LEARNED LEARNING RATE SCHEDULES FOR DEEP NEURAL NETWORK TRAINING USING REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Stochastic gradient descent (SGD) is a popular optimization algorithm for training deep neural networks (DNNs), but it requires careful selection of the learning rate schedule. In this paper, we present a novel strategy to generate learned schedules for SGD using reinforcement learning (RL). Our approach trains a Proximal Policy Optimization (PPO) model to predict optimal learning rate schedules for SGD based on the training loss and progress of the training process. The RL agent is incentivized to generate schedules with low validation loss. Experiments on image classification tasks demonstrate that the schedules generated by the RL agent lead to faster convergence compared to popular fixed learning rate schedules.

1 INTRODUCTION

The optimization of deep neural networks (DNNs) has been a long-standing challenge in the field of machine learning. One of the critical hyperparameters is the learning rate schedule, which determines the step size for each iteration of the optimization process. A common approach for setting the learning rate schedule is to use a fixed schedule, such as a step decay or a cosine decay Loshchilov & Hutter (2016). However, these fixed schedules may not be optimal for all DNNs and tasks, and they may require careful tuning to achieve good performance Ruder (2016).

Recently, there has been growing interest in learned optimization algorithms, which rely on a meta-optimization process to learn an optimization algorithm from data Wichrowska et al. (2017); Metz et al. (2022). These algorithms have been shown to improve the stability and generalization performance of DNNs compared to traditional optimization algorithms Andrychowicz et al. (2016); Metz et al. (2019). Additionally, learned optimization algorithms have the ability to adapt to different tasks and architectures, making them a versatile tool for optimizing DNNs Andrychowicz et al. (2016). In recent years, several works have demonstrated the effectiveness of learned optimization algorithms for training DNNs Wichrowska et al. (2017); Metz et al. (2019). Reinforcement learning (RL) has also been applied to learned optimization, with the goal of learning an optimal policy for adjusting the optimization parameters during training Bello et al. (2017); Metz et al. (2021; 2020).

In this paper, we develop a novel learned schedule for SGD using RL. Our approach trains a Proximal Policy Optimization (PPO) model to predict the optimal learning rate schedules for SGD by using the training loss and other information that encapsulates the state of the training. Our results demonstrate that the schedules generated by the RL agent result in faster convergence and lower validation loss compared to popular fixed learning rate schedules, and that the learned optimizer can be used as a computationally efficient alternative to line-search methods.

2 LEARNED SCHEDULES USING RL

In this work, we develop a novel learned schedule for SGD. To this end, we train a RL agent to predict the optimal step size of SGD at each iteration (the learning schedule). The agent observes the following signals 1) *validation loss from previous epoch*, 2) *distance to the end of training* ($epoch_{\max} - epoch$) and 3) layer-wise gradient norms of the deep learning model. We device the following simple reward function to encourage faster convergence and penalize loss divergence.

$$r = \gamma(\texttt{best_loss} - \texttt{val_loss}) + \lambda(epoch_{\max} - epoch), \tag{1}$$

where λ, γ are hyperparameters and `best_loss` is the lowest validation loss achieved so far.

3 EXPERIMENTS AND RESULTS

Image classification. Via deep learning [1] for the MNIST and CIFAR-100 dataset, we compare the results obtained by the RL-learned schedule with SGD to that of SGD, Adadelta and Adam with a fixed learning rate found by grid search (shown in Figure 2 for SGD). As shown in Table 1 for CIFAR-100, the learned schedule leads to loss values comparable to best loss of other algorithms, and also outperforms optimizers with known schedules. Further, we observe (see Figure 3 in the Appendix) that some optimizer-scheduler combinations can lead to divergence or sub-optimal final loss values. This highlights the advantage of our proposed RL strategy as it automatically finds a good schedule without the need for the manual tuning trial and error effort. We also note that upon investigating the RL action history, we find that the agent often finds a non-trivial schedule (see Figures 4 to 6 in Appendix) where dramatic increases and decreases in the learning rate frequently take place, alluding to the complexity of the alternative manual tuning task. Lastly we see that the learned policies for the learning rate schedule are sensitive the RL agent hyperparameters, such as PPO learning rate and number of iterations.

	Agent	SGD	Adadelta	Adam	SGD-Sch	Adadelta-Sch	Adam-Sch
Final Loss	0.033	0.026	0.026	0.030	0.39	0.044	2.30

Table 1: Final Loss for different methods; "Sch" is OneCycleLR from Smith & Topin (2019)

Comparison with full line-search methods. Recently Vaswani et al. (2019) (called `SLS`) developed an optimizer for SGD that automatically sets the step size of SGD by performing a line search at each iteration of training. Given that both `SLS` and the RL agent have access to similar information at inference time, we wanted to evaluate the performance of the agent with the `SLS` optimizer and compare their respective learning rate schedules. In Figure 1 we compare `SLS` with our learned optimizer and observe that the loss curve upper bounds that of `SLS`. However, we see no common discernible pattern in the step sizes proposed by the two algorithms. We made the following observations from this experiment **a)** step sizes set by the two optimizers were very different, despite having similar performances could suggest that there can be several "good" learning schedules for a given task; **b)** it is not clear as to how to compare two learning schedules rigorously for stochastic optimizers. We delegate these questions for future investigation.

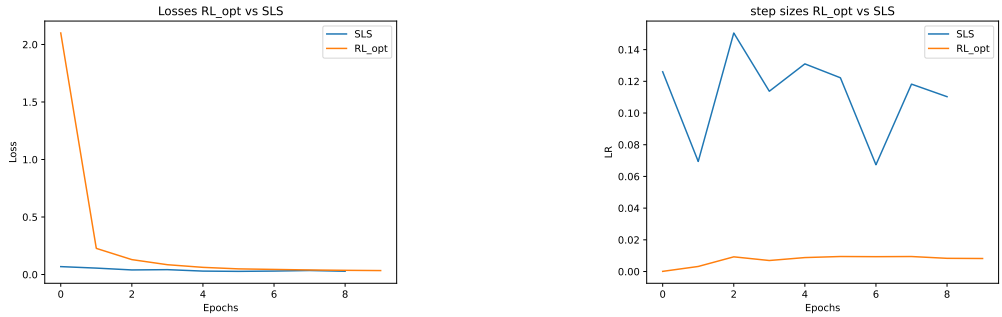

Figure 1: Our learned optimizer vs full line-search method of Vaswani et al. (2019) (`SLS`). (Left) Comparison of validation losses (right) comparison of learning rates predicted by the optimizers.

4 CONCLUSION

This paper presents a novel method for finding learned schedules for SGD using reinforcement learning. Experiments on image classification tasks showed that the schedules generated by the RL agent resulted in faster convergence and lower validation loss compared to popular fixed learning rate schedules.

[1] deep neural network similar to `https://github.com/pytorch/examples/blob/main/mnist/main.py`

URM STATEMENT

The authors acknowledge that at least one key author of this work meets the URM criteria of ICLR 2023 Tiny Papers Track.

REFERENCES

Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.

Irwan Bello, Barret Zoph, Vijay Vasudevan, and Quoc V Le. Neural optimizer search with reinforcement learning. In *International Conference on Machine Learning*, pp. 459–468. PMLR, 2017.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Luke Metz, Niru Maheswaranathan, Jonathon Shlens, Jascha Sohl-Dickstein, and Ekin D Cubuk. Using learned optimizers to make models robust to input noise. *arXiv preprint arXiv:1906.03367*, 2019.

Luke Metz, Niru Maheswaranathan, C Daniel Freeman, Ben Poole, and Jascha Sohl-Dickstein. Tasks, stability, architecture, and compute: Training more effective learned optimizers, and using them to train themselves. *arXiv preprint arXiv:2009.11243*, 2020.

Luke Metz, C Daniel Freeman, Niru Maheswaranathan, and Jascha Sohl-Dickstein. Training learned optimizers with randomly initialized learned optimizers. *arXiv preprint arXiv:2101.07367*, 2021.

Luke Metz, James Harrison, C Daniel Freeman, Amil Merchant, Lucas Beyer, James Bradbury, Naman Agrawal, Ben Poole, Igor Mordatch, Adam Roberts, et al. Velo: Training versatile learned optimizers by scaling up. *arXiv preprint arXiv:2211.09760*, 2022.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pp. 369–386. SPIE, 2019.

Sharan Vaswani, Aaron Mishkin, Issam Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. *Advances in neural information processing systems*, 32, 2019.

Olga Wichrowska, Niru Maheswaranathan, Matthew W Hoffman, Sergio Gomez Colmenarejo, Misha Denil, Nando Freitas, and Jascha Sohl-Dickstein. Learned optimizers that scale and generalize. In *International conference on machine learning*, pp. 3751–3760. PMLR, 2017.

A ADDITIONAL DEEP LEARNING EXPERIMENTS

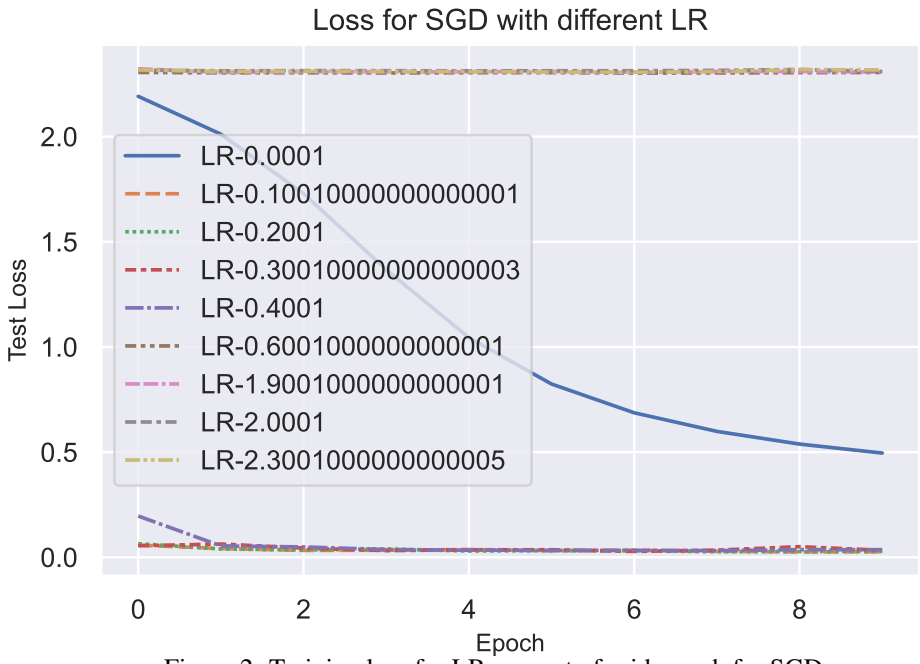

Figure 2: Training loss for LRs as part of grid search for SGD.

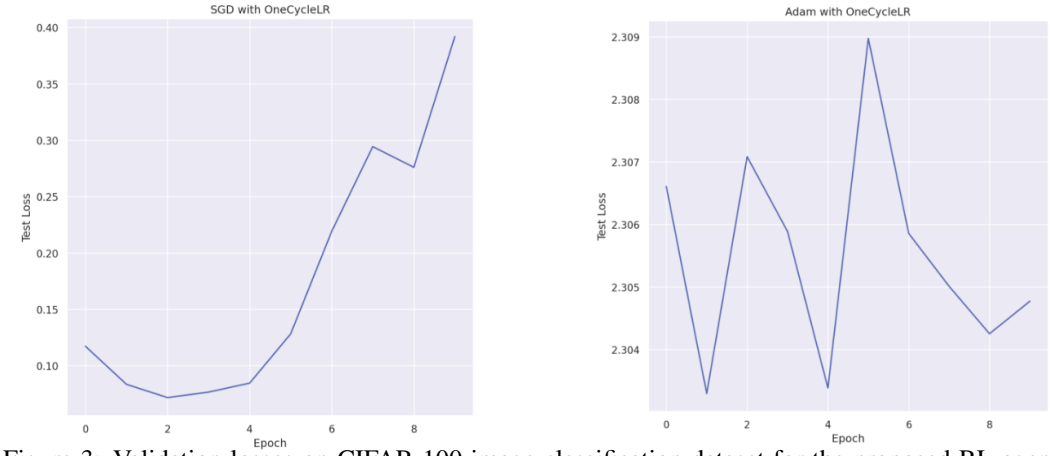

Figure 3: Validation losses on CIFAR-100 image classification dataset for the proposed RL agent along with popular optimizers with different learning rates.

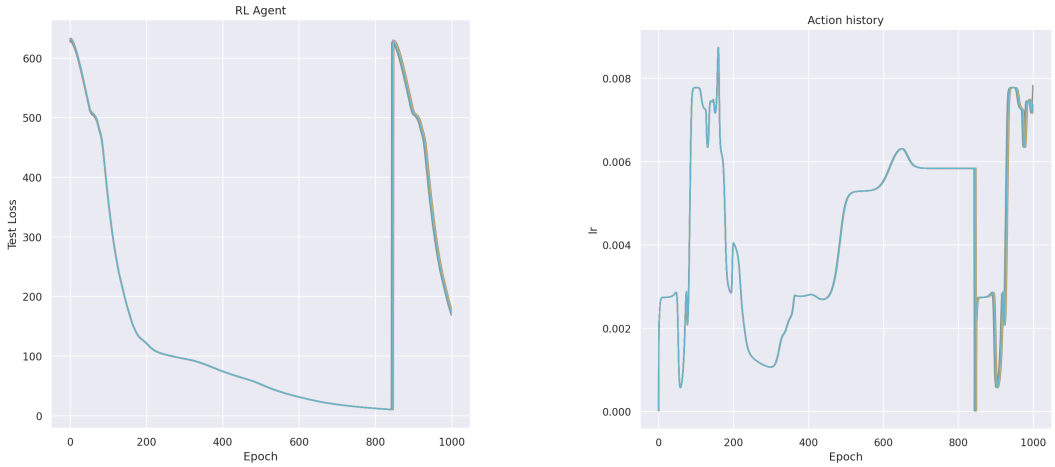

Figure 4: Validation loss and action histories of the proposed agent after $10K$ training iterations on MNIST image classification task.

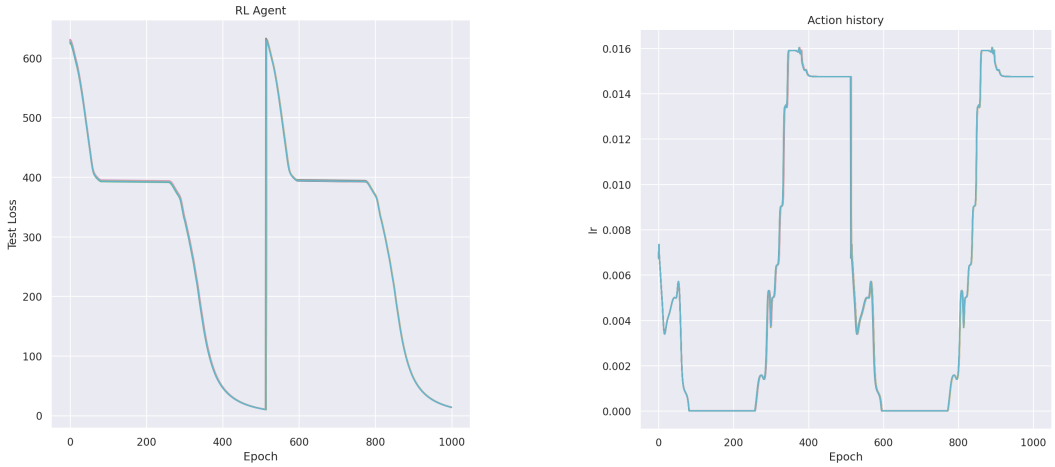

Figure 5: Validation loss and action histories of the proposed agent after $20K$ training iterations on MNIST image classification task.

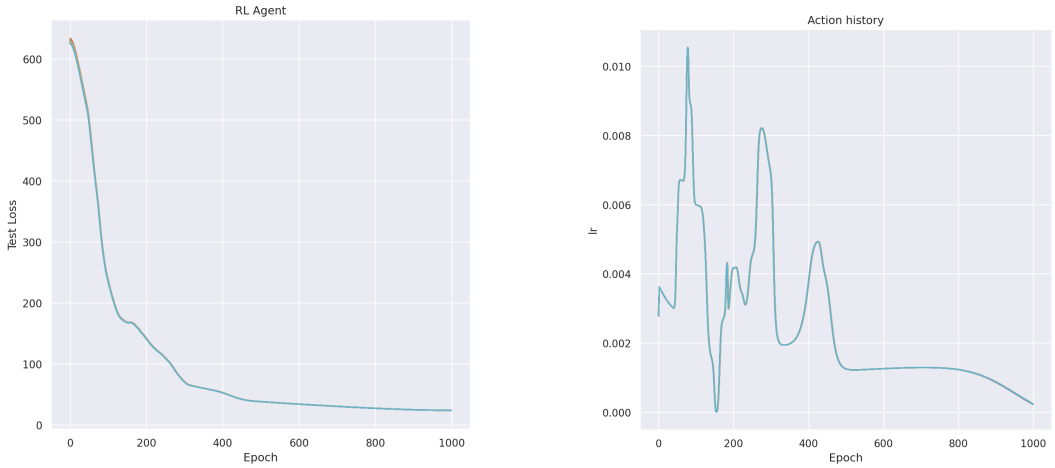

Figure 6: Validation loss and action histories of the proposed agent after $50K$ training iterations on MNIST image classification task.