Neon: Nuclear Norm to Beat Muon

Alexey Kravatskiy, Ivan Kozyrev, Nikolay Kozlov, Alexander Vinogradov, Daniil Merkulov, Ivan Oseledets

Optimization Class Project, MIPT

Лето с AIRI

Introduction

Recent advances in optimization techniques have highlighted the benefits of leveraging the matrix structure of neural network weights during training. Optimizers like Muon (Jordan et al.) and Shampoo (Gupta, Koren and Singer) have shown promise, but at a significantly higher computational cost than traditional methods like Adam. To bridge this gap, we propose Neon, a new optimizer that builds upon the framework of Bernstein and Newhouse. By using alternative norms, such as nuclear norm (Nuclear-Neon) or custom $F*$ norm ($F*$-Neon), we obtain low-rank update matrices that enable more efficient computation. We evaluate the performance of Neon, Muon, and Adam on training MLP and CNN on CIFAR-10 and on finetuning NanoGPT.

Neon's update rule

Bernstein and Newhouse suggest obtaining the update step for a weight matrix W as a solution to the optimization problem:

$$\langle G, \delta W\rangle + \frac{\lambda}{2}\|\delta W\|^2 \to \min_{\delta W}, \tag{1}$$

where G is a gradient-like matrix obtained via backpropagation. Setting norm to RMS-to-RMS norm (scaled version of Spectral norm) produces Muon. We consider two different choices instead:

1. Choosing nuclear norm, $\|\cdot\|_*$, produces rank-1 update, defined by

$$\delta W = -\frac{1}{2\lambda} u_1 \sigma_1 v_1^T, \tag{2}$$

   where σ_1 is largest singular value of G, and u_1, v_1 are corresponding singular values.

2. Choosing $F*$ norm, defined by $\|\cdot\|_{F*} = (\|\cdot\|_* + \|\cdot\|_F)/2$, produces a relatively small-rank update, defined by

$$\delta W = -\frac{1}{\lambda} U D V^T \tag{3}$$

   with $D = \mathbf{diag}(d_i)$, where $d_i = [\sigma_i - \tau]_+$ and τ is given by

$$\sum_{i=1}^{n} [\sigma_i - \tau]_+ = \tau\,.$$

Efficient update computation

We use cupy's svds routine to obtain gradients' matrices' SVD approximation formed by largest singular values and corresponding vectors. By applying the Lanczos process to either A^TA or AA^T, it generates a sequence of orthogonal vectors (Lanczos vectors) that capture the dominant spectral properties of the original matrix. The singular values and vectors are then extracted from the tridiagonal matrix using standard eigenvalue techniques like QR iteration.

Jeremy Bernstein and Laker Newhouse. Old optimizer, new norm: An anthology (2024)

Jeremy Bernstein. Deriving Muon (2025)

Keller Jordan et al. Muon: An optimizer for hidden layers in neural networks (2024)

Algorithms

Now we can write down pseudocode for both versions of Neon.

Algorithm 1 Nuclear-Neon update step for linear layer

Input: λ, gradient-like matrix G.
Output: Weight matrix update δW.

1. $U, \Sigma, V := \mathrm{Lanczos}(G, 1)$.

Return $-\frac{1}{2\lambda} U \Sigma V^T$.

For $F*$-Neon it is a bit trickier (3), we need to compute τ and know number of singular values larger then τ, which we denote by r. Assuming that singular values spectrum of G changes little between iterations (which was true in our experiments) we propose the following algorithm.

Algorithm 2 $F*$-Neon update step for linear layer

Input: λ, r, gradient-like matrix G.
Output: Weight matrix update δW.

1. $U, \Sigma, V := \mathrm{Lanczos}(G, r+1)$.
2. $s := \sum_{i=1}^{r} \sigma_r$.
3. **If** $(r+1)\sigma_{r+1} > s$:
   4. $r := r+1$. 5. $\tau := \frac{s+\sigma_r}{r+1}$.
6. **Else if** $(r+1)\sigma_{r-1} < s$:
   7. $r := r-1$. 8. $\tau := \frac{s-\sigma_r}{r+1}$.
9. **Else:**
   10. $\tau := \frac{s}{r+1}$
11. $D = [\Sigma - \tau I]_+$.

Save r for the next iteration.
Return $-\frac{1}{\lambda} U D V^T$.

MLP tests

We test Muon, SGD, and Neon on a simple MLP (2 linear layers) that solves CIFAR-10 classification problem. Muon converges faster than Neon and reaches higher accuracy.

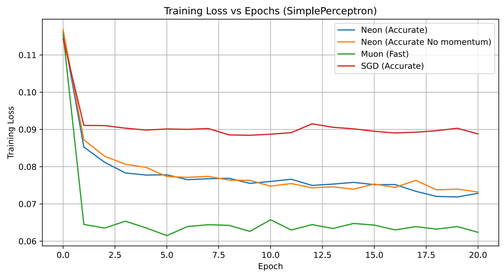

Рис. 1: MLP validation loss

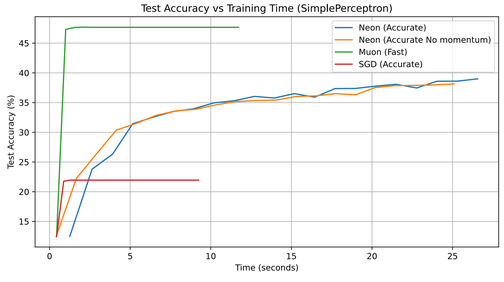

Рис. 2: MLP Accuracy vs wallclock time

CIFAR-10 tests

Now we compare Muon and Neon on ResNet from Keller's cifar10-aribench at GitHub. On RTX-4050, Muon is approximately twice as fast as Neon. That is likely caused by ineffective implementation of Neon's update, because in theory updates of Neon are more light and simple, especially for Nuclear Neon. Interestingly, Neon has a ceiling of convergence about 70% which is not observed for Muon. Moreover, Neon requires far smaller learning rates to converge. All these aspects should be considered when we improve the algorithm.

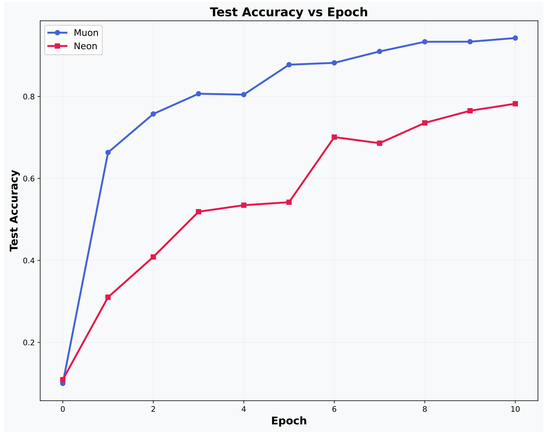

Рис. 3: Accuracy vs Epoch for Muon vs Neon

NanoGPT tests

We finetuned NanoGPT by Alex Karpathy on two NVIDIA RTX 4090 GPUs (24 GB each) on tiny stories dataset via 200 iterations. Both Neon and Muon showed better convergence then AdamW. Neon showed similiar convergence rate to Muon

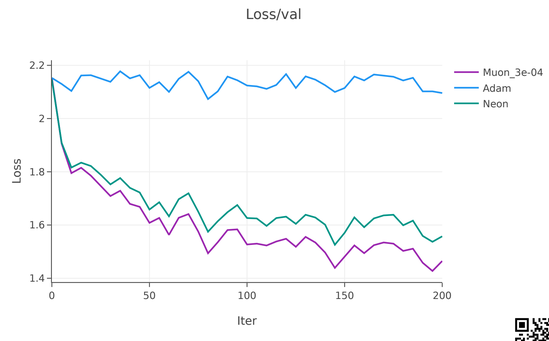

Рис. 4: NanoGPT validation loss

Conclusion

A new norm-motivated algorithm proved to be not so effective as Muon, at least from the first sight. However, its updates seem to be much faster to compute, and more so for large matrices that have not been tested yet. We believe our algorithm is not an alternative to Muon, but probably a more effective tool for solving some other optimization problems. A deep inspection of nuclear norm and its properties might give us an insight into Neon's future.