Neural Network with Backpropagation

Axel Ind (*4563539*)

November 6, 2017

Abstract

This document details work to create a neural network that successfully classifies the MNSIST database of images into the numbers which each image represents. In testing the Sigmoid, Relu, and Tanh activation functions were used. Testing error of 3.32% is achieved.

1 Activation Functions

Sigmoid

Equation

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

Derivative

$$f'(x) = f(x)(1 - f(x)) \tag{2}$$

Tanh

Equation

$$g(x) = tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3}$$

Derivative

$$g'(x) = 1 - g(x)^2 \tag{4}$$

Relu

Equation

$$h(x) = \max(0, x) \tag{5}$$

Derivative

$$h'(x) = \begin{cases} 0 \text{ for } x < 0 \\ 1 \text{ for } x >= 1 \end{cases} \tag{6}$$

2 Dataset

The data set used is the MNIST dataset of hand-written numbers represented by 28px $\times$ 28px grayscale images with intensity values in the range $(0, 256)$.

3 Network Design

Inputs

Stochastic Gradient Descent: $784 \times n$ (where n is the batch size).
Gradient Descent: $784 \times N$ (where N is the number of samples).

Hidden Layers

Some number of hidden layers ($|h| > 0$), with some number of weights (W_i) and biases (b_i) at each layer (h_i). Each layer makes use of one of the described activation functions.

Outputs

Some linear or unhotted array that uniquely identifies one of 10 classes.

4 Training

All training results are achieved using backpropagation to update weights.

General Observations

- Changing the number of hidden layers did not consistently imrpove network performance.
- As expected, increasing the number of entries in the data set significantly improved performance.

4.1 General Training

Without Optimization

Without any form of regularization, the greatest results in the table were above the required accuracy threshold.

Activation	Num Layers	Hidden Nodes	Learning Rate	Epochs	Error
Relu	3	94	0.1	110	3.32%
Relu	4	200	0.31	30	3.47%

Table 1: Unoptimized training results for the MNIST Test Set

With Optimization

Optimizing Steps

- Consensus training: three networks were trained simultaneously on the training data and the class which two or more agreed on was used as the prediction when the testing data was used.
- This approach, though it seemed promising, simply served to significantly slow computation time and did not improve accuracy on the most successful unoptimised networks.
- The failure of this technique suggests that the network tends to become trapped in the same small set of local minima.

Activation	Num Layers	Hidden Nodes	Learning Rate	Epochs	Error
Relu	4	84	0.05	170	3.52%
Relu	4	98	0.11	45	3.56%

Table 2: Optimized training results for the MNIST Test Set

5 Conclusion

The created neural network is able to classify numbers in the MNIST dataset to a very high degree of accuracy. The addition of a consensus optimizing technique did not significantly significantly reduces the observed error rate.

It also appears clear that the sigmoid and tanh functions are consistently outperformed by the Relu function.

Further, the randomness inherent in the neural network algorithm may have obscured configurations which were more optimal