Automated learning rate scheduling with in-training information

Yongjun Kim
20200418

Youngjun Yu
20220665

Abstract

Learning rate is considered as the most important hyperparameter in deep neural network. It is common to adjust learning rate in training procedure by using pre-determined decay schemes like linear, cosine, step, or constant to reflect in-training dynamics. However, simply applying pre-determined strategy for learning rate is not sufficient; to accurately reflect the dynamics, consideration of various factors such as model architecture and dataset is necessary, which is not trivial. Our goal in this project is to propose an automatic and flexible method using in-training information, loss, which indirectly reflects those factors.

1 Introduction

1.1 Motivation

Choosing an appropriate learning rate is most important and difficult process in deep neural networks(DNNs). If the selected learning rate is too large, there can be instability in convergence. Conversely, if the learning rate is set too low, the network would fail to escape from local minima. To address such problems, many forms of learning rate decay have been proposed, such as linear decay, exponential decay, and time-based decay. However, simply reducing the learning rate is not sufficient; it is essential to consider various factors such as model architecture, dataset, and the number of parameters. Thus, learning rate scheduling is a fundamental technique that can significantly enhance overall performance. Our project aims to model an automatic and flexible method that can be applied regardless of various factors.

1.2 Related work

There has been tremendous attempts to make automatic and flexible learning rate scheduler using in-training information. Takase et al [6] introduced adaptive learning rate based on training loss using tree search. It choose the actual learning rate that has resulted in the smallest training loss by using search algorithm based on breadth-first beam search. Xu et al [7] introduced an adaptive learning rate schedule based on reinforcement learning (RL). It uses the validation loss as the reward, and the scaling factor as the action. Proximal policy optimization is used to automatically learn the learning rate. Some works parameterized learning rate to apply gradient descent. Baydin et al [1] introduced learning rate adaptation with hypergradient descent proposes using the hypergradient to dynamically update the learning rate. Furthermore, Retsinas et al [5] incorporated newton-based method to find appropriate learning rate scheduler during training procedure.

2 Methodology

We developed the idea of Takase et al [6], by smoothly reflect all of candidate learning rate. We assigned scores based on the normalized training losses of learning rate values applied with various

scale factors, and used thoes scores to calculate a weighted geometric mean. This weighted geometric mean is then used as the scale factor for the current epoch. Our algorithm can be formulated as Algorithm 1, and represented as in Figure 1. We used a scale factor for our algorithm to ensure it can be applied generally to various datasets, rather than selecting one among several learning rates. This is because different datasets may require different learning rate scales. We used exponential score function to amplify the influence of factors that results in significantly smaller losses. Also, we use the geometric mean because it can prevent the average from being skewed by extreme values, leading to more stable and gradual changes in the learning rate. ALR technique [6] requires calculating the training loss for beam size * number of factors, but this algorithm only needs to calculate N training losses, making it computationally less expensive.

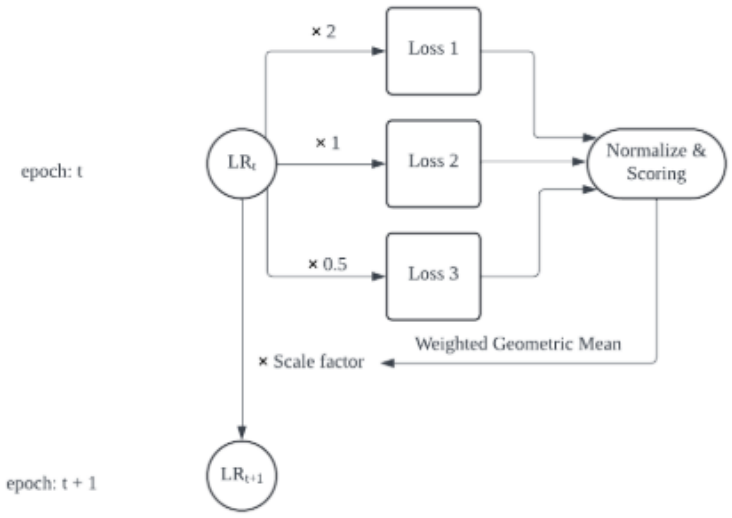

Figure 1: Schematic diagram of our method

Algorithm 1 Automated learning rate scheduler

Require: model: θ_t, factor list, learning rate: η_t, score function: $score$, loss function: L, Randomly selected batch: B
Ensure: η_{t+1}
for f in factor list **do**
  $\theta'_t = \text{copy}(\theta_t)$
  $\nabla_{\theta_t} L \leftarrow$ forward&backprop
  $\theta'_{t+1} = \theta'_t - f \cdot \eta_t \cdot \nabla_{\theta_t} L$
  Loss_list $\leftarrow L(\theta'_{t+1}, B)$
end for
Normalized_Loss_list = Normalize(Loss_list)
for loss in Normalized_Loss_list **do**
  score_list $\leftarrow$ score($loss$)
end for
$\eta_{t+1} =$ weighted_geometry_average(factor list, score list) $\cdot \eta_t$
return

3 Experiments

3.1 Experiment setting

We used Resnet18 [2] model for all of our experiments, and target dataset is CIFAR10 and CIFAR100 [3]. We used CIFAR10 dataset otherwise specified. Hyperparameters of already existing learning rate

schedulers were initialized with pytorch's default value except for learning rate, and we set learning rate as 0.005, batch size with 32. As we have limited resources, we evaluate results after 100 epochs. We use factors scaling with the power of 2. For example, for factor_7, we use following scaling values: [8, 4, 2, 1, 0.5, 0.25, 0.125].

3.2 Experiment results

3.2.1 Comparing with pre-determined learning rate schemes

First of all, we compared our proposed learning rate scheduler with existing pre-determined learning rate schemes, linear, constant, cosine, and step schemes. As shown with 1, our scheduler with 5 factors shows better test accuracy on CIFAR-10 dataset compared with pre-determined schedulers.

scheduler	CIFAR-10 test accuracy(%)
Constant	83.5
linear	85.2
cosine	85.0
step	84.4
ours	**87.0**

Table 1: Test accuracies on CIFAR-10 using different schedulers

3.2.2 Exploring effect of number of factors

We explored the effect of number of factors, which indicates number of candidate learning rates. For this experiment, we used CIFAR-100 dataset. As shown in Figure 2, increasing the number of factors can improve generalization performance, but it can also make the model stuck in suboptimal local minimum. Interestingly, we can see epoch-wise double descent phenomenon [4] for learning scheduler with factor_5, which might lead factor_5 scheduler to a better point.

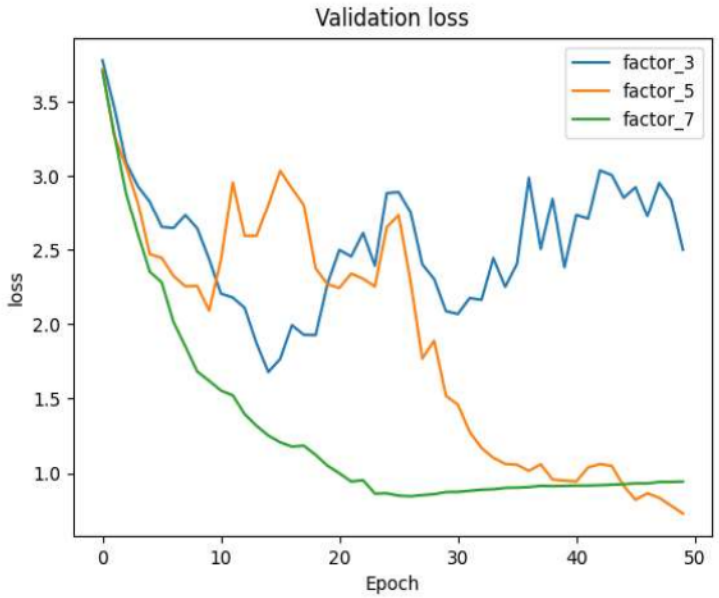

Figure 2: Validation loss with respect to the number of factors, CIFAR-100

3.2.3 Effects of number of batch size used for factor scoring

We also explored the effect of number of batch size which is used for factor scoring. As shown in Figure 3, smaller batch size shows stable learning dynamics, without showing epoch-wise double

descent phenomenon. At the end of training, all of them shows similar validation loss results. We expect as we use more complex dataset other than CIFAR-10, the effect of double descent phenomenon for larger batch size would lead to better generalization effect.

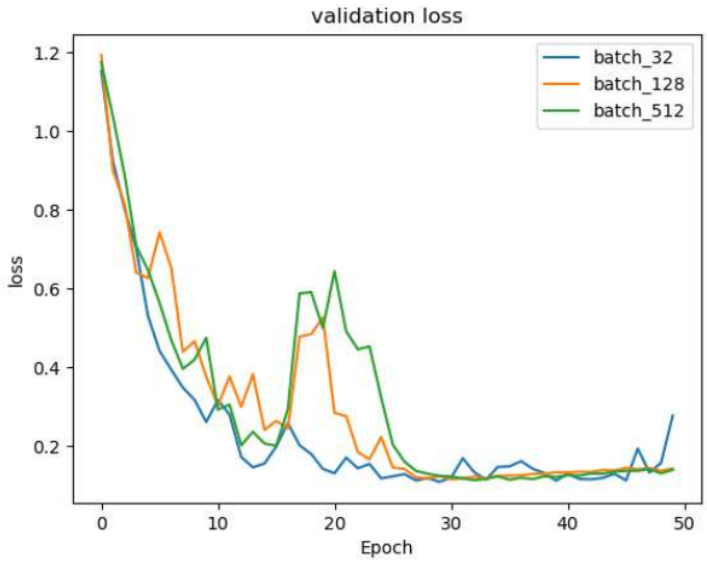

Figure 3: Validation loss with respect to the number of batch size used for factor scoring

3.2.4 Can our scheduler find appropriate learning rate with abnormally initialized learning rate?

We set initial learning rate as 1, which is 200 times larger than regular learning rate, and conducted experiment if our scheduler can find appropriate learning rate and make the result converge. However, as we use factor_7 scheduler, the result does not converge. This means that our method is not strong enough to find appropriate learning rate during training without prior knowledge.

4 Conclusion

We proposed a flexible, and automatic learning rate scheduler to adjust learning rate with regarding of loss, which indirectly reflects in-training dynamics. As a result, our method can reflect in-training dynamics more accurately compared with predetermined scheduler such as constant, linear, step, and cosine scheduler. However, since our method needs to calculate next step's training losses of candidate learning rates, the entire model should be copied with the number of learning rate candidates, which results in tremendous usage of GPU memory. Therefore, for further research, there needs to estimate next step's training loss without calculate actual gradient.

References

[1] Baydin, A.G., Cornish, R., Rubio, D.M., Schmidt, M., Wood, F.: Online learning rate adaptation with hypergradient descent. arXiv preprint arXiv:1703.04782 (2017)

[2] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR **abs/1512.03385** (2015), `http://arxiv.org/abs/1512.03385`

[3] Krizhevsky, A.: Learning multiple layers of features from tiny images (2009), `https://api.semanticscholar.org/CorpusID:18268744`

[4] Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., Sutskever, I.: Deep double descent: Where bigger models and more data hurt. Journal of Statistical Mechanics: Theory and Experiment **2021**(12), 124003 (2021)

[5] Retsinas, G., Sfikas, G., Filntisis, P.P., Maragos, P.: Newton-based trainable learning rate. In: ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1–5. IEEE (2023)

[6] Takase, T., Oyama, S., Kurihara, M.: Effective neural network training with adaptive learning rate based on training loss. Neural Networks **101**, 68–78 (2018)

[7] Xu, Z., Dai, A.M., Kemp, J., Metz, L.: Learning an adaptive learning rate schedule. arXiv preprint arXiv:1909.09712 (2019)