Document Classification using Stochastic Gradient Descent

Ramabhadra V [1]

1. Local Implementation

The steps followed in predicting the class of a document are as follows.

- Find the 'term frequency' of a term in a document, i.e. the number of times that term t occurs in document d. If we denote the raw count by $\mathrm{f}_{\mathrm{t,d}}$,

$$tf(t, d) = log(1 + f_{\mathrm{t,d}})$$

- Find the 'Inverse Document Frequency of a term in the corpus.

$$idf(t, D) = N/|d \in \ D : t \in d|$$

- Then tf-idf(w, d, D) is calculated as,

$$tfidf(w, t, D) = tf(w, d) * idf(w, D)$$

- Now, we will train as many classifiers as the number of classes.
- Consider a particular document, for whichever classes it belong, for them target = 1. For rest of the classes target = 0. Now do Binary classification.[2]
- Now, we have as many classifiers as the number of classes. For each document in the test set, find the probability of the document belonging to a particular class by passing it over that class's classifier. the class with highest probability is the winner. Calculate accuracy accordingly.
- I have used n-gram model. I observed that increasing n increases accuracy.
- Fig.1 shows the plot of loss against epochs.

$$loss = (y - 1) * log(1 - p) - ylog(p)$$

- The learning rates are changed as follows.

$$Increasing : lr = lr + 0.1 * lr * epoch$$

$$Decreasing : lr = lr/(1.5^{\mathrm{epoch}})$$

- The reason for 'decreasing' to perform better is to not skipping the optimal value.

Table 1. Total time taken(in sec) vs number of workers

No. of Workers	Time Taken
1	1347
2	694
3	562

2. Parameter Server Implementation

I implemented Asynchronous SGD in tensorflow. Steps I followed:

- Implemented Parameter Server for MNist data using this link[1].
- Modified the code to suit for DBPedia Dataset.
- Since the data is multi-labelled, I modied the dataset into this.

$$k1, k2data- > k1data, k2data$$

- I could not implement the tf-idf model. Hence I implemented just term frequency model.
- I used GradientDescentOptimiser with cross-entropy loss.
- Even though the loss decreases over epochs the accuracy was not high.
- I observed that the time taken for each epochs decreases as the number of workers increase.

3. Help Sought

1. Vadiraj helped me in setting Parameter Server
2. Akash clarified some doubts.

4. References

1. http://ischlag.github.io/2016/06/12/async-distributed-tensorflow
2. https://www.youtube.com/watch?v=-EIfb6vFJzc

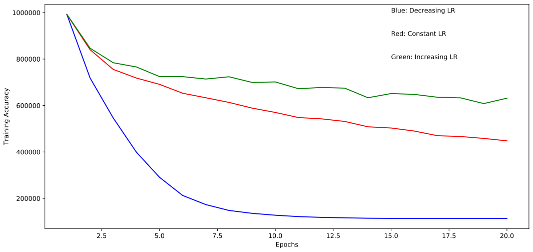

Figure 1. Loss vs Epochs

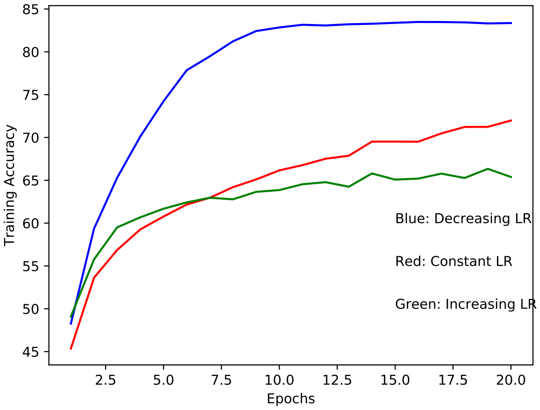

Figure 2. Training Accuracy

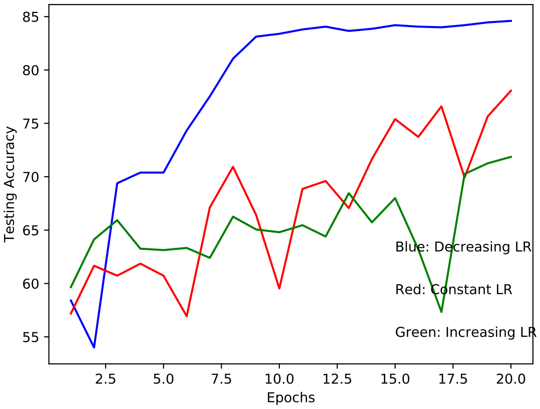

Figure 3. Testing Accuracy

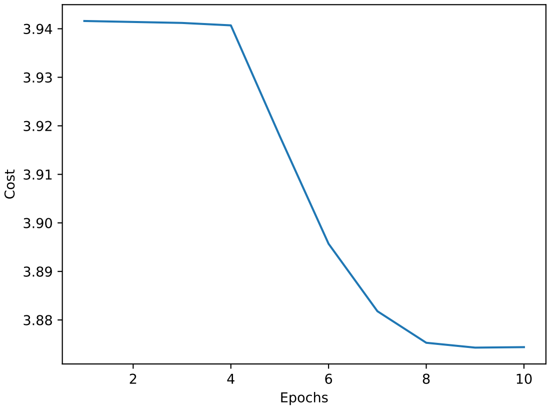

Figure 4. Loss vs Epochs in Parameter Server