Event-Based Control for Online Training of Neural Networks

Zilong Zhao, Sophie Cerf, Bogdan Robu, Nicolas Marchand

Event-Based Control for Online Training of Neural Networks

Zilong Zhao[1], Sophie Cerf[1], Bogdan Robu[1], and Nicolas Marchand[1]

***Abstract*—Convolutional Neural Network (CNN) has become the most used method for image classification tasks. During its training the learning rate and the gradient are two key factors to tune for influencing the convergence speed of the model. Usual learning rate strategies are time-based i.e. monotonous decay over time. Recent state-of-the-art techniques focus on adaptive gradient algorithms i.e. Adam and its versions. In this paper we consider an online learning scenario and we propose two Event-Based control loops to adjust the learning rate of a classical algorithm E (Exponential)/PD (Proportional Derivative)-Control. The first Event-Based control loop will be implemented to prevent sudden drop of the learning rate when the model is approaching the optimum. The second Event-Based control loop will decide, based on the learning speed, when to switch to the next data batch. Experimental evaluation is provided using two state-of-the-art machine learning image datasets (CIFAR-10 and CIFAR-100). Results show the Event-Based E/PD is better than the original algorithm (higher final accuracy, lower final loss value), and the Double-Event-Based E/PD can accelerate the training process, save up to 67% training time compared to state-of-the-art algorithms and even result in better performance.**

***Index Terms*—Event Based Control; Gradient Methods; Neural Networks**

I. Introduction

Convolutional Neural Network (CNN) is a popular machine learning algorithm for image classification because it outperforms any other network architecture on visual data. In this paper, we focus on an online learning scenario where data used for training the CNN comes in batches over time [1], [2]. A CNN model is a neural network structure with a set of weights which are iteratively learned from training data using methods such as Stochastic Gradient Descent (SGD). The SGD algorithm is parametrized with a learning rate λ. A large λ helps the model to converge faster but increases the risk of diverging [3]. A small λ slows the convergence but may lead to a local minimum.

There are two main learning rate evolution strategies: time-based or adaptive. In most time-based learning rate strategies, λ decreases following a predefined decay function [4]. Cyclical strategies have also been developed, where two boundaries are defined and λ cyclically varies between them. The disadvantage of these algorithms is that the learning rate path is fixed before training, it cannot be adjusted when necessary.

Adaptive learning rate algorithms such as Adam [5], Nadam (Adam with Nesterov momentum) [6] and AMSGrad [7] are recent state-of-the-art algorithms which mainly focus on the convergence speed. Different from SGD which uses only the current value of the gradient to update weights, these algorithms use squared gradient to scale the learning rate and take advantage of momentum by using moving average of the gradients. Nevertheless, Wilson et al. [8] suggested that adaptive gradient methods do not generalize as well as SGD. These methods tend to perform well in the initial portion of training but are outperformed by SGD at later stages of training [9]. To address this issue, AdaBound [10] employs dynamic bounds on learning rates to achieve a gradual and smooth transition from adaptive methods to SGD.

Up to our knowledge, E (Exponential)/PD (Proportional Derivative) control [11] is the first adaptive learning rate algorithm which uses control theory to dynamically adapt the learning rate during the learning process. It uses only current gradient as in SGD, but its learning rate λ is dynamically calculated based on the loss value. During the E phase, that corresponds to the beginning of the training when the loss value is continuously decreasing, λ is increased each time step by a factor of two. Once the loss stops decreasing, the PD phase takes over and, considering CNN as a dynamic system, computes the control input (i.e. λ) based on the CNN's output (i.e. the loss value).

The above-mentioned algorithms are time-based, in the sense of a periodic computation of the control law regardless its utility. In this paper, we propose two event-based control strategies to reduce the time CNN spends learning "inefficiently" from data, as well as an extensive evaluation. Moreover, while using event-based mechanisms we should expect for a reduction in the use of resources [12], [13], without degrading performances [14] and with stability and robustness guarantees [15]. Numerous Event-Based control strategies in the literature are focusing on stability and performance guarantees. Most event-based PID controllers are based on level-crossing triggering of some measuring error (see for instance [13], [16]) or more generally rely on an event-function based on Lyapunov functions (see for instance [15], [17]).

The two introduced Event-Based control algorithms are: (i) Event-Based Learning Rate control, which will be implemented to prevent sudden drop of the learning rate when the model is approaching the optimum; (ii) Event-Based Learning Epochs control, which will decide based on the learning speed when to switch to the next data batch.

Our algorithm is evaluated on two classical machine learning image datasets CIFAR-10 and CIFAR-100 [18]. The results are compared with four best state-of-the-art algorithms: Adam, Nadam, AMSGrad and AdaBound. Our results show that the E/PD combined with the two introduced Event-Based control not only outperforms original E/PD but also converges faster than any other state-of-the-art counterpart.

The article is organised as follows: after a brief introduction of the problem in Section I, we detail the scenario and the system to be controlled (i.e. a CNN) with its input and output metrics in Section II. The contribution, i.e., the two event-based mechanisms, is described in Section III.Section IV

[1] Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France firstname.lastname@gipsa-lab.fr

contains the experimental setup, results and analysis. The article ends with a conclusion and perspectives for further work in Section V.

II. Background

A. Classical Online Learning Scenario

We consider a dataset $\mathscr{T}$ with a total number of training instances T, each one belonging to a class $c : \mathbb{Z}^+ \to [1, C]$. The whole dataset is composed of B subsets (i.e. batches), T_i is the i^{th} batch where $i : \mathbb{Z}^+ \to [1, B]$. Each batch equally contains S data instances and will be used to train the model for N epochs (i.e. N times). At the reception of a new batch, the learning rate algorithm is reset with initial values. Classical online learning scenario is illustrated in Fig. 1.

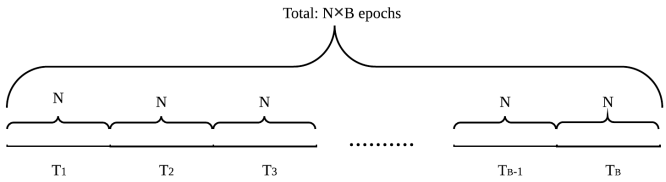


Fig. 1: Classical Online Learning Scenario.

B. Convolutional Neural Network and Gradient

Convolutional Neural Network (CNN) is the state-of-the-art learning mechanism for image classification [19]. CNN neurons functions are parameterized with weights and, eventually, bias. The objective of the learning phase is to make iterative adjustments to these biases and weights to better fit the data. These weights in the CNN are usually updated using Stochastic Gradient Descent techniques (SGD):

$$\theta_j = \theta_{j-1} - \lambda \frac{\partial L}{\partial \theta}$$

where vector θ_j represents the weights vector computed at j^{th} discrete time instant, λ is positive and denotes the learning rate. L is the loss function. As we are always trying to minimize the loss function, we suppose that there exists an optimal solution of parameters θ_j^*.

C. Performance Metrics

There exists many metrics to evaluate the performance of a CNN model [20], we used two of the most classical: classification accuracy and loss value.

For evaluating, machine learning researchers typically prepare a testing dataset which will not be used during the training process. At the end of each training phase (called from now on epoch), the testing dataset is used to evaluate the model by measuring the classification accuracy and the loss value. Accuracy is defined as:

$$\text{Accuracy} = \frac{\text{Number of correct prediction}}{\text{Total number of prediction}} \tag{1}$$

The loss L is defined as the difference between the predicted value by the model and the true value. The most common definition of L used for classification problems is cross-entropy [21]:

$$L = -\frac{1}{V} \sum_{p=1}^{V} \sum_{q=1}^{C} y_{p,q} \log(\hat{y}_{p,q}) + (1 - y_{p,q}) \log(1 - \hat{y}_{p,q}) \tag{2}$$

where V is the size of testing dataset and C is the total number of classes and also the length of the prediction vector which is a probability vector. $\hat{y}_{p,q}$ denotes the q^{th} bit value of prediction vector for data sample p while $y_{p,q}$ is the ground truth, indicating if data p belongs to class q ($y_{p,q} = 1$) or not ($y_{p,q} = 0$).

III. Event-Based Control Laws

In [11], an E/PD control of the learning rate is proposed consisting of an increasing phase followed by a PD phase. However, if an increase of the performance can be achieved on both the loss and the accuracy, the learning rate is progressively decreased by the E/PD control in the PD phase, even though a larger value of learning rate would be more efficient in term of performance. Since event-based PID have shown to be more efficient in terms of convergence [13], we propose here to implement an event-based E/PD controller to control the learning rate. [11] also shows that significant improvements in terms of accuracy and loss only occurred at the first epochs of training each data batch, so after this stage there is no limited interest into continuing the learning on further epochs. Therefore, we propose a second event-based control to adapt the data batch loading process.

A. Event-Based Learning Rate

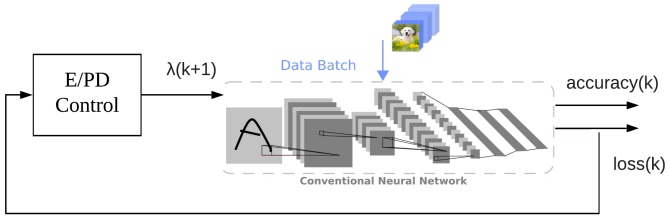


Fig. 2: E/PD Control Structure.

A recall of the E/PD Control algorithm from [11] is schematicly presented in Fig. 2. We suggest to look at a CNN training as a dynamical system with the learning rate as controlled input and the loss as measurable output. Initial weights of the CNN are chosen randomly and the initial learning rate $\lambda(0)$ is fixed. E/PD learning rate strategy is defined as:

$$\lambda(k+1) = 2\lambda(k) \tag{3}$$

as long as $L(k) < L(k-1)$ (E phase) and

$$\lambda(k+1) = K_P \frac{L(k)}{L(0)} - K_D \frac{L(k) - L(k-1)}{L(0)} \tag{4}$$

from the first instant $k = k^*$ when $L(k^*) > L(k^* - 1)$ to the end of learning process for the data batch (i.e. the PD phase). For the sake of simplicity the loss values are normalized with

respect to the initial epoch loss value $L(0)$. K_P and K_D are the proportional and derivative gain detailed in [11].

On top of the PD phase we consider the following event base mechanism where instead of letting the PD-Control compute the rate each time (which might be lowering the learning rate), we propose to update the learning rate only if the loss value increases during the PD-Control phase.

Let us define the event function $e_1 : \mathbb{R}^+ \to \{0, 1\}$ by:

$$e_{1_k} = \begin{cases} 1 & \text{if } L(k) - L(k-1) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The proposed PD event-triggered control output $\lambda(k+1)$ at time $k+1$ is then:

$$\lambda(k+1) = \begin{cases} K_P \frac{L(k)}{L(0)} - K_D \frac{L(k)-L(k-1)}{L(0)} & \text{if } e_{1_k} = 1 \\ \lambda(k) & \text{otherwise} \end{cases} \tag{6}$$

where $\lambda(k+1)$ is the calculated learning rate for epoch $k+1$, $L(k)$ is the corresponding loss for epoch k.

Note that the stability of CNN is ensured by E/PD, whose stability analysis is provided in [11]. Proposed event-based control does not introduce any instability because if $e_1 = 0$, which means the loss is decreasing, model is converging, and if $e_1 = 1$, the learning rate strategy returns to E/PD.

B. Event-Based Learning Epochs

1) Controller Design: As observed in [11], significant improvement in the learning only occurs at the beginning when loading a new batch, the accuracy and loss value evolve slowly afterwards. This motivates the use of an event-based strategy on the loss value record.

Consider a maximum of N training epochs within each batch. Let X_k vector contains the latest m epochs numbers and Y_k vector contains the m latest corresponding normalized loss values:

$$X_k = \begin{bmatrix} k-m & \cdots & k-2 & k-1 & k \end{bmatrix}$$
$$Y_k = \begin{bmatrix} \frac{L(k-m)}{L(0)} & \cdots & \frac{L(k-2)}{L(0)} & \frac{L(k-1)}{L(0)} & \frac{L(k)}{L(0)} \end{bmatrix}$$

where $k \in [1, N-1]$. One can use least squares estimation to fit a regression line with X_k and Y_k:

$$Y_k = \alpha_k X_k + \beta_k \tag{7}$$

The purpose of this is that if the training process goes well the loss value should always decrease, therefore α_k should always be negative. Even with the presence of loss variations during the training, as long as the decreasing trend doesn't change, α_k should still be negative. Nevertheless, in the moment the loss trend becomes flat or even is increasing, α_k will become 0 or positive.

We define the event mechanism by the event function $e_2 : \mathbb{R}^+ \to \{0, 1\}$ by:

$$e_{2_k} = \begin{cases} \text{call new batch} & \text{if } \alpha_k > \alpha_{thld} \text{ or } k = N \\ \text{remain on same batch} & \text{if } \alpha_k \leq \alpha_{thld} \text{ and } k < N \end{cases} \tag{8}$$

which enables to switch to new data batch when the learning speed is too low, i.e. the training is not efficient anymore.

The threshold α_{thld} can be adjusted in order to control the efficiency of learning. This threshold should never be positive as an increasing curve of the loss value is not desirable. With enough computing resources and no time constraints, the threshold can be set close to 0, and the training will last even though it makes very small improvement. Nevertheless, for online learning the time interval between two data batches can be short compared to the training time and we could encounter the scenario when before we finish the current training epochs the next data batch is already available. In this case, cutting off some useless training can be very useful. Therefore α_{thld} should also be chosen depending on the frequency of batch arrival. The choice of m is based on the constraints imposed by the CNN (or the application using CNN). A large value of m would imply a long time of inactivity as the controller would react only after m epochs (consecutive tests). A small value of m would imply that the algorithm is very sensitive to each epoch thus if $m = 0$ the event based algorithm becomes a time based one.

2) Online Learning Scenario: Recall the online learning scenario defined in Sec. II-A and Fig. 1, the difference for Event-Based Learning Epochs is that the training epochs for each batch could be varied but no larger than N, but the total training epochs are the same for both scenario for all the experiments of the same dataset. So here we could cyclically learn the data batches until it reaches the total epochs limit. The online learning arrangement for Event-Based Learning Epochs is illustrated in Fig. 3.

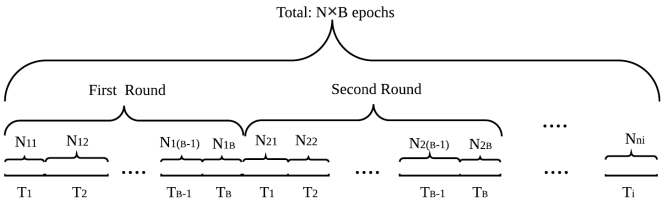


Fig. 3: Event-Based Learning Epochs Online Learning Scenario.

IV. Experimental Evaluation

A. Experimental Setup

The experiments are implemented on two state of the art machine learning datasets: 1) CIFAR-10 (a natural image data set with 10 categories) and 2) CIFAR-100 (a natural image data set with 100 categories) [18] with 3 different initial learning rate. The characteristics of the two data-sets are given in Table I. As the CIFAR-100 dataset has more classes, we use a deeper CNN: ResNet [22] than the one used for CIFAR-10 VGG [23]. Due to the computational resource limitation, for ResNet with CIFAR-100, we train 30 epochs per data batch instead of 60 for CIFAR-10.

All the experiments are implemented with Keras [24] and are carried out on Google Cloud Compute-Engine using 8 virtual CPU with 30 GB memory and one P100 GPU. Each experiment is repeated 5 times.

The parameters α_{thld} and m are selected through a process of cross validation on a subset of CIFAR-10. As a small value for m leads to high sensitivity and a large m slows down

TABLE I: Experiments configuration

Use case	CIFAR-10	CIFAR-100
#data instances to train T	50,000	50,000
#data instances to test V	10,000	10,000
#classes C	10	100
data batch size S	10000	10000
total batches B	5	5
#trainng epochs per batch N	60	30

the detection of the situation, we predefined a reasonable list of choice $m \in [4; 5; 6; 7; 8]$. Due to similar consideration of sensibility, we also predefined a list for the learning rate threshold $\alpha_{thld} \in [-0.1; -0.01; -0.001; -0.0001]$. Each possible pair from these two lists is tested, a good compromise between reactivity and noise sensitivity was found for $m = 4$ and $\alpha_{thld} = -0.001$.

B. Evaluation Metrics

The final loss and final validation accuracy (hereinafter referred to as FVA) reveal the performance of the final model. Nevertheless, stability metrics are also important: if accuracy curve experiences a big variance near the end of training process, even we could have a good final result, we could not assure that we always get this result. Thus, in our evaluation, we include standard deviation of the accuracy of the last 10% training epochs [25] (hereinafter referred to as FASD (Final Accuracy Standard Deviation)). Convergence speed of accuracy is another metric to evaluate the performance, as we will focus on online learning scenario, the interval between two batch data can be short. With a limited time, a faster accuracy convergence could lead to a better model performance comparing to other algorithms. Therefore, we will report the first epoch when the experiment reaches the 95% of best final accuracy among all the experiments.

C. Evaluation of Event-Based E/PD

Event-Based E/PD (hereinafter referred to as EB E/PD) refers to the E/PD control combined with Event-Based Learning Rate control (Sec. III-A). We implement the online training experiments with E/PD and EB E/PD on CIFAR-10. From Fig. 5 we can first see the comparison between EB E/PD and original E/PD (only yellow and dotted blue line for now). For the first 60 epochs, we can see that EB E/PD is more stable than E/PD, then their curves are quite overlapped. The averaged comparison results are showed in Table. II. EB E/PD performs better than E/PD in almost all metrics for all initial learning rate group. Even though EB E/PD has a higher FASD under 0.01 and 0.05 initial learning rate, but the minimum value of FVA(±FASD) range of EB E/PD is higher than the maximum value of the range of E/PD.

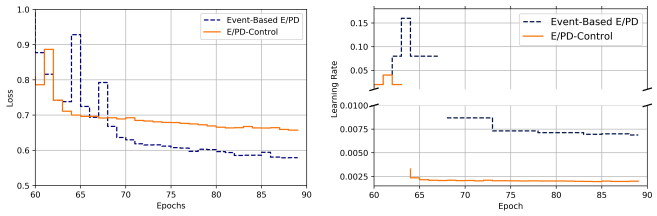
For the sake of visibility, we zoom into the 60th to 90th training epochs from our two experiment runs and show the evolution of the loss value and learning rate in Fig. 4. According to the learning rate curve, we know that E phase ends at 62th epoch for E/PD-Control curve, and at 64th epoch for EB E/PD. E/PD-Control curve clearly shows the problem we mentioned above, we can observe that from 62th epoch, the loss of E/PD is continuously decreasing until 70th epoch, and its learning rate is also decreasing during this period. If

TABLE II: Experiments with varying initial learning rate $\lambda(0)$ on CIFAR-10. Mean value over 5 runs are reported.

Algorithm	$\lambda(0)$	Final loss	FVA[1] (± FASD[2]) (%)	1st epoch to 81.66%[3]
E/PD	0.002	0.58	83.17(±0.08)	124/300
EB E/PD	0.002	**0.56**	**83.81(±0.03)**	**93/300**
E/PD	0.01	0.55	84.35(±0.07)	88/300
EB E/PD	0.01	**0.54**	**84.91(±0.10)**	**75/300**
E/PD	0.05	0.56	85.06(±0.12)	73/300
EB E/PD	0.05	**0.50**	**85.96(±0.26)**	**63/300**

1. FVA: Final Validation Accuracy
2. FASD: Final Accuracy Standard Deviation
3. 81.66%: 85.96%(best final accuracy among all the experiments)×95%

the learning rate could stay constant during these 9 epochs, its loss would decrease sharply and that would improve the convergence speed. In contrast, EB E/PD keeps the learning rate when the loss continuously decreases which helps to accelerate the convergence. We can also notice that with the drop of the loss, each time when we update the learning rate for EB E/PD, its trend is also decreasing which will guarantee the stability of EB E/PD near the optimum.



(a) Loss value (b) Learning rate

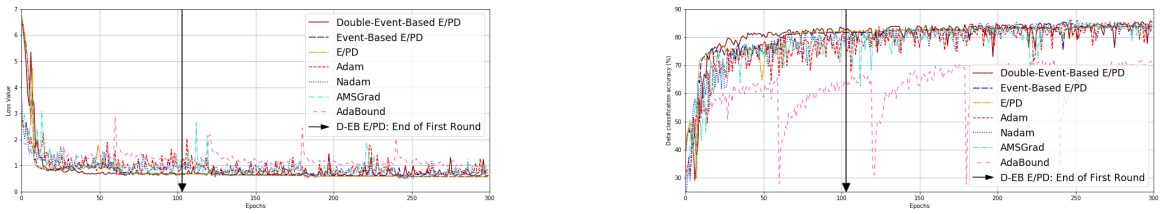
Fig. 4: Performances of E/PD and EB E/PD on CIFAR-10

D. Evaluation of Double-Event-Based E/PD

Double-Event-Based E/PD-Control (hereinafter referred to as D-EB E/PD) refers to the E/PD control combined with Event-Based Learning Rate control (Sec. III-A) and Event-Based Learning Epochs control (Sec. III-B). To ensure the need of the Event-Based Learning Rate control, we implemented E/PD with only Event-Based Learning Epochs control; results showed that Double Event-Based E/PD always has a better performance in Final loss and FVA. Due to the page limitation, we exclude these results from the main manuscript, however they are available online as appendices.

D-EB E/PD-Control has been tested on CIFAR-10 and CIFAR-100 and compared with 4 best state-of-the-art adaptive optimization algorithms: Adam, Nadam, AMSGrad and AdaBound. For these 4 learning rate strategies, except varying initial learning rate, all the other parameters remain as default as they mentioned in their paper or coded in Keras. As we adopt Event-Based Learning Epochs control into D-EB E/PD, the training epochs for each data batch is not fixed, we may also iterate each data batch several times. Therefore, we will not only report the results at the end of whole training process, but also the results after first round training (i.e. the training process iterates, for the first time, all the data batches, refer to Fig. 3).

Experimental results on CIFAR-10 are showed in Fig. 5, all the curves are generated with the same initial learning



(a) Loss value

(b) Accuracy

Fig. 5: Performance comparison on CIFAR-10 with $\lambda(0) = 0.01$ initial learning rate. Compact view of the results in Table III.

rate 0.01. Between 25th and 60th epoch, D-EB E/PD largely outperforms all the counterparts. The vertical line with arrow at 104th epoch indicates that our D-EB E/PD algorithm has finished its first round learning of the whole 5 batches after this epoch. There are two reasons that we can achieve this performance: (i) EB E/PD converges very fast, (ii) during these epochs, our D-EB E/PD algorithm have trained with later batches data, while other 4 algorithms, they are still working on the first batch data. Diversity of training data helps to reach better performance.

More detail of results on CIFAR-10 is reported in Table. III. D-EB E/PD reaches a higher final accuracy and lower final loss no matter $\lambda(0)$. Even though D-EB E/PD has a higher FASD than AdaBound with $\lambda(0) = 0.01$ and $\lambda(0) = 0.05$, the FVA($\pm$FASD) range of D-EB E/PD is always higher than the range of AdaBound. Additionally it only takes about 32 to 38 epochs to reach 95% best accuracy in any group. All the indicators are very stable across different groups for D-EB E/PD. One can also note that for all the 4 state-of-the-art algorithms, they all perform very bad with $\lambda(0) = 0.05$, they cannot even reach the 95% best accuracy. We also implemented the same experiments with $\lambda(0) = 0.25$. Except our algorithm, no other one reaches a reasonable accuracy value, which can be explained by the fact that during the PD phase of E/PD control our learning rate can decrease to a low level while the counterparts can not. Those results are available as appendices.

CIFAR-100 results are reported in Table. IV. According to the FVA, we know that all the algorithms did not totally converge in the end of training process, but that does not influence our conclusion of analysis. D-EB E/PD outperforms other algorithms in almost all the metrics, when its FASD is higher than others in certain groups, its FVA($\pm$FASD) range is always higher than others. As the algorithms are not totally converged, the trend of accuracy curve is still increasing, therefore, the higher the initial learning rate, the faster the 1st epoch to reach 95% best accuracy.

Table. V shows the results of D-EB E/PD in the end of first round learning. All the final loss after first round learning in this table is lower than all the state-of-the-art algorithms in their end of whole training process comparing to their own group. Except CIFAR-100 for $\lambda(0) = 0.002$, all the FVA after first round learning in this table exceed the 95% best accuracy in Table. III and Table. IV, respectively. As the learning process on CIFAR-100 is not totally converged, we can notice that the ending epoch of their first round is near the end of whole training process, our event-based control did not cut off many epochs. But for CIFAR-10, event-based control helps to massively cut off around 62% to 67% training epochs meanwhile guarantee a very good result.

TABLE III: Double-Event-Based E/PD algorithm experiments with varying initial learning rate $\lambda(0)$ on CIFAR-10. Mean value over 5 runs are reported.

Algorithm	$\lambda(0)$	**Final loss**	**FVA $\pm$FASD**	**1st epoch to** 80.94%[1]
D-EB E/PD	0.002	**0.58**	**84.50($\pm$0.59)**	**38/300**
Adam	0.002	0.73	84.14($\pm$1.34)	64/300
Nadam	0.002	0.71	83.29($\pm$1.11)	66/300
AMSGrad	0.002	0.67	84.21($\pm$1.65)	65/300
AdaBound	0.002	0.81	84.31($\pm$0.96)	75/300
D-EB E/PD	0.01	**0.61**	**84.83($\pm$1.29)**	**37/300**
Adam	0.01	0.79	83.98($\pm$1.58)	64/300
Nadam	0.01	0.75	84.15($\pm$1.29)	65/300
AMSGrad	0.01	0.65	84.21($\pm$1.50)	72/300
AdaBound	0.01	0.84	79.22($\pm$1.21)	-
D-EB E/PD	0.05	**0.60**	**85.20($\pm$3.14)**	**32/300**
Adam	0.05	5.98	48.93($\pm$14.06)	-
Nadam	0.05	7.74	42.27($\pm$13.95)	-
AMSGrad	0.05	2.69	59.74($\pm$12.43)	-
AdaBound	0.05	1.03	71.49($\pm$1.65)	-

1. 80.94%: 85.20%(best final accuracy among all the experiments)$\times$95%

TABLE IV: Double-Event-Based E/PD algorithm experiments with varying initial learning rate $\lambda(0)$ on CIFAR-100. Mean value over 5 runs are reported

Algorithm	$\lambda(0)$	**Final loss**	**FVA ($\pm$FASD) (%)**	**1st epoch to** 46.56%[1]
D-EB E/PD	0.002	**2.59**	**45.69($\pm$1.94)**	-
Adam	0.002	3.40	31.29($\pm$3.23)	-
Nadam	0.002	3.18	35.66($\pm$3.35)	-
AMSGrad	0.002	3.13	35.38($\pm$4.02)	-
AdaBound	0.002	3.29	39.87($\pm$4.42)	-
D-EB E/PD	0.01	**2.41**	**48.14($\pm$3.34)**	**111/150**
Adam	0.01	4.94	8.11($\pm$2.04)	-
Nadam	0.01	4.55	9.70($\pm$2.32)	-
AMSGrad	0.01	4.79	8.16($\pm$0.50)	-
AdaBound	0.01	3.51	30.98($\pm$3.08)	-
D-EB E/PD	0.05	**2.38**	**49.01($\pm$10.52)**	**100/150**
Adam	0.05	4.72	2.64($\pm$0.58)	-
Nadam	0.05	4.74	1.88($\pm$0.79)	-
AMSGrad	0.05	4.68	1.98($\pm$0.56)	-
AdaBound	0.05	3.69	19.03($\pm$2.42)	-

1. 46.56%: 49.01%(best final accuracy among all the experiments)$\times$95%

E. Trade-offs and limitations

The addition of event-based mechanisms improves the performance in terms of final accuracy and loss, however at the cost of two sacrifices: (i) Event-Based Learning Epochs accelerate the speed of learning each data batch. However, if we are not allowed to keep in cache any data batch locally, i.e. only allowed to learn each data batch once, the performance of Double Event-Based E/PD after first round is slightly worse than the performance after all the training epochs. (ii) Double

Event-Based E/PD will cyclically learn all data batches, and it will need to load and unload data batch more times than classical online learning setting. Loading (unloading) data into (from) memory needs time. These are extra costs for Double Event-Based E/PD, however negligible compared to the computing intensity of CNNs.

Regarding the limitation of the presented D-EB E/PD, we identified one potential case for which our algorithm will fail: if the training data contains mislabeled data. These data will lead the model to converge to a wrong optimum, and as the algorithm minimizes faster the loss function, it will be faster over-fitting to the noisy data than other algorithms. However, this fail is caused by poor data selection, and is not specific to our algorithm.

TABLE V: Double Event-Based E/PD experiments on CIFAR-10 and CIFAR-100 in the End of First Round. Mean value over 5 runs are reported.

Dataset	$\lambda(0)$	**EE of FR**[1]	**FL after FR**[2]	**FVA after FR**[3] **(%)**
CIFAR10	0.002	99/300	0.60	82.47
CIFAR10	0.01	104/300	0.62	82.36
CIFAR10	0.05	113/300	0.62	82.75
CIFAR100	0.002	148/150	2.61	44.98
CIFAR100	0.01	148/150	2.44	48.04
CIFAR100	0.05	146/150	2.41	48.95

1. EE of FR: End Epoch of First Round
2. FL after FR: Final loss after First Round
3. FVA after FR: Final Validation Accuracy after First Round

V. Conclusion and future work

Due to the limitation of computing resource or short interval time between two data batches, convergence speed of the loss and accuracy becomes especially important for online learning. E/PD control is a powerful learning rate algorithm when training neural network on an online learning scenario. Based on E/PD, this paper proposes two algorithms: (i) Event-Based Learning Rate algorithm and (ii) Event-Based Learning Epochs algorithm.

The new algorithm firstly introduces an Event-Based control on PD phase of E/PD, when the loss continuously decreases, we prevent the learning rate to decrease during this period. Second Event-Based control is implemented to inspect the record of the loss value. If the loss record has the tendency to increase, showing little learning efficiency, we will drop the rest learning epochs for current data batch.

Results show that Double-Event-Based E/PD can massively cut off training epochs, and even results in a lower loss value. For instance with CIFAR-10 dataset, it could save up to 67% training epochs.

As the Event-Based Learning Epochs control is independent from learning rate algorithm and dataset, this work could be further extended by implementing this control with language, image and numeric datasets on time-based decay SGD, Adam, Nadam, AMSGrad and AdaBound learning rate algorithms, to prove that by simply adding this event-based control, all the learning rate algorithms on any dataset can improve their performance on online learning scenario.

References

[1] D. Tien Nguyen, S. Joty, M. Imran, H. Sajjad, and P. Mitra, "Applications of Online Deep Learning for Crisis Response Using Social Media Information," *arXiv e-prints*, p. arXiv:1610.01030, Oct 2016.

[2] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *International conference on machine learning*, 2015, pp. 597–606.

[3] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural networks: Tricks of the trade*. Springer Berlin Heidelberg, 2012, pp. 437–478.

[4] W. An, H. Wang, Y. Zhang, and Q. Dai, "Exponential decay sine wave learning rate for fast deep neural network training," in *IEEE Visual Communications and Image Processing*, Dec 2017, pp. 1–4.

[5] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014.

[6] T. Dozat, "Incorporating Nesterov momentum into Adam," in *4th International Conference on Learning Representations*, San Juan, Puerto Rico, May 2016.

[7] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," in *International Conference on Learning Representations*, Vancouver, Canada, Apr 2018.

[8] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning," in *Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA*, 2017, pp. 4148–4158.

[9] N. Shirish Keskar and R. Socher, "Improving Generalization Performance by Switching from Adam to SGD," *arXiv e-prints*, p. arXiv:1712.07628, Dec. 2017.

[10] L. Luo, Y. Xiong, Y. Liu, and X. Sun, "Adaptive gradient methods with dynamic bound of learning rate," in *7th International Conference on Learning Representations, New Orleans, LA, USA, May 6-9, 2019*.

[11] Z. Zhao, S. Cerf, B. Robu, and N. Marchand, "Feedback control for online training of neural networks," in *IEEE Conference on Control Technology and Applications*, Hong Kong, China, 2019, pp. 136–141.

[12] K. J. Aström, "Event based control," in *Analysis and Design of Nonlinear Control Systems*, A. Astolfi and L. Marconi, Eds. Springer Berlin Heidelberg, 2008, pp. 127–147.

[13] S. Durand and N. Marchand, "Further results on event-based PID controller," in *Proceedings of the European Control Conference*, 2009.

[14] J. Lunze and D. Lehmann, "A state-feedback approach to event-based control," *Automatica*, vol. 46, pp. 211–215, 2010.

[15] N. Marchand, S. Durand, and J. F. Guerrero-Castellanos, "A general formula for event-based stabilization of nonlinear systems," *IEEE Transactions on Automatic Control*, no. 5, pp. 1332–1337, 2013.

[16] K. E. Årzén, "A simple event-based PID controller," in *Preprints of the 14th World Congress of IFAC*, 1999.

[17] M. Velasco, P. Martí, and E. Bini, "On Lyapunov sampling for event-driven controllers," in *Proceedings of the 48th IEEE Conference on Decision and Control*, 2009.

[18] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25 (NIPS 2012), Lake Tahoe, Nevada USA*, 2012, pp. 1097–1105.

[20] X. Li, G. Zhang, H. H. Huang, Z. Wang, and W. Zheng, "Performance analysis of Gpu-based convolutional neural networks," in *45th IEEE International Conference on Parallel Processing*, 2016, pp. 67–76.

[21] R. Rubinstein, "The cross-entropy method for combinatorial and continuous optimization," in *Methodology And Computing In Applied Probability*, 1999, p. 127–190.

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

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014.

[24] F. Chollet, *Deep Learning with Python*. Greenwich, CT, USA: Manning Publications Co., 2017.

[25] S. Minaee, "20 popular machine learning metrics. part 1: Classification & regression evaluation metrics," 2019. [Online]. Available: https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-regression-evaluation-metrics-1ca3e282a2ce