LRTuner: A Learning Rate Tuner for Deep Neural Networks

Nikhil Iyer*, Thejas Venkatesh*, Nipun Kwatra, Ramachandran Ramjee, Muthian Sivathanu
Microsoft Research, India

* denotes equal contribution

Motivation

- Learning rate is one of the most important hyperparameters for generalization
- The state space of learning rate schedules is infinite
- Standard learning rate schedules do not fully exploit the properties of neural loss landscapes
- SGD requires higher learning rates while Adam typically operates with a lower learning rate
- Automatic schedules do not generalize well, or take more time to generalize, costing precious GPU compute

Questions we ask:

- Can we design an automatic tuner that tunes the learning rate optimally during the current time step?
- Can we show both generalization capabilities as well as wall clock time savings by incorporating certain methods that take advantage of the properties of loss landscapes?

Quadratic Approximation

- We approximate the loss function for the next time step as a quadratic polynomial by applying Taylor series expansion.
- A small perturbation is applied to the current learning rate and the loss is expanded as a function of the perturbation.

$$\eta \implies \eta + \epsilon$$

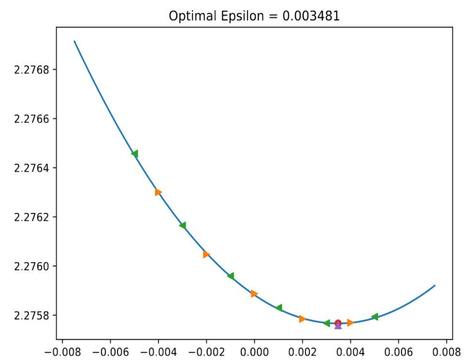
$$\hat{L}(\epsilon) = L(\theta - (\eta+\epsilon)\vec{d}) = L(\theta - \eta\vec{d} - \epsilon\vec{d})$$

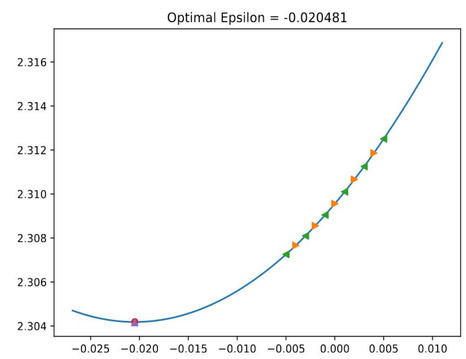
$$= L(\theta - \eta\vec{d}) - \epsilon\vec{d}^{T}\vec{g} + \frac{1}{2}\epsilon^2\vec{d}^{T}H\vec{d} + O(\epsilon^3)$$

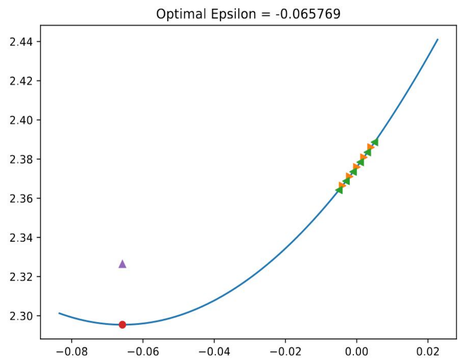
$$= k_0 + k_1\epsilon + k_2\epsilon^2 + O(\epsilon^3)$$

- To evaluate the above coefficients, we compute the loss at a few values of epsilon and fit a quadratic curve.
- We find the optimal epsilon value for the next time step by minimizing the quadratic.
- This method is independent of any optimizer, as we only need to access the search direction to compute the loss samples.
- We do the quadratic approximation once every few minibatches to reduce computational cost.

Accuracy of Quadratic Approximation



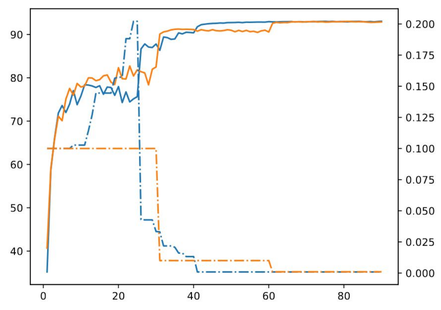




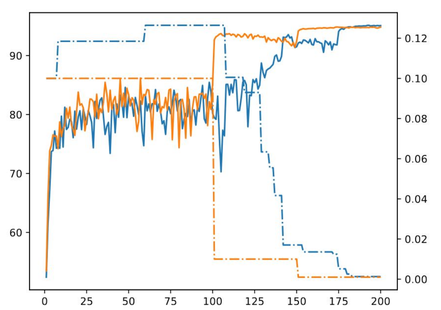
We demonstrate the quality of our quadratic estimation approach on Cifar-10 with Resnet-18. The quadratic approximation is valid only for small errors. The left and the middle figures have small perturbations and the quadratic approximation is accurate. The right figure shows a large optimal perturbation, causing the approximation error to blow up. To combat this, we clip ϵ as shown below

$$O(\epsilon^3) \approx |\epsilon^3| \implies |\epsilon^3| < r * L(\theta)$$

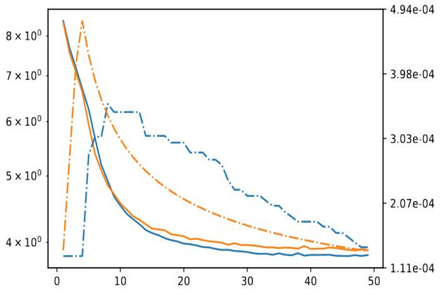
Performance of LRTuner



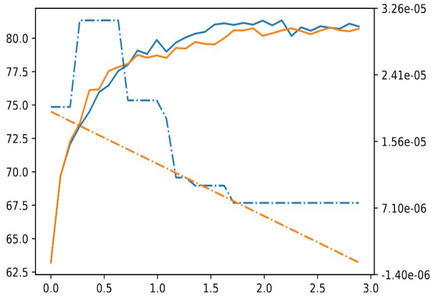
Imagenet Top-5 Accuracy with Resnet-50



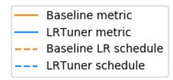
Cifar-10 Top-1 Accuracy with Resnet-18



IWSLT'14(DE-EN) validation perplexity with Transformer-BASE



SQuAD v1.1 EM score with BERT-BASE



Plots for the metrics computed by the baseline schedule and LRTuner. X-axis denotes the total epochs for training, Y-axis left denotes the absolute value of the metric and Y-axis right denotes the LR range.

To improve generalization, we employ an explore-exploit scheme within LRTuner. The explore duration only allows an increase in learning rate if suggested by LRTuner. This large learning rate phase allows the optimizer to escape narrow minima and land in wider minima, which are known to generalize well. We usually set explore to 20-30% of the total training budget.

The exploit phase only allows a decrease in learning rate, if suggested by LRTuner. This allows the optimizer to descend into the wider minimum found during explore. This two-phase scheme improves generalization significantly. We also show that we can generalize as well as baseline schedules in fewer training steps, leading to wall clock time savings.