ECE 1504 Statistical Learning

Programming Assignment 1

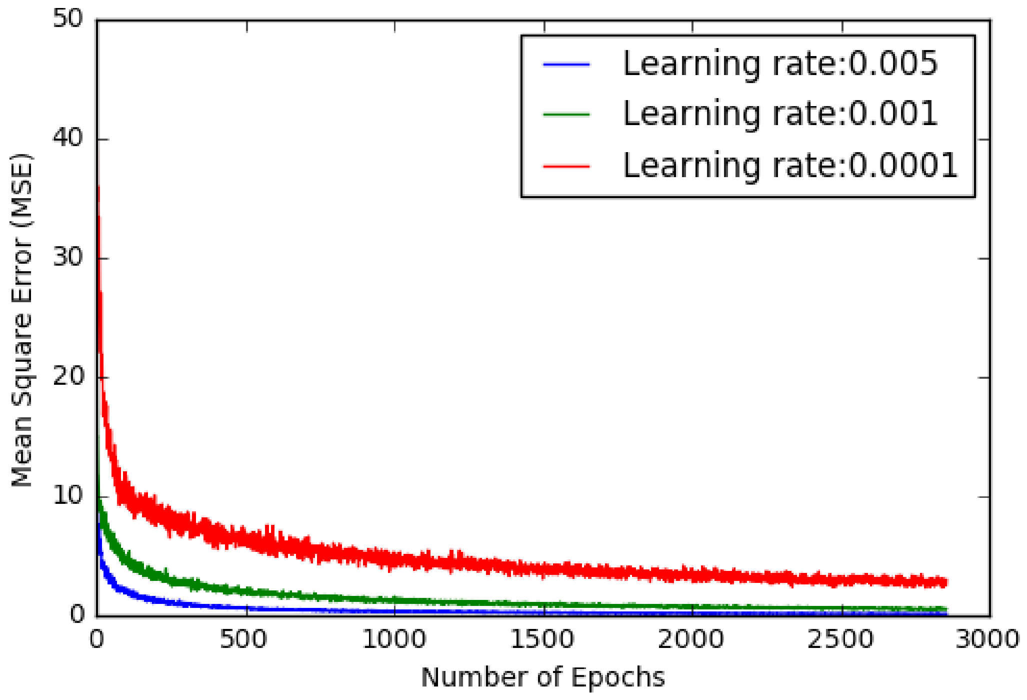
Hitarth Choubisa (1004965479) Alex Labach (1000698248)

October 29, 2018

The contribution from each group member was 50%.

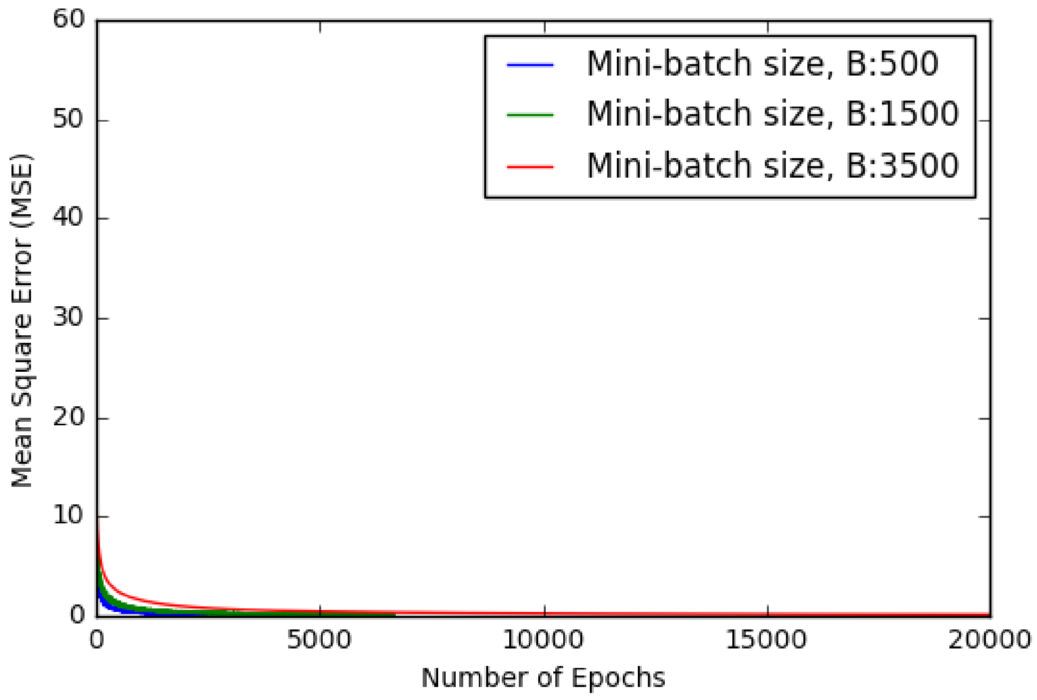
1. Linear Regression

1. We ran stochastic gradient descent algorithm for linear regression with mini batch size B = 500, number of iterations = 20000 and weight decay coefficient $\lambda = 0$ for 3 learning rates $\eta = 0.005, 0.001, 0.0001$. We obtained the following plot between training mean square loss error and number of epochs:



The best learning rate is 0.005 among the three. As we can see, increasing the learning rate enables us to rapidly reduce in-sample error and achieve convergence.

2. We chose the learning rate $\eta = 0.005$. However, we vary the mini batch size B as 500,1500,3500. Number of iterations is still similar. We get the following plots:



The final mean squared error for the three cases is:

Mini-batch Size(B)	MSE
500	0.12
1500	0.13
3500	0.126

Convergence is faster for B=500 and hence, if training time is the constraint, B=500 is the best mini-batch size. The observation is supported by the fact that smaller batches will require lesser computational time and hence, our model will get trained several times more than the one where we train our model with a larger batch size.

3. Validation errors for varying λ is listed below:

λ	Validation accuracy(%)	Test accuracy(%)
0	87%	80%
0.001	88%	83%
0.1	97%	96%
1	95%	95%

Regularization parameter value 0.1 yields the best validation error. Test accuracy for this particular value is: 96%.

Training accuracy is best for $\lambda = 0.1$. As we increase beyond this value, the reduced training accuracy is an indicative that we are underfitting our model. For values of λ smaller than 0.1, the smaller training accuracy are indicative of overfitting; hence our model doesn't perform well over the unseen training & validation data.

λ is intended to reduce overfitting. To measure this effect, we need data outside of what the neural network is training on. So, we use the validation set rather than the training set to tune it.

4. Using the closed form mathematical equation for $\lambda = 0$, we get the training data mean square error as 0.0033. This is much better than MSE we obtained when we used SGD with $\lambda = 0$. Training classification accuracy was 99.3%, which is higher than the best training accuracy for SGD, which was 96.6%. These results make sense since the closed form solution is optimal with respect to MSE.
In this particular case, training takes 0.05 seconds if we use closed form expression whereas SGD takes 0.05 seconds per

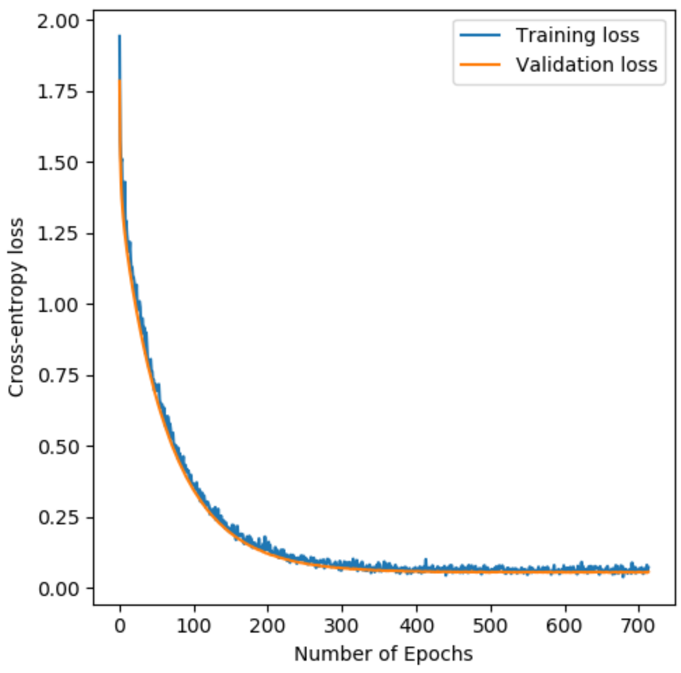
iteration.
However, the memory and computational cost associated with closed form expression is high as we have to store a matrix of dimension $(d+1) \times (d+1)$ and also invert it. So, in the cases where the number of features is a large number, we should resort to SGD for training our models. The analytical solution will also cause overfitting of the training data in most cases whereas SGD is more robust. Another added advantage of SGD implementation over closed form method is that we can easily parallelize our implementation on multiple cores.

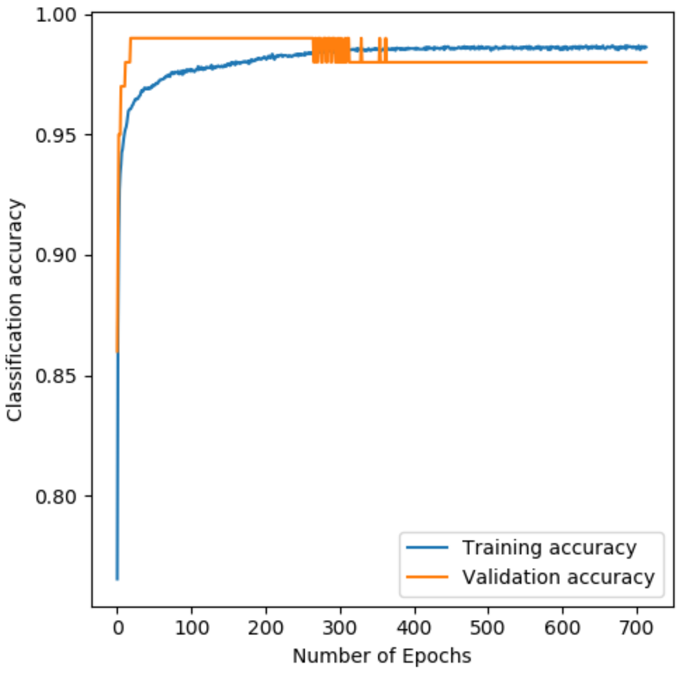
2 Logistic Regression

2.1 Binary cross-entropy loss

1.

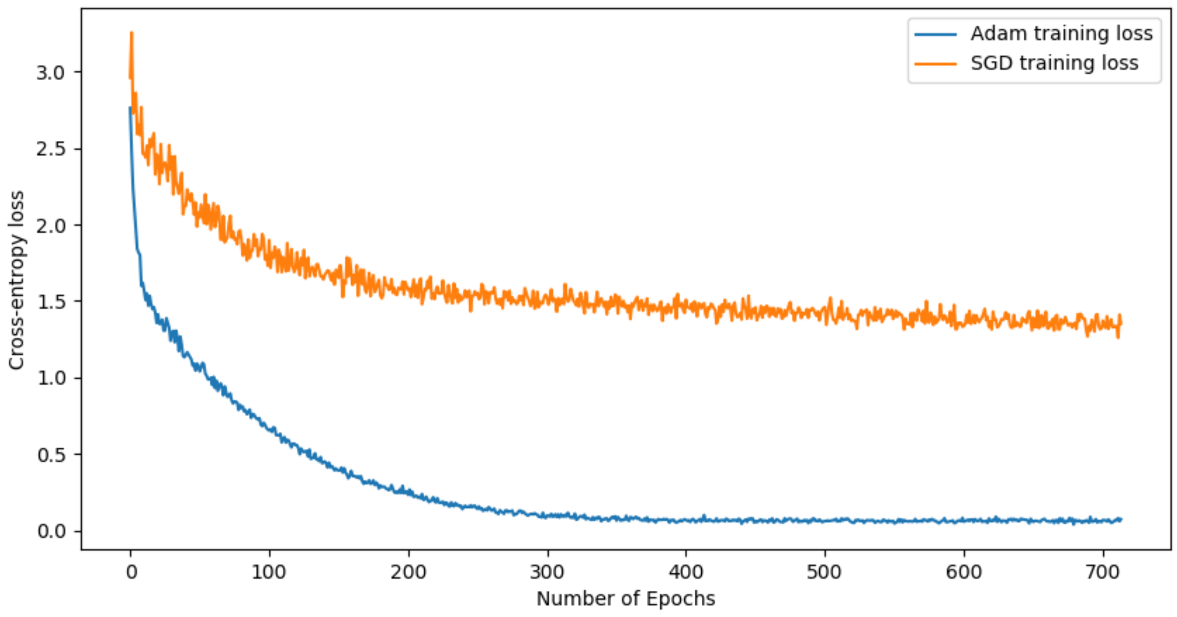
The tuned learning rate used was $\eta = 0.1$. This achieved a test classification accuracy of 97.9%. The following plot shows the curves for training and validation cross-entropy loss and accuracy.





2.

The following plot shows the training loss curves for the Adam optimizer and the SGD optimizer.



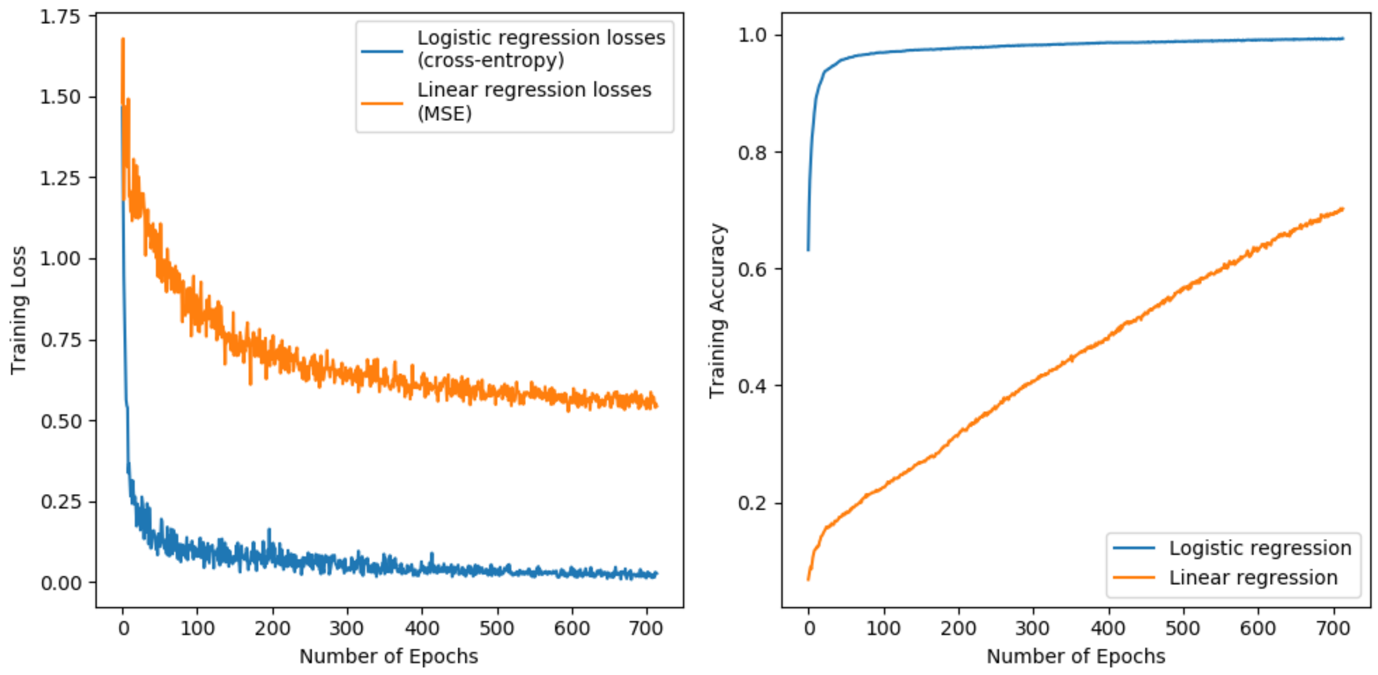
Learning with the Adam optimizer converged much more quickly than with the SGD optimizer. This is because the Adam optimizer takes into account both weight velocity and momentum, encouraging it to follow the gradient in directions that offer continued improvement while avoiding those that do not.

3.

Experiment	Training accuracy (%)	Validation accuracy(%)	Test accuracy(%)
Linear regression (closed form)	99.3	98.0	95.9
Logistic regression	98.8	99.0	97.9

The closed form solution for linear regression had a higher training accuracy than logistic regression, reflecting the fact that it is an optimal solution for the training data. However, logistic regression generalized better for this classification task, achieving higher validation and test accuracies.

The following plot shows the training loss and accuracies for both logistic and linear regression using the same learning rate and the Adam optimizer.

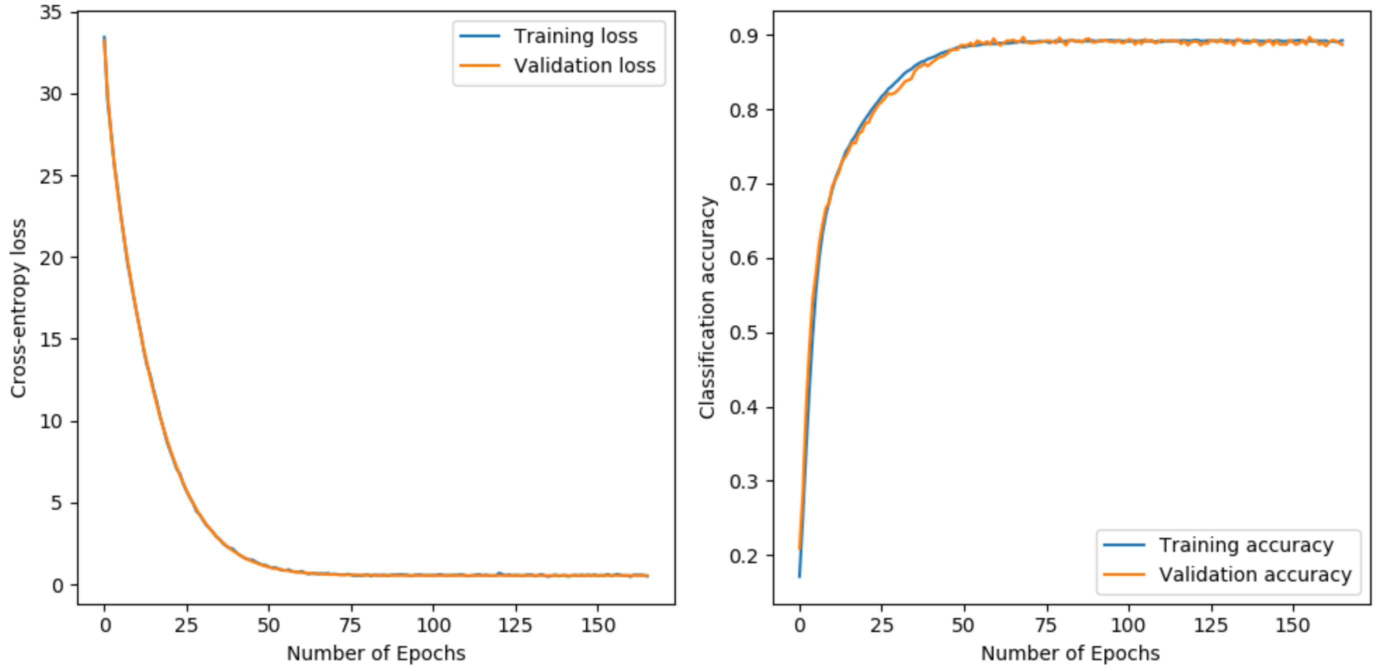


The cross-entropy loss caused the classifier to converge much more quickly, both in terms of loss and in terms of accuracy over the training set. This is because the $-\log$ function in cross-entropy loss grows much more quickly than the square function, causing it to penalize mistakes more harshly.

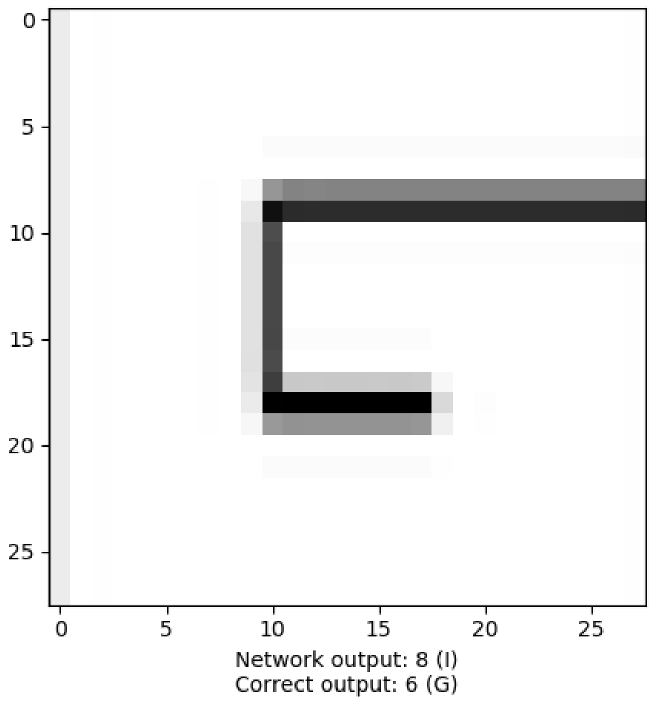
2.2 Multi-class classification

1.

The tuned learning rate and weight regularization parameter were $\eta = 0.001, \lambda = 0.01$. The following plot shows the training and validation loss and accuracy curves.



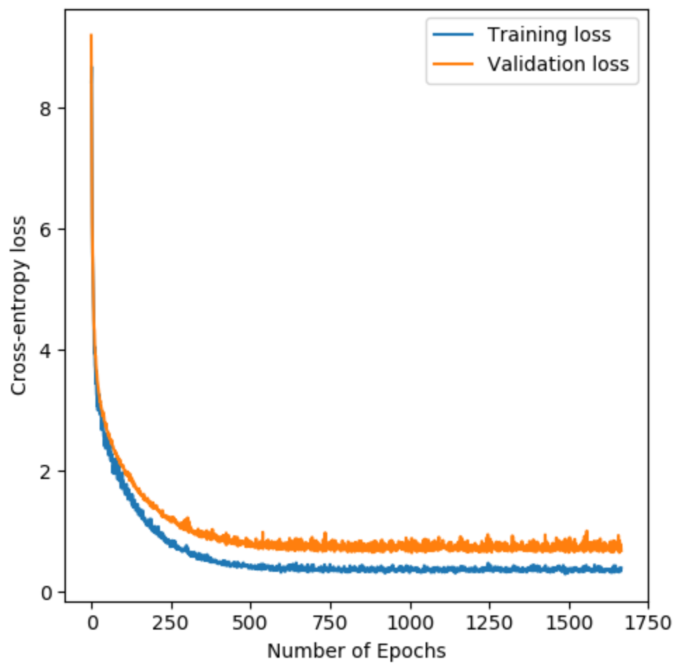
The test classification accuracy was 89.0%. This is much lower than the accuracy achieved with two classes, which makes sense since classifying ten kinds of letters is more difficult than classifying two, with only 1/10 rather than 1/2 answers being correct for each letter.

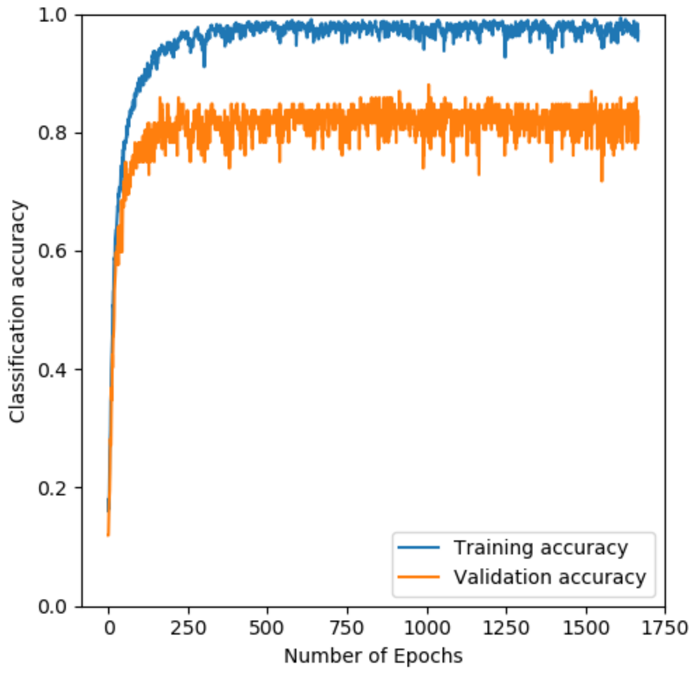


This is an example of a letter that was incorrectly classified by the network. It seems likely that the network was confused by the straight lines in the G—normally a more curved letter.

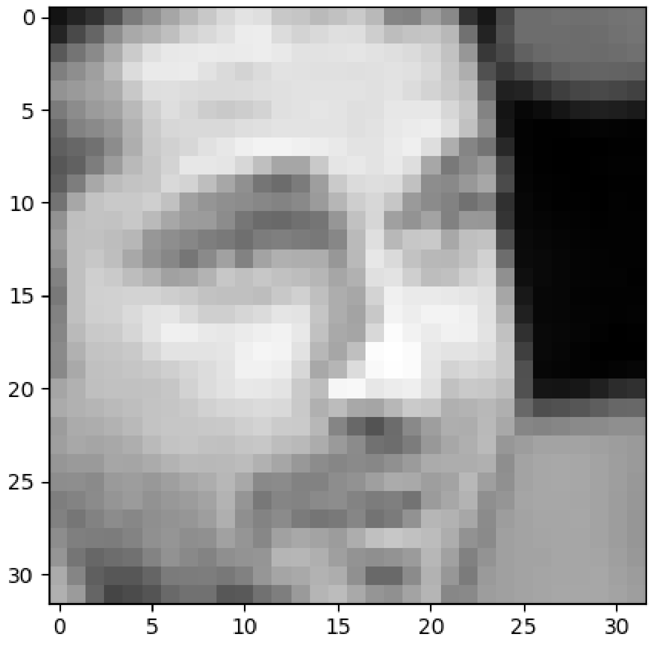
2.

Since the size of the training set was not a multiple of the minibatch size, we ran a smaller minibatch at the end of each epoch to ensure that every training sample was used. The following plot shows the training and validation loss and accuracy curves.





The tuned learning rate and weight regularization parameter were $\eta = 0.01, \lambda = 0.001$. The test classification accuracy was 83.9%.



Network output: 4 (Daniel Radcliffe)
Correct output: 1 (Gerard Butler)

This is an example of a face that was incorrectly classified by the network. While the picture shows Gerard Butler, the network chose Daniel Radcliffe, who looks similar. The photo being in 3/4 profile with the hair cropped out may have also made it more difficult to classify.