Assignment2

Benjamin Klybor

September 20, 2018

My repository for this assignment can be found here:
`https://github.com/bklybor/Assignment2.git`

Trial	Layer 1	Layer 2	Layer 3	Accuracy	Loss
1	linear	none	none	0.0982	nan
2	sigmoid	none	none	0.208	0.873
3	softmax	none	none	0.910	0.0179
4	linear	sigmoid	none	0.778	0.0457
5	sigmoid	linear	none	0.8246	0.0834
6	linear	softmax	none	0.912	0.0176
7	softmax	linear	none	0.828	0.0334
8	sigmoid	softmax	none	0.863	0.0209
9	softmax	sigmoid	none	0.114	0.0899
10	linear	softmax	sigmoid	0.114	0.0899
11	linear	sigmoid	softmax	0.874	0.0211
12	softmax	linear	sigmoid	0.403	0.104
13	softmax	sigmoid	linear	0.395	0.0700
14	sigmoid	softmax	linear	0.708	0.458
15	sigmoid	linear	softmax	0.8625	0.0211

Figure 1: A table of accuracies for the MNIST data set. All trials used a batch size of 128, 20 epochs, a learning rate of 0.50, and used normalized data.

Questions

1. Is there a relationship between using or not using an activation function and normalizing the data as it pertains to accuracy?

    - Yes there is a relationship between which activation function is used and accuracy and between whether the data are normalized and accuracy. Regardless of the activation function, normalizing the data seems to have a profound impact on the accuracy of the model. On most counts we can increase the accuracy of the model from 10% to over than 50% at least simply by normalizing the data. There is an especially marked difference between using the softmax function with normalized versus non-normalized data.

2. For the single layer perceptron, which activation makes the most sense to use in the output layer?

    - The softmax activation function makes the most sense to use as a single layer. Since the data being used are the greyscale pixelation of natural, hand-written numbers written by different peoepl, the model has to give greater weight to the darkest pixels than if the images were, say, all written by the same person. Since the softmax function exponentiates the weights, those weights that correspond to the darkest pixels will weigh far more heavily on the activation than if they were taken at their non-eponentiated values.

3. What kind of hidden layer architecture worked best for you? A graph or a table, along with an explanation, would likely be the best way to justify your answer.

    - As we can see from Fig. 1 the best layerd architecture that was used has a linear activation as the first layer and a softmax activation as the second layer (Trial 6), although the softmax by itself performs nearly as well as this two-layer architecture. Again, since the data being processed are the natural hadwriting of many different people, the differences between different numbers and not different styles of handwriting need to be emphasized extremely well in order for the model to perform well, so the softmax function is expected to provide the greatest accuracy.

4. (Harder) Suppose we would like to predict more than one class at once (multi-label classification)–for example, "cat" and "sleeping", but we want the classes to be independent of each other, i.e., a high probability of one class should not lower the probability of the other classes. How might one approach this? Would a softmax activation at the prediction layer be appropriate? If so, explain why. If not, what would be more appropriate?

    - Perhaps we could run the data set through one model which uses a softmax activation function and then simultaneosly run it through another model wich uses another softmax activation function, the difference between the two being that the first ignores the characteristics of the learning examples which would activate the function of the second. For example if we wanted to teach a model to learn what a picture of a sleeping cat is, then we would give the first learning example to a model which is learning what a cat is, then give the first learning example to a model which is learning what sleeping is, then combine the two activation functions to make a final activation function for the whole model.