International Journal of Science and Research (IJSR)
ISSN: 2319-7064
ResearchGate Impact Factor (2018): 0.28 | SJIF (2019): 7.583

Effect of Local Dynamic Learning Rate Adaptation on Mini-Batch Gradient Descent for Training Deep Learning Models with Back Propagation Algorithm

Joyce K. Ndauka[1], Dr. Zheng Xiao Yan[2]

[1]Postgraduate Student at the Tianjin University of Technology and Education, China
[2]Tianjin University of Technology and Education, China

Abstract: ***Back propagation has gained much popularity for training neural network models including deep learning models. Despite its popularity, ordinary back propagation suffers from low convergence rate due to existence of constant learning rate. Since error surface is not smooth, the learning rate needs to be dynamically adapted to speed up the rate of convergence. Much work has been done showing benefits of employing dynamic learning rate on back propagation algorithm to speed up the rate of convergence focusing on global learning rate adaptation and local adaptive learning rate on batch gradient descent. In this work we explore the effect of local dynamic learning rate adaptation using improved iRprop- algorithm with mini batch gradient descent to improve convergence rate of back propagation. Experiment was conducted in python using cifar 10 dataset. Results show that the proposed algorithm outperforms the ordinary back propagation algorithm in terms of speed when the batch size is large.***

Keywords: Mini batch gradient descent, Learning rate, Back propagation, Deep leaning

1. Introduction

Deep learning

Deep learning is an artificial neural network with many stacks of layer which nonlinearly extracts complex features representation of data set. It is inspired by the working principle of the brain by extracting potential features representation from the data to make decision [1].

Deep learning has shown great improvement to most of complex applications on machine learning such as computer vision object recognition, speech recognition, speech and image features coding, information retrieval, analysis of molecules to discover drugs and audio processing.

In deep learning information passes from the first layer called input layer then neurons make some guess and pass information to next layers. The process continues until it reaches last layer known as output layer. On each layer the output of the previous layer becomes input to the next layer and layers between the input and output are called hidden layers.

Back propagation

Back propagation is common technique used for training deep learning models. It uses gradient descent optimization method to update weights on direction of steepest descent during training process. The step size taken while moving down the slope are known as learning rate [2].

There are three categories of gradient: Stochastic Gradient Descent, Batch Gradient Descent and Mini-batch Gradient Descent. Mini-batch gradient descent takes advantage of less computational cost and memory resource required compared to other invariants. During training with back propagation, step size (learning rate) controls the extent to which connection weights are adjusted during back propagation algorithm.

Learning rate has also great influence on the convergence speed and accuracy of back propagation algorithm. If the learning rate is large the convergence speed is large, but it can cause divergence. Likewise, if the learning rate is small the algorithm can take a long time to converge. Nonetheless, although it can locally guarantee convergence, it can be a poorly local minimal. Therefore, training deep learning with constant learning rate is found to be trivial to most of deep learning models.

Various dynamically adaptive learning rate techniques have been developed to improve the convergence rate of back propagation algorithm [3]. This has been mostly done through global learning rate in which single learning rate is used to adjust all weight parameters. Only a few studies such as [2], [3] and [4]have used local adaptive learning rate where each weight parameter has a separate learning rate with batch gradient descent [4].Consequently, local adaptive learning rate proved to be superior to global adaptive learning rate but with batch gradient descent.

This study, therefore, is aimed at narrowing the gap in improving back propagation algorithm literature by focusing on exploring the effect of using local dynamic learning rate adaptation on mini batch gradient decent to train deep learning models with back propagation algorithm. To ensure effective learning, data preprocessing technique was implemented as an important initial step to improve the quality of data for effective learning.

The rest of the paper is organized as follows: Section 2 provides an overview of the related literature. Section 3 describes the methods and materials used. Section 4 presents the experimental results and Section 5 concludes.

2. Related Work

In gradient descent-based algorithm determination of proper learning rate is one of major problem for training neural network based model. Various dynamically adaptive

learning rate such as AdaError eliminate the problem of tuning learning rate manual so as to find proper learning rate [5]

A lot of research on optimization of back propagation algorithm in case of convergence speed have been conducted and various optimization algorithms have been proposed, ranging from global dynamic learning rate such as exponential decay to local dynamic learning [6].Previous proposed methods have high convergence speed but results from one classification task cannot be compared to other classification tasks.

Barnard and Holm [7] describe deferent approaches to optimize back propagation algorithm such adaptive step size, Leap Frog and Conjugate gradient. Evaluation was done on classification task of three-color based features to distinguish three classes of animals. Results show that training speed was higher on gradient descent with adaptive step size and a need to research more on generalization ability of leaf frog was recommended in this work.

Smith [6] proposed cyclical learning rates for deep learning. The author described cyclical learning rate method which works as global adaptation of learning rate cyclical within certain reasonable bound during training to avoid the need of tuning optimal learning rate. The work was done by using cifar-10 dataset and cifar-100 with mini batches of 100 datasets. In this work author explained local adaptive learning rate to be superior to global adaptive learning rate.

Florescu and Igel [8] implemented Rprop invariant on tensor Flow framework. The experiment was conducted using Human Activity recognition using smartphone dataset. Results showed that on batch gradient descent iRprop+ performed better compared to mini-batch gradient descent [8]. The authors did not provide enough explanation about the size of batch used to implement the model, whereas batch size is also important parameter to determine accuracy of the model [9].

3. Method and Materials

3.1. Network structure

In this study cifar 10 dataset where trained on deep learning model of full connected layer with three hidden layers. The first hidden layer has 2048 number of neurons, second 1024 third 512 and the last(output layer)has10 number of neurons. All were activated by rectifier unit (ReLu) except the output layer that was activated by Softmax that represents probability distribution of images from each class label. Figure 1 shows full connected network architecture having three hidden layers with respective number of neurons.

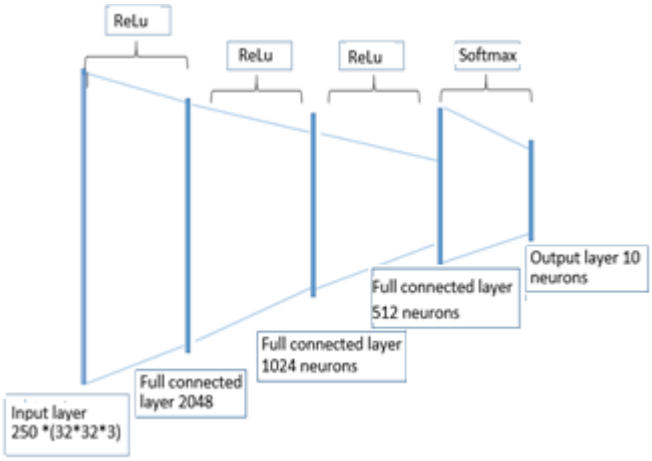
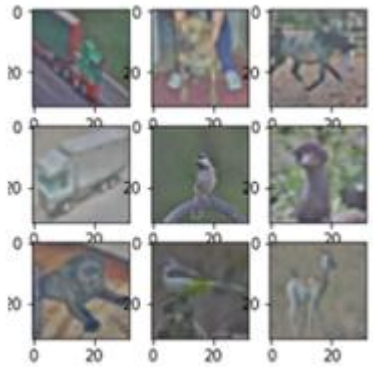


Figure 1: Feed forward neural network

3.2. Dataset

Full connected layer was trained with 10000 cifar 10 dataset having 10 data labels. Before training the model with proposed algorithm, data set were preprocessed in order to ensure our model does not mislead due to data inequality. Zero-phase Component Analysis (ZCA) whitening techniques were used to normalize and enhance the quality of data. Figure 2 shows some of cifar 10 image dataset. The left panel of Figure 2 shows un-processed dataset while the right panel show processed dataset.

1. Un-preprocessed dataset



2. Preprocessed dataset:

3.3. Weight initialization

Assigning initial values of connecting weights has great impact of improving convergence speed of deep learning model. In this work, Kaiming normal distribution was used to initialize weight randomly with mean of 0 and variance given by:

$V^2 = 2/(inputsize)$

3.4. Batch Normalization

Batch Normalization was used to avoid exploding of the gradient descent in the hidden layers hence improved the efficiency of our model.

3.5. The proposed algorithm

The ordinary Back propagation algorithm for training deep learning model was modified with dynamic changing learning rate to improve convergence speed. The adaptation considered the change on the sign of the gradient where If no change on sign of gradient for each individual weight indicates that it was on same surface plane of the slope then its learning step size was increased, Else if sign of gradient for each individual weight change it indicates that it was on different surface plane of the slope then its learning step size was decreased and if gradient value is equal to zero then learning step size will not be change.

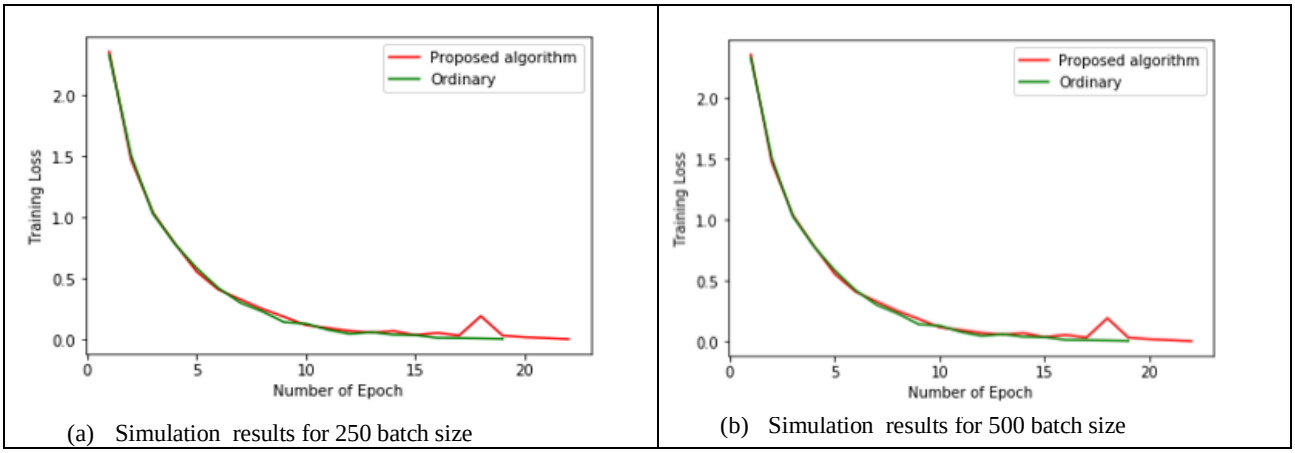
Algorithm: Improved iRprop- algorithm

Set $\Delta_{ij}(0) = small\ value$
Set previous loss= large value
For each batch
If $\frac{\partial E(t)}{\partial w_{ik}} * \frac{\partial E(t-1)}{\partial w_{ik}} > 0$
$\Delta_{ij}\left(tw_{ij}(t+1)\right) = \min\left(\Delta_{ij}(t-1) * \mu, \max\right)$
$\Delta w_{ik}(t) = -sign\left(\frac{\partial E(t)}{\partial w_{ik}}\right) * \Delta_{ij}(t)$
$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ik}(t)$
$\frac{\partial E(t-1)}{\partial w_{ik}} = \frac{\partial E(t)}{\partial w_{ik}}$
Else if $\frac{\partial E(t)}{\partial w_{ik}} * \frac{\partial E(t-1)}{\partial w_{ik}} < 0$
$\Delta_{ij}(t) = \max\left(\Delta_{ij}(t-1) * e, \min\right)$
$\Delta w_{ik}(t) = -sign\left(\frac{\partial E(t)}{\partial w_{ik}}\right) * \Delta_{ij}(t)$
$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ik}(t)$
$\frac{\partial E(t-1)}{\partial w_{ik}} = 0$ *0 to allow changed learning rate to take effect on next learning iteration*
Else
$\Delta w_{ik}(t) = -sign\left(\frac{\partial E(t)}{\partial w_{ik}} * \Delta_{ij}(t)\right)$
$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ik}(t)$
$\frac{\partial E(t-1)}{\partial w_{ik}} = \frac{\partial E(t)}{\partial w_{ik}}$

The process repeated until the error cost was less than threshold value of 0.01. The value of $\mu$was set to 1.2 and $e$to 0.5. The learning step was set on limit; min and max values 0.001 to 0.006 respectively to avoid oscillation when the value grew higher.

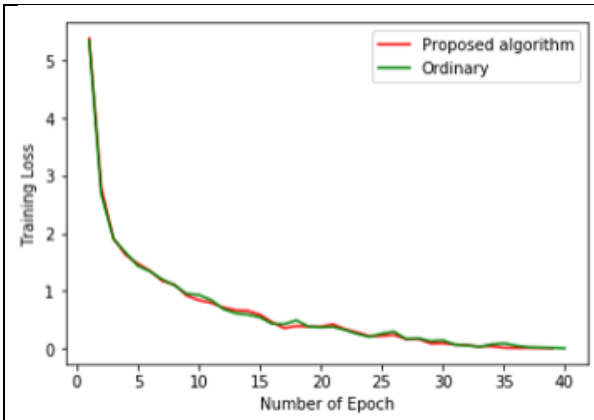
4. Experimental Results

The trained data were divided into mini batch of 250, 500, 1000 and 2000 and fed into the model batch by batch. The simulation results were programmed in Python using personal computer without GPU. To verify effectiveness of proposed algorithm, comparison was made to ordinary back propagation algorithm in terms of number of epoch. Figure 3 shows comparison between proposed algorithm and ordinary back propagation algorithm on each batch size.

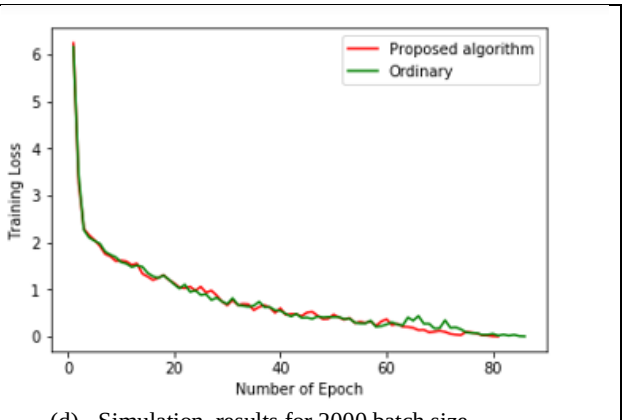


(a) Simulation results for 250 batch size

(b) Simulation results for 500 batch size



(c) Simulation results for 1000 batch size



(d) Simulation results for 2000 batch size

Result shows that for large batch size iRprop- local dynamic learning rate proved to be superior to ordinary back propagation. That is opposite to small batch size where local dynamic learning shows to have small convergence speed compare to ordinary back propagation but has poor generalization ability.

5. Conclusion

The work aimed at exploring the effect of implementing independent dynamic learning rate on each weight parameter with batch gradient descent on back propagation algorithm. The algorithm was implemented only with full connected layer and it was observed that the proposed algorithm can converge faster compared to ordinary back propagation algorithm when the batch size is large enough.

References

[1] N. . Lewis, *Deep Learning from Data Made Easy with R: A Gentle Introduction for Data Science*. 2016.
[2] D. P. Mandic and J. A. Chambers, "Towards the optimal learning rate for back propagation," *Neural Process. Lett.*, vol. 11, no. 1, pp. 1–5, 2000, doi: 10.1023/A:1009686825582.
[3] E. D. Di Claudio, R. Parisi, and G. Orlandi, "LS-Back propagation Algorithm for Training Multilayer Perceptrons," *Icann '93*, pp. 768–771, 1993, doi: 10.1007/978-1-4471-2063-6_213.
[4] Z. Zainuddin, N. Mahat, and Y. A. Hassan, "Improving the Convergence of the Back propagation Algorithm Using Local Adaptive Techniques," *World Acad. Sci. ...*, no. 4, pp. 79–82, 2005.
[5] D. Li *et al.*, "AdaError: An Adaptive Learning Rate Method for Matrix Approximation-based Collaborative Filtering ACM Reference Format," vol. 11, 2018, doi: 10.1145/3178876.3186155.
[6] L. N. Smith, "Cyclical learning rates for training neural networks," *Proc. - 2017 IEEE Winter Conf. Appl. Comput. Vision, WACV 2017*, no. April, pp. 464–472, 2017, doi: 10.1109/WACV.2017.58.
[7] E. Barnard and J. E. W. Holm, "A comparative study of optimization techniques for back propagation," vol. 6, pp. 19–30, 1994.
[8] C. Florescu and C. Igel, "R ESILIENT B ACKPROPAGATION (R PROP) LEARNING IN T ENSOR F LOW FOR B ATCH -," no. 1997, pp. 1–5, 2018.
[9] E. Hoffer, I. Hubara, and D. Soudry, "Train longer, generalize better: Closing the generalization gap in large batch training of neural networks," *Adv. Neural Inf. Process. Syst.*, vol. 2017-December, pp. 1732–1742, 2017.