COSC 522 Machine Learning Project 4 MNIST Digit Recognition Using Multi-Layer Neural Networks

Yangsong Gu

October 24 2021

1 Introduction

1.1 Objective

The objective of this project is to have a thorough understanding of the traditional multi-layer perceptron and back propagation.

2 Technical approach

2.1 batch vs. online processing

2.1.1 batch processing

- the parameters e.g., biases and weights, are updated based on a batch of training samples. For instance, the weights and bias of all mini batches (i.e., whole training set) are added up together and averaged before activating sigmoid function.
- when batch size is 1, the batch learning becomes online learning.
- the batch learning is not fit for huge dataset training since it has to deal with huge data samples each time.

2.1.2 online processing

- update parameters like weights and biases immediately after processing each training sample.
- It could speed up the convergence or slow down the convergence. As one of online learning approach, wta cost much more time than kmeans in image compression (Project 3). This issue is prone to happen when algorithm is feed with bad data (i.e., outliers).
- it needs a learning rate to update parameters. If the learning rate is too high, the algorithm will be difficult to converge. Conversely, the algorithm can converge while it is time-consuming if learning rate is too small.
- compared to batch learning, it is more time effective.

2.2 gradient descent vs. stochastic gradient descent

The goal of gradient descent is to compute the descent ∇C of training data. The differences are:

2.2.1 Gradient descent

- use the loss and derivative of all training samples.
- It becomes time-consuming when handling huge training set. It is slower than Stochastic gradient descent.
- gradient descent can obtain the optimal solution.

2.2.2 Stochastic gradient descent

- use a small sample of randomly chosen training samples.
- it can speed up the convergence when dealing with a large number of training samples.
- by averaging this small portion of training set each time, it turns out that we can quickly get a good estimate of true gradient.
- due to the random selection, the solution of stochastic gradient descent may keep oscillating around the true solution.

Mini-batch processing is an approach that dealing with a small portion of training samples each time.

2.3 perceptron vs. sigmoid neurons

2.3.1 perceptron

- a perceptron is a single layer nerual network.
- can only do binary classification.
- it is similar to linear classifier generated from weighted features.
- the perceptron makes decisions by weighing up evidence whose rule can be formulated as:

$$output = \begin{cases} 0 & if\ w \cdot x + b \leq 0 \\ 1 & if\ w \cdot x + b > 0 \end{cases} \tag{1}$$

  where b can be viewed as bias or threshold.

2.3.2 sigmoid neurons

- sigmoid neurons have some same qualitative behavior as perceptrons, for instance, they can figure out how changing the weight and biases will change the output. In comparison, the small changes in weights and bias cause only a small change in sigmoid function output.
- the output of a sigmoid neuron is $(w + b)$, where σ is defined by:

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

- with the inputs $x_1, x_2, ...$, weights $w_1, w_2, ...$ and bias b, the sigmoid function can be rewritten as

$$\frac{1}{1 + exp(-\sum_j w_j x_j - b)}$$

- above sigmoid function has two advantages, one is that it is differentiable, which made the gradient descent possible. The other is that it has as output any real number between 0 and 1.
- Figure 1 shows the mapping of sigmoid neuron and perceptron, given the input.

(a) $y = x$

(b) perceptron

Figure 1: Comparison

2.4 feedforward vs. backpropagation

2.4.1 feedforward

- output from one layer is used as input to the next layer.
- No loops in the nework, the information is always fed forward, never fed back.
- A single-layer perceptron is the simplest feedforward network.
- different from recurrent neural networks.
- the input is transformed and forwarded.

2.4.2 back propagation

- A fast algorithm for computing the gradient of the cost function with respect to parameters.
- the problem is essentially "how to choose weigth w to minimize the error E between expected output and the actual output"
- the basic idea behind backpropagation is gradient descent and chain rule.
- the error (i.e., difference between target and output) is back propagated.

2.4.3 Why do we need to introduce "bias" when training a neural network?

- for a perceptron, th bias is a measure of how easy it is to get the perceptron to fire.
- Bias is a constant which helps the model in a way that it can fit best for the given data.
- bias can be used to delay the triggering of the activation function.

3 Task 2

3.1 Task 2.1 Base model

Figure 2 shows the convergence curve of neural network NN1 with parameters network size = [784, 30,10], eta = 1, which serves as the baselin. It takes around 20 epochs to converge on test set. The test accuracy oscillates around 95%, and the test loss floats around 500.

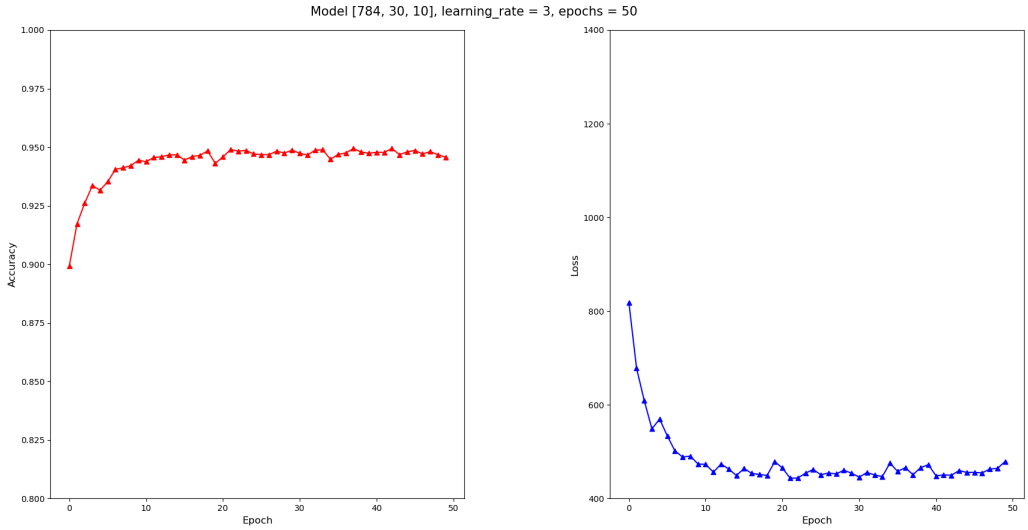

Figure 2: Convergence curve of NN1 [784,30,10], eta = 3, epochs = 50

3.2 Task 2.2 different hyper parameters setup

1. effect of network structure.

Figure 3: Caption