Lookahead Optimizer: k steps forward, 1 step back

Michael R. Zhang James Lucas Geoffrey Hinton

Jimmy Ba

Department of Computer Science, University of Toronto, Vector Institute michael, jlucas, hinton,jba@cs.toronto.edu

Introduciton

Recent attempts to improve SGD can be broadly categorized into two approaches:

(1) adaptive learning rate schemes,such as AdaGrad and Adam,

(2) accelerated schemes, such as Polyak heavyball and Nesterov momentum.

Both approaches make use of the accumulated past gradient information to achieve faster convergence. However, to obtain their improved performance in neural networks often requires costly hyperparameter tuning.

Algorithm 1 Lookahead Optimizer:

Require: Initial parameters ϕ_0, objective function L
Require: Synchronization period k, slow weights step size α, optimizer A
for $t = 1, 2, \ldots$ **do**
  Synchronize parameters $\theta_{t,0} \leftarrow \phi_{t-1}$
  for $i = 1, 2, \ldots, k$ **do**
    sample minibatch of data $d \sim \mathcal{D}$
    $\theta_{t,i} \leftarrow \theta_{t,i-1} + A(L, \theta_{t,i-1}, d)$
  end for
  Perform outer update $\phi_t \leftarrow \phi_{t-1} + \alpha(\theta_{t,k} - \phi_{t-1})$
end for
return parameters ϕ

Figure 1: Lookahead psuedocode

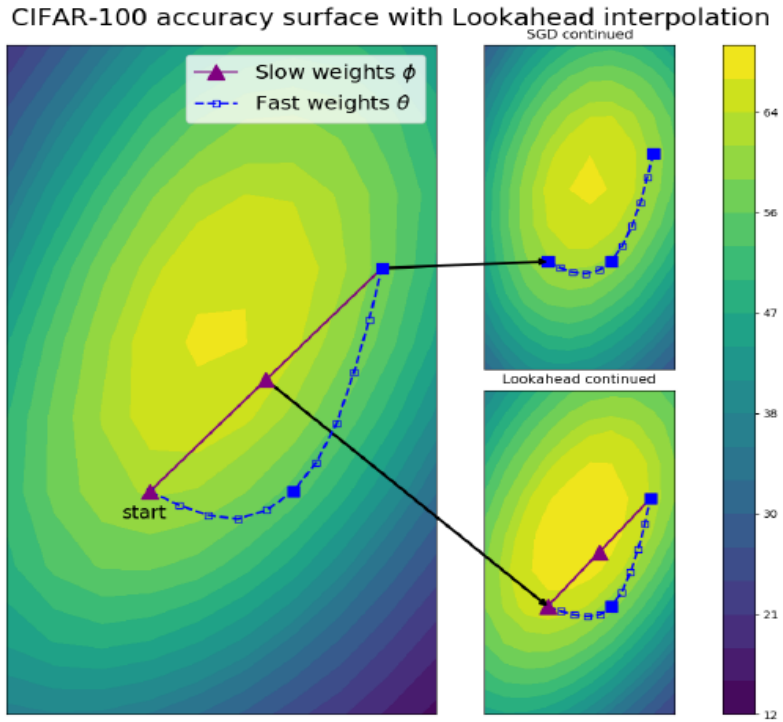


Figure 1 shows the trajectory of both the fast weights and slow weights during the optimization of a ResNet-32 model on CIFAR-100. While the fast weights explore around the minima, the slow weight update pushes Lookahead aggressively towards an area of improved test accuracy.

Slow weights trajectory

The slow weights as an exponential moving average (EMA) of the final fast weights within each inner-loop, regardless of the inner optimizer. After k inner-loop steps we have:

$$\begin{aligned}\phi_{t+1} &= \phi_t + \alpha(\theta_{t,k} - \phi_t) \\ &= \alpha[\theta_{t,k} + (1-\alpha)\theta_{t-1,k} + ... + (1-\alpha)^{t-1}\theta_{0,k}] + (1-\alpha)^t\phi_0 \quad (0.1)\end{aligned}$$

Fast weights trajectory

Within each inner-loop, the trajectory of the fast weights depends on the choice of underlying optimizer. Given an optimization algorithm A that takes in an objective function L and the current mini-batch training examples d, we have the update rule for the fast weights:

$$\theta_{t,i+1} = \theta_{t,i} + A(L, \theta_{t,i}, d) \tag{0.2}$$

Computational complexity

Lookahead has a constant computational overhead due to parameter copying and basic arithmetic operations that is amortized across the k inner loop updates. The number of operations is $O((k + 1/k))$ times that of the inner optimizer. Lookahead maintains a single additional copy of the number of learnable parameters in the model.

Selecting the Slow Weights Step Size

The step size in the direction $\theta_{t,k} - \theta_{t,0}$ is controlled by α. By taking a quadratic approximation of the loss, we present a principled way of selecting α.

Proposition 1 (Optimal slow weights step size). For a quadratic loss function$L(x) = \frac{1}{2}x^T Ax - b^T x$ the step size α^* that minimizes the loss for two points $\theta_{t,0}$ and $\theta_{t,k}$;k is given by:

$$\alpha^* = arg\min_{\alpha} L(\theta_{t,0} + \alpha(\theta_{t,k} - \theta_{t,0})) = \frac{(\theta_{t,0} - \theta^*)^T A(\theta_{t,0} - \theta_{t,k})}{(\theta_{t,0} - \theta_{t,k})^T A(\theta_{t,0} - \theta_{t,k})} \quad (0.3)$$

where $\theta^* = A^{-1}b$ minimizes the loss.

Proof Compute the dderivative with respect to α

$$\nabla L(\theta_{t,0}+\alpha(\theta_{t,k}-\theta_{t,0})) = (\theta_{t,k}-\theta_{t,0})^T A(\theta_{t,0}+\alpha(\theta_{t,k}-\theta_{t,0}))-(\theta_{t,k}-\theta_{t,0})^T b \tag{0.4}$$

Setting the derivative to 0 and using $b = A\theta^*$:

$$\alpha[(\theta_{t,k} - \theta_{t,0})^T A(\theta_{t,k} - \theta_{t,0})] = \theta_{t,k} - \theta_{t,0})^T A(\theta^* - \theta_{t,0})$$

$$\Rightarrow \alpha^* = arg\min_{\alpha} L(\theta_{t,0} + \alpha(\theta_{t,k} - \theta_{t,0})) = \frac{(\theta_{t,0} - \theta^*)^T A(\theta_{t,0} - \theta_{t,k})}{(\theta_{t,0} - \theta_{t,k})^T A(\theta_{t,0} - \theta_{t,k})} \tag{0.5}$$

Convergence Analysis

Model:Noisy quadratic model

$$\hat{L} = \frac{1}{2}(x-c)^T A(x-c) \tag{0.6}$$

with $c \sim N(x^*, \sum)$. A and $\sum$ is diagonal,$x^* = 0$,We use a_i and σ_i^2 to denote the diagonal elements of A and $\sum$. Taking the expectation over c, the expected loss of the iterates $\theta^{(t)}$ is,

$$\begin{aligned} L(\theta^{(t)}) = E[\hat{L}(\theta^{(t)})] &= \frac{1}{2}E[\sum_i a_i(\theta_i^{(t)^2} + \sigma_i^2)] \\ &= \frac{1}{2}\sum_i a_i(E[\theta_i^{(t)}]^2 + V[\theta_i^{(t)}] + \sigma_i^2) \end{aligned} \tag{0.7}$$

Proposition 2 (Lookahead variance reduction). Let $0 < \gamma < \frac{2}{L}$ be the learning rate of SGD and Lookahead where $L = \max a_i$. In the noisy quadratic model, the iterates of SGD and Lookahead with SGD as its inner optimizer converge to 0 in expectation and the variances converge to the following fixed points:

$$V^*_{SGD} = \frac{\gamma^2 A^2 \sum^2}{I - (I - \gamma A)^2} \tag{0.8}$$

$$V^*_{LA} = \frac{\alpha^2(I - (I - \gamma A)^{2k})}{\alpha^2(I - (I - \gamma A)^{2k}) + 2\alpha(1 - \alpha)(I - (I - \gamma A)^k)} V^*_{SGD} \tag{0.9}$$

Stochastic dynamics of SGD From Wu et al. [42], we can compute the dynamics of SGD with learning rate γ as follows:

$$E[x^{t+1}] = (I - \gamma A)E[x^(t)] \tag{0.10}$$

$$V[x^{t+1}] = (I - \gamma A)^2 V[x^(t)] + \gamma^2 A^2 \sum \tag{0.11}$$

The dynamics of the slow weights of Lookahead.

Lemma 1 The Lookahead slow weights have the following trajectories:

$$E[\phi_{t+1}] = [1 - \alpha + \alpha(I - \gamma A)^k]E[\phi_t] \tag{0.12}$$

$$v[\phi_{t+1}] = [1 - \alpha + \alpha(I - \gamma A)^k]^2 V[\phi_t] + \alpha^2 \sum_{i=0}^{k-1} (I - \gamma A)^{2i} \gamma^2 A^2 \sum \tag{0.13}$$

Proof

$$
\begin{aligned}
E[\phi_{t+1}] &= (1-\alpha)E[\phi_t] + \alpha E[\theta_{t,k}] \\
&= (1-\alpha)E[\phi_t] + \alpha(I-\gamma A)^k E[\phi_t] \\
&= [1-\alpha+\alpha(I-\gamma A)^k]E[\phi_t] \qquad (0.14)
\end{aligned}
$$

For the variance:

$$
V[\phi_{t+1}] = (1-\alpha)^2 V[\phi_t] + \alpha^2 V[\theta_{t,k}] + 2\alpha(1-\alpha)cov(\phi_t, \theta_{t,k}) \qquad (0.15)
$$

For simplicity, we work with a single element, θ, of the vector θ (as A is diagonal, each element evolves independently).

$$\begin{aligned} cov(\theta_{t,k-1}, \theta_{t,k}) &= E[(\theta_{t,k-1} - E[\theta_{t,k-1}])(\theta_{t,k} - E[\theta_{t,k}])] \\ &= E[((\theta_{t,k-1} - E[\theta_{t,k-1}])(\theta_{t,k} - (1-\gamma a)E[\theta_{t,k-1}]))] \\ &= E[\theta_{t,k-1}\theta_{t,k}] - (1-\gamma a)E[\theta_{t,k-1}]^2 \\ &= E[(1-\gamma a)\theta_{t,k-1}^2] - (1-\gamma a)E[\theta_{t,k-1}]^2 \\ &= (1-\gamma a)V[\theta_{t,k-1}] \end{aligned} \tag{0.16}$$

$$cov(\phi_t, \theta_{t,k}) = (I - \gamma A)^k V[\phi_t] \tag{0.17}$$

$$V[\phi_{t+1}] = (1-\alpha)^2 V[\phi_t] + \alpha^2 V[\theta_{t,k}] + 2\alpha(1-\alpha)cov(\phi_t, \theta_{t,k}) \tag{0.18}$$

We now proceed with the proof of Proposition 2.

Proof. First note that if the learning rate is chosen as specified, then each of the trajectories is a contraction map. By Banach's fixed point theorem,they each have a unique fixed point. Clearly the expectation trajectories contract to zero in each case. For the variance we can solve for the fixed points directly. For SGD,

$$V^*_{SGD} = (1-\gamma A)^2 V^*_{SGD} + \gamma A^2 \sum \Rightarrow V^*_{SGD} = \frac{\gamma^2 A^2 \sum}{I-(I-\gamma A)^2} \tag{0.19}$$

For Lookahead,we have,

$$V_{LA}^{*} = [1-\alpha+\alpha(I-\gamma A)^{k}]^{2} V_{LA}^{*} + \alpha^{2} \sum_{i=0}^{k-1} (I-\gamma A)^{2i} \gamma^{2} A^{2} \sum$$

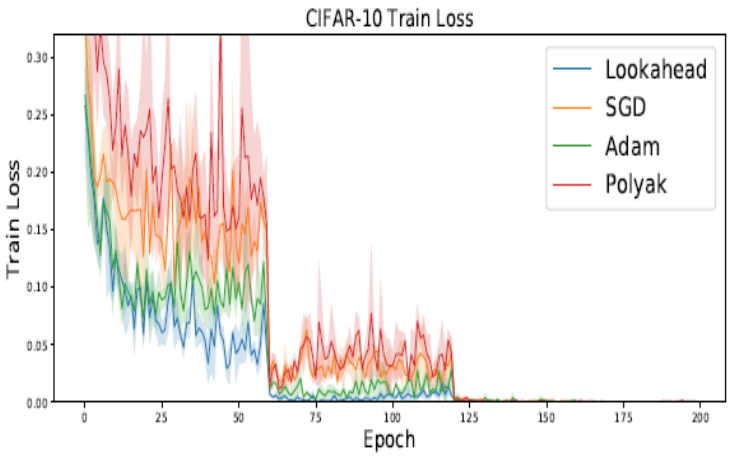
$$\Rightarrow V_{LA}^{*} = \frac{\alpha^{2} \sum_{i=0}^{k-1} (I-\gamma A)^{2i}}{I-[(1-\alpha)I+\alpha(I-\gamma A)^{k}]^{2}} \gamma^{2} A^{2} \sum$$

$$V_{LA}^{*} = \frac{\alpha^{2}(I-(I-\gamma A)^{2}k)}{I-[(1-\alpha)I+\alpha(I-\gamma A)^{k}]^{2}} \frac{\gamma^{2} A^{2} \sum}{I-(I-\gamma A)^{2}}$$

$$V_{LA}^{*} = \frac{\alpha^{2}(I-(I-\gamma A)^{2}k)}{\alpha^{2}(I-(I-\gamma A)^{2k}) + 2\alpha(1-\alpha)(I-(I-\gamma A)^{k})} \frac{\gamma^{2} A^{2} \sum}{I-(I-\gamma A)^{2}} \tag{0.20}$$

Experiments

CIFAR-10 and CIFAR-100



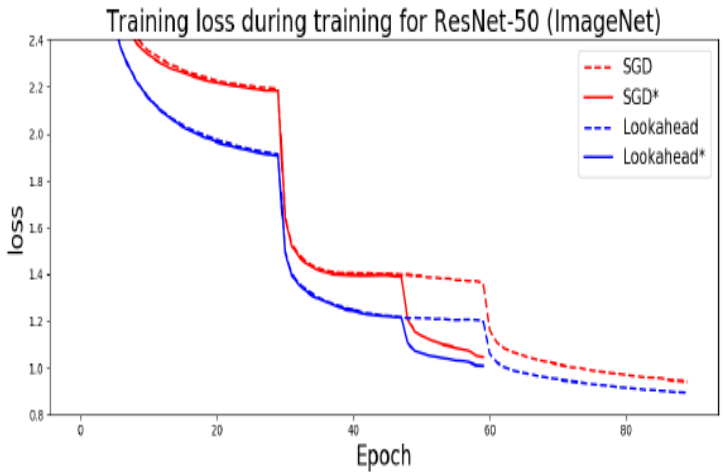
Optimizer	CIFAR-10	CIFAR-100
SGD	$95.23 \pm .19$	$78.24 \pm .18$
Polyak	$95.26 \pm .04$	$77.99 \pm .42$
Adam	$94.84 \pm .16$	$76.88 \pm .39$
Lookahead	$95.27 \pm .06$	$78.34 \pm .05$

Table 1: CIFAR Final Validation Accuracy.

Lookahead achieves significantly faster convergence throughout training even though the learning rate schedule is optimized for the inner optimizer.

Experiments

ImageNet

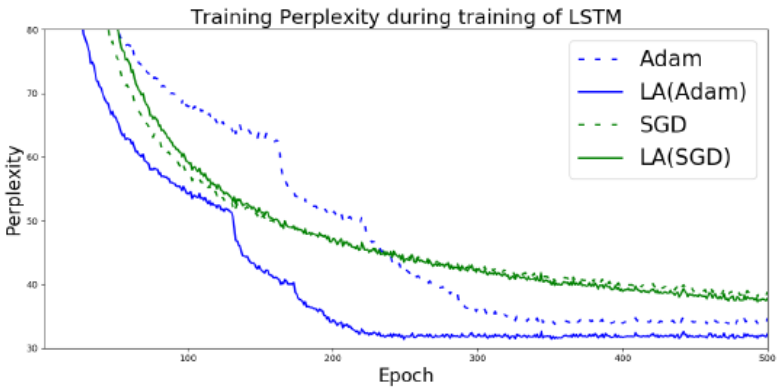


Optimizer	LA	SGD
Epoch 50 - Top 1	75.13	74.43
Epoch 50 - Top 5	92.22	92.15
Epoch 60 - Top 1	75.49	75.15
Epoch 60 - Top 5	92.53	92.56

Table 2: Top-1 and Top-5 single crop validation accuracies on ImageNet.

Experiments

LSTM



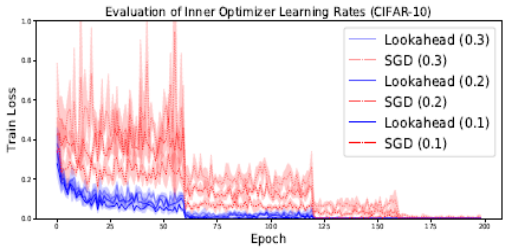
(a) Training perplexity of LSTM models trained on the Penn Treebank dataset

Table 3: LSTM training, validation, and test perplexity on the Penn Treebank dataset.

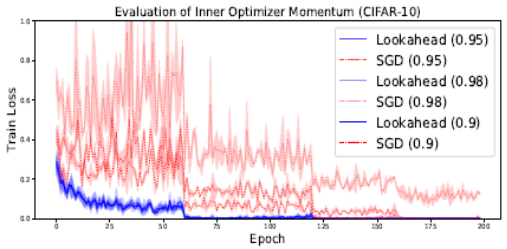
OPTIMIZER	TRAIN	VAL.	TEST
SGD	43.62	66.0	63.90
LA(SGD)	35.02	65.10	63.04
ADAM	33.54	61.64	59.33
LA(ADAM)	**31.92**	**60.28**	**57.72**
POLYAK	-	61.18	58.79

Empirical analysis

Robustness to inner optimization algorithm,*k* and α

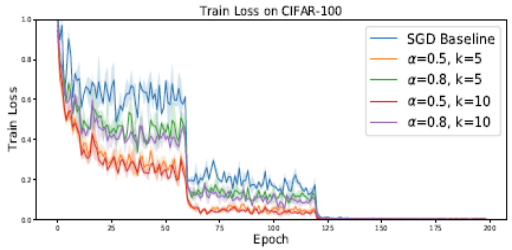


(a) CIFAR-10 Train Loss: Different LR



(b) CIFAR-10 Train Loss: Different momentum

Figure 8: We fix Lookahead parameters and evaluate on different inner optimizers.



K \ α	0.5	0.8
5	$78.24 \pm .02$	$78.27 \pm .04$
10	$78.19 \pm .22$	$77.94 \pm .22$

Table 5: All settings have higher validation accuracy than SGD (77.72%)

Figure 9: CIFAR-100 train loss and final test accuracy with various k and α.

Lookahead can train with higher learning rates on the base optimizer with little tuning on *k* and α.

Empirical analysis

Inner loop and outer loop evaluation

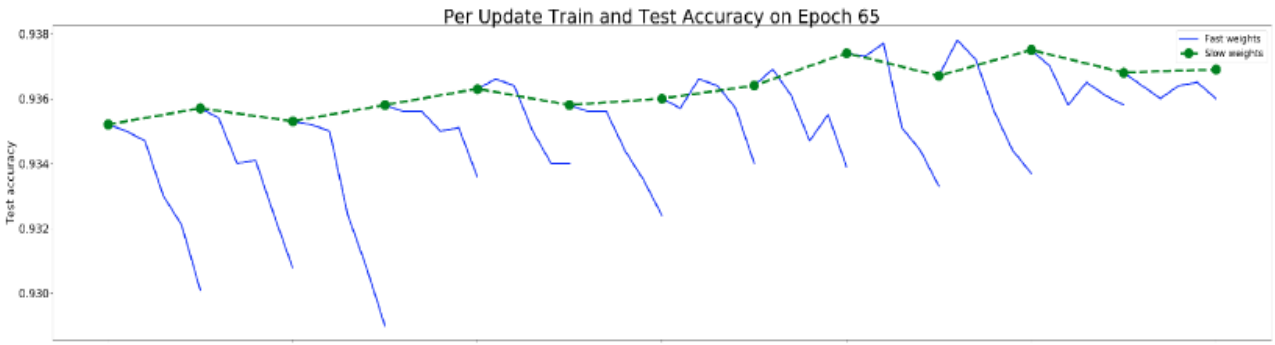


Figure 10: Visualizing Lookahead accuracy for 60 fast weight updates. We plot the test accuracy after every update (the training accuracy and loss behave similarly). The inner loop update tends to degrade both the training and test accuracy, while the interpolation recovers the original performance.

Thank you!