Deep Learning Project 1: MNIST Digit Recognition Using Multi-Layer Neural Networks

Elliot Greenlee

December 18, 2018

1 Subject Comparison

I know this is not right, but I didn't at the time so I am leaving it Batch vs. Online Processing: In batch processing, a a random selection of samples are all fed through the network to establish an error that is used for training the weights. This occurs multiple times with different random samplings in "epochs" of learning. Online processing can be thought of as an extreme case of this, where the batch size is only one sample.

Gradient Descent vs. Stochastic Gradient Descent: Gradient descent attempts to optimize the weights of the network by traveling towards an error minimum from running the training data through the network. Unfortunately, when the training data is large, this is computationally expensive, and can be approximated by randomly sampling some parts of the training data for the calculation instead. Using this random sampling is stochastic gradient descent.

Perceptron vs. Sigmoid Neurons: Perceptrons use a harsh division between classifications as output, causing small changes in weights to possibly produce a large difference in output. This creates a problem when trying to use gradient descent, so sigmoid neurons with a sigmoid function are used as a smooth function instead.

Feedforward vs. Backpropagation: In the feedforward operation, inputs from the previous layer are fed into the next layer, multiplied by the weights and added to the bias, continuing forward until a final output is reached. This method occurs during both training, testing, and final network operation. Backpropagation computes the gradient of the cost function for each weight, so that they can be adjusted properly during training.

2 Tensorflow Graphs

For each setup, the final function is a softmax to better generate probabilities.

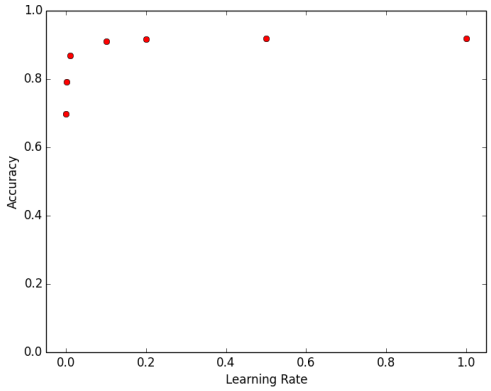

Figure 1: Accuracy when varying the learning rate for a (784, 10) perceptron with 1000 iterations and mini-batch size 100. Learning rates 0.0001, 0.001, 0.01, 0.1, 0.2, 0.5, 1.0

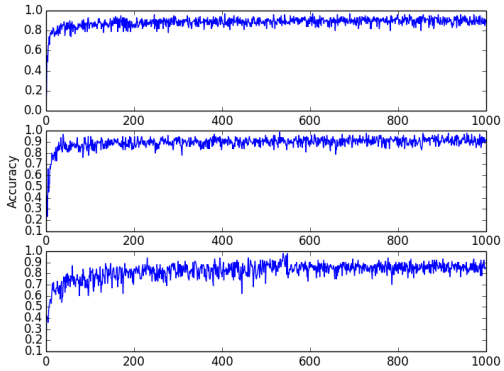

Figure 2: Accuracy across 1000 iterations for a (784, 10) perceptron with batch size 100. Learning rates 1.0, 0.1, 0.01 from top to bottom.

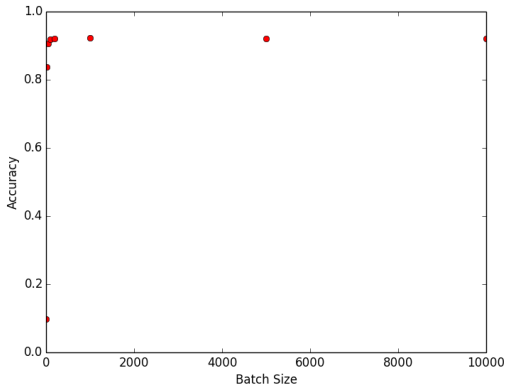

Figure 3: Accuracy when varying the batch size for a (784, 10) perceptron with 1000 iterations and learning rate 0.5. Batch sizes 1, 10, 50, 100, 200, 1000, 5000, 10000

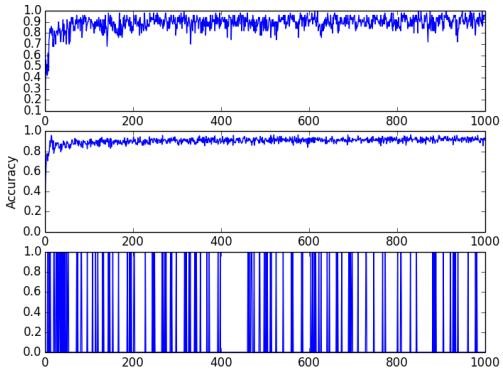

Figure 4: Accuracy across 1000 iterations for a (784, 10) perceptron with learning rate 0.5. Batch sizes 50, 200, 1 from top to bottom.

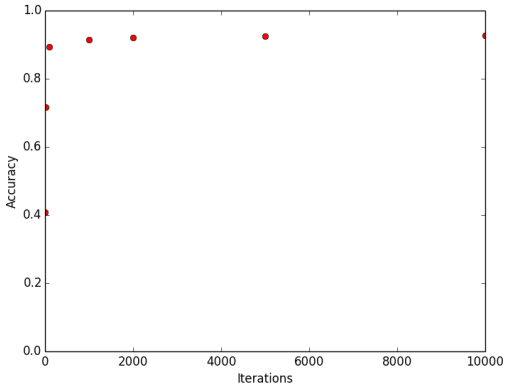

Figure 5: Accuracy when varying the iterations for a (784, 10) perceptron with batch size 100 and learning rate 0.5. Iterations 1, 10, 100, 1000, 2000, 5000, 10000

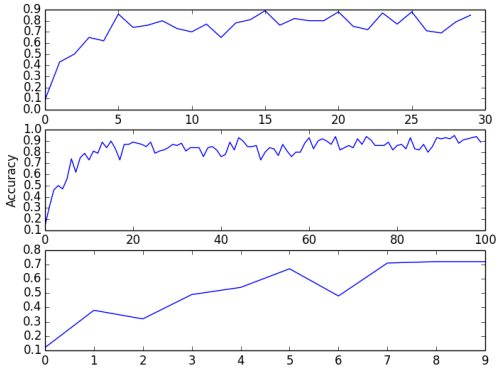

Figure 6: Accuracy across variable iterations for a (784, 10) perceptron with batch size 100 and learning rate 0.5. Iterations 10, 100, 30 from top to bottom.

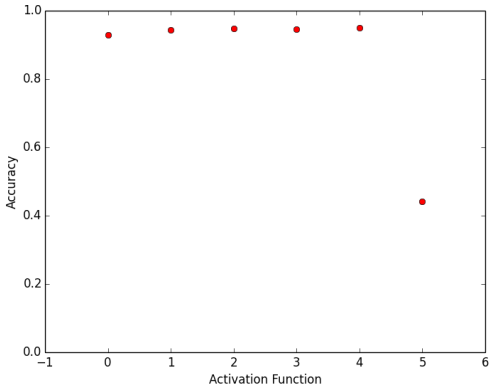

Figure 7: Accuracy when varying the activation function for a (784, 30, 10) neural network with 1000 iterations, batch size 100 and learning rate 0.5. Activation functions are sigmoid, tanh, elu, softplus, relu, softmax from left to right

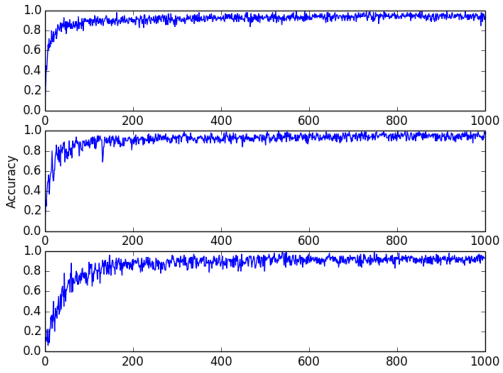

Figure 8: Accuracy across 1000 iterations for a (784, 30, 10) neural network with batch size 100 and learning rate 0.5. Activation functions are relu, tanh, sigmoid from top to bottom.

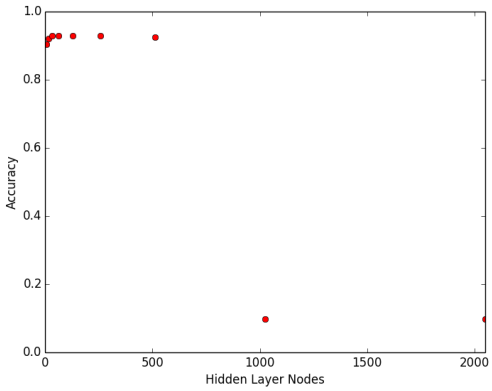

Figure 9: Accuracy when varying the number of nodes in a (784, x, 10) neural network with 1000 iterations, batch size 100 and learning rate 0.5 with a sigmoid activation function. Nodes 8, 16, 32, 64, 128, 256, 512, 1024, 2048

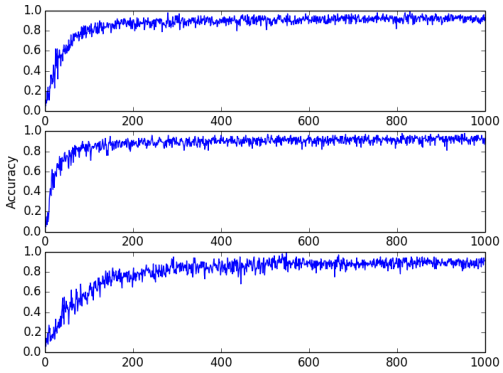

Figure 10: Accuracy across 1000 iterations for a (784, x, 10) neural network with batch size 100 and learning rate 0.5 using a sigmoid activation function. Nodes 32, 256, 8 from top to bottom.

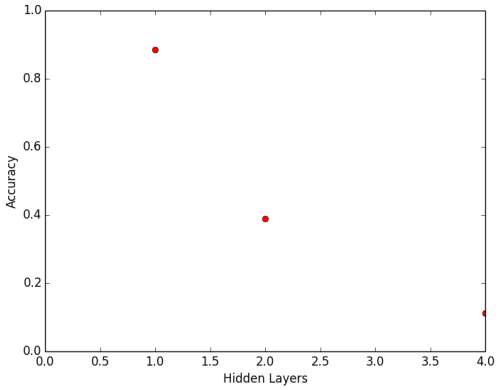

Figure 11: Accuracy when varying the number of layers in a (784, 30, ..., 30, 10) neural network with 10000 iterations, batch size 100 and learning rate 0.01 with a sigmoid activation function. Layers 1, 2, 4

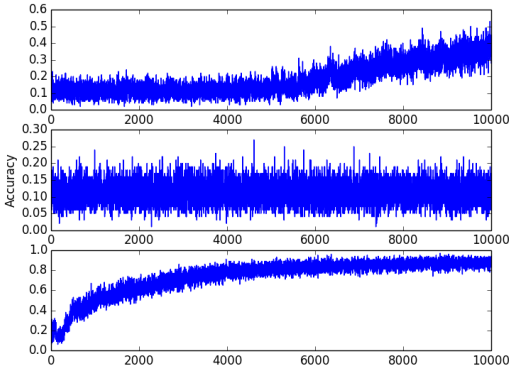

Figure 12: Accuracy across 10000 iterations for a (784, 30, ..., 30, 10) neural network with batch size 100 and learning rate 0.01 using a sigmoid activation function. Layers 2, 8, 1 from top to bottom.

3 Numpy Graphs

Because of the difference is speed of Numpy vs Tensorflow, much smaller values were chosen when varying using Numpy. The activation function is always sigmoid.

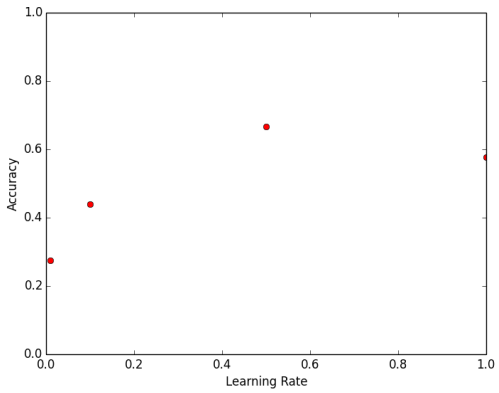

Figure 13: Accuracy when varying the learning rate for a (784, 10) perceptron with 30 iterations and mini-batch size 10. Learning rates 0.01, 0.1, 0.5, 1.0

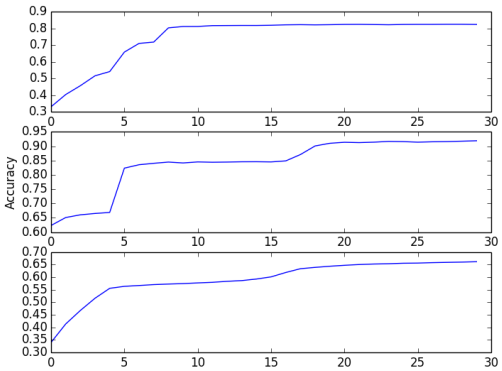

Figure 14: Accuracy across 30 iterations for a (784, 10) perceptron with batch size 10. Learning rates 1.0, 0.1, 0.01 from top to bottom.

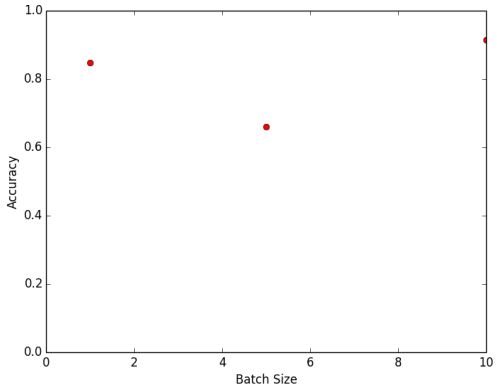

Figure 15: Accuracy when varying the batch size for a (784, 10) perceptron with 30 iterations and learning rate 0.5. Batch sizes 1, 5, 10

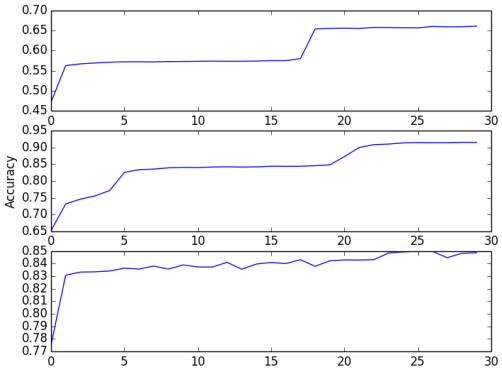

Figure 16: Accuracy across 30 iterations for a (784, 10) perceptron with learning rate 0.5. Batch sizes 1, 5, 10 from top to bottom.

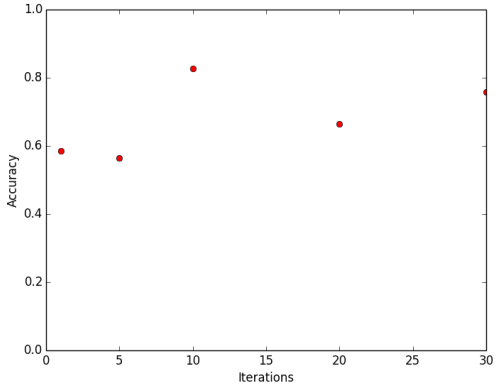

Figure 17: Accuracy when varying the iterations for a (784, 10) perceptron with batch size 10 and learning rate 0.5. Iterations 10, 20, 30

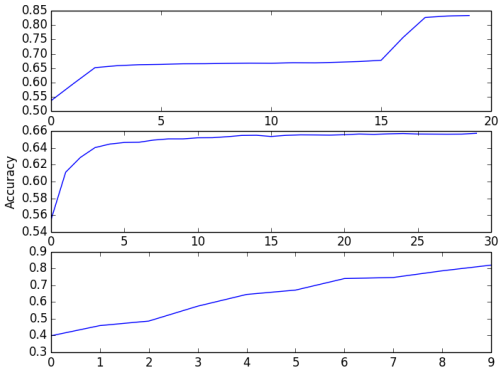

Figure 18: Accuracy across variable iterations for a (784, 10) perceptron with batch size 10 and learning rate 0.5. Iterations 20, 30, 10 from top to bottom.

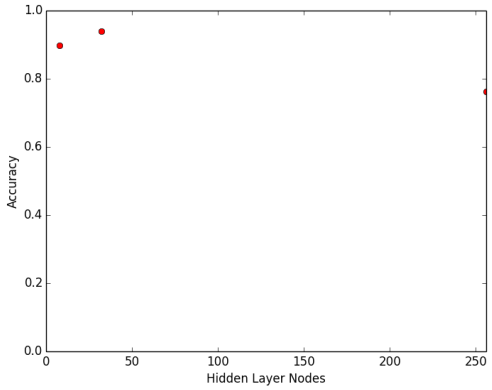

Figure 19: Accuracy when varying the number of nodes in a (784, x, 10) neural network with 30 iterations, batch size 10 and learning rate 0.5 with a sigmoid activation function. Nodes 8, 32, 256

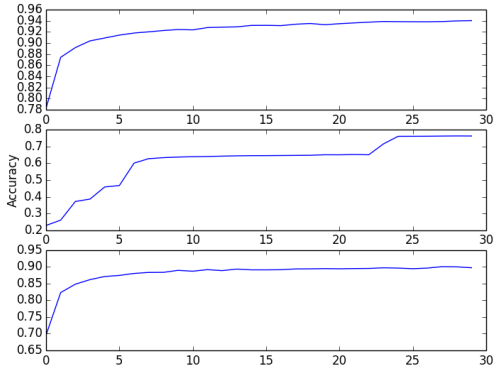

Figure 20: Accuracy across 30 iterations for a (784, x, 10) neural network with batch size 10 and learning rate 0.5 using a sigmoid activation function. Nodes 256, 8, 32 from top to bottom.

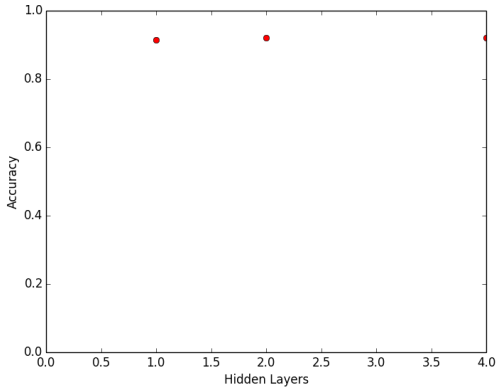

Figure 21: Accuracy when varying the number of layers in a (784, 30, ..., 30, 10) neural network with 30 iterations, batch size 10 and learning rate 0.1 with a sigmoid activation function. Layers 1, 2, 4

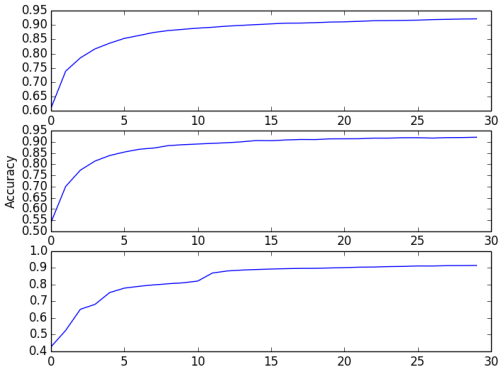

Figure 22: Accuracy across 30 iterations for a (784, 30, ..., 30, 10) neural network with batch size 10 and learning rate 0.1 using a sigmoid activation function. Layers 4, 2, 1 from top to bottom.