Nested Lookahead: A Lookahead Optimizer that Uses Lookahead

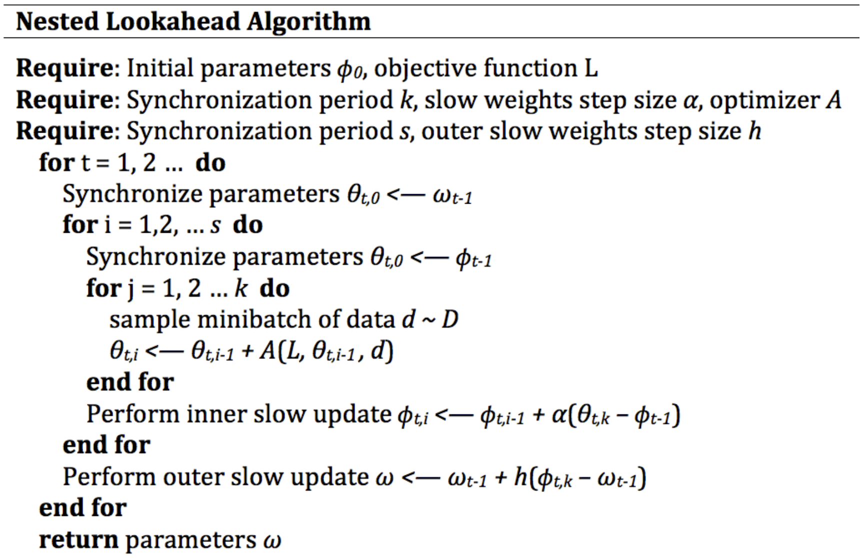
John Gilbert

Abstract: The Lookahead optimizer by Zhang et al. [1] has been shown to improve performance when used in conjunction with other deep learning optimizers such as Stochastic Gradient Descent (SGD) and Adam. Lookahead involves the use of an inner loop with fast weights to guide an outer loop of slower but more stable weights. This paper examines whether Lookahead can be improved by using Lookahead inside its inner loop. Overall, results indicate that Lookahead cannot easily be directly improved, with various hyperparameter combinations of Lookahead and Nested Lookahead yielding similar results on the CIFAR-10 dataset with ResNet-18.

Code: https://github.com/johngilbert2000/nested_lookahead

Method

The algorithm of Nested Lookahead involves an inner loop within an inner loop, illustrated below. The innermost loop involves fast weights θ updating for k steps with any given inner optimizer A. After k steps, the current location of the fast weights is used to update slower weights, ϕ, with stepsize α. The fast weights θ are pulled to the slow inner weights ϕ, which are then cached. After updating the slow inner weights ϕ for s steps, they are pulled to a final set of outer weights, ω, using step size h, which are also cached, and iterate until training is complete. Ultimately, the weights θ guide ϕ, which in turn guide ω.



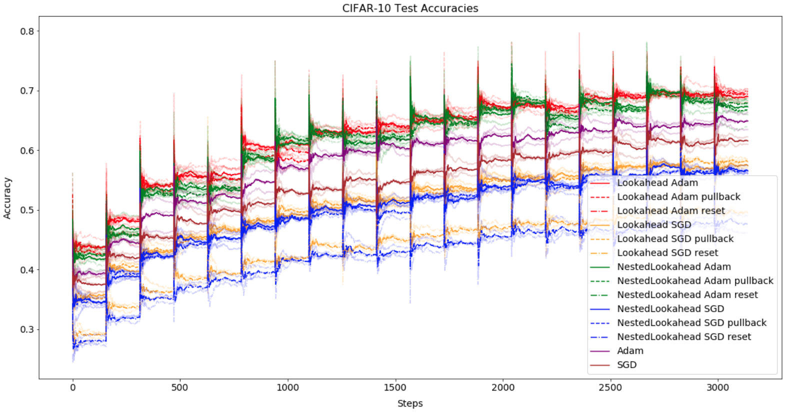
Nested Lookahead Algorithm

```
Require: Initial parameters $\phi_0$, objective function L
Require: Synchronization period $k$, slow weights step size $\alpha$, optimizer $A$
Require: Synchronization period $s$, outer slow weights step size $h$
  for t = 1, 2 ... do
    Synchronize parameters $\theta_{t,0}$ <— $\omega_{t-1}$
    for i = 1,2, ... $s$ do
      Synchronize parameters $\theta_{t,0}$ <— $\phi_{t-1}$
      for j = 1, 2 ... $k$ do
        sample minibatch of data $d \sim D$
        $\theta_{t,i}$ <— $\theta_{t,i-1} + A(L, \theta_{t,i-1}, d)$
      end for
      Perform inner slow update $\phi_{t,i}$ <— $\phi_{t,i-1} + \alpha(\theta_{t,k} - \phi_{t-1})$
    end for
    Perform outer slow update $\omega$ <— $\omega_{t-1} + h(\phi_{t,k} - \omega_{t-1})$
  end for
  return parameters $\omega$
```

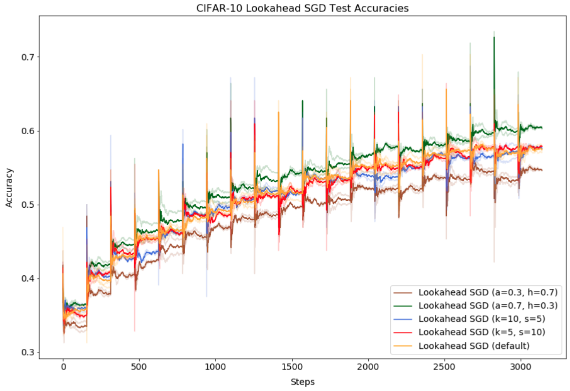
Two different image classification experiments were run using Nested Lookahead with ResNet-18 on CIFAR-10, which contains 10 class labels, a training set of 50,000 images, and a test set of 10,000 images. The first experiment ran SGD with 0.9 Polyak momentum, Adam, Lookahead, and Nested Lookahead for 20 epochs, 3 times each. Pullback momentum and reset momentum were also assessed for both Lookahead and Nested Lookahead, where momentum was pulled back to a cached state or reset to 0, respectively, upon finishing an inner loop. The second experiment involved the use of hyperparameter tuning to assess the effect of k, s, α, and h on Lookahead and Nested Lookahead. Additionally, learning rate, momentum, and weight decay were also modified for SGD and Adam.

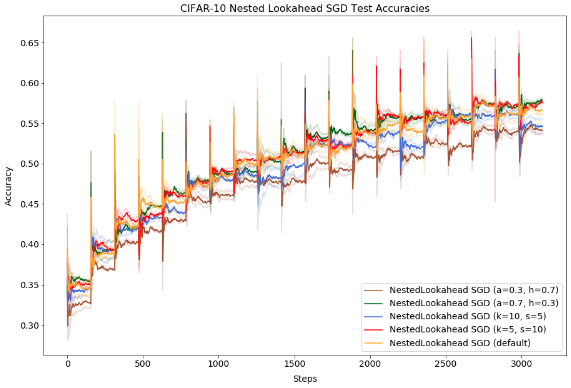
Results

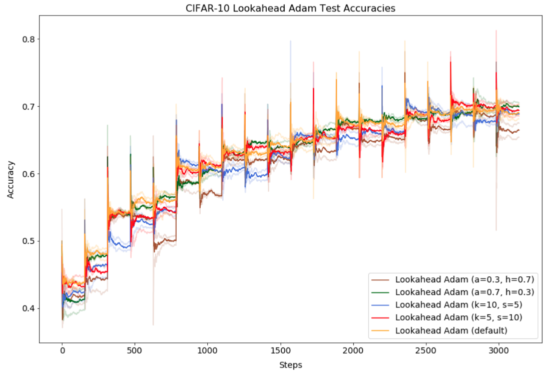
The test accuracy results of the first experiment are shown below. Accuracy is displayed as a value between 0 and 1, with 1 indicating a perfect score or 100% accurate. Steps indicate the number of steps taken by the innermost optimizer A over the course of 20 epochs.

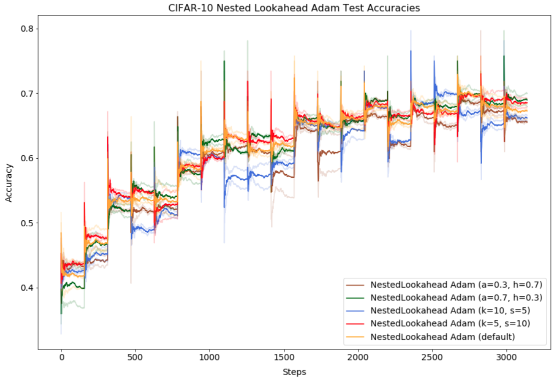


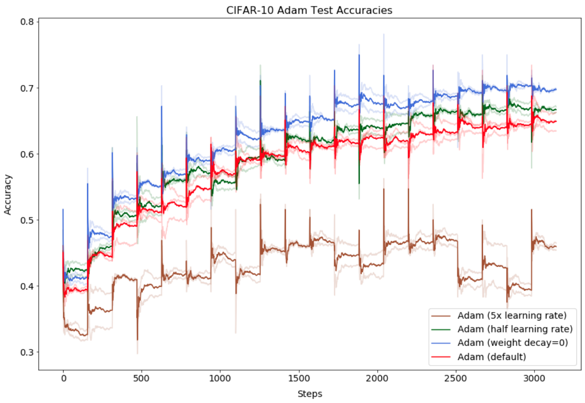
Results of the second experiment are shown below:

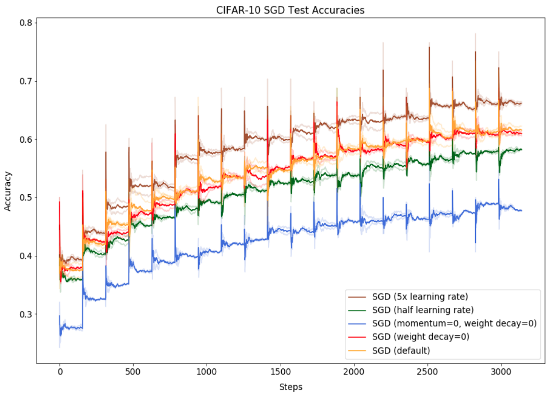












Discussion

The results of the first experiment indicate that Nested Lookahead and Lookahead have similar performance, and that combined with Adam as the inner optimizer, they outperform Adam, SGD with momentum, and Lookahead and SGD. Pullback momentum did not offer any benefit to Lookahead or Nested Lookahead, and resetting the momentum appeared to hurt performance. Curiously, SGD outperformed Lookahead + SGD and Nested Lookahead + SGD. While the cause of this is unclear, it is possible it is due to SGD with momentum already being relatively slow and stable, hence making it not particularly useful as the fast-weight inner optimizer A, such that pulling weights towards a cached state hurt performance.

The results of the second experiment indicate that both Lookahead and Nested Lookahead have similar performances both for different steps k and s, and for different step sizes α and h. This coincides with the findings of Zhang et al., and when considered with the results of the first experiment, it implies that there may be little that can be done to directly improve the Lookahead algorithm. Additionally, the use of Polyak momentum was beneficial for SGD, weight decay had little effect, and a learning rate of 0.001 was suitable for at least the first 20 epochs.

Conclusion

In conclusion, Nested Lookahead provided the same performance benefits as regular Lookahead. The results imply that Lookahead can not easily be improved directly, and they confirm that Lookahead paired with the right inner optimizer can provide a boost in performance. Even though Nested Lookahead provided no breakthroughs, it provided supporting evidence that Lookahead is an already robust and useful optimization technique.

References

[1] Michael Zhang, James Lucas, Geoffrey Hinton, Jimmy Ba. Lookahead Optimizer: k steps forward, 1 step back. 2019. arXiv:1907.08610