Learning Rate Schedules

Deep Learning Research Kitchen

Robin Uhrich

June 6, 2024

Outline

1. Background - Why do we schedule learning rates in DL?
2. Which LR schedules are used in practice?
3. Experiments
4. Scheduling other Hyperparameter
5. Conclusion

Outline

1 Background - Why do we schedule learning rates in DL?

2 Which LR schedules are used in practice?

3 Experiments

4 Scheduling other Hyperparameter

5 Conclusion

Background

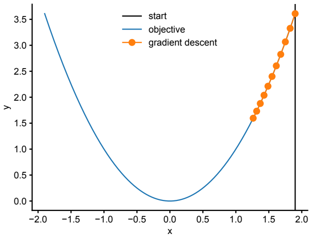
Motivation; Murphy 2022, Goodfellow *et al.* 2016

- Empirical risk minimization

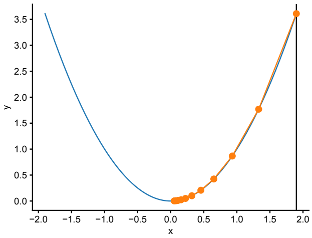
$$\theta^* = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i, f(x_i; \theta))$$

- In deep learning it is a non convex optimization problem
- Find a solution through gradient descent.

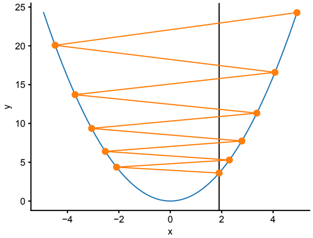
$$\theta_{t+1} = \theta_t - \alpha \nabla_{\theta_t} f(\boldsymbol{x}, \theta_t)$$



α is very small



α is reasonably big

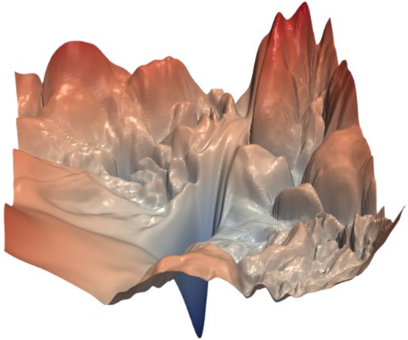


α is to big big

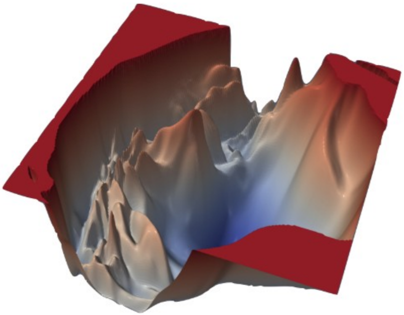
Background

Motivation

Loss surfaces from Li *et al.* 2018



ResNet-56



ResNet-110

Background

The problem with constant learning rates; Zhang *et al.* 2023

The idea is: a preset constant learning rate throughout the training is not optimal.

- Too high: Large jumps, potential for instability and divergence.
- Too low: Slow convergence, risk of getting stuck in local minima.

Background

what is a learning rate scheduler?; Zhang *et al.* 2023

Approach: take the learning parameter α not as a constant but as a function of t

$$\theta_{t+1} = \theta_t - \alpha(t)\nabla_{\theta_t} f(\mathbf{x}, \theta_t)$$

With this we try to enable:

- Early rapid exploration with high learning rates.
- Finer adjustments and convergence with small learning rates.
- Higher stability and avoids getting stuck.

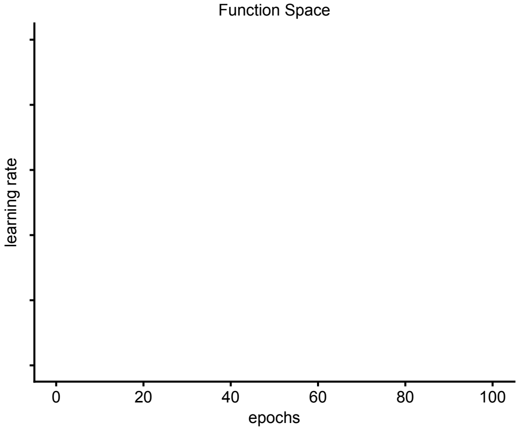
Outline

1. Background - Why do we schedule learning rates in DL?
2. Which LR schedules are used in practice?
3. Experiments
4. Scheduling other Hyperparameter
5. Conclusion

Commonly used Learning-Rate Scheudles

Function space

We have a whole function space for $\alpha : \mathbb{R} \to \mathbb{R}$
Which functions are reasonable?



Commonly used Learning-Rate Scheudler

ConstantLR Schedule

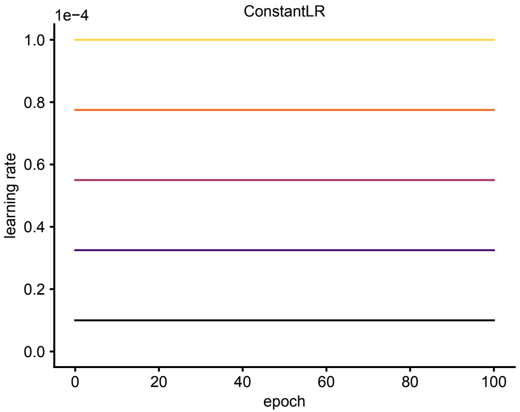
constant learning rate over the whole training.

Advantages

- Trivial implementation
- Useful baseline

Disadvantages

- More sensitive to high learning rates
- Has to comprehend for fast exploration and refinement



Commonly used Learning-Rate Scheudler

StepLR Schedule

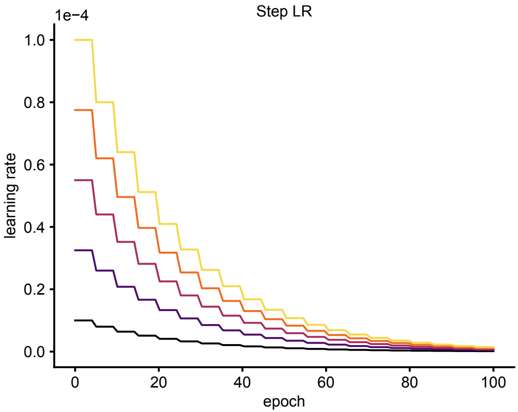
Multiply learning rate by constant factor every k epochs.

Advantages

- Aggressive initial learning rate
- Smooth decay

Disadvantages

- Potentially overshoots the solution
- Limited adaptability



Commonly used Learning-Rate Scheudler

Cosine Annealing Schedule

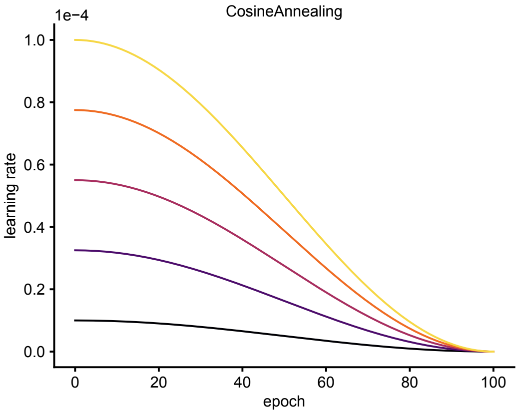
Work done by: Loshchilov and Hutter 2016 Follow the first half of a cosine period

Advantages

- Reduces the overshot problem
- Smooth decay

Disadvantages

- Potentially slow convergence (not so aggressive lr in the beginning)
- More parameters to tune



Commonly used Learning-Rate Scheudler

Warmup Schedules

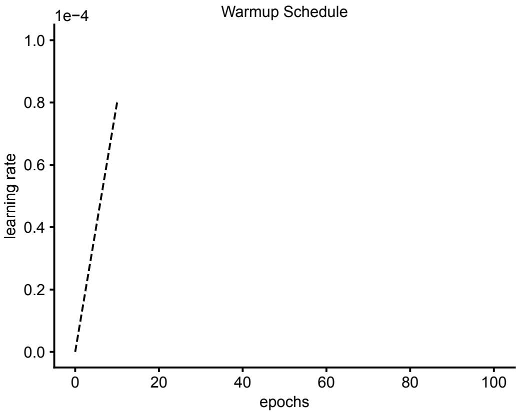
One of the first appearance w.r.t DL in He *et al.* 2016.

1. Gradually increasing the learning rate until a maximum
2. Continue with any schedule you like

Advantages (Gotmare *et al.* 2018):

- Lowers the amount of divergence of parameters by the end of training
- More stable training for higher learning rates
- Can improve training

But it introduces also new hyperparameter



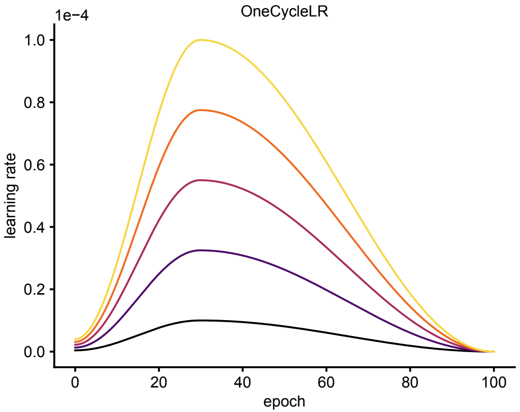
Commonly used Learning-Rate Scheudler

OneCycle Schedules

Proposed by: L. N. Smith and Topin 2019

1. Gradually increasing the learning rate until a maximum
2. Staying on a plateau for exploration
3. Decreasing the learning rate for fine-tuning

Comes with the same advantages as warmup schedules



Commonly used Learning-Rate Scheudler

Cyclic Learning Rate Schedules

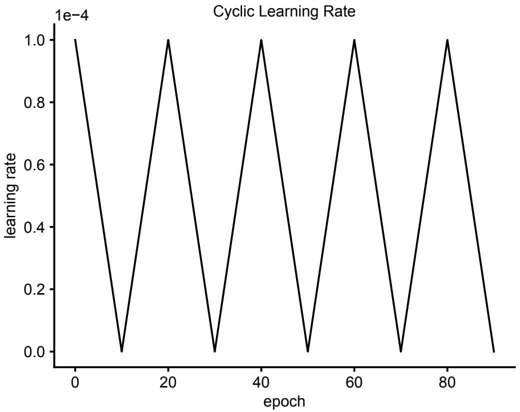
Proposed by: Schaul *et al.* 2013, L. N. Smith 2017
oscillate between min and max learning rate.

Advantages

- Explores broader region
- More flexible
- Avoiding getting stuck

Disadvantages

- More complex to tune
- Slow convergence
- Prone to over-fit

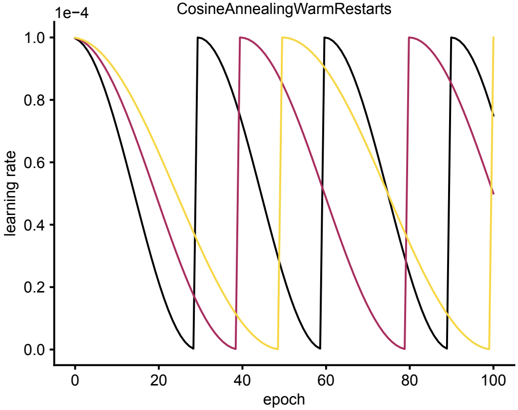


Commonly used Learning-Rate Scheudler

Cosine Annealing with Warm Restarts

Proposed by: Loshchilov and Hutter 2016
Cyclical learning rate schedule with cosine as decay.

- Single decaying cycle
- Less parameters to tune
- Focuses on smooth learning rate decay
- Works well with an over all decay trend Gotmare *et al.* 2018



Commonly used Learning-Rate Scheudler

Wrapup

Decaying Schedules:

- Step Decay: Periodical exponential decay
- Exponential Decay: Continuously differentiable version of step decay
- Cosine Annealing: Reduce learning rate following a cosine curve
- Linear Decay: Linear decay

Option to expand those with warmup

Cyclic:

- Cyclic Learning Rate: Bouncing between max mal and minimal learning rate
- Cosine Annealing with Warm Restarts: Decay with cosine and jump back to maximal learning rate
- OneCycle: combines the content of cyclic lr schedules in one cycle

Others like piecewise constant or something else are also valid

Outline

1 Background - Why do we schedule learning rates in DL?
2 Which LR schedules are used in practice?
3 Experiments
4 Scheduling other Hyperparameter
5 Conclusion

Experiments

Setup

- Task: Image Classification
- Dataset: CIFAR10 Krizhevsky, Hinton, *et al.* 2009
- architecture: small ViT (approx. 12.5M params scaled down version from Dosovitskiy *et al.* 2020 at omihub777 2021)
- optimizer: AdamW with weight decay 10^{-4} and different learning rates Loshchilov and Hutter 2017
- Training for 100 epochs
- Batch size: 512 training, 2048 validation and testing

Metrics:

- Cross Entropy
- Accuracy
- Test best model w.r.t. validation accuary

Various configured learning rate scheduler

Experiments

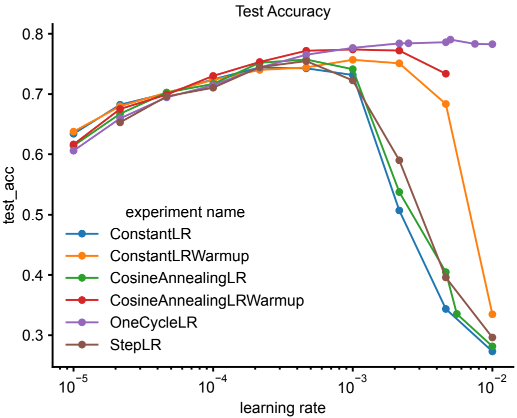
Scheduler Search Space

Scheduler	Parameter	Search Space
ConstantLR (warmup)	factor = 1, *	lr $\in [10^{-6}, 10^{-2}]$
CosineAnnealingLR (warmup)	$T_{\text{max}} = 100$, $\eta_{\text{min}} = 10^{-6}$, *	lr $\in [10^{-5}, 10^{-2}]$
StepLR	step size = 5, $\gamma = 0.9$	lr $\in [10^{-5}, 10^{-2}]$
OneCycleLR	total steps = 100	max_lr $\in [10^{-5}, 10^{-2}]$
CosineAnnealingWarmRestarts	lr $\approx 4.6 \cdot 10^{-4}$	$T_0 \in [10, 50]$

*: for the warmup version we choose 6 warmup epochs

Test accuracy per deployed learning rate factor

- While increasing the maximal learning rate, predictor performance varies across different schedules
- But we can spot similarities between schedules



Experiments

Test Accuracy Groups

OneCycle Learning Rate:

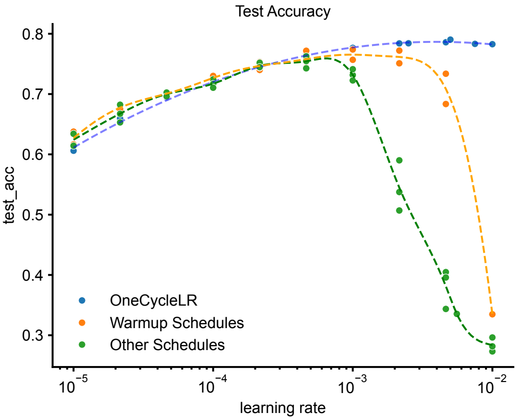
- Seems to work best in the current setup
- Able to handle high maximal learning rate

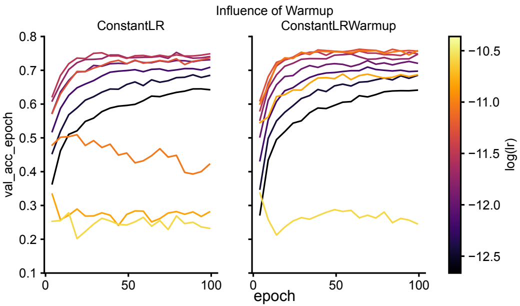
Purly Decaying / Constant

- Similar to OneCycle for small learning rates
- Begin to diverge very early

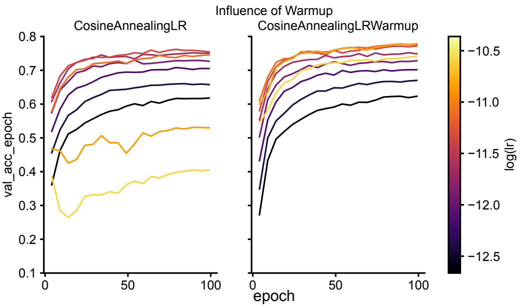
Warmup Schedules:
(ConstantLR and CosineAnnealingLR with Warmup)

- Seems to produce slightly better results
- Makes training more stable





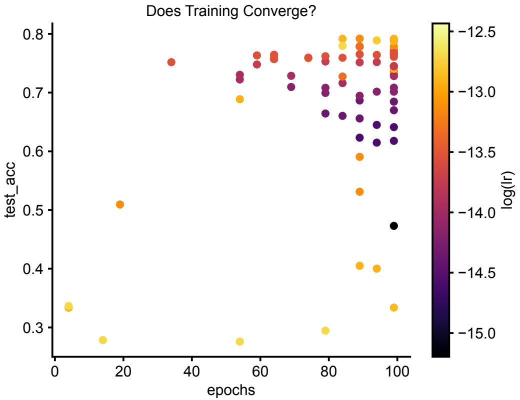
Constant Learning Rate



Cosine Annealing

At which epoch was the model at its best and how high was the learning rate?

- Convergence region (upper right): smooth gradient towards maximal accuracy late in the training process
- Divergence / under-fit region (lower left): high learning rates and low accuracy early in training

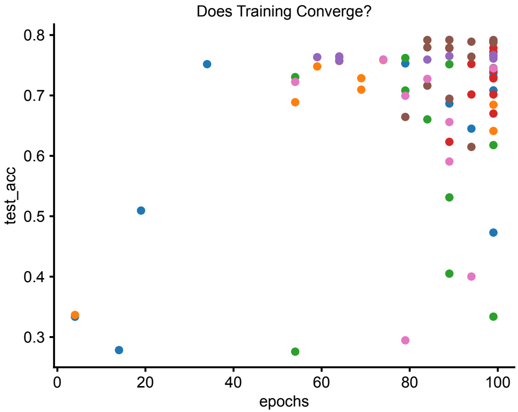


Experiments

Convergence / Divergence

At which epoch was the model at its best and how high was the learning rate?

- Convergence region (upper right): smooth gradient towards maximal accuracy late in the training process
- Divergence / under-fit region (lower left): high learning rates and low accuracy early in training



Experiments

Limitations

Setup

- Dataset size. Other datasets might reveal a higher variance in training behavior
- Model Complexity: The simplified ViT could be not as sensitive as large scale models

Training

- Limited exploration space
- Tuning only for scheduler hyperparameter, leaving other parameters constant
- Training duration
- Used metrics could be expanded also F1 score, …
- No statistical significance conducted.

Outline

1 Background - Why do we schedule learning rates in DL?

2 Which LR schedules are used in practice?

3 Experiments

4 Scheduling other Hyperparameter

5 Conclusion

Scheduling other Hyperparameter

Overview

Learning rate is not the only hyperparameter that benefits from scheduling. Benefits:

- Fine-tuning hyperparameter values throughout training can lead to better performance and stability.
- Allows for more sophisticated training strategies that adapt to the learning process.

Exmaples of current research schedules 3 other parameter:

- Batch Size: S. L. Smith *et al.* 2017
- Momentum: Sun *et al.* 2021
- Weight Decay: Xie *et al.* 2024

But you can basically schedule everything you want.

Scheduling other Hyperparameter

Batch Size

Work done by: S. L. Smith *et al.* 2017

Instead of increasing the learning rate they propose to increase the batch-size.

- More accurate estimate of the true gradient
- Update step size is proportional to both the learning rate and the batch size $\rightarrow$ batch size effectively reduces the learning rate

Advantages

- Reduced the number of parameter-updates required
- Their scaling rules enable them to use existing hyperparameter-configurations

Scheduling other Hyperparameter

Momentum

Work done by Sun *et al.* 2021

Problem: Momentum β as fixed hyperparameter. Setting it could be quite challenging

$$\theta_{t+1} = \theta_t - \alpha \nabla_{\theta_t} f(x; \theta_t) + \beta(\theta_t - \theta_{t-1})$$

Solution: Adaptive heavy ball momentum (Polyak momentum), inspired by the optimal choice of momentum for quadratic optimization problems. Adjusts automatically based on past gradients $\rightarrow$ no manual tuning needed Advantages:

- Convergences faster than those with fixed momentum.
- More robust w.r.t. large learning rates
- Might generalize better to unseen data.

Scheduling other Hyperparameter

Weight Decay

Work done by: Xie *et al.* 2024

Problem: Weight decay is a regularization technique, helps prevent over-fitting. But large weight decay can lead to large gradient norms during the final stages of training. This could lead to: Destabilize training, Hinder convergence
Solution: Paper proposes Scheduled-Weight-Decay (SWD), dynamically adjusts the weight decay strength based on the gradient norm.

- High Gradient Norm - Lower Weight Decay
- Low Gradient Norm - Higher Weight Decay

This feedback loop leads to:

- Simpler Hyperparameter Tuning
- Improved Convergence
- Better Generalization

Outline

1 Background - Why do we schedule learning rates in DL?
2 Which LR schedules are used in practice?
3 Experiments
4 Scheduling other Hyperparameter
5 Conclusion

Conclusion

- Learning rate scheduler are a reasonable aspect in improving training neural networks
  - They can speed up training
  - Find better optima
  - Stabilize training
- But the also open up a huge parameter space to optimize.
- Tuning one schedule does not mean we can map those result on any other schedule.
- There is no single 'best' schedule for every model.
- Scheduling other parameters could be use-full for boosting performance. But still come at the cost of tuning additional parameters.

References I

- K. P. Murphy, *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. [Online]. Available: `probml.ai`.
- I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
- H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, “Visualizing the loss landscape of neural nets,” *Advances in neural information processing systems*, vol. 31, 2018.
- A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*. Cambridge University Press, 2023, `https://D2L.ai`.
- I. Loshchilov and F. Hutter, “Sgdr: Stochastic gradient descent with warm restarts,” *arXiv preprint arXiv:1608.03983*, 2016.
- K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- A. Gotmare, N. S. Keskar, C. Xiong, and R. Socher, “A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation,” *arXiv preprint arXiv:1810.13243*, 2018.

References II

- L. N. Smith and N. Topin, “Super-convergence: Very fast training of neural networks using large learning rates,” in *Artificial intelligence and machine learning for multi-domain operations applications*, SPIE, vol. 11006, 2019, pp. 369–386.
- T. Schaul, S. Zhang, and Y. LeCun, “No more pesky learning rates,” in *International conference on machine learning*, PMLR, 2013, pp. 343–351.
- L. N. Smith, “Cyclical learning rates for training neural networks,” in *2017 IEEE winter conference on applications of computer vision (WACV)*, IEEE, 2017, pp. 464–472.
- A. Krizhevsky, G. Hinton, *et al.*, “Learning multiple layers of features from tiny images,”, 2009.
- A. Dosovitskiy *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.
- omihub777, “Vit-cifar,” 2021. [Online]. Available: `https://github.com/omihub777/ViT-CIFAR`.
- I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” *arXiv preprint arXiv:1711.05101*, 2017.

References III

- S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le, “Don’t decay the learning rate, increase the batch size,” *arXiv preprint arXiv:1711.00489*, 2017.
- T. Sun, H. Ling, Z. Shi, D. Li, and B. Wang, “Training deep neural networks with adaptive momentum inspired by the quadratic optimization,” *arXiv preprint arXiv:2110.09057*, 2021.
- Z. Xie, Z. Xu, J. Zhang, I. Sato, and M. Sugiyama, “On the overlooked pitfalls of weight decay and how to mitigate them: A gradient-norm perspective,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- Y. Jin *et al.*, *Autolrs: Automatic learning-rate schedule by bayesian optimization on the fly*, 2021. arXiv: `2105.10762 [cs.LG]`.

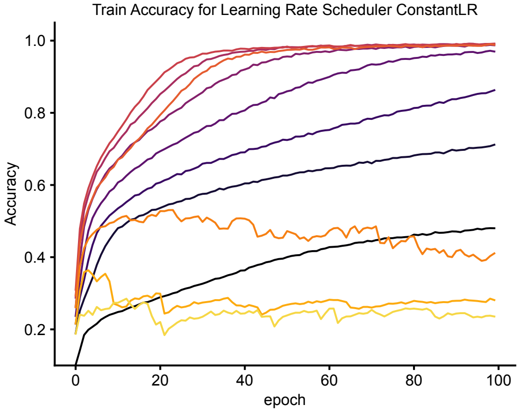
Thank you for your attention

Thank you for your attention.

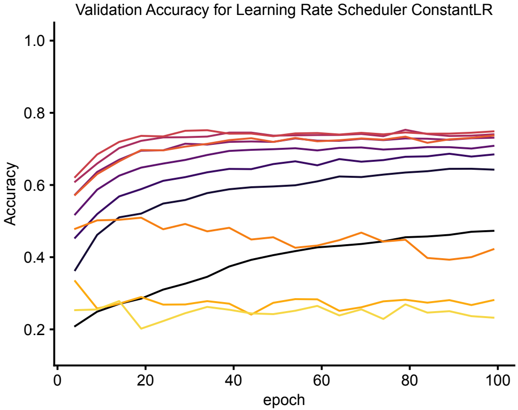
Questions?

GitHub:
https://github.com/RobinU434/DeepLearningResearchKitchen.git

Appendix

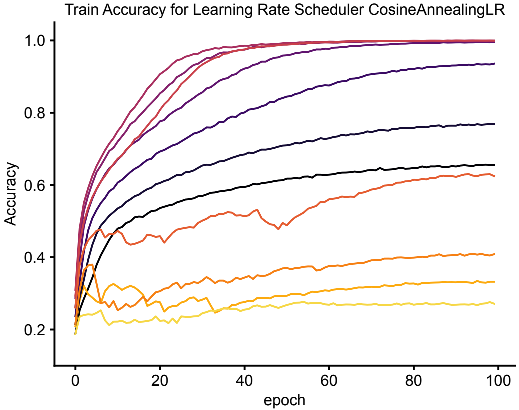


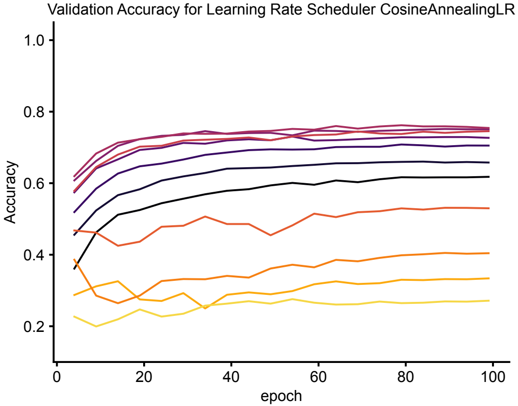
Train Accuracy for Learning Rate Scheduler ConstantLR
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch



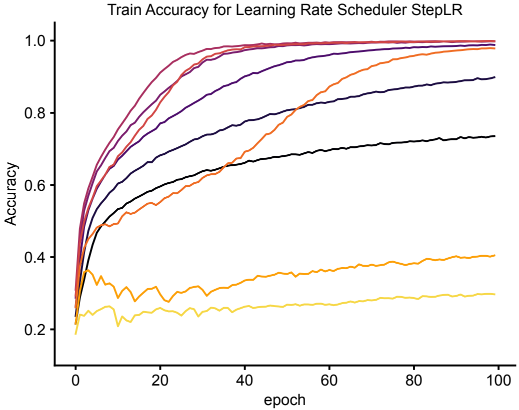
Validation Accuracy for Learning Rate Scheduler ConstantLR
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch

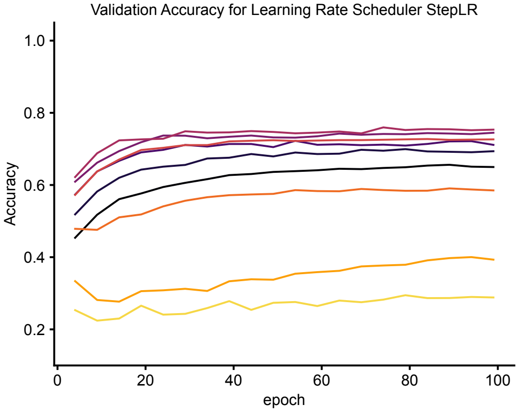
Appendix


Train Accuracy for Learning Rate Scheduler CosineAnnealingLR
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch

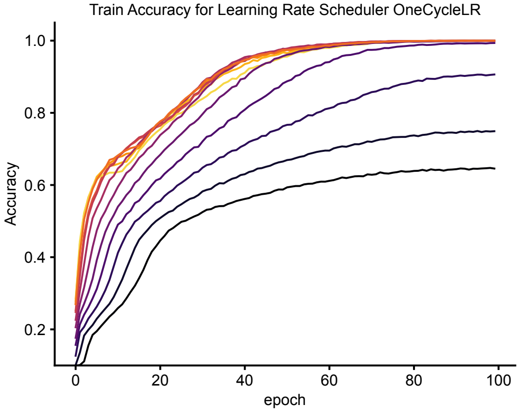

Validation Accuracy for Learning Rate Scheduler CosineAnnealingLR
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch

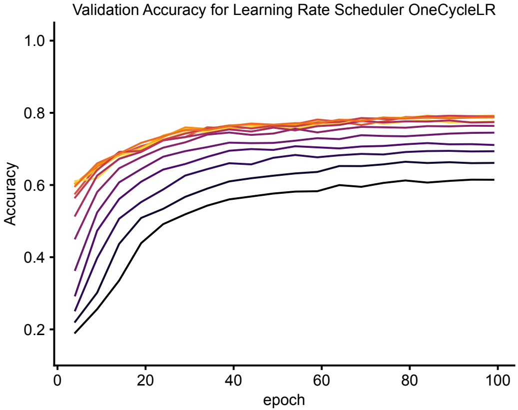
Appendix


Train Accuracy for Learning Rate Scheduler StepLR
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch

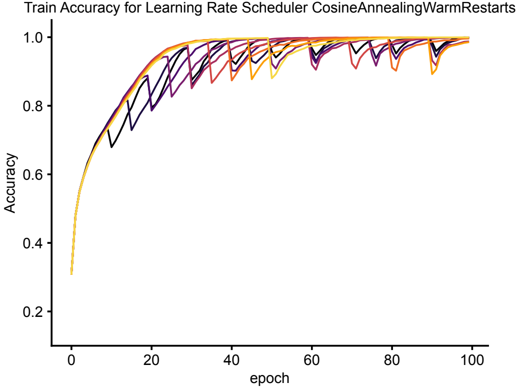

Validation Accuracy for Learning Rate Scheduler StepLR
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch

Appendix

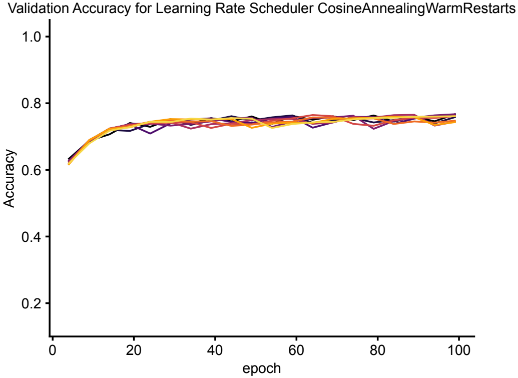

Train Accuracy for Learning Rate Scheduler OneCycleLR
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch


Validation Accuracy for Learning Rate Scheduler OneCycleLR
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch

Appendix



Train Accuracy for Learning Rate Scheduler CosineAnnealingWarmRestarts
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch



Validation Accuracy for Learning Rate Scheduler CosineAnnealingWarmRestarts
Accuracy
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch

$$
\begin{aligned}
\mathbf{x}^{k+1} &= \mathbf{x}^{k} - \gamma \mathbf{g}^{k} + \beta_k (\mathbf{x}^{k} - \mathbf{x}^{k-1}), \\
\beta_{k+1} &= \mathbf{Proj}_{[0,1-\delta]} \left(\left[1 - \sqrt{\gamma \frac{\|\mathbf{g}^{k} - \mathbf{g}^{k-1}\|}{\|\mathbf{x}^{k} - \mathbf{x}^{k-1}\|}} \right]^2 \right),
\end{aligned}
$$

When your training budget is infinite you can follow approaches like:

- Cyclical Learning Rates (CLR) for Long-Range Optimization: multiple cycles of increasing and decreasing the learning rate, allowing the model to explore a wider range of learning rates and potentially avoid getting stuck in local minima.
- AutoLRS: Automatic Learning-Rate Schedule by Bayesian Optimization on the Fly Jin *et al.* 2021