Biologically Inspired Computation - Coursework 1

Scott Condron

October 2018

0.1 Introduction

This report will outline variations decisions made during the development of a feed forward neural network. A number of decisions needed to be made regarding topology, activation functions, weight initialisation technique, regularisation strength, sample size etc. Experimentation of these parameters was done systematically and variations for each choice were thoroughly explored.

0.2 Topology

The topology chosen was one hidden layer with 128 neurons. The input shape is N0x784 where N is the number of samples used and 784 is the number of pixels, the input features. N was experimented with to find an acceptable speed of training and is described further in this report. The weights were of shape 784x128 and 128x10. The output layer is of shape Nx10 so that the output scores is a vector of 10 class scores.

The algorithm computes a forward pass through the network and then computes gradients for each layer using backpropagation. These gradients are used to update the weights and biases of neural network. Different methods of regularisation were chosen to test performance increases, both L2 regularisation and dropout were tested.

0.3 Methodology

The training and testing data were loaded, the images were normalised, the data was shuffled and the labels were converted to one hot vectors. Biases were initialized at zero. Validation Accuracy describes the percentage of correct predictions on the test set, while Training Accuracy describes the correct predictions on the training set.

0.4 Choosing Weights Initialisation

Many different methods were used for initialising the weights. Zero initialized weights stopped any learning from happening. Random initial-

ized weights proved better but He-et-al initialized weights helped learning slightly better. (See Figure 1.) He-et-al initialized weights are still random but the range takes into account the size of the previous layer. Improvements from He-et-al initialized weights can be seen better with deeper networked. He et al. (2015)

0.5 Choosing Hyper Parameters

A heuristic approach was taken to find the learning rate, iterations on a log scale with all other hyper parameters being the same showed that $1x10^-2$ would increase in accuracy rapidly but did not reach very high accuracy and caused over-fitting and $1x10^-4$ was too slow. A learning rate of size $1x10^-3$ was chosen.

0.6 Choosing Activation Functions

The activation function chosen after the hidden layer is a ReLU. The activation function at the output was chosen to be a sigmoid.

ReLU was chosen as the activation function after the hidden layer because it is easily differentiable and it is less computationally expensive than a tahn or sigmoid activation functions. "Neural networks with rectified linear unit (ReLU) non-linearities have been highly successful for computer vision tasks and proved faster to train than standard sigmoid units, sometimes also improving discriminative performance." **?**

0.7 Choosing Cost Function

The log loss function was chosen as an appropriate function to use for multi-label classification problems. This was chosen because it suits problems where the output of the model is a probability value between 0 and 1, as is the case with the softmax activation function used at the output of the model. It also takes into account the size of the difference between the output and the actual label, increasing greatly when there is a large difference between the predicted value and the desired value. Nielsen (2018)

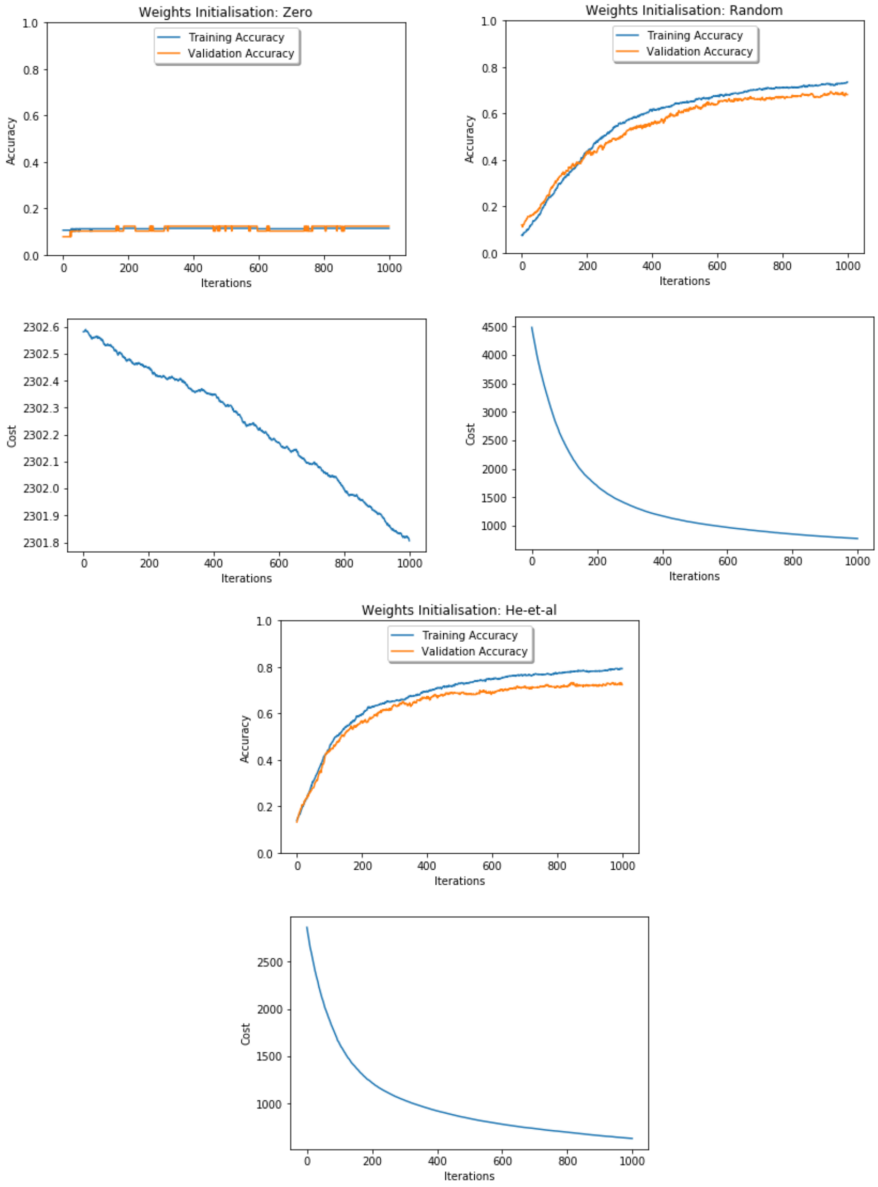

Figure 1: Weight inialisated at Zero, Normal and He-et-al

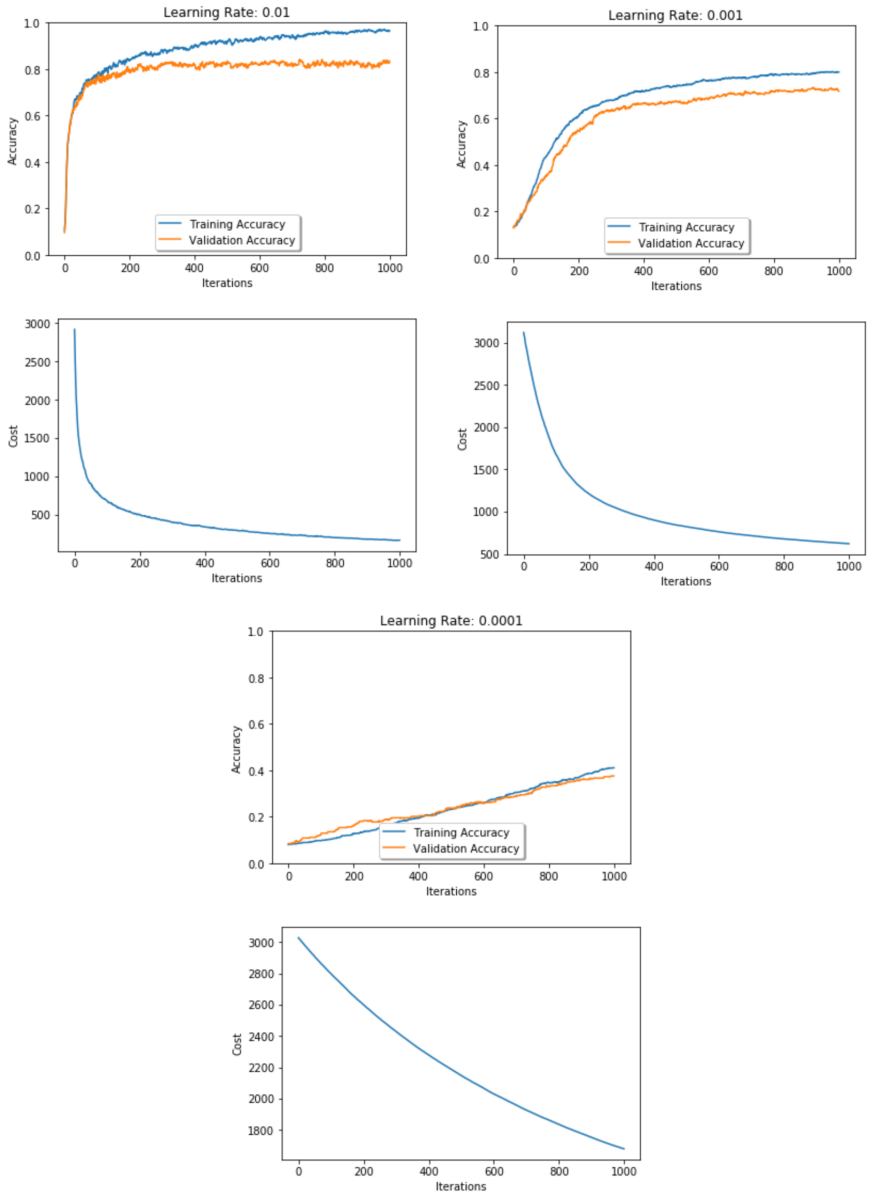

Figure 2: Performance at Learning Rate = 0.01, 0.001, 0.0001

0.8 Balancing training time and accuracy

To test different configurations that would speed up training time, variations of sample size N, number of iterations through training cycles and use of minibatches were tested. Below are some of the results of these tests.

Sample Size	Iterations	Minibatch Size	Accuracy (%)	Training time (secs)
10000	1000	50	73	56
1000	10000	n/a	79	140
1000	10000	100	78	62
1000	10000	50	80	62
1000	10000	20	82	68

This showed that using a large sample size and small number of iterations took a similar amount of time but with worse accuracy. Using a smaller sample size and larger number of iterations improved accuracy. Using minibatches helped training time considerably which helped when experimenting for future tests. Decreasing minibatch size increased accuracy, settling on a minibatch size of 20.

0.9 Regularisation

Both L2 regularisation and dropout were experimented with. Both improved learning and increased the training and validation accuracy of the model.

0.10 Dropout

Experimentation with dropout helped the model generalise and made the validation accuracy increase. ”The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units from co-adapting too much.” Srivastava et al. (2014). Variations of the percentage of dropped units was experimented with.

There was a varying level of improvement when using dropout, it increases accuracy but can reduce accuracy when too much is used.

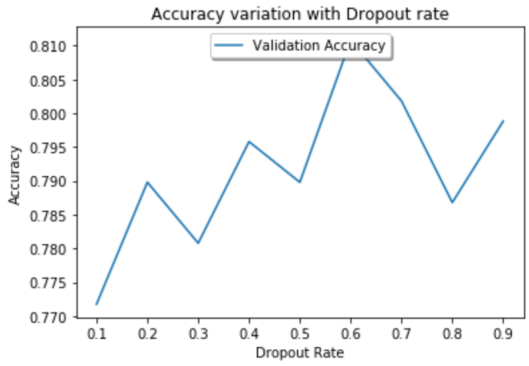

Figure 3: Dropout from 0.1 - 0.9

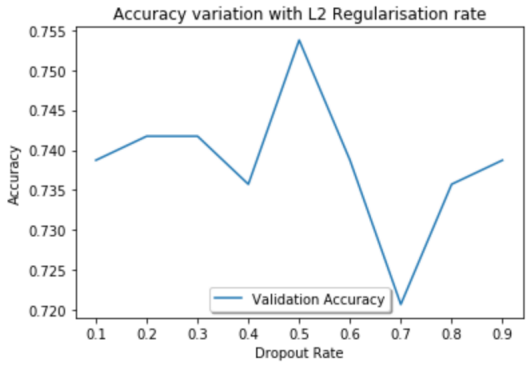

Figure 4: L2 Regularisation from 0.1 - 0.9

0.11 L2 Regularisation

”A different way to constrain a network, and thus decrease its complexity, is to limit the growth of the weights through some kind of weight decay. It should prevent the weights from growing too large unless it is really necessary. ” Krogh & Hertz (1991). Varying levels of L2 regularisation was used to test if weight decay would improve generalization alongside dropout.

This second graph shows both L2 Regularisation and Dropout rates being increased side by side. There was continued improvement in accuracy as they were added until there was too much penalizing and learning was being limited.

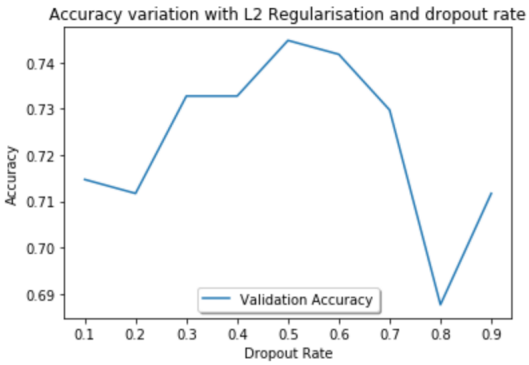

Figure 5: L2 Regularisation and Dropout from 0.1 - 0.9

0.12 Conclusion

A two layer neural network was built to classify images into 10 classes. Methods to speed up training time were tested, including varying sample size, number of iterations and using minibatches. Variations on weight initialisations were tested, with He-et-al initialization being chosen. To try improve generalisation of the model, 2 different methods of regularisation were tested, L2 and Dropout. Both improved generalization and accuracy of the model, with dropout being slightly more successful. When tried side-by-side, there was little improvement. Further experiments would be to try a deeper network with more layers and use more samples during training with more computational power.

Bibliography

He, K., Zhang, X., Ren, S. & Sun, J. (2015), 'Deep residual learning for image recognition', *CoRR* **abs/1512.03385**.
URL: *http://arxiv.org/abs/1512.03385*

Krogh, A. & Hertz, J. A. (1991), A simple weight decay can improve generalization, *in* 'Proceedings of the 4th International Conference on Neural Information Processing Systems', NIPS'91, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 950–957.
URL: *http://dl.acm.org/citation.cfm?id=2986916.2987033*

Nielsen, M. A. (2018), 'Neural networks and deep learning'.
URL: *http://neuralnetworksanddeeplearning.com/*

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014), 'Dropout: A simple way to prevent neural networks from overfitting', *J. Mach. Learn. Res.* **15**(1), 1929–1958.
URL: *http://dl.acm.org/citation.cfm?id=2627435.2670313*