Project 1: MLP Implementation

In this project a 2-layer neural net with SoftMax Classifier was implemented.

Forward pass

This section of the code computes the scores from the equation W_2 (ReLU ($W_1x + b_1$)) + b_2, which will be the input for the SoftMax classifier. The result from this section of the code seems correct, as the difference from the correct results is 3.68 * e^{-08}.

Finish forward pass and compute loss

The forward pass then moves on to the SoftMax classifier, where the output is computed using the SoftMax function. Total loss is then computed from SoftMax and log likelihood loss, in addition to L-2 regularization loss to the weights. There seems to be some error in this section, as the difference between the correct loss is 0.019, which is bigger than 1e-12.

Backward pass

Using backpropagation, the gradients for W1, W2, b1 and b2 are calculated as follows:

$$\frac{\partial z_2}{\partial p_1} = W_2 \qquad \frac{\partial p_1}{\partial z_1} = ReLU'(z1) \qquad \frac{\partial z_2}{\partial b_2} = \frac{\partial z_1}{\partial b_1} = 1 \qquad \frac{\partial z_1}{\partial W_1} = X \qquad \frac{\partial z_2}{\partial W_2} = p_1$$

$$\frac{\partial L}{\partial z_2} = (p_2 - y) \qquad \frac{\partial L}{\partial p_1} = \frac{\partial L}{\partial z_2} W_2^T \qquad \frac{\partial L}{\partial z_1} = \frac{\partial L}{\partial p_1} * (1\ if\ z1 > 0, otherwise\ 0)$$

$$\frac{\partial L}{\partial W1} = X^T \frac{\partial L}{\partial z_1} \qquad \frac{\partial L}{\partial W2} = p_1^T \frac{\partial L}{\partial z_2} \qquad \frac{\partial L}{\partial b_1} = \frac{\partial z_1}{\partial b_1}\frac{\partial L}{\partial z_1} = \frac{\partial L}{\partial z_1} \epsilon\, \mathbb{R}^{X,1} \qquad \frac{\partial L}{\partial b_2} = \frac{\partial z_2}{\partial b_2}\frac{\partial L}{\partial z_2} = \frac{\partial L}{\partial z_2} \epsilon\, \mathbb{R}^{Y,1}$$

The result seems to be correct as the relative errors were in the range e^{-09} to e^{-11}.

Random mini-batch creation

As random training data is provided in array X and its corresponding label in Y, creation of a random mini-batch is done by generating a random index. This index then determines the training data to be used, obtained from X and Y. When training the network, the loss decreases which must mean this code is properly providing random mini-batches and training successfully.

Update parameters

The parameters are updated with the loss gradients by the learning rate (alpha). This is already computed, so the parameters are simply updated by subtracting the new value multiplied by the learning rate from the old value. When training the network, the loss decreases which must mean this code works properly as the parameters are being updated to decrease the loss.

Predict labels

This code is very similar to the forward pass code as it simply performs forward pass, but in order to predict the label it returns the argmax(y) of the scores instead of the scores themselves. This code is used when determining the validation accuracy, which seems to be working correctly.