Implementation of a simple neural network

Steven Luu

Abstract

We implement a simple neural network consisting of a single hidden layer and use this for the purpose of classifying images of digits.

1 Introduction

In this task we implement a simple neural network from first principles and use this to classify images of digits ranging from 0-9. The neural network we construct consists of a single hidden layer, whose activation function is the sigmoid function. As this is a classification problem, we use the softmax function as the activation function of the output layer. As such, given some input data, the neural network will output a vector of values consisting of 10 elements/values. These values correspond to the models estimated probabilities of the image being classified as either a $0, 1, 2, \ldots, 9$.

To train the algorithm, we use the cross-entropy loss function as a measure of the model's accuracy. Moreover, we compare the performance of the neural network with and without the use of L2 regularization.

2 Results

The code used in this task is contained in the file named 'NN_code'. The training dataset was pre-processed as follows. The dataset containing 60000 examples was permuted to create a randomized ordering, which was then split into a training and development datasets containing 50000 and 10000 examples, respectively. For compute time considerations, the training and development datasets were then normalized using their means and standard deviations.

Instead of training the neural network over the complete training dataset we train over its batches. To do this we simply divide the training dataset into n equal-sized batches. This way, the network is trained over the first batch, say, B_1, and back-propagation is then used to find the optimal parameters. This procedure is repeated up until the last batch, ultimately training over the complete training dataset; the first epoch. We repeat this complete procedure for some specified number of epochs; in this task we chose 30 epochs.

After the neural network finishes training under an epoch, the value of the cross-entropy loss function and its accuracy against the development set is then computed.

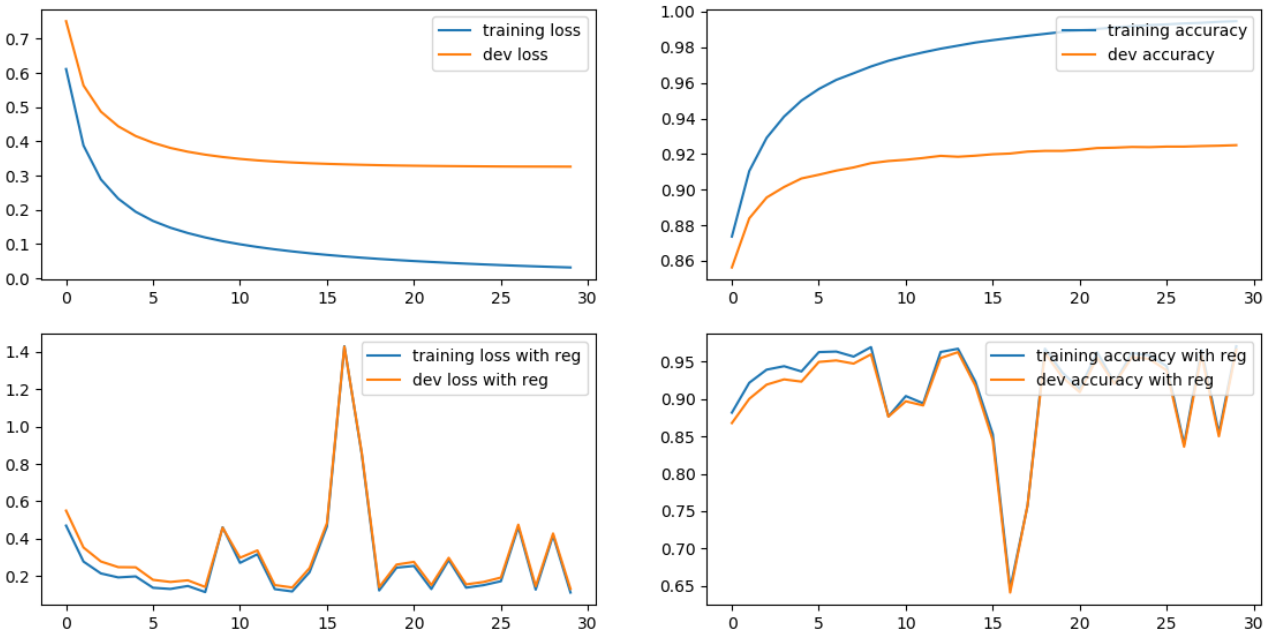

Figure 2.1: The figure above shows the behaviour of the cross-entropy loss function and the accuracy of the neural network as a function of the epochs. Specifically, the first and second row of images illustrate the behaviour of the loss function and accuracy of the neural network when it is trained without and with (L2) regularization, respectively. The value of the regularization parameter used here was 0.5. These images show that the behaviour of the loss and accuracy functions over the development dataset does not follow the behaviour of the model when trained over the training dataset. Contrast to training with regularization, the performance of the model over the development dataset follows closely to that over the training dataset. This may be an indication that the model is overfitting the data when trained without regularization. Under this hypothesis, the inclusion of the regularization term allows the model to avoid overfitting with the added benefit of increasing its accuracy.