Distributed Logistic Regression

DS 222: ML with Large Datasets (2018)
Assignment 2
Swapnil Gupta, 14890

1. Introduction

In this assignment we have performed a multi-label classification task using logistic regression algorithm. Logistic Regression with cross-entropy loss is a linear classifier aiming to maximize the likelihood of posterior probability. For handling the multi-label case during the training, corresponding to each sample equal probability mass is assigned to each positive class and during testing if the class assigned by the classifier belongs to the list of true classes it is considered as a true class.

For all the experiments, distributed tensorflow is used. The key motivation behind choosing distributed tensorflow over Hadoop is because of the iterative nature of the optmization problem where Hadoop requires data loading from the disk after every epoch such requirements are not there in a platform like distributed tensorflow.

2. Dataset

Given is a multi-label document classification task with the following characteristics:

- No. of training set samples: 214998
- No. of validation set samples: 61497
- No. of testing set samples: 29997
- No. of distinct classes: 50
- Vocabulary Size(after preprocessing): 11412

For the above dataset the total no. tuning parameters is $(No.ofclasses) * Vocab_size + (No.ofclasses) = 50 * 11,412 + 50 = 5,82,012$.

2.1. Data Preprocessing

The following preprocessing steps have been performed on the data prior to training:

- Removing punctuation marks attached to the words.
- Removing words with total counts in the corpus less than 75 and greater than 20,000.
- Removing non-alpha-numeric words.

3. In Memory Logistic Regression

These experiments are done on the local pc with moderate specification.

3.1. Training Phase

The following parameter values are kept constant across all the experiments:

- Batch Size: 2048
- Optimizer: ADAM
- No. of Epochs: 50
- L2 regularization constant: 0.005

3.2. Results

The below table summarizes the train and test accuracy measures for three different strategies of varying the the learning rate.

Strategy	TrainAccu	TestAccu	TrainTime(PerEpoch)
Constant LR	82.4	72.5	25.2
Decaying LR	83.5	74.1	25.3
Increasing LR	80.2	72.3	25.8

Constant learning rate: 0.003
Decaying learning rate initialized with 0.001 and decreased exponentially at the rate of 0.95
Increasing learning rate initialized with 0.005 and increased exponentially at the rate of 1.05

The **Figure 1. and 2.** below depict the variations in training loss and test accuracy for the three cases respectively.

The convergence of the model is quite fast and in all the three cases the model converges to almost similar generalization performance. Though it is expected that for higher no. of epochs both the training loss and test accuracy for the case of exponentially increasing learning rate should deteriorate.

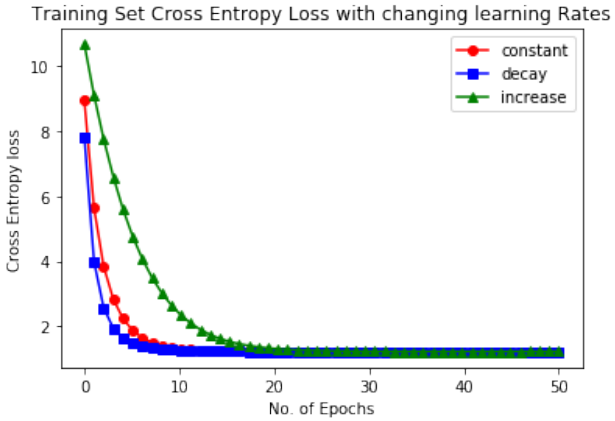

Figure 1. Training loss vs. epochs

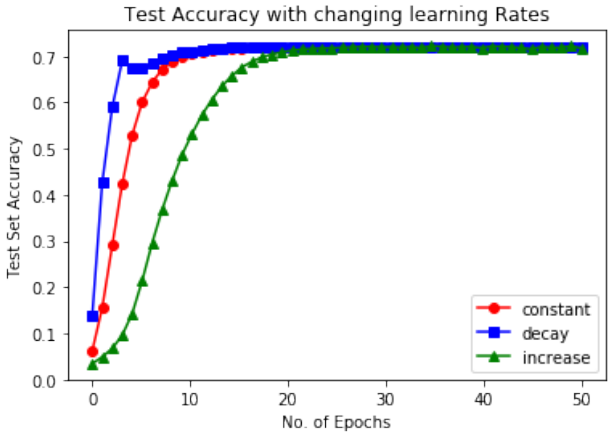

Figure 2. Test set accuracy with no. of epochs

4. Distributed Logistic Regression

In this section distributed training of the Logistic Regression algorithm is performed for three different distributed paradigm namely: **Synchronous, Asynchronous and Stale Sychronous**. For all these experimentation the no. of parameter servers and worker nodes is kept fixed at 2.

Two separate experiments have also been performed to examine the impact varying no. of workers in the asynchronous case and varying the staleness parameter in the stale synchronous case and the results are reported.

For all the experimentations, **Turing Cluster** at CDS department in IISC Bangalore is used.

4.1. Results

For all the reported results and figures at each epoch the average of training loss and test set accuracy across all the workers is taken. The below table summarises the performance of the models trained under three different strategies on the train and test dataset.

Strategy	TrainAccu	TestAccu	TrainTime(perEpoch)
Bulk Synchro	73.9	63.8	21.3
Asynchro	73.4	62.9	18.5
Stale Synchro (10)	73.5	63.5	19.1

A key observation is the significant decrease in the performance on the distributed system. It is most definitely unexpected. The gain in speed is also considerably less considering the fact that two workers are working on the same data simultaneously. This might be due to communication overhead.

Figure 3. and 4. corresponds to varying train loss and test accuracy with different machine learning distribution paradigms.

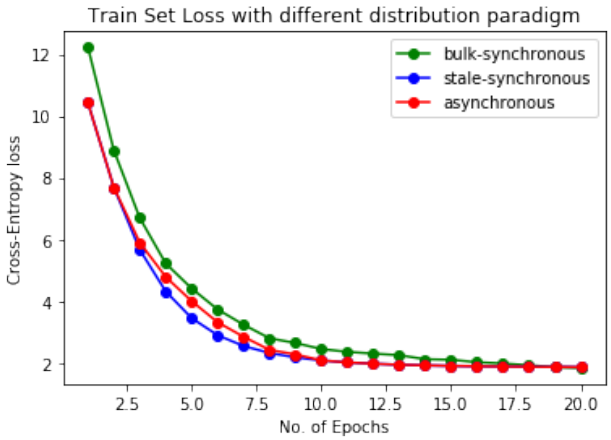

Figure 3. Training loss vs. epochs

Figure 5. captures the decrease in training loss with epochs when different no. of workers are used.

Here, while with more workers the decay is faster in both the cases the convergence point remains the same.

5. Conclusions

The above experiments examine the impact of doing distributed computation over single machine computation. The experiments suggest an unexpected decrease in the performance for the case of distributed computation with slight speed up in per epoch time.

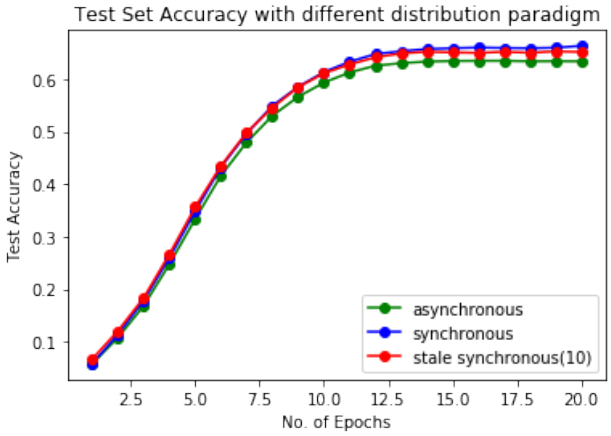

Figure 4. Test set accuracy with no. of epochs

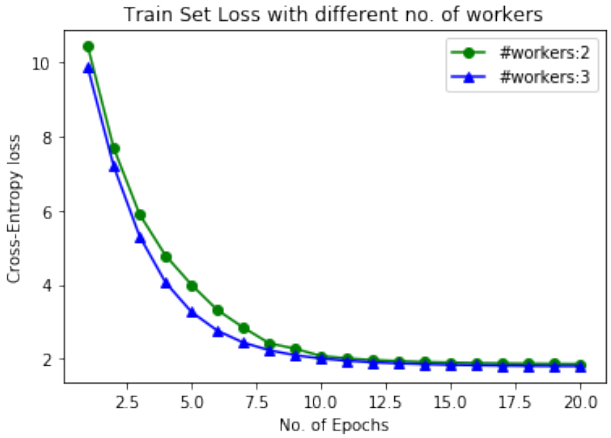

Figure 5. Training loss vs. epochs

Within the distributed computing paradigm as expected bulk synchronous mode is most accurate but since the workers need to work in coherence the communication overhead is most visible which will only increase with increased no. of workers. On the other hand stale synchronous maintains a nice balance between the performance and per epoch time.