Project1 Report

Akanksha Bhattacharya
anbhatta@ncsu.edu

Nishanth Vimalesh
nvimale@ncsu.edu

Vandan Phadke
vphadke@ncsu.edu

I. Gradient of Cross-Entropy Function

Assume $\sigma(a) = \hat{y}$ which is the predicted value and y is the actual label.
We can write the loss function as following:

$$L(y, \hat{y}) = \sum_k -(y_k \ln \hat{y_k}) - ((1 - y_k) \ln (1 - \hat{y_k}))$$

We have to find the gradient of this function,

$$\nabla_a L(y, \hat{y})$$

where $\hat{y} = \sigma(a)$

$$\frac{dL(y,\hat{y})}{da} = \frac{dL(y,\hat{y})}{d\hat{y}} \frac{d\hat{y}}{da}$$

Considering a single neuron j, we can find

$$\frac{\partial L(y,\hat{y})}{\partial \hat{a_j}}$$

The partial differential of all the other terms in the summation will become 0 with respect to $\hat{y_j}$ as they will be treated as constants. One term remains in which $k = j$

$$\frac{\partial L(y,\hat{y})}{\partial \hat{a_j}} = \frac{\partial(-(y_j \ln \hat{y_j}) - ((1 - y_j) \ln (1 - \hat{y_j}))}{\partial \hat{y_j}}) * \frac{\partial \sigma(a_j)}{\partial \hat{a_j}}$$

Simplifying this, we get

$$\frac{\partial L(y,\hat{y})}{\partial \hat{a_j}} = (\frac{1 - y_j}{1 - \hat{y_j}} - \frac{y_j}{\hat{y_j}}) * (\sigma'(a_j))$$

$$\frac{\partial L(y,\hat{y})}{\partial \hat{a_j}} = (\frac{1 - y_j}{1 - \hat{y_j}} - \frac{y_j}{\hat{y_j}}) * (\sigma(a_j) * (1 - \sigma(a_j)))$$

But $\sigma(a_j) = \hat{y_j}$

$$\frac{\partial L(y,\hat{y})}{\partial \hat{a_j}} = (\frac{1 - y_j}{1 - \hat{y_j}} - \frac{y_j}{\hat{y_j}}) * (\hat{y_j} * (1 - \hat{y_j}))$$

$$\frac{\partial L(y,\hat{y})}{\partial \hat{a_j}} = \frac{\hat{y_j} - y_j\hat{y_j} - y_j + y_j\hat{y_j}}{\hat{y_j} * (1 - \hat{y_j})} * (\hat{y_j} * (1 - \hat{y_j})$$

Simplifying,

$$\frac{\partial L(y,\hat{y})}{\partial \hat{a_j}} = \hat{y_j} - y_j$$

This is the partial derivative with respect to a single neuron j. We can extend this to a matrix form

$$\frac{\partial L(y,\hat{y})}{\partial \hat{a}} = \hat{y} - y$$

II. Derivative of sigmoid function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Differentiating this function wrt x

$$\frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2}$$

$$\sigma(x) * (1 - \sigma(x)) = \frac{1}{1 + e^{-x}} * (1 - \frac{1}{1 + e^{-x}})$$

$$\sigma(x) * (1 - \sigma(x)) = \frac{e^{-x}}{(1 + e^{-x})^2}$$

Thus,

$$\sigma'(x) = \sigma(x) * (1 - \sigma(x))$$

III. Structure of the Neural Network

A. Structure

Our current neural network contains all fully connected layers. It has three layers (one input, one output and one hidden)

As the image is an vector of size $784(28 * 28)$, the input layer contains 784 neurons.

The hidden layer contains 20 neurons,

The output layer contains 10 neurons which correspond to digit (0-9) labels of the MNIST dataset.

B. Other hyperparameters

The following is the value of hyperparameters which was set during the training of the NN:

1) **Number of Epochs:** 100
2) **Mini Batch Size:** 128
3) **Learning Rate:** $0.5 * 10^{-3}$ (this was initially set to $1 * 10^{-3}$)
4) **Regularization Constant (λ):** 0

IV. Learning Curves

A. Loss

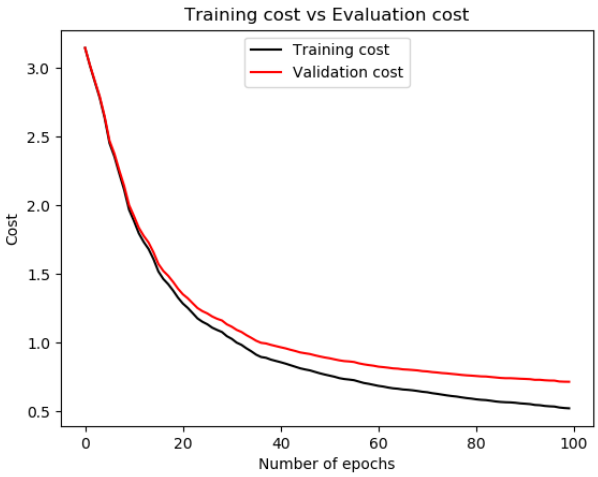

Fig. 1. Loss function vs the number of epoch trained

B. Accuracy

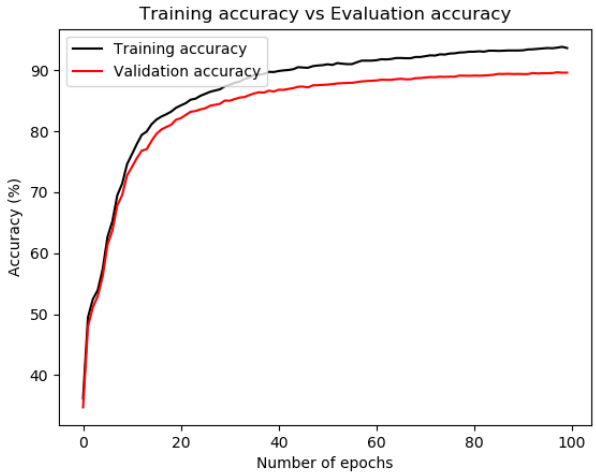

Fig. 2. Accuracy vs the number of epoch trained

C. Explanation of the above plots

As expected, with each epoch the training error drops as the weights and biases get updated due to backpropogation.

If we continue training for more epoch, we will see further decrease in the training and validation error until a point where the validation error starts rising up again whereas the training error continues to decrease. This is the point at which we can stop the training.

V. Accuracy on test set

We obtained an accuracy of 89.6% on the provided test set