2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS) 978-1-7281-4862-5/20/$31.00 ©2020 IEEE 10.1109/SCEECS48394.2020.94

COMPARISON OF VARIOUS LEARNING RATE SCHEDULING TECHNIQUES ON CONVOLUTIONAL NEURAL NETWORK

Jinia Konar
C.S.E Department
N.I.T Bhopal
Bhopal, India
konar.riku@gmail.com

Prerit Khandelwal
C.S.E Department
N.I.T Bhopal
Bhopal, India
preritkhandalwal@gmail.com

Rishabh Tripathi
C.S.E Department
N.I.T Bhopal
Bhopal, India
imristri@gmail.com

***Abstract*—The learning rate is a hyperparameter which determines how much the model should change concerning the error each time the model parameters are updated. It is important to tune the learning rate properly because if it is set too low, our model will converge very slowly and if set too high, our model may diverge from the optimal error point. Some conventional learning rate tuning techniques include constant learning rate, step decay, cyclical learning rate and many more. In this paper, we have implemented some of these techniques and compared the model performances gained using these techniques.**

***Index Terms*—Learning Rate, Convolutional Neural Network (CNN), Constant learning rate, Step Decay learning rate, Exponential Decay learning rate, Differential learning rate, Cyclical learning rate, Stochastic Gradient Descent with Warm Restarts (SGDR).**

I. INTRODUCTION

A Deep Neural Network (DNN) is an Artificial Neural Network (ANN) which consists of multiple neural layers between its input and output layers. They are used in fields like image recognition, object detection, face recognition, speech recognition, etc. However, tuning its parameters and hyper-parameters is a challenging problem [1].

Hyper-parameters are settings that need to be tuned to steer any machine learning algorithm. They cannot be learned from regular training processes. Instead, it is a variable whose value is not estimated from the data given but explicitly defined by the model trainer and it is an important value in estimating model parameters [2]. They are used to control properties like time and space complexity of the model. These variables don't affect the performance of the model but affect the speed and quality of the training process.

While training a neural network, we seek to minimize the error between actual and predicted values. We define a loss function for a neural network which helps in improving its weights. During the training of the neural network, we try to reduce this loss function only. The loss function is calculated by calculating the difference between the target(actual) values and the predicted values. We use the gradient descent technique to decrease this loss function.

Mathematically, if the loss function is L(X; W,b), then our goal is to minimize L. Here, X is input to the neural network, W is the model parameter, and b is the bias value. The model weights are updated using the following equation:-

$$W_t = W_{t-1} - \alpha(\frac{\partial L}{\partial W}) \tag{1}$$

Here, α is called the learning rate which determines how quickly or slowly we want to update the parameters.

In other words, it determines how much the model should change concerning the error each time the model parameters are updated [3].

There are different ways to select the initial learning rate. A naive approach is to randomly select different values and choose the one which gives minimum loss without relinquishing speed of training. Another approach is to start training with a large learning rate, and then lower the value gradually using some mechanism.

Choosing an optimal learning rate is a challenging task because a too small learning rate, as shown in Fig. 1(a), may result in a very long and very slow training process which may get stuck, whereas a too-large learning rate value, as shown in Fig. 1(c), may result in diverging away from the optimal point rather than converging towards it [4].

In this paper, we are going to compare different learning rate techniques by applying them to convolutional neural network [5]. We will work on two different datasets: Cifar-10 and MNIST. In the end, we'll compare these techniques based on metrics like precision, recall, f1-score, and accuracy.

II. LEARNING RATE TUNING MECHANISMS

In this paper, we are considering 6 different techniques for setting the learning rate. They are described below:-

A. Constant Learning Rate

In the constant learning rate mechanism, we fix the learning rate of the model to be a fixed number. Choosing that fixed number is a difficult task but it can be determined through

978-1-7281-4862-5/20/$31.00 ©2020 IEEE

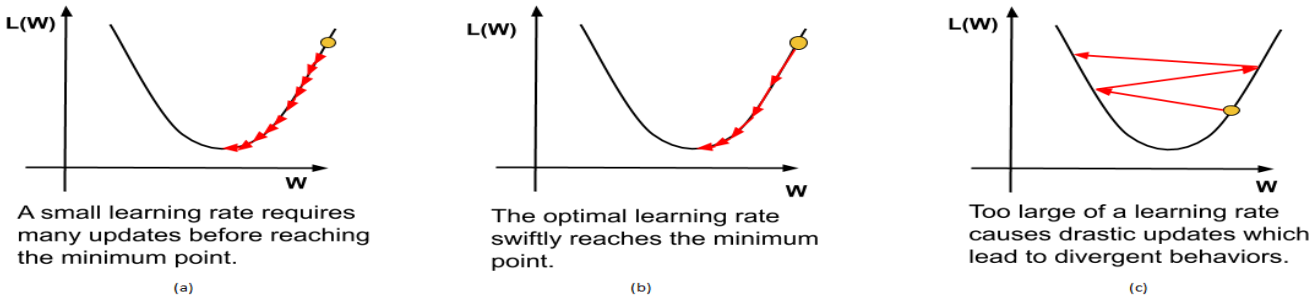

Fig. 1. Different scenarios with different learning rate values (a) When learning is too small, (b) When learning rate is optimal, (c) When learning rate is too large.

experiments [6]. Experiments have shown that starting the learning rate from 0.1 gives a relatively good performance.

If setting the learning rate to 0.1 does not give good accuracy then we choose another constant number based on experiments again.

B. Step Decay Learning Rate

Step decay learning rate is also known as learning rate annealing. In this technique, we start from a relatively high learning rate and then gradually decrease it during the training as shown in Fig. 2. The decrease is done after every few epochs.The mathematical form of step decay is:-

$$\alpha = \alpha_0 * drop^{floor(\frac{epoch}{epochs\_drop})} \tag{2}$$

Here, α is the learning rate of current time stamp, α_0 is the learning rate of previous time stamp, drop is the constant factor by which learning rate drops each time, epochs_drop is the number of epochs after which learning rate will drop, epoch is the number of epoch.

The reason behind doing this decrease is that initially, we would like to traverse quickly to a range of good parameter values and after we achieve that we would like to traverse slowly so that we can explore deeper and narrower parts also of the loss function. It means that we initially take higher jumps and as we approach the minima point, we make our jumps smaller so that we don't get diverged from the minima point.

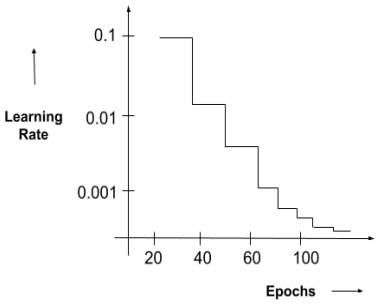

Fig. 2. Step Decay Schedule

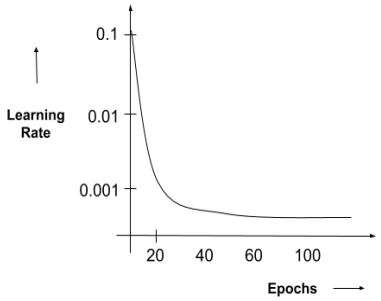

Fig. 3. Exponential Decay Schedule

C. Exponential Decay Learning Rate

It is similar to step decay learning rate mechanism. But here we decrease the learning rate exponentially as shown in Fig. 3 [7]. The learning rate is changed by the following mathematical equation:-

$$\alpha = \alpha_0 * e^{-kt} \tag{3}$$

Here, α = Learning rate of current timestamp, α_0 = Learning rate of previous timestamp, e = Napier's constant, k = Constant factor, t = Time stamp

D. Differential Learning Rate

The differential learning rate is a technique to make transfer learning faster and more efficient. Here, the learning rate for the layers which are in the starting of the model is kept low and then gradually increased for the further layers.

The layers closer to the input layer mostly learns the general features and hence they needn't be changed much, so, we keep learning rate for these layers very low and then moderately increase the learning rate as we go deeper in the network as shown in Fig. 4.

E. Cyclical Learning Rate

In the cyclical learning rate mechanism, we vary the learning rate between two boundary values i.e. upper bound and lower bound [8]. The learning rate is updated generally by the triangular update rule but we can also use exponential or sinusoidal update rule. The triangular update rule is shown in Fig. 5.

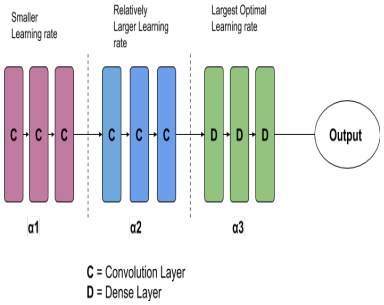

Fig. 4. Differential Learning Rate Schedule

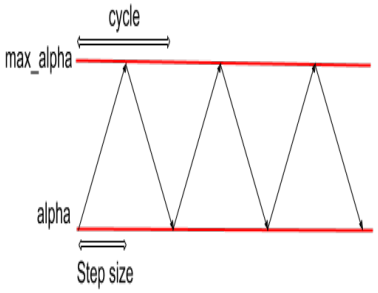

Fig. 5. Triangular Schedule

The upper bound and lower bound values are calculated by the LR range test. In the LR range test, we plot a curve between loss and learning rate. The range of the learning rate where we get a smooth decline in the curve is chosen as the optimal range for varying the learning rate. The learning rate is varied in the optimal range using the following equations:-

$$\eta_t = \eta_{min} + (\eta_{max} - \eta_{min})(max(0, 1 - x)) \quad (4)$$

Where x is defined as,

$$x = abs(\frac{iterations}{stepsize} - 2 * (cycle) + 1) \quad (5)$$

and cycle can be calculated as,

$$cycle = floor(\frac{1 + iterations}{2(stepsize)}) \quad (6)$$

Here, η_{min} and η_{max} = the bounds of our learning rate, iterations = the number of completed mini-batches, stepsize = one half of a cycle length, $1 - x$ = always a positive value.

The benefit of using the cyclical learning rate over other learning rate techniques is that other mechanisms don't give assurity that the model will not get stuck in a plateau region because we might not have that value of learning rate which can take it away from that region. But in the case of cyclical learning rate, if we got stuck to such a place then since we're increasing the learning rate in a cycle, this will take us out of the plateau region. And since we are decreasing the learning rate in the other half of a cycle, we can also explore the narrower parts of the loss function. So in this way cyclical learning rate converges more quickly.

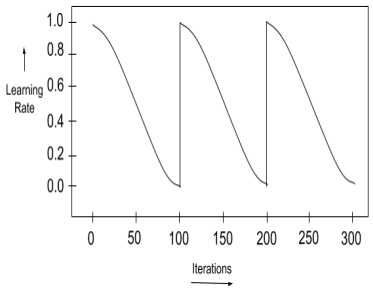

Fig. 6. Learning Rate Schedule with Restarts

F. Stochastic Gradient Descent with Warm Restarts

This technique is an aggressive annealing schedule. In this technique, we vary the learning rate like a cosine function and after certain number of epochs, we restart the learning rate from the initial learning rate as shown in Fig. 6. The word "warm" is used because each time the model weights learned are used as initial weights in the next restart instead of starting from the scratch [9].The schedule can be written using the following equations:-

$$a_t = a^i_{min} + \frac{1}{2}(a^i_{max} - a^i_{min})(1 + cos(\frac{T_{current}}{T_i} * \pi)) \quad (7)$$

Here, a_t = the learning rate at timestep t, a_{max} and a_{min} = the range of desired learning rates, $T_{current}$ = the number of epochs since the last restart, and T_i = the number of epochs in an cycle.

III. EXPERIMENTS

We applied each of these techniques on a convolutional neural network (whose architecture is described ahead) using two datasets: Cifar-10 and MNIST.

A. Datasets

a) Cifar-10: The Cifar-10 dataset has 10 mutually exclusive classes (like an airplane, automobile, birds, cats, etc). Each class consists of 6,000 colored images of size 32x32 making a total of 60,000 images. Among these 60,000 images, 50,000 images belong to the train set and the rest 10,000 belongs to the test set.

The train set and test set are divided into 5 batches. A train batch consists of 10,000 images whereas a test batch consists of 2,000 images.

b) MNIST: The MNIST dataset consists of 70,000 greyscale images of handwritten digits. Each image is of size 28x28. Among these 70,000 images, 60,000 belong to the train set and the rest 10,000 belongs to the test set.

The dataset consists of 10 classes i.e. images of digits from 0-9. Each class consists of 6,000 and 1,000 images in their train and test set respectively.

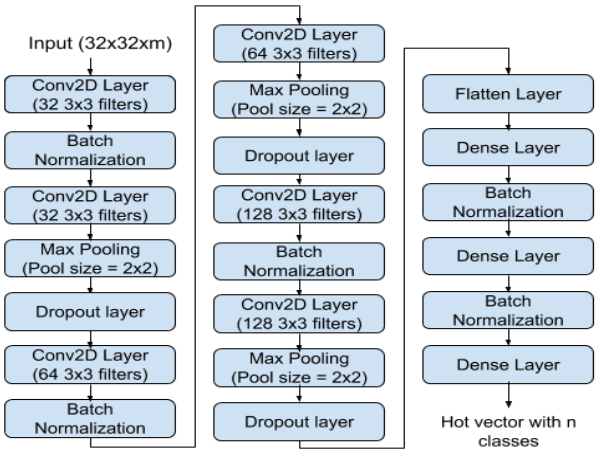

Fig. 7. CNN architecture used

B. Architecture of Convolutional Network used

We have trained the model using convolutional neural network architecture (CNN). This architecture uses 6 convolutional layer which uses 'relu' activation function. Each convolutional layer uses filters of size 3x3. After each convolutional layer, we have used the Max Pooling layer which has a pool size of 2x2. We have used Dropout layers to drop some neurons each time to avoid overfitting. At last, we have dense layers that use sigmoid activation function and a hot vector of size equal to the number of classes in the dataset is produced as shown in Fig 7.

C. Results & Discussion

After implementing the model using different learning rates on two datasets, we have obtained the following results. The results for the Cifar-10 dataset are described in Table I and for MNIST dataset is described in Table II.

TABLE I
PERFORMANCE METRICS OF DIFFERENT LEARNING RATES ON CIFAR-10

Learning Rate Used	*Weighted Average Precision*	*Weighted Average Recall*	*Weighted Average F1-Score*	*Accuracy*
Constant lr	0.85	0.84	0.84	0.8822
Step Decay lr	0.84	0.84	0.84	0.8686
Exponential lr	0.79	0.79	0.79	0.7773
Differential lr	0.76	0.75	0.75	0.7673
Cyclical lr	0.88	0.89	0.89	0.8996
Warm Restarts	0.77	0.76	0.76	0.7628

Here we have compared different learning rates on the basis of precision, recall, f-score and accuracy [10]. These values are calculated using the equations described below:-

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \quad (8)$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \quad (9)$$

TABLE II
PERFORMANCE METRICS OF DIFFERENT LEARNING RATES ON MNIST

Learning Rate Used	*Weighted Average Precision*	*Weighted Average Recall*	*Weighted Average F1-Score*	*Accuracy*
Constant lr	0.98	0.98	0.98	0.9516
Step Decay lr	0.99	0.99	0.99	0.9551
Exponential lr	0.99	0.99	0.99	0.9642
Differential lr	0.99	0.99	0.99	0.9705
Cyclical lr	0.99	0.99	0.99	0.9940
Warm Restarts	0.99	0.99	0.99	0.9789

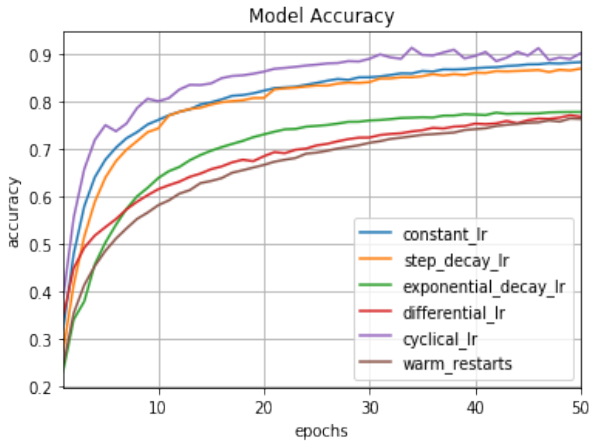

Fig. 8. Accuracy vs Epochs for learning rates on Cifar-10

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (10)$$

$$Accuracy = \frac{TruePositive + TrueNegative}{TotalNumberofExamples} \quad (11)$$

Here, True Positive = Condition detected when the condition exists, True Negative = Condition not detected and the condition doesn't exist, False Positive = Condition detected

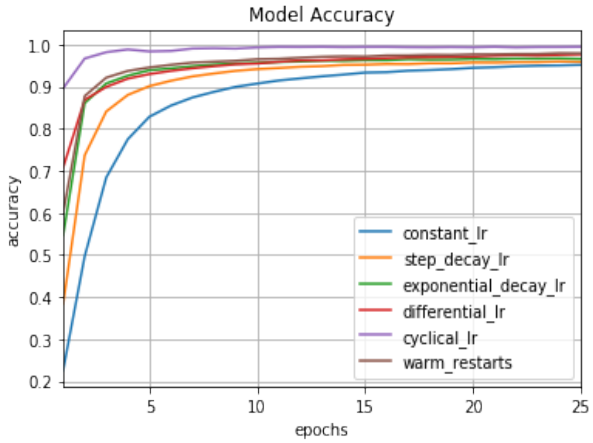

Fig. 9. Accuracy vs Epochs for learning rates on MNIST

but the condition doesn't exist, False Negative = Condition not detected when the condition exists.

In Table I, we see that cyclical learning produced the best accuracy and after that, the constant learning rate technique was in the second position. Then we have step decay at third position and the remaining techniques did not give that much good accuracy like the above three techniques. The warm restarts technique gave the worst accuracy among these.

In Table II, we see that the cyclical learning rate again gave the best accuracy. But this time the positions of other techniques changed. Warm restarts technique gave the second-best accuracy and continuing in this list, we find the constant learning rate at the end.

If we look into the graphs, cyclical learning rates produced good accuracy at very fewer epochs only when compared to others. For instance, if we look into Fig. 8, it converged at about 30 epochs for the Cifar-10 dataset whereas others took much more iterations. And if we look into Fig. 9, for the MNIST dataset, it converged at about 10 epochs only which is much faster than others.

Here, after analyzing the results from the above tables, we can see that the cyclical learning rate produced the best accuracy among all the other techniques. The cyclical learning rate mechanism turns out to be a trustworthy technique in comparison to rest. For the Cifar10 dataset, it gave an accuracy of 89.96% and for the MNIST dataset, the accuracy gained was 99.40% which is highest among all others.

IV. CONCLUSION & FUTURE SCOPE

After studying the results, we concluded that the cyclical learning rate performs better than the rest of the other 5 learning rates, with the datasets and CNN model architecture used here. The cyclical learning technique produced this accuracy at a much lesser epoch than the other techniques. So it is time and space-efficient than others. Its time complexity is highly less than the other techniques.

In the future, the accuracy of the model can be increased by properly tuning other hyper-parameters (like dropout, the number of neurons in a layer, etc) also. Other model architectures can be also used which may produce better results than the above-used architecture.

REFERENCES

[1] L. L. Lima, J. R. Ferreira and M. C. Oliveira, "Efficient Hyperparameter Optimization of Convolutional Neural Networks on Classification of Early Pulmonary Nodules," 2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS), Cordoba, Spain, 2019, pp. 144-149.

[2] S. Khandelwal, S. Rana, K. Pandey and P. Kaushik, "Analysis of Hyperparameter Tuning in Neural Style Transfer," 2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC), Solan Himachal Pradesh, India, 2018, pp. 36-41.

[3] N. Kamiyama, H. Izumi, N. Iijima, H. Mitsui, M. Yoshida and M. Sone, "Tuning of learning rate and momentum on backpropagation," IJCNN-91-Seattle International Joint Conference on Neural Networks, Seattle, WA, USA, 1991, pp. 963 vol.2-.

[4] S. Roy, "Factors influencing the choice of a learning rate for a back-propagation neural network," Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94), Orlando, FL, USA, 1994, pp. 503-507 vol.1.

[5] T. Guo, J. Dong, H. Li and Y. Gao, "Simple convolutional neural network on image classification," 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)(, Beijing, 2017, pp. 721-724.

[6] S. C. Ng, S. H. Leung and A. Luk, "Convergence of the generalized back-propagation algorithm with constant learning rates," 1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36227), Anchorage, AK, 1998, pp. 1090-1094 vol.2.

[7] W. An, H. Wang, Y. Zhang and Q. Dai, "Exponential decay sine wave learning rate for fast deep neural network training," 2017 IEEE Visual Communications and Image Processing (VCIP), St. Petersburg, FL, 2017, pp. 1-4.

[8] L. N. Smith, "Cyclical Learning Rates for Training Neural Networks," 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, 2017, pp. 464-472.

[9] P. Mishra and K. Sarawadekar, "Polynomial Learning Rate Policy with Warm Restart for Deep Neural Network," TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), Kochi, India, 2019, pp. 2087-2092.

[10] T. Villmann, M. Kaden, M. Lange, P. Stürmer and W. Hermann, "Precision-Recall-Optimization in Learning Vector Quantization Classifiers for Improved Medical Classification Systems," 2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), Orlando, FL, 2014, pp. 71-77.