T-LRA: Trend-based Learning Rate Annealing for Deep Neural Networks

Samira Pouyanfar and Shu-Ching Chen
School of Computing and Information Sciences
Florida International University
Miami, FL 33199, USA
Email: {spouy001, chens}@cs.fiu.edu

***Abstract*—As deep learning has been widespread in a wide range of applications, its training speed and convergence have become crucial. Among different hyperparameters existed in the gradient descent algorithm, the learning rate has an essential role in the learning procedure. This paper presents a new statistical algorithm for adapting the learning rate during the training process. The proposed T-LRA (trend-based learning rate annealing) algorithm is calculated based on the statistical trends seen in the previous training iterations. The proposed algorithm is computationally very cheap and applicable to online training for very deep networks and large datasets. This efficient, simple, and well-principled algorithm not only improves the deep learning results, but also speeds up the training convergence. Experimental results on a multimedia dataset and deep learning networks demonstrate the effectiveness and efficiency of the proposed algorithm.**

1. Introduction

Multimedia big data provides precious insights into many real-world problems [1], [2], [3], [4], [5]. Nowadays, machine learning and data mining techniques have been significantly leveraged in different multimedia applications [6], [7], [8], [9], [10]. Among various learning approaches, deep learning, a buzzword of neural networks, has shown its supremacy in representing very complex data [11], [12].

Backpropagation (BP) is the classical algorithm to train feedforward neural networks and it has solved numerous problems such as image processing, video analysis, speech recognition, and text mining [13], [14], [15]. This simple algorithm consists of local computations which avoid the large storage need. Nevertheless, it still has difficulty to handle large datasets due to its slow convergence and high dependency to its hyperparameters, such as the learning rate and momentum.

In BP, there are two general versions of the gradient descent that can be used to minimize an objective (loss) function: batch and online [16]. In the batch version, all the training data is used to minimize the loss function; while in online training, the network weights are updated after each training pattern presentation. Stochastic Gradient Descent (SGD) is the online process of minimizing the loss function in BP [17]. In this technique, the learning rate plays a critical role to achieve faster training and higher performance.

Handling the irregularities in the shape of the multi-dimensional error functions is very difficult in the standard SGD algorithms with a fixed step size. In addition, it is very common that the search procedure traps in the regions with local minima and stops there rather than continuing toward a global minimum. Although several studies suggest to use local minima since they are easy to find in larger networks [18], it is still possible to reach to the global optima while decreasing the convergence time on deep networks. Learning rate is an essential and key factor to avoid trapping in such local minima and adapting the learning rate during the training process has shown promising results on many problems. However, conducting an algorithm to automatically adjust this parameter is still an open issue [19]. Generally, there are two methods (time-based and drop-based [20]) to schedule the learning rate parameters which are also known as learning rate annealing, i.e., decreasing the learning rate based on a predefined threshold. In the first technique, the learning rate is incrementally reduced based on the training epoch using a fixed decay parameter. Another schedule algorithm is drop-based, which systematically drops the learning rate in every fixed number of epochs. However, the question is how to determine the number of epochs to drop the learning rate. This is another hyperparameter that must be initialized before the learning process starts.

In this paper, an automatic drop-based learning rate scheduling is proposed to improve the SGD algorithm in deep learning. This work alleviates the task of selecting an appropriate learning rate by analyzing the statistical trends of the training process in an online manner. It automatically decides when to drop the learning rate based on the losses in the previous training iterations. The trivial computational costs of the trend analysis is ignorable compared to the gradient descent computation. The proposed T-LRA (trend-based learning rate annealing) algorithm is applied to the Convolutional Neural Networks (CNNs), a popular deep learning algorithm, on a classification task to evaluate its effectiveness. To our best knowledge, this is the first paper that schedules the learning rate using statistical trend analysis. The advantages of this approach are: 1) automatic setting of a learning rate based on the previous training trends; 2) negligible computation over gradient descent; and 3) applicable to the deep neural networks and large datasets.

The remaining of the paper is organized as follows. In section 2, the existing work in the learning rate adaptation and statistical trend analysis is discussed. Section 3 presents the proposed learning rate annealing for deep neural networks. In section 4, experimental results and observations are provided. Finally, the paper is concluded in section 5.

2. Related Work

2.1. Learning Rate Adaptation for Neural Networks

Artificial neural networks have been widely used in a wide range of applications. Specifically in recent years, it has been extended to a successful learning technique called deep learning, which has shown its strength in dealing with real world problems [11], [21]. Till now, several algorithms have been proposed to train neural networks and minimize the loss functions, including Stochastic Gradient Descent (SGD) [22], AdaDelta [19], AdaGrad [23], Adam [24], to name a few. In a gradient descent process, selecting the appropriate parameters such as the learning rate (step size) is crucial for a better and faster learning. It also needs expert knowledge and differs for each problem. LeCun et al. [25] proposed an online estimation of principal eigenvalues and eigenvectors of the loss function's Hessian or second derivative matrix. Generally, the optimal learning rate can be selected as the inverse of the largest eigenvalue of the Hessian matrix. However, since computing the Hessian matrix for large learning algorithms (e.g., backpropagation) with thousands of parameters is computationally expensive, an online version of this algorithm was proposed by LeCun et al. In another work [26], the learning rate is changed in each epoch based on the weights and gradient values of the previous epoch in order to minimize the loss function. This method is inspired by the learning rate adaptation proposed in [27], which derives two-point step sizes approximation to the secant equation.

AdaGrad [23] has shown promising results on the large learning tasks. This method utilizes the first order information but relies on some second order features and annealing. In this method, small gradients have large learning rates and vise versa. However, it is very sensitive to the initial conditions and network hyperparameters. AdaDelta [19], an extension of AdaGrad, is a dynamic method adapting the learning rate over time using only the first order information. This approach overcomes some of the AdaGrad problems such as continual decay of the learning rate during the training process and the need for manually selecting the global learning rate. Adam [24] is another algorithm for the first-order optimization of gradient descent. It combines the advantage of AdaGrad and RMSProp [28], an optimization for online and non-stationary environments. Currently, Adam has been applied in popular deep learning architectures [21] and has shown its effectiveness.

However, all these methods somehow modify the learning algorithm, which requires extra computation per iteration over the gradient descent process. Due to such an increase in the computational costs, they may not be applicable for large scale datasets and deep neural networks.

Annealing is a simple and powerful learning rate adaptation which has shown its effectiveness in many learning algorithms such as traditional neural networks and current deep neural networks (e.g., CNNs) [11], [21]. A simple annealing schedule is called "search-then-converge" [29] which keeps the learning rate constant for a fixed amount of time T, and then starts to anneal it slowly. However, there is a new free parameter T that should be determined by experiments.

In this paper, a novel learning rate annealing is proposed, which is much faster on big data problems. In addition, the learning rate is updated only if the statistics from the previous epochs are not significantly changed, at the same time, the continual decrease of the learning rate is avoided. Finally, it does not need any manual setting of the learning rate. It is worth mentioning that this algorithm can be run simultaneously with an SGD algorithm (e.g., with the CNN algorithm) and updates the learning rate in an online manner without extra computation.

2.2. Statistical Trend Analysis

Trend analysis is a statistical process to evaluate the data collected over time or to analyze the relationship between two quantitative variables [30]. It is usually used as either a regression analysis or analysis of variance (ANOVA) [31]. Statistical trend analysis has been widely applied on different applications, especially environmental changes [32], [33]. A trend analysis of the insured damages caused by the natural disasters is presented in [34]. In another work, a trend analysis is used for remote sensing phenology which needs a long period temporal observations [32]. In that work, the slope of the linear regression line which is fitted to the phrenologic metrics is tested using a t-test to find significant differences from zero.

Mann-Kendall, a non-parametric test for trend analysis in time series, has been widely applied to various applications including water quality, rain fall, meteorology, etc. [35], [36], [37]. Gocic et al. [36] leverage Mann-Kendall together with another non-parametric method called Sen's slope estimator [38] to determine the significant trends in weather data. Another simple trend analysis algorithm is sign tests presented by Cox and Stuart [39], which can be used in situations that simplicity and speed are important. This non-parametric method is less sensitive to outliers, compared to the parametric regression algorithms. It is used for climate change analysis in [40] to analyze the impact of changes in the radiation patterns and temperature.

In this paper, the aforementioned trend analysis techniques are utilized to see if there is a significant change in the training curve, and the learning rate is reduced whenever the loss plateaus.

3. The Proposed Algorithm

In this paper, SGD is utilized for the training of deep learning networks and two well-known trend analysis techniques (i.e., the Mann-Kendall and Cox-Stuart tests) are leveraged for automatic learning rate adaptation as described in the following sections.

3.1. Stochastic Gradient Descent

Supervised learning is the most popular machine learning technique for either deep or shallow networks. In a supervised learning, an objective function measuring the error or distance between the actual and desired outputs is computed. The learning process includes adapting its internal parameters in order to minimize this error. Such adjustable parameters are also called "weights". Deep neural networks may contain millions of weights which need to be updated during the training process. In order to update the weights properly, in each iteration, a gradient vector is computed which measures the error when the weight is increased by a very small factor. Then, using the opposite direction of the gradient vector, the weights are updated. If the gradient vector is negative, the direction of the steepest descent takes the objective function to the average low output error or its minimal point [41].

SGD is one of the most common procedures used to minimize the objective function, especially for neural networks and deep learning. In overall, it consists of computing the outputs, errors, and the average gradient for a few input examples. Accordingly, the weights are updated based on such information. After repeating this process for many small training sets, it stops when the error or loss stops decreasing. This process is surprisingly fast compared to its batch version which employs all training examples in each iteration [25].

Generally, a neural network, deep or not, consists of an input layer $X = \{x_1, ..., x_i, ..., x_N\}$ including N input examples, the hidden layers containing K neurons $H = \{h_1, ..., h_k, ..., h_K\}$, and an output layer including M outputs $Y = \{y_1, ..., y_j, ..., y_M\}$. The neurons are connected to each other with w_{ik} or w'_{kj} which indicates the weights between the i^{th} input and the k^{th} hidden neuron or the k^{th} hidden neuron and the j^{th} output neuron. To simplify, the whole weights can be considered as the entries of a general weight matrix W. A simple version of this network is shown in Figure 1. An output of a neuron y_j is calculated using Equation (1) and the optimization function L is defined in Equation (2) [42].

$$y_j = f(\sum w'_{kj} * h_k); \quad (1)$$

$$L(W) = \frac{1}{N}\sum_{i=1}^{N} E(x_i) + \lambda r(W); \quad (2)$$

where f is the activation function (e.g., sigmoid, tanh, or ReLU) which produces the non-linearity in a neural network. E is the loss on data instance x_i (e.g., $E = 1/2\sum_{j=1}^{M}(y_j - t_j)^2$, where t_j is the actual output and y_j is the predicted one). The regularization term is $r(W)$ with the weight λ. Since the input size $|D|$ may be very large in practice, a stochastic approximation of the optimization function is used ($N \ll |D|$) as shown in Equation (2). The loss function E is computed in the forward pass of the backpropagation neural network, while its gradient ΔE_w is calculated in the backward pass. In particular, SGD updates the weights using a linear combination of the previous weight update V_t and the negative gradient $\Delta L(W_t)$ (given in Equation (4)), where W_t and W_{t+1} are the previous and updated weight matrices, respectively.

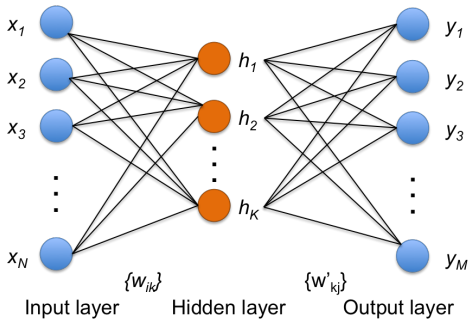

Figure 1. Neural network layers

$$V_{t+1} = \mu V_t - \alpha \Delta L(W_t); \quad (3)$$

$$W_{t+1} = W_t + (V_{t+1}). \quad (4)$$

There are two important hyperparameters in the weight update formula that need to be assigned carefully: (1) the learning rate α or the negative gradient weight, and (2) the momentum μ or the weight of the previous update. Regarding the learning rate which heavily affects the network performance in practice, it has been shown that the drop-based techniques are slightly preferable and efficient for deep neural networks [21], [22]. Therefore, an automatic and fast learning rate schedule is proposed in this paper, so that there is no need to manually select the number of training iterations in which the learning rate should be decreased.

3.2. Non-Parametric Trend Analysis

In a non-parametric trend test, no assumption of normality is required and a null hypothesis, H_0, is that the data population is identically distributed and comes from an independent population [43]. Since the errors in a deep neural network may not be normally distributed, two well-known non-parametric trend tests are utilized in this paper, which are described in the following sections. In a hypothesis test, the $p\text{-}value$ determines the statistical significance and plays a key role in interpreting the data statistics. To find a specific difference in an experiment, it is assumed that the null hypothesis H_0 is true. If the $p\text{-}value$ is small (less than a significant value φ), then the null hypothesis is rejected which shows a significant change in the data observations; while a large $p\text{-}value$ (greater than φ) indicates the acceptance of the H_0.

3.2.1. Mann-Kendall Trend Test. The Mann-Kendall is a popular non-parametric trend test commonly used to detect monotonic trends in series, especially for climate or environmental data [35], [36]. Suppose a set of observations are denoted as $\{x_1, x_2, ..., x_N\}$ ordered in time. The Mann-Kendall statistic is given as Equation (5) [33], [43]:

$$S = \sum_{k=1}^{N-1} \sum_{j=k+1}^{N} sgn(x_j - x_k) \tag{5}$$

where

$$sgn(x_j - x_k) = \begin{cases} 1, & if\ x_j - x_k > 0 \\ 0, & if\ x_j - x_k = 0 \\ -1, & if\ x_j - x_k < 0 \end{cases} \tag{6}$$

The mean of the Mann-Kendall statistic is $E[S] = 0$ and its variance σ^2 is calculated as:

$$\sigma^2 = \frac{N(N-1)(2N+5) - \sum_{j=1}^{q} t_j(t_j-1)(2t_j+5)}{18} \tag{7}$$

where the number of data points is denoted as N, the number of the tied groups in the data set is q, and the number of data points in the j^{th} tied group is t_j.

Finally, the test statistic Z is calculated using S and σ^2 (given in Equation (8)):

$$Z = \begin{cases} \frac{S-1}{\sigma}, & if\ S > 0 \\ 0, & if\ S = 0 \\ \frac{S+1}{\sigma}, & if\ S < 0 \end{cases} \tag{8}$$

Similar to other two-sided tests, the $p\text{-}value$ in the Mann-Kendall represents the probability of the error regarding the null hypothesis H_0. This probability shows whether there is no trend or a significant change in the time series. If Z is negative (positive) and the probability is greater than the level of significance, there is a decreasing (increasing) trend in the data series, otherwise there is no significant trend.

3.2.2. Cox-Stuart Trend Test. Another trend analysis utilized in this paper is a sign test proposed by Cox and Stuart [39]. This is a simple test for a monotonic (increasing or decreasing) trend analysis. Using the notions of the previous section, the data observations are paired as:

$$\{x_1, x_{c+1}\}, \{x_2, x_{c+2}\}, ..., \{x_{N-c}, x_N\}. \tag{9}$$

where $c = \frac{N}{2}$ if N is even and $c = \frac{N+1}{2}$ if N is odd. Then, a sign test is applied as follows. First, the differences between each pair is taken. In an increasing trend, it is expected most of the differences to be positive. On the contrary, a predominance of negative differences demonstrates a decreasing trend. Specifically, the Cox-Stuart test for $N > 30$ is calculated as [43]:

$$Z = \frac{\left|Sg - \frac{N}{6}\right|}{\sqrt{\frac{N}{12}}} \tag{10}$$

where the maximum number of the signs is denoted as Sg. Again, $p\text{-}value$ represents the probability of the error regrading the selected significance level.

3.3. Online Learning Rate Schedule

In a training process, the loss function could be subjected to gradual changes or decay. In the drop-based techniques, it is desirable to drop the learning rate after several iterations. To automatically determine the number of iterations, an online learning rate schedule is proposed using the non-parametric trend analysis techniques. Figure 2 presents the overall training algorithm. The input of this algorithm contains the training input X and initial weight matrix W which will be iteratively updated using the SGD algorithm as shown in line 5. There are three hyperparameters: the learning rate α, the decay factor θ, and a new hyperparameter φ which will be initialized in this algorithm. Decay is the factor to be used for learning rate annealing. Similar to other learning parameters, the initialization values (α_0 and θ_0) may be altered for different datasets. The φ is defined as the level of significance for the trend analysis. After initialization, the training is started and the network weight is updated in each iteration. Suppose the network is going to be trained for I iteration (e.g., 100,000). Thus, the whole learning process is divided into I' steps (e.g., 20 steps with each 5,000 iterations). In each iteration, SGD is employed on the data using the current weight matrix W and the learning rate value. Then the corresponding loss (error) is extracted (line 6 in Figure 2) as explained in Section 3.1. This process will be executed for $\frac{I}{I'}$ iterations. After that, a scheduler is employed to update the hyperparameters (α, θ, and φ) as shown in Figure 3. This algorithm illustrates the whole procedure of the proposed online learning rate annealing. First, a time-series object TS of the losses is created as shown in line 3 in Figure 3. This object is used as the input of the trend analysis function (e.g., Mann-Kendall or Cox-Stuart) in order to detect the trends in the losses curves. The learning rate is updated if there is no significant change in the losses or if the trend is positive, which means the losses are increasing. The significant level is defined by φ and is updated whenever the learning rate is decreased. If the loss plateaus, the learning rate is divided by θ. In addition, the decay factor θ is divided by 2 to reduce the learning rate decay over time. Similarly, the significance level φ is multiplied by 2 as the loss changes are reduced exponentially over time. In other words, the losses are reduced very fast in the early iterations so the $p\text{-}value$ should be very small (e.g., less than 0.05), while in the last iterations where the loss curve is going to be flatten, the $p\text{-}value$ threshold should be increased (e.g., 0.1).

4. Experiments

In this paper, the main focus is to show the functionality and effectiveness of the proposed Trend-Based Learning Rate Annealing (T-LRA) on the stat-of-the-art algorithms and large public datasets. For this purpose, a popular deep learning algorithm, called CNNs (Convolutional Neural Networks), is used and the experiments are conducted on a challenging multimedia task, namely concept and image classification. SGD algorithm is highly used in CNNs to

procedure DPLEARNING(X, W) ▷ Training the network
  $\alpha = \alpha_0$, $\theta = \theta_0$, $\varphi = \varphi_0$;
  for all iterations $i \in (1, \cdots, I')$ **do**
    for $j = 1$ to I/I' **do**
      $W = SGD(X, W, \alpha)$; ▷ Update the weight
      $L[j] = ExtractLoss(W)$;
    end for
    $LRScheduler(L, \alpha, \theta, \varphi)$;
  end for
end procedure

Figure 2. Training deep neural network

procedure LRSCHEDULER($L, \alpha, \theta, \varphi$) ▷ Schedule the learning rate
  $SampleSet = L$;
  $TS = TimeSeries(SampleSet)$;
  $Tr = MannKendall(TS)$; ▷ Tr=CoxStuart(TS)
  if $sign > 0$ or $Tr.p - value > \varphi$ **then**
    $\alpha = \frac{\alpha}{\theta}$;
    $\theta = \frac{\theta}{2}$;
    $\varphi = \varphi * 2$;
  end if
end procedure

Figure 3. The proposed online learning rate annealing algorithm

optimize the network and improve the losses. Therefore, the proposed T-LRA can automatically schedule the learning rate in CNNs.

4.1. Convolutional Neural Networks (CNNs)

The proposed T-LRA is a general learning rate scheduler that can be used for different classifiers and applications. Specifically, in this paper, it is applied on a successful CNN architecture called Network In Network (NIN) [44]. This architecture is selected as it has shown very promising performance on image recognition and classification, while it has a very simple and straightforward architecture. The difference between NIN and the conventional CNNs is the utilization of advanced micro neural networks to abstract the data within the receptive field. In overall, The Deep NIN consists of the following layers (as also shown in Figure 4). First, a stack of mlpconv layers is used which replaces the Generalized Linear Model (GLM) in traditional CNNs with MultiLayer Perceptron (MLP) to convolve over the input. Then, oversampling layers are used after each mlpconv and followed by a dropout layer. The dropout layer can somehow prevent from overfitting in fully connected layers [11], [45]. Finally, the global average pooling and cost layers are added in the last layer. It is worth mentioning that a simple architecture of the NIN network is used to only focus on the functionality of the proposed learning rate scheduler. Therefore, other techniques such as augmentation, ensemble, and sampling have not been conducted in these experiments.

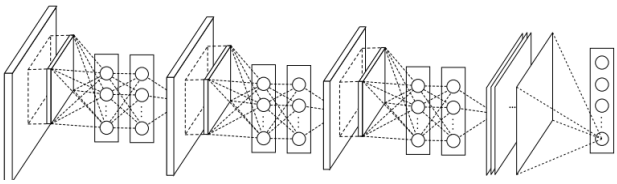

Figure 4. The Network In Network structure [44]

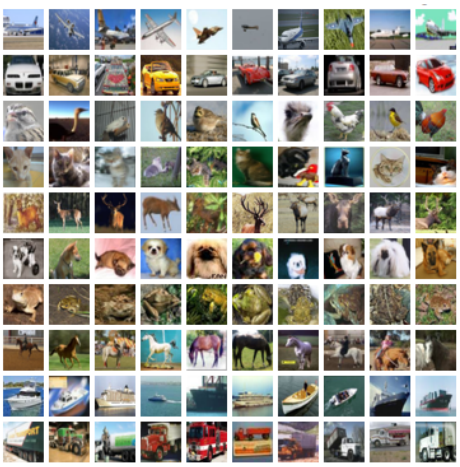

Figure 5. CIFAR-10 image dataset including 60,000 tiny images (32×32) and 10 classes

As mentioned earlier, the T-LRA algorithm can be run in parallel with the NIN algorithm. Every 5K iterations of the NIN training, T-LRA is called and calculates the trend in the current training curve. If a significant trend is observed in the curve, the learning rate remains constant, otherwise it is updated as explained in Figure 3. The 5K-iteration criterion is selected because enough data (losses in this paper) is needed for a statistical test. In addition, based on the experiments, it is an adequate and reasonable range to find significant changes in the training curve.

4.2. Results and Discussions

The proposed T-LRA is evaluated on a public multimedia dataset called CIFAR-10 [46]. This dataset includes 60,000 images (50,000 for training and 10,000 for testing) with labels and is used for image classification and object recognition. It is also composed of 10 classes as shown in Figure 5. The main challenge in this dataset is that it includes tiny color images with 32×32 resolution, taken from the dataset of 80 million tiny images [47]. Therefore, it may be thought it does not include enough information to detect objects effectively. However, this public dataset is used for many multimedia and computer vision competitions every year and even a small improvement in accuracy (e.g., 1%) can distinguish the proposed model.

Several experiments have been conducted to demonstrate the effectiveness of the proposed T-LRA. Both Cox-Stuart and Mann-Kendall trend analysis techniques are used to

schedule the learning rate. The results are also compared to the benchmark technique with the original SGD. Although there are other learning rate scheduling techniques (e.g., drop-based or time-based schedules), they are not used as the comparison benchmarks because this is a subjective problem and there is no specific rule when the learning rate should be dropped. Different studies use different epochs to reduce or increase the rate which is completely based on trial and error. This is another reason why an automatic scheduler is needed to handle this issue in a general manner. The evaluation criteria include common metrics such as accuracy and losses (as explained in Section 3.1 Equation (2)).

A popular deep learning framework called Caffe [48] is used to train the NIN network on the CIFAR-10 dataset. This C++ library contains the state-of-the-art deep learning techniques (e.g., CNNs) and several pre-trained reference models. The NIN network is trained on CPU mode using 6 servers with 64 processor.

The network input for this dataset is 32×32 with mean subtraction. The solver parameters include the base learning rate α_0 of 0.01 and SGD with momentum 0.9. In total, the network is trained for more than 100,000 iterations. The initial decay factor θ is selected as 10, which means the learning rate is divided by 10 in the first steps, while this factor is reduced gradually (i.e., divided by 2 each time when α changes) to lessen the effects of the learning rate changes in the final stages. Another important parameter is the significance level φ which controls the range of $p\text{-}value$ in the trend analysis algorithms. It is initialized to 0.05 and increased gradually due to the significant loss changes in early iterations and small loss changes in the final iterations.

Figure 6 shows the behavior of the training and testing in three methods. The first model is the original SGD without the learning rate scheduling algorithm. The second and third models are the proposed learning rate annealing based on the Cox-Stuart and Mann-Kendall trend analysis techniques, respectively. The first plot (Figure 6a) shows the training losses of these three models for more than 70,000 iterations. As can be inferred from this plot, both Mann-Kendall and Cox-Stuart have lower training losses compared to the original SGD algorithm. Specifically, Cox-Stuart losses are decreased around 30K iterations compared to both original and Mann-Kendall approaches, while Mann-Kendall training losses are suddenly decreased around 40K iterations and stays as low as possible compared to Cox-Stuart and the original one. On the other hand, the test losses plot (Figure 6b), shows the supremacy of the Cox-Stuart than Mann-Kendall as it has lower losses, especially after 30K iterations. In this plot, though, the average losses for Mann-Kendall are much smaller than the original algorithm. Finally, the last plot (Figure 6c) shows the comparison between the test accuracy of each algorithm in different iterations. As can be seen from this plot, all these three algorithms have the same accuracy in the first steps. At 30K iterations, the Cox-Stuart $p\text{-}value$ is no longer smaller than the selected significance level (e.g., 0.05) which causes a drop in the learning rate value. This drop improves the accuracy from 0.844 to 0.860 as shown in the second row of Table 1. Around 40K iterations, a similar case happens for Mann-kendall where the accuracy increases significantly from 0.834 to 0.863 as depicted in both Figure 6c and the third row of Table 1. It can be seen that the original SGD does not have this improvement even after many iterations because it may be trapped in a local minimum. The final results during the 70K iterations are shown in Table 1. The accuracy of the proposed learning rate annealing could reach to 0.862 and 0.867 using the Cox-Stuart and the Mann-Kendall trend analysis techniques, respectively.

One improvement is to increase the initial value of the significance factor φ_0 from 0.05 to 0.1 since the loss trend usually changes very slowly compared to the other trend analysis applications (e.g., environmental changes). However, this improvement causes more computational cost because it needs more iterations to reach its maximum performance. To show the effects of the significance factor, another experiment is conducted using the NIN network and the Mann-Kendall trend analysis (because it shows a higher accuracy than the Cox-Stuart one in the previous results). For this purpose, φ_0 is initialized to 0.1 and multiplied by 2 every time the learning rate is decreased. The results are shown in the last row of Table 1. As can be seen from this table, the proposed algorithm has similar results as the original SGD until iteration 50K. Then, it shows a significant increase around 60K and finally reaches to 0.87 at 70K, which is higher than all the other methods in this iteration. The learning rate annealing is executed every 5K iterations during the training of the deep network. Finally, the maximum accuracy of 0.879 is obtained, which is 3.3% higher than the one in the SGD algorithm.

The aforementioned results show the effectiveness of the proposed learning rate annealing compared to the algorithms that do not leverage any scheduling method. In addition, it automates the process of learning rate modifications, which means it can reach to the highest accuracy without manually changing the hyperparameters. Moreover, due to the very light processing of the proposed algorithm, its computational time can be completely disregarded compared to the heavy SGD costs. Therefore, it can be easily integrated with online algorithms and applications. Regarding the efficiency of the proposed method, as explained in Table 1, the proposed method can achieve a higher performance in fewer iterations. For example in 50k iterations, the accuracy of 0.863 is obtained using the Mann-Kendall trend analysis with $\varphi_0 = 0.05$; while the original SGD achieves the accuracy of 0.845 and can only increase it by 0.1% after 10k iterations. Moreover, for the proposed method, the accuracy is increased by 0.3% after 10k iterations, which shows the effectiveness and efficiency of the proposed method.

5. Conclusion

This paper proposes a novel learning rate annealing (scheduling) using two light and efficient trend analysis approaches (namely, Cox-Stuart and Mann-Kendall). This automatic and online drop-based technique reduces the learning rate value gradually to avoid trapping in a local

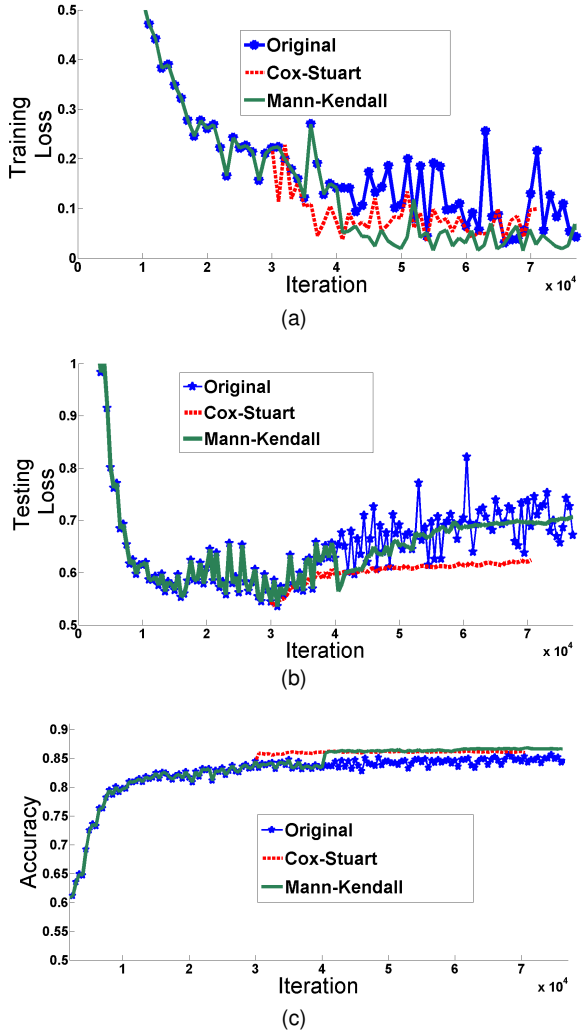

Figure 6. Performance comparison of the original SGD algorithm and the proposed algorithm using both Cox-Stuart and Mann-Kendall tests with $\varphi_0 = 0.05$. (a) shows the training losses vs iterations. Similar phenomena are shown in (b), presenting the testing losses vs iterations. The accuracy plot is demonstrated in (c).

minimum in a training loss curve, where there may exist a global minimum. Specifically, it is applied on a popular deep learning architecture called Network In Network and evaluated using a public large-scale image dataset called CIFAR-10. The proposed algorithm improves the results of the original stochastic gradient descent, used in many learning algorithms such as backpropagation. In overall, the classification accuracy on the testing data is increased by 3.3% compared to the SGD algorithm. In addition, the proposed algorithm reaches the highest accuracy in a smaller number of iterations and reduces the computational costs of training.

In future, the proposed learning rate annealing will be extended using more trend analysis techniques. In addition, it will be applied on deeper network architectures and larger datasets with multi-modality.

TABLE 1. ACCURACY COMPARISON ON CIFAR-10 FOR DIFFERENT ITERATIONS AND TWO DIFFERENT VALUES FOR THE SIGNIFICANCE FACTOR (φ_0)

	Iterations				
Algorithm	30K	40K	50K	60K	70K
Original	0.844	0.834	0.845	0.846	0.845
Cox-Stuart ($\varphi_0 = 0.05$)	0.844	0.860	0.861	0.862	0.862
Mann-Kendall ($\varphi_0 = 0.05$)	0.844	0.834	0.863	0.867	0.867
Mann-Kendall ($\varphi_0 = 0.1$)	0.844	0.834	0.845	0.867	0.870

Acknowledgments

This research is partially supported by DHSs VACCINE Center under Award Number 2009-ST-061-CI0001 and NSF HRD-0833093, HRD-1547798, CNS-1126619, and CNS-1461926. This is contribution number 820 from the Southeast Environmental Research Center in the Institute of Water & Environment at Florida International University.

References

[1] X. Chen, C. Zhang, S.-C. Chen, and S. Rubin, "A human-centered multiple instance learning framework for semantic video retrieval," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 39, no. 2, pp. 228–233, 2009.

[2] L. Lin, G. Ravitz, M.-L. Shyu, and S.-C. Chen, "Video semantic concept discovery using multimodal-based association classification," in *Proceedings of the IEEE International Conference on Multimedia & Expo (ICME)*. IEEE, 2007, pp. 859–862.

[3] X. Li, S.-C. Chen, M.-L. Shyu, and B. Furht, "Image retrieval by color, texture, and spatial information," in *Proceedings of the 8th International Conference on Distributed Multimedia Systems (DMS)*, 2002, pp. 152–159.

[4] S.-C. Chen, S. Sista, M.-L. Shyu, and R. L. Kashyap, "Augmented transition networks as video browsing models for multimedia databases and multimedia information systems," in *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 1999, pp. 175–182.

[5] S.-C. Chen and R. Kashyap, "Temporal and spatial semantic models for multimedia presentations," in *International Symposium on Multimedia Information Processing*, 1997, pp. 441–446.

[6] S. Pouyanfar and S.-C. Chen, "Semantic concept detection using weighted discretization multiple correspondence analysis for disaster information management," in *The 17th IEEE International Conference on Information Reuse and Integration (IEEE IRI)*, 2016, pp. 556–564.

[7] M.-L. Shyu, C. Haruechaiyasak, S.-C. Chen, and N. Zhao, "Collaborative filtering by mining association rules from user access sequences," in *International Workshop on Challenges in Web Information Retrieval and Integration*. IEEE, 2005, pp. 128–135.

[8] X. Chen, C. Zhang, S.-C. Chen, and M. Chen, "A latent semantic indexing based method for solving multiple instance learning problem in region-based image retrieval," in *Proceedings of the IEEE International Symposium on Multimedia (ISM)*. IEEE, 2005, pp. 37–44.

[9] X. Huang, S.-C. Chen, M.-L. Shyu, and C. Zhang, "User concept pattern discovery using relevance feedback and multiple instance learning for content-based image retrieval," in *Proceedings of the Third International Conference on Multimedia Data Mining (MDM/KDD)*. Springer-Verlag, 2002, pp. 100–108.

[10] S.-C. Chen, M.-L. Shyu, and C. Zhang, "An intelligent framework for spatio-temporal vehicle tracking," in *IProceedings of the 4th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2001, pp. 213–218.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[12] S. Pouyanfar and S.-C. Chen, "Semantic event detection using ensemble deep learning," in *The IEEE International Symposium on Multimedia (IEEE ISM)*, 2016, pp. 203–208.

[13] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[14] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1746–1751.

[15] M. Chen, C. Zhang, and S.-C. Chen, "Semantic event extraction using neural network ensembles," in *First International Workshop on Semantic Computing and Multimedia Systems, in conjunction with First IEEE International Conference on Semantic Computing (IEEE ICSC2007)*, 2007, pp. 575–580.

[16] V. Plagianakos, G. Magoulas, and M. Vrahatis, "Learning rate adaptation in stochastic gradient descent," in *Advances in convex analysis and global optimization*. Springer, 2001, pp. 433–444.

[17] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *19th International Conference on Computational StatisticsParis France (COMPSTAT'2010)*. Springer, 2010, pp. 177–186.

[18] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks." in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

[19] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," *CoRR*, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[20] J. Brownlee, "Using learning rate schedules for deep learning models in python with keras," http://machinelearningmastery.com/using-learning-rate-schedules-deep-learning-models-python-keras/, 2016.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[22] L. Bottou, "Stochastic gradient tricks," *Neural Networks, Tricks of the Trade, Reloaded*, vol. 7700, pp. 430–445, 2012, lecture Notes in Computer Science (LNCS 7700).

[23] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[25] Y. LeCun, P. Y. Simard, and B. Pearlmutter, "Automatic learning rate maximization by on-line estimation of the Hessian's eigenvectors," *Advances in neural information processing systems*, vol. 5, pp. 156–163, 1993.

[26] V. Plagianakos, D. Sotiropoulos, and M. Vrahatis, "Automatic adaptation of learning rate for backpropagation neural networks," *Recent Advances in Circuits and Systems*, vol. 337, 1998.

[27] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, 1988.

[28] T. Tieleman and G. Hinton, "Lecture 6.5 - RMSProp," COURSERA: Neural networks for machine learning, Tech. Rep., 2012.

[29] C. Darken and J. E. Moody, "Towards faster stochastic gradient search," in *Advances in Neural Information Processing Systems 4 NIPS Conference*, 1991, pp. 1009–1016.

[30] D. W. Meals, J. Spooner, S. A. Dressing, and J. B. Harcum, "Statistical analysis for monotonic trends," Tetra Tech, Tech. Rep., 2011, developed for U.S. Environmental Protection Agency.

[31] R. G. Miller Jr, *Beyond ANOVA: basics of applied statistics*. CRC Press, 1997.

[32] B. C. Reed, "Trend analysis of time-series phenology of north america derived from satellite data," *GIScience & Remote Sensing*, vol. 43, no. 1, pp. 24–38, 2006.

[33] J. M. Kampata, B. P. Parida, and D. Moalafhi, "Trend analysis of rainfall in the headstreams of the zambezi river basin in zambia," *Physics and Chemistry of the Earth, Parts A/B/C*, vol. 33, no. 8, pp. 621–625, 2008.

[34] F. Barthel and E. Neumayer, "A trend analysis of normalized insured damage from natural disasters," *Climatic Change*, vol. 113, no. 2, pp. 215–237, 2012.

[35] M. Shadmani, S. Marofi, and M. Roknian, "Trend analysis in reference evapotranspiration using Mann-Kendall and Spearman's Rho tests in arid regions of Iran," *Water resources management*, vol. 26, no. 1, pp. 211–224, 2012.

[36] M. Gocic and S. Trajkovic, "Analysis of changes in meteorological variables using Mann-Kendall and Sen's slope estimator statistical tests in Serbia," *Global and Planetary Change*, vol. 100, pp. 172–182, 2013.

[37] K. H. Hamed, "Trend detection in hydrologic data: the Mann-Kendall trend test under the scaling hypothesis," *Journal of Hydrology*, vol. 349, no. 3, pp. 350–363, 2008.

[38] P. K. Sen, "Estimates of the regression coefficient based on Kendall's tau," *Journal of the American Statistical Association*, vol. 63, no. 324, pp. 1379–1389, 1968.

[39] D. R. Cox and A. Stuart, "Some quick sign tests for trend in location and dispersion," *Biometrika*, vol. 42, no. 1/2, pp. 80–95, 1955.

[40] I. Supit, C. Van Diepen, A. De Wit, P. Kabat, B. Baruth, and F. Ludwig, "Recent changes in the climatic yield potential of various crops in europe," *Agricultural Systems*, vol. 103, no. 9, pp. 683–694, 2010.

[41] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[42] "Solver methods," retrieved at: 2016-12-20. [Online]. Available: http://caffe.berkeleyvision.org/tutorial/solver.html

[43] T. Pohlert, "Non-parametric trend tests and change-point detection," *CC BY-ND 4.0*, 2016.

[44] M. Lin, Q. Chen, and S. Yan, "Network in Network," *CoRR*, vol. abs/1312.4400, 2013. [Online]. Available: http://arxiv.org/abs/1312.4400

[45] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, 2012. [Online]. Available: http://arxiv.org/abs/1207.0580

[46] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009. [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[47] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 11, pp. 1958–1970, 2008. [Online]. Available: http://groups.csail.mit.edu/vision/TinyImages/

[48] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.