effect of regularization with a L2 penalty of 0.001 on the network. L2 penalty was chosen as 0.001 and a test accuracy of 92.67% was obtained with the addition of same.

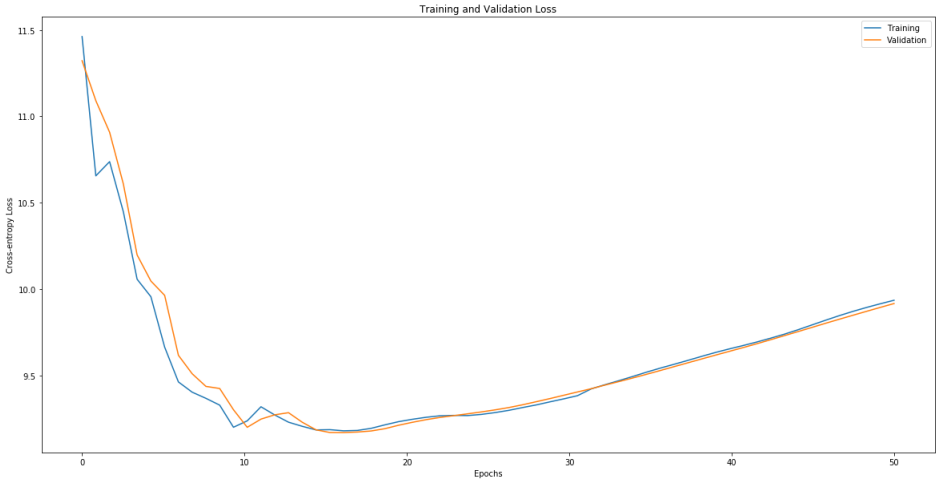

Figure 5: Figure showing Training, Validation loss with L2 penalty=0.001 and activation function as tanh

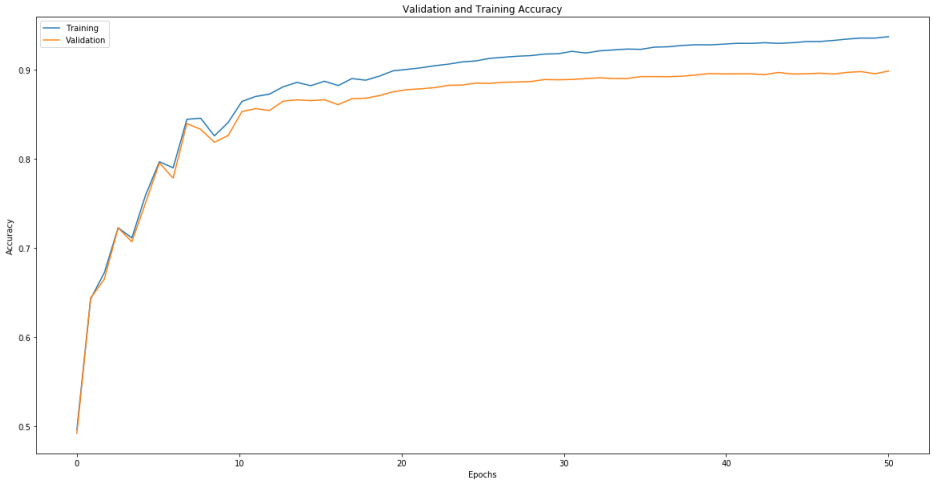

Figure 6: Figure showing Training, Validation accuracy with L2 penalty=0.001 and activation

function as tanh

Addition of L2 penalty increases the test accuracy as it avoids overfitting on the training data and generalizes well.

2.5 e)Training,Validation and Test accuracy with different Activation functions

2.5.1 ReLU

The ReLU activation function was used on the hidden layer instead of the tanh function from part c). The initial weights were scaled by a factor of $\frac{1}{fan_{in}+fan_{out}}$ and the learning rate lowered by a similar factor. The results are shown in Figure 5 and 6. An test accuracy of 96.38% with early stopping on the 6th epoch was obtained

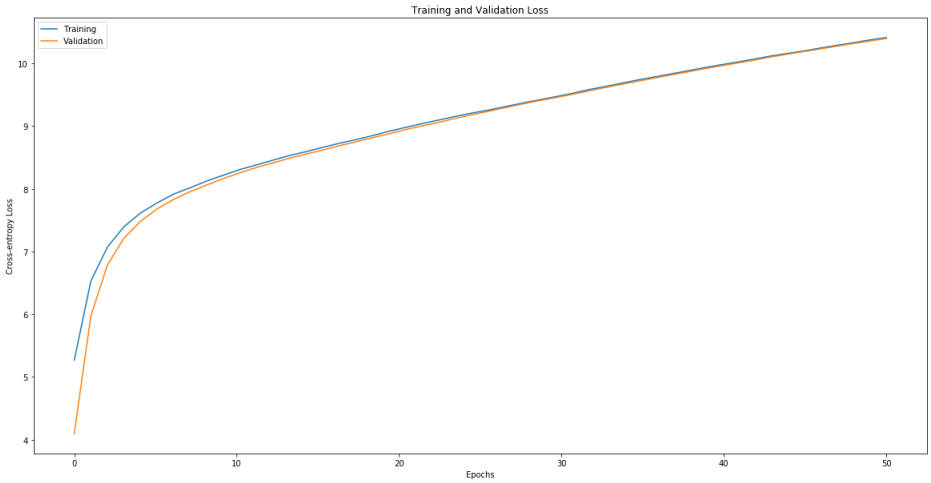

Figure 7: Figure showing Training and Validation Loss for ReLU

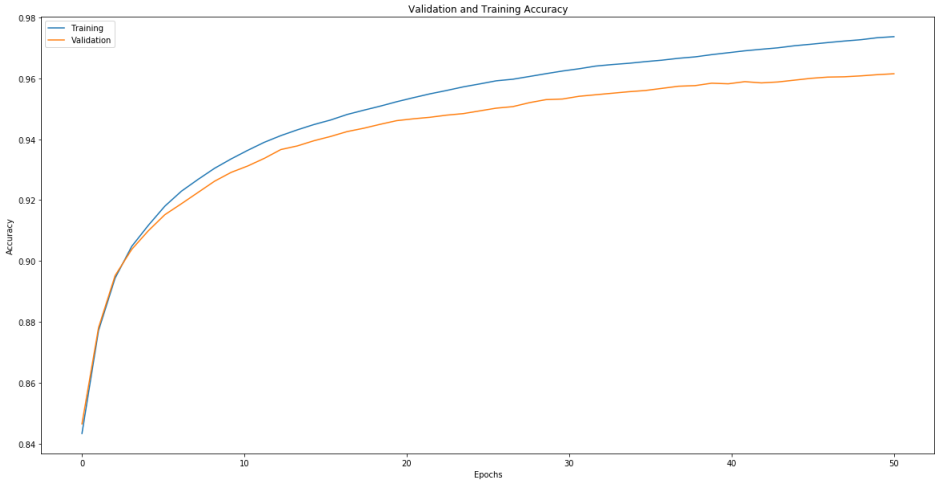

Figure 8: Figure showing Training and Validation Accuracy for ReLU

It can be seen that for single layer network with only 100 units, the ReLU function attain lower cross-validation loss across training epochs than the tanh activation function. The loss on both the training and validation sets increased over the training epochs.

The accuracy of the model over the 50 epochs increased, with a higher accuracy on the training data (the seen data) than on the validation data.The accuracy obtained from the ReLU was the highest among the activation functions used.

2.5.2 Sigmoid

A sigmoid activation function was used instead of a tanh function in the part c) set up. A test accuracy of 89.1% was achieved, this was the lowest accuracy achieved for any activation function. An early stop was implemented.

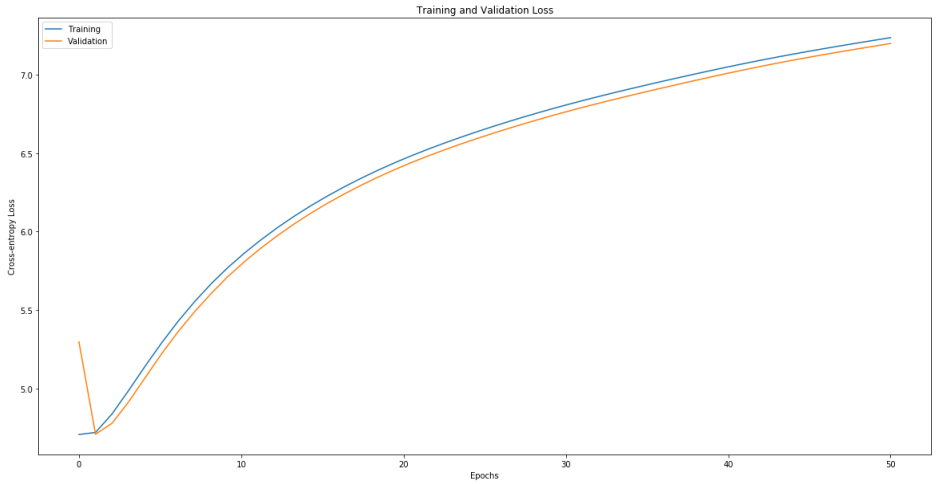

Figure 9: Figure showing Training and Validation Loss for Sigmoid

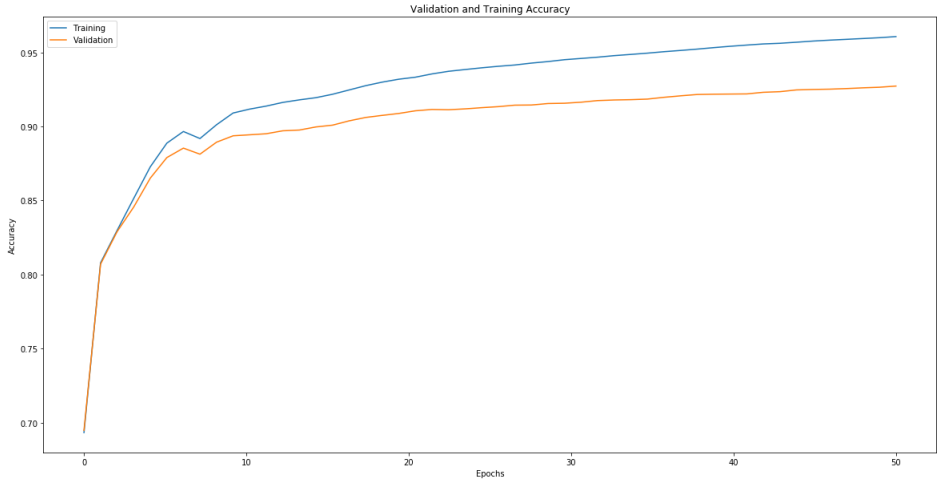

Figure 10: Figure showing Training and Validation Accuracy for Sigmoid

The sigmoid loss function for both the training and validation sets are similar and dip slightly before the 5th epoch and then rise steadily. The magnitude and range of the loss for the sigmoid function across the training epochs was the lowest among the three different activation functions.

The accuracy of the model over the 50 epochs increased, with a higher accuracy on the training data (the seen data) than on the validation data.

2.6 f) Training,Validation and Test accuracy with changing the number of hidden units and the number of hidden layers

2.6.1 Changing the number of hidden units

Using the setup from the c model, the number of hidden units was double (to 200) and halved (to 50) and the curves in Figure 11 to 14 were obtained.
For the case where the units were halved, the test accuracy obtained was 90.59%. This was the lowest accuracy in comparison to the two other hidden unit configurations.The loss and accuracy over

training epochs are shown in figures 11 and 12.

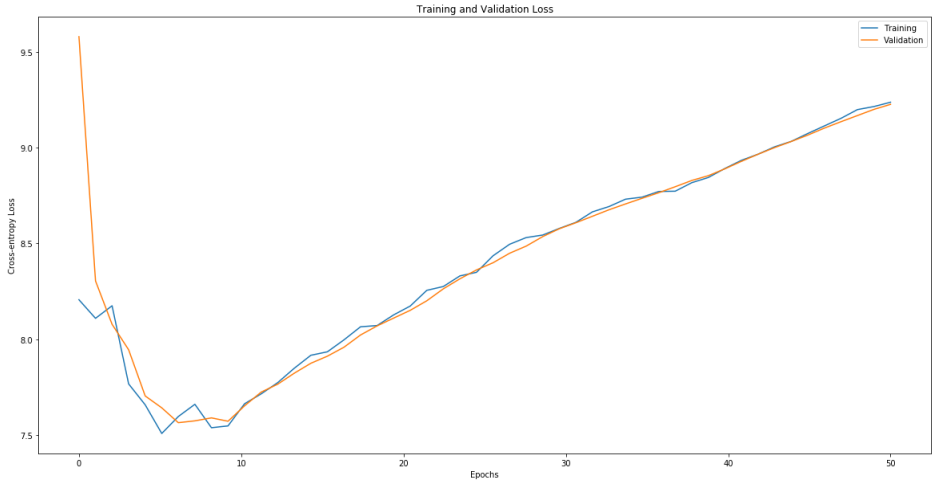

Figure 11: Figure showing Training and Validation Loss for 50 hidden units

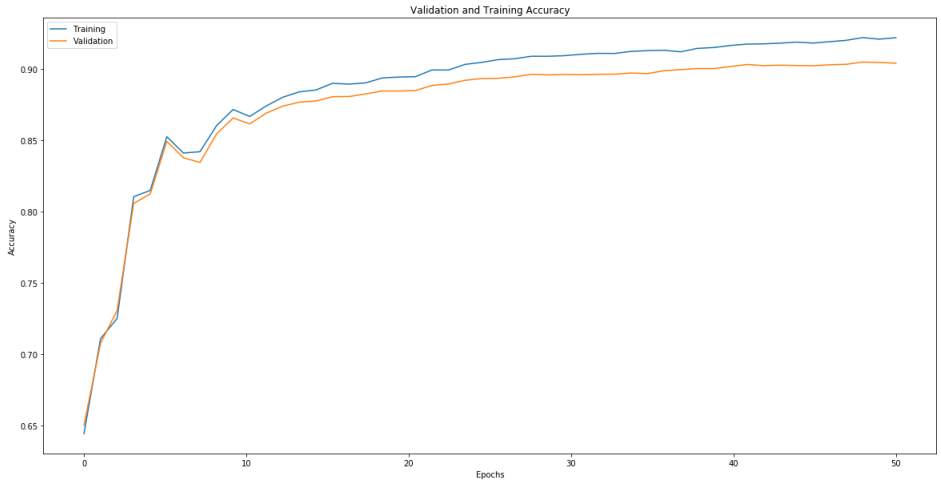

Figure 12: Figure showing Training and Validation Accuracy for 50 hidden units

The minimum cross-entropy loss was obtained earlier (in fewer epochs) than when there were 100 units. This means the weights were learnt faster.
The accuracy for both sets increased over the training epochs and the training accuracy was higher than the validation accuracy across the epochs.
For the case where the units were doubled, the test accuracy obtained was 93.53%. This was the highest accuracy in comparison to the two other hidden unit configurations.The loss and accuracy over training epochs are shown in figures 13 and 14.

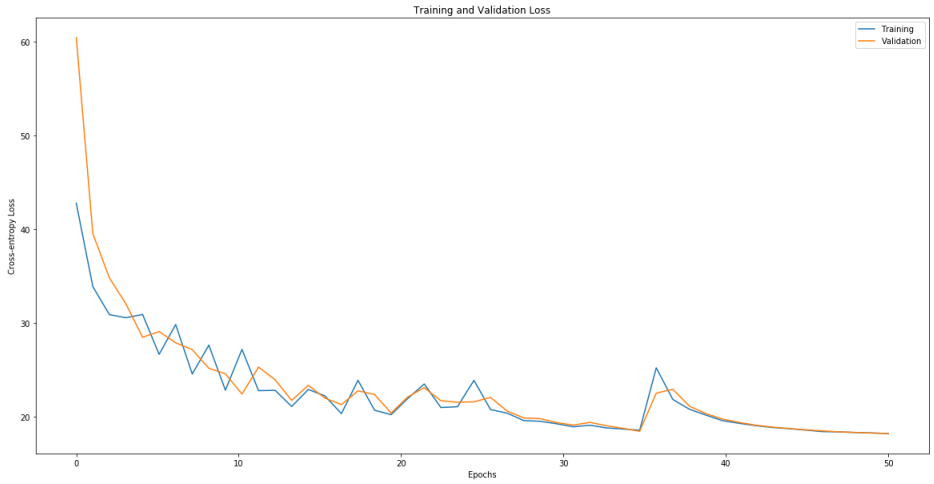

Figure 13: Figure showing Training and Validation Loss for 200 hidden units

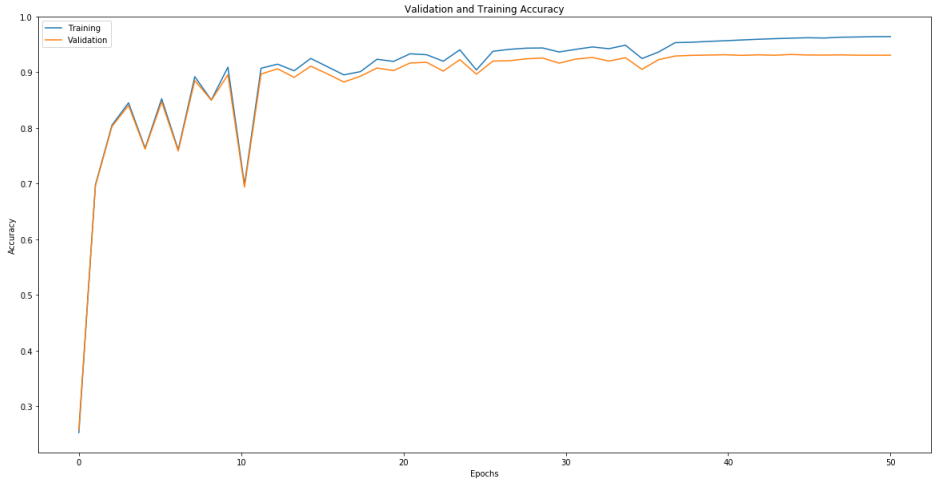

Figure 14: Figure showing Training and Validation Accuracy for 200 hidden units
The minimum cross-entropy loss was obtained later (in more epochs) than when there were 100 units. This means the weights were learnt slower.
The accuracy for both sets increased over the training epochs and the training accuracy was higher than the validation accuracy across the epochs.

2.6.2 Two hidden layers of 50 units each

Using the setup from part c) the number of hidden layers was increased to two layers of 50 units each to keep the number of parameters to be learnt similar. A test accuracy of 91.28% was obtained which is slightly lower than in the single layer configuration.

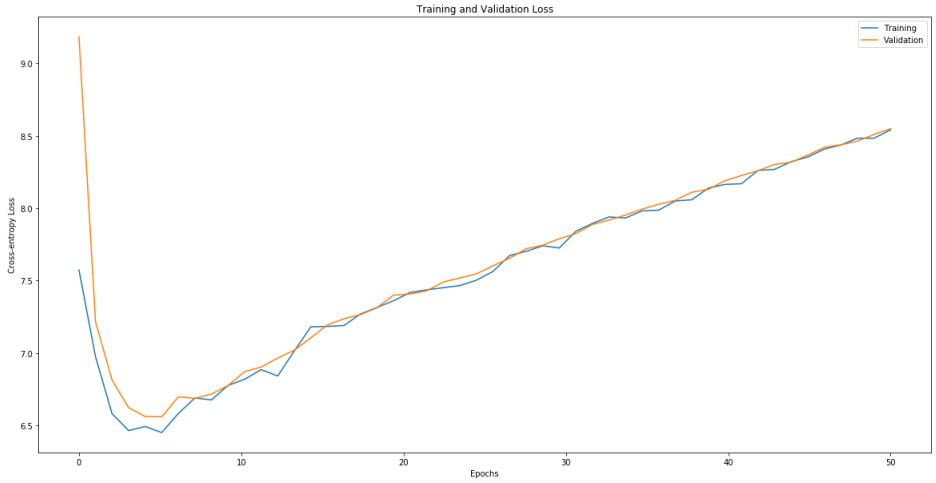

Figure 15: Figure showing Training and Validation Loss for Two Hidden Layers

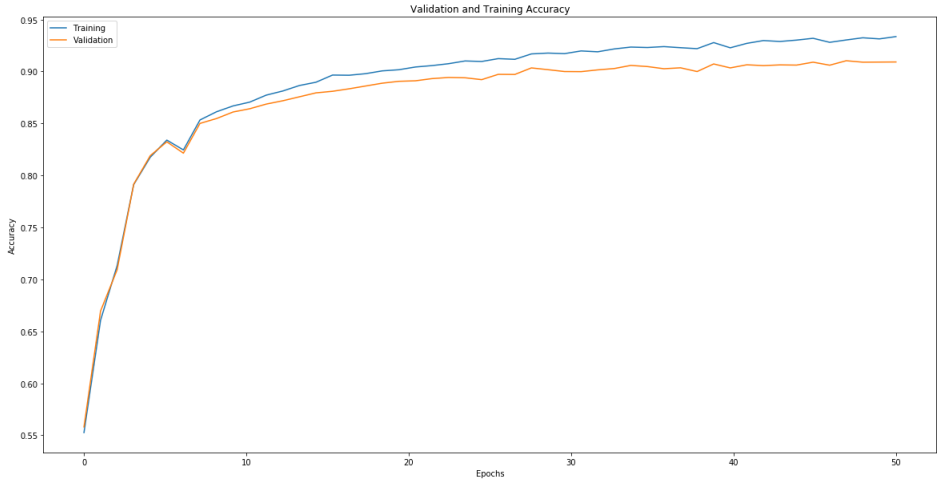

Figure 16: Figure showing Training and Validation Accuracy for Two Hidden Layers

The cross-entropy loss was lower over all training epochs. It reached a minimum within the first 10 epochs after which it steadily increased. The accuracy on both the validation and training sets increased over the training epochs, with higher accuracy on the training set.

3 Individual Contribution

Gitika Meher Karumuri - Parts a), b), c) and d)

Nasha Meoli - Parts c), e) and f)