Machine Learning for Large Datasets - Assignment 2

D Ravi Shankar (14169) [1]

1. Logistic Regression with L2 Regularization using SGD

Logistic Regression with L2 regularization is used. Parameters are updated using SGD. One hot encoding for words are used as features. The top 9999 words are considered for this and rest are discarded i.e each word is represented as one hot encoded vector of dimension 10000. If an unknown word occurs it is represented with 1 in $10000^{th} position$.

No.of epochs = 30.
Learning Rate = 1

Training Time = 949.4 sec
Testing Time = 47.08 sec
Training Accuracy = 76.81
Test Accuracy = 72.81

Different models with constant, increasing and decreasing learning rates are experimented and results are shown in the table. Exponential decay is applied to the learning rate. It is computed as:

$$decayedLearningRate = learningRate * beta^{x}$$

where x = globalRtep/decaySteps.
If beta is less than 1, then it is decayed learning rate and if beta is greater than 1, then it is increasing learning rate.

Learning Rate = 1					
	Decay Rate =0.6	Decay Rate =0.8	Constant Learning Rate = 1	Inr Rate =1.8	Inr Rate =2.5
Training Time	995.48	983.5	949.4	884.69	852.27
Testing Time	21.61	22.93	23.48	21.14	22.94
Training Accuracy	76.48	75.42	76.81	77.78	69.25
Test Accuracy	66.2	72.27	72.81	72.74	67.21

In figure 1, starting with learning rate as 1, learning is decayed and increased and loss is plotted. As learning rate is increased, the model reaches saturation quickly. But the accuracy decreases as shown in the table. This is because, as the no.of epochs are increased, the model will be near to a local minima and learning rate should be slow so that it reaches the minima.If the learning rate is high, it might move away from the local minima. So initially the learning rate should be high and as the training period increases, the learning should decrease.

Various learning rates are experimented and their losses are shown in figure 2.If the learning rate is less (example 0.01), the model takes lot of time to train. As learning rate is increased, the loss is saturated quickly but at the same time the accuracy also decreases, shown in figure 3. For this model, the optimal learning rate is found out to be 1.

2. Using TensorFlow

I have used TensorFlow for parallelization. Even though for distributed computations both spark and tensorflow are good, tensor-flow inherently parallelizes the computations. It will be faster than spark with respect to computations (matrix multiplications, etc). We cannot consider hadoop for iterative models because we have to run a map-reduce for each iteration which significantly slow downs the computations.

[1]Indian Institute of Science. Correspondence to: D Ravi Shankar <dravishankar7@gmail.com>.

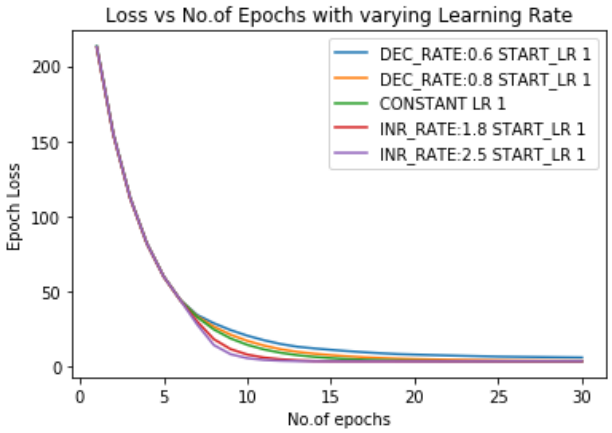

Figure 1. Loss as no.of epochs are increased for decaying and increasing learning rate with starting learning rate = 1.

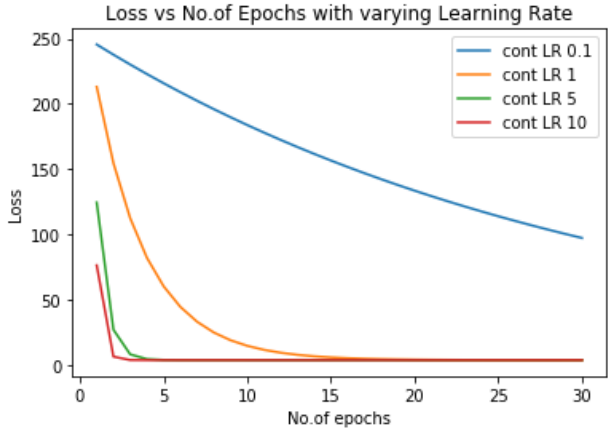

Figure 2. Loss as no.of epochs are increased for different learning rates (kept constant).

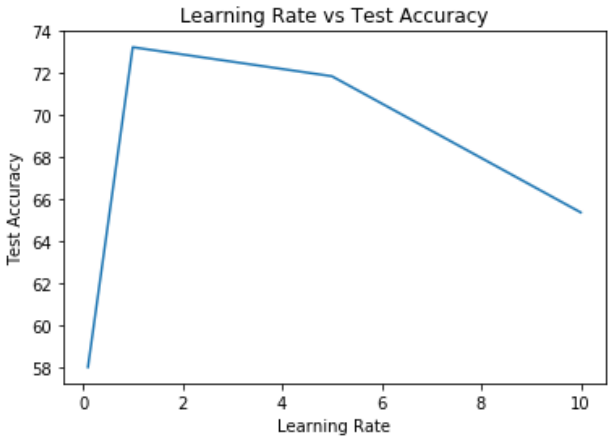

Figure 3. Test accuracy as learning rate is increased.

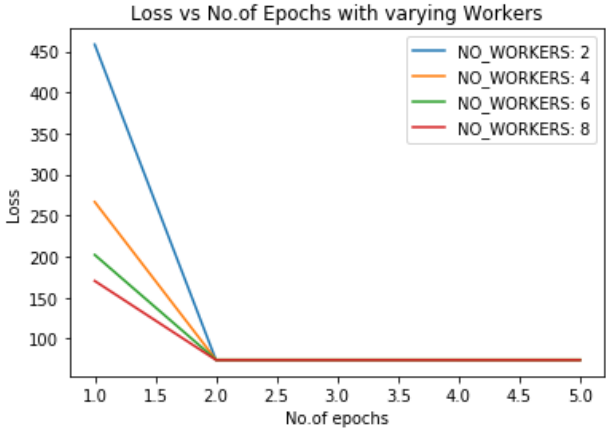

Figure 4. Test accuracy as learning rate is increased.

3. Logistic Regression with L2 Regularization using Asynchronous SGD

In Asynchronous SGD, as soon the worker computes the gradients, the respective parameters are updated without waiting for other worker nodes. Here the model is copied to each worker node and the gradients are computed batch wise by each worker node. As soon the worker computes the gradients, the weights and biases are updated immediately. Because of this the model reaches the saturation level quickly (within 2-3 epochs).

No.of Workers = 4
No.of Epochs = 5

Training Time = 346
Testing Time = 23.48
Training Accuracy = 78.45
Testing Accuracy = 74.36

No.of workers are varied and the corresponding loss is plotted as shown in figure 4.After 1st epoch itself, the loss is reduced as we increase number of workers.This because as no.of workers are increased, the no.of updates made increases and thus moves the model closer to the minima much faster.