Stochastic Line Search

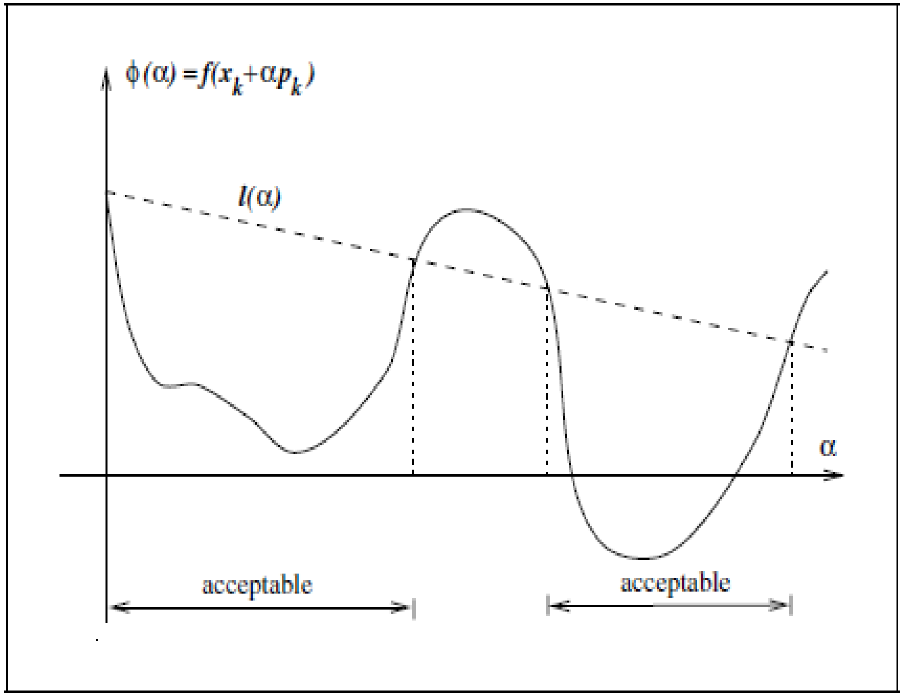
by
Philip Kenneweg, Leonardo Galli, Tristan Kenneweg
and Barbara Hammer

UNIVERSITÄT
BIELEFELD

What is Stochastic Line Search

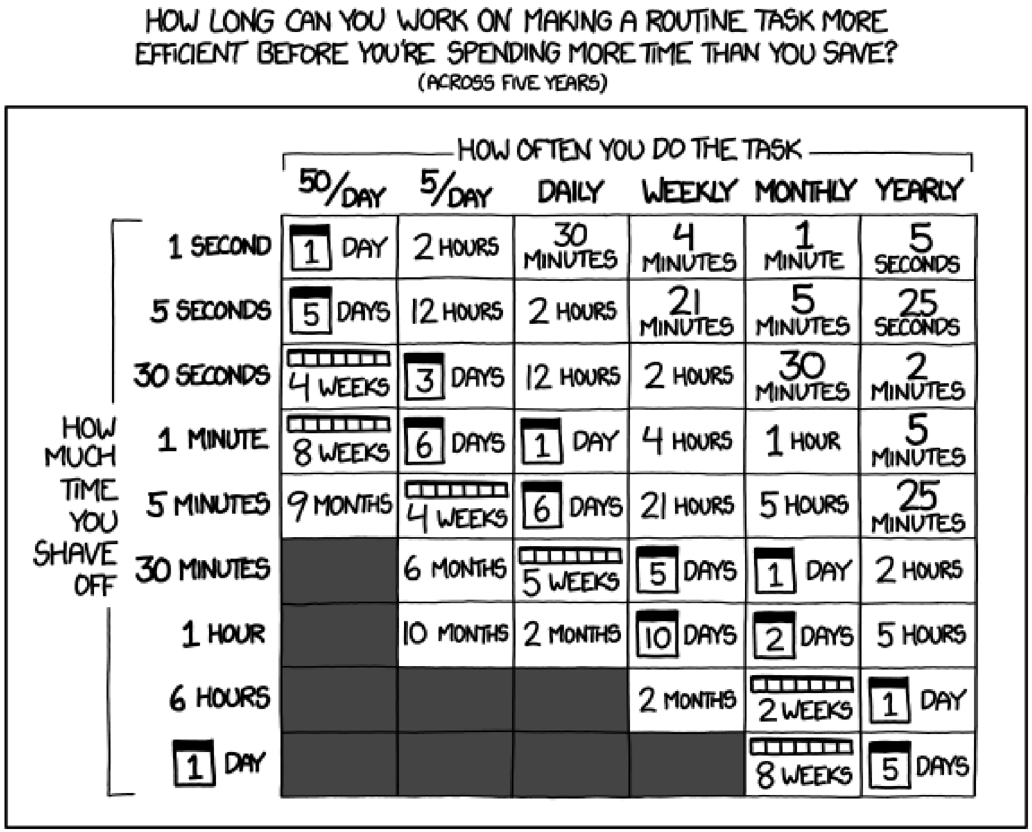
- Method to find optimal learning rate
- Key idea:

  1. Take a step along the gradient direction
  2. compute the loss
  3. if loss did not reduce (enough) try different lr



Advantages

- No learning rate tuning anymore
- Faster convergence
- Better generalization performance (surprisingly)



Disadvantages

- Uses more compute

  -a lot more for un-optimized approaches

  -about 1 extra forward pass per loss computation/step for our implementation

- Can not easily incorporate human expert knowledge

Challenges

- Where to start checking for reasonable learning rate?
- How much reduction in loss is reasonable?
- Many more implementation details

Solutions (existing)

- Lipschitz Line Search criterion:

$$f_k(w_k) - f_k(w_k + \eta_k d_k) \geq c \cdot \eta_k ||\nabla f_k(w_k)||^2,$$

Loss reduction Gradient norm

- Guaranteed to converge with SGD (under some assumptions) if step size is lowered until this criterion is fullfilled
- Can be extended to Armijo Line Search for adaptive optimizers (Adam, Adagrad etc)

Solutions (existing)

- Initial Step size for the search is the last step size.
- To facilitate growing learning rates, double lr every n(in practice 300) steps.

Our Idea

- Adapt learning rates per Layer.
- Evaluate this and the Armijo approach combined with Adam on modern architectures. (Transformers)

Math

Change:

$$f_k(w_k) - f_k(w_k + \eta_k d_k) \geq c \cdot \eta_k ||\nabla f_k(w_k)||^2,$$

To:

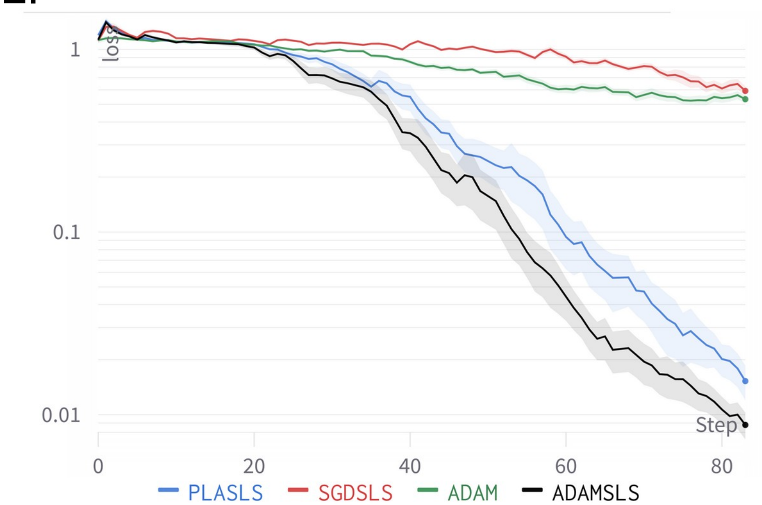
$$f_k(w_k + \eta_k^{(l)} \bar{d}_{k,l}) \leq f_k(w_k) - c \cdot \eta_k^{(l)} ||\nabla f_k(w_k)^{(l)}||^2,$$

$$\bar{d}_{k,l} := (0, \ldots, d_k^{(l)}, \ldots, 0)$$

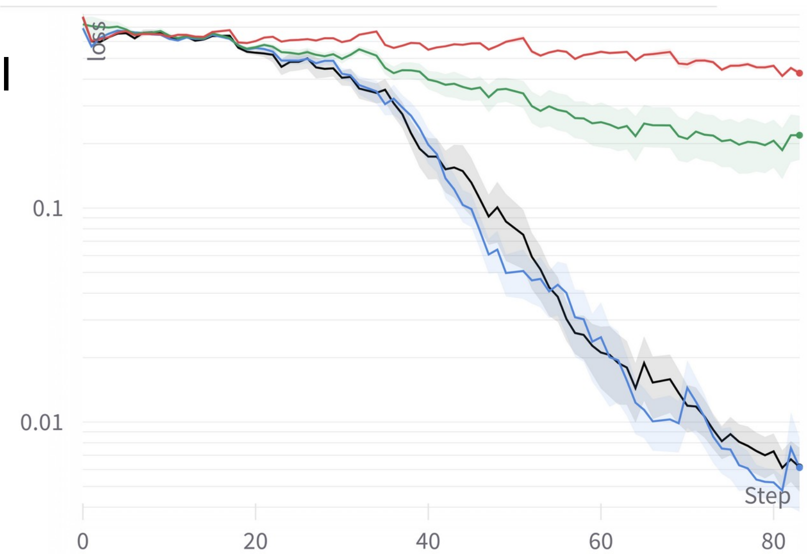
Actually not quite correct, but correct one would need 2 forward passes per check
→ too expensive

Results small Dataset

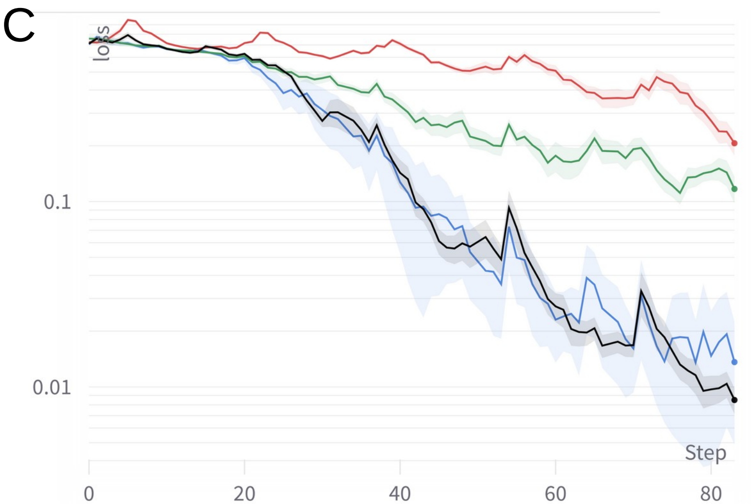
MNLI



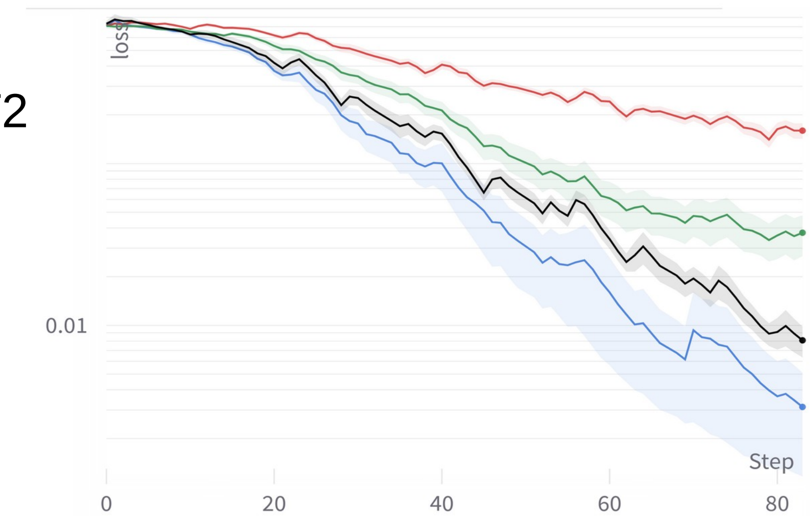
QNLI



MRPC



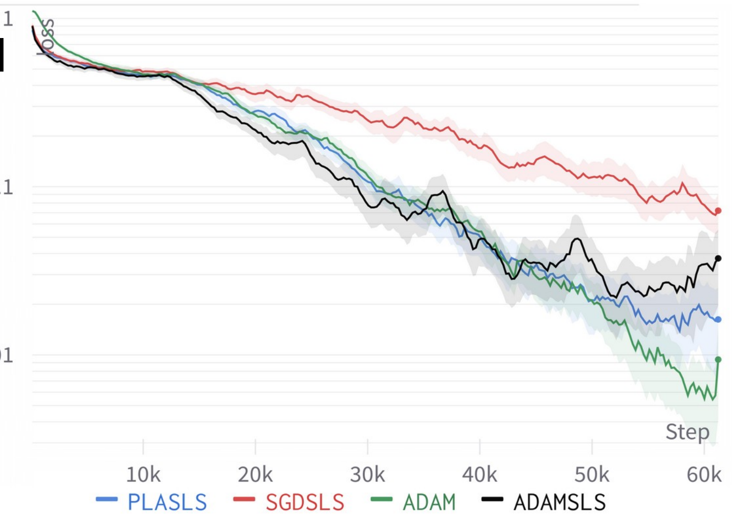
SST2



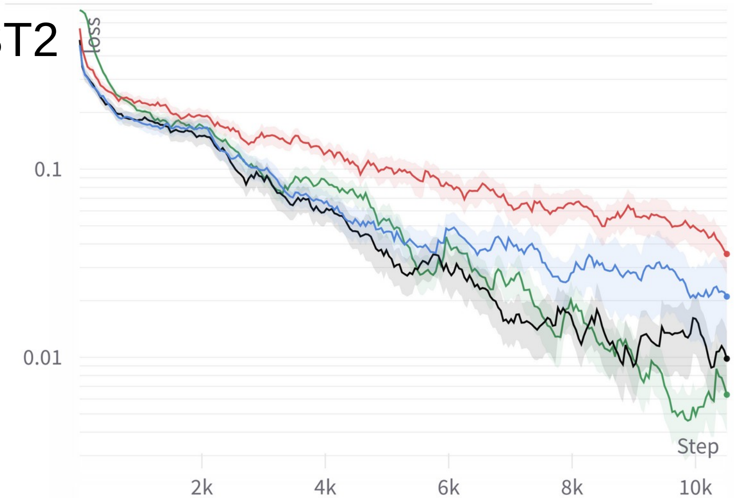
UNIVERSITÄT BIELEFELD

Results

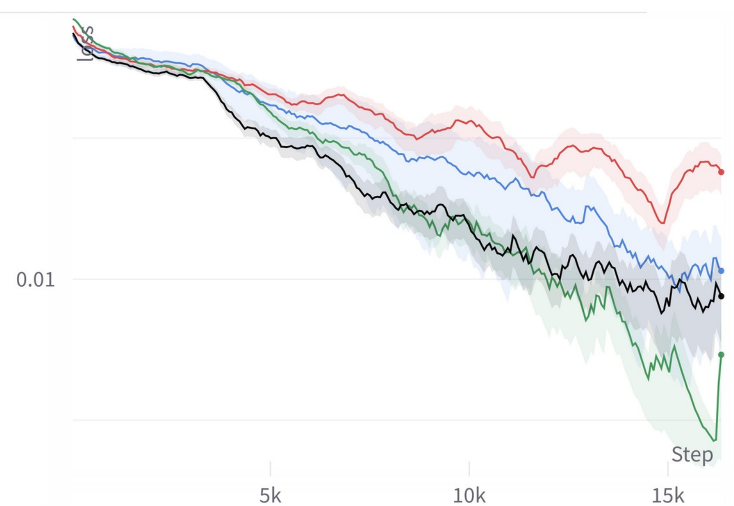
MNLI



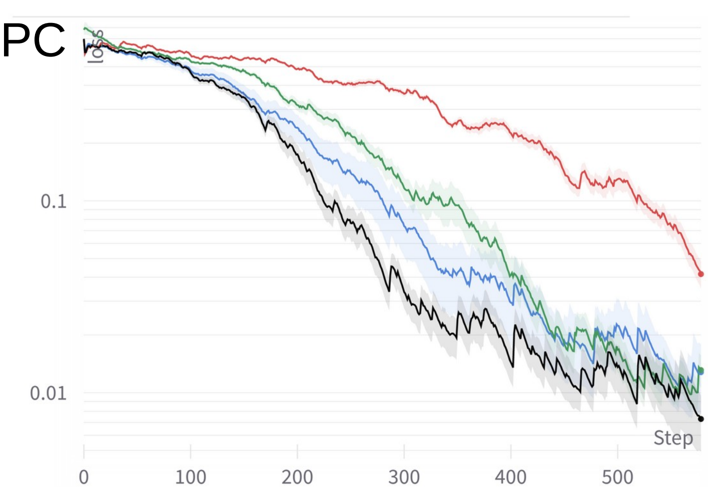
SST2



QNLI



MRPC

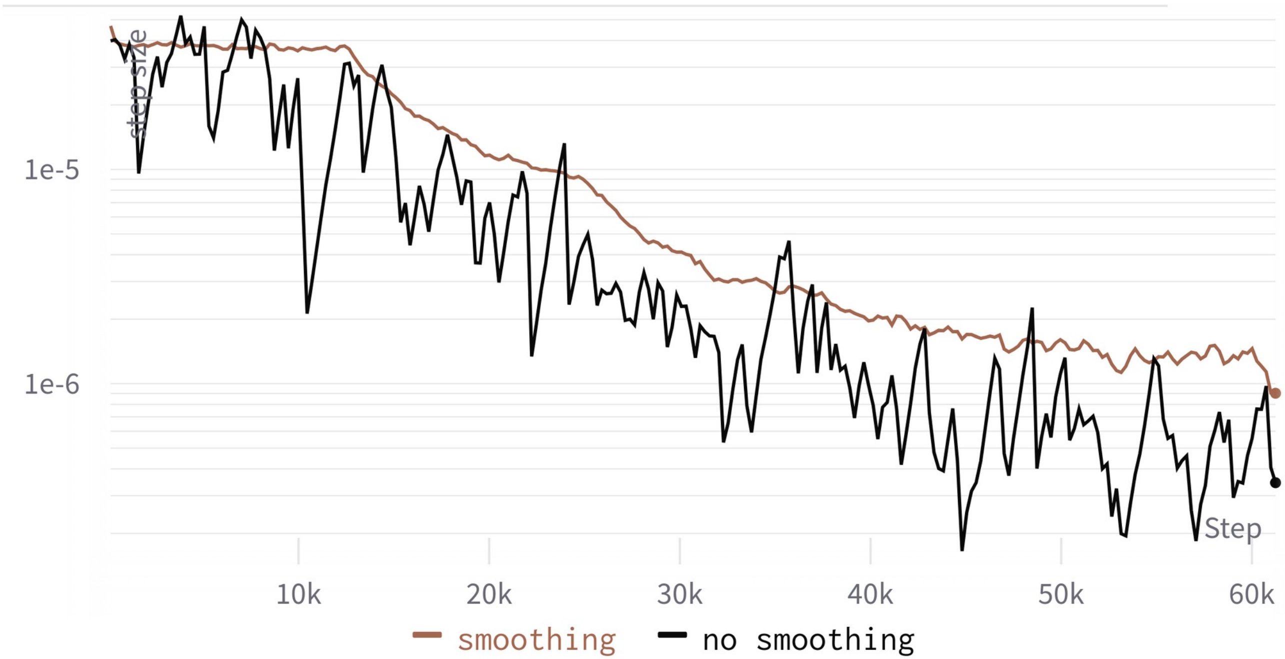


Results

- PLASLS and ADAMSLS perform a lot better than ADAM or SGD even with ADAMs tuned learning rate schedule.
- PLASLS does not significantly outperform ADAMSLS
- → this was a very short summary of the IJCNN paper

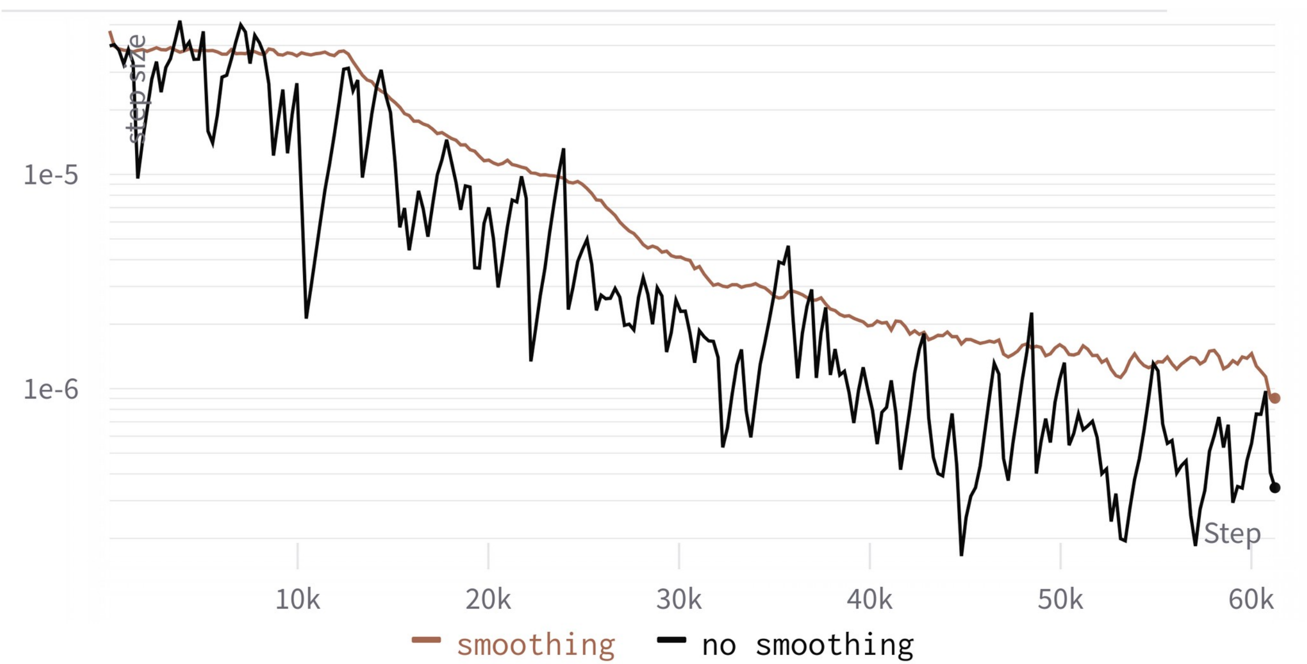
New Idea

- We saw during training that the learning rate is mostly determined by a small percentage of batches which lower it significantly.



New Idea

- So smoothing seems to make intuitive sense.



Math

$$f_k(w_k) - f_k(w_k + \eta_k d_k) \geq c \cdot \eta_k ||\nabla f_k(w_k)||^2, \qquad (3)$$

We call $f_k(w_k) - f_k(w_k + \eta_k d_k)$ the decrease h_k and $||\nabla f_k(w_k)||^2$ the sufficient decrease s_k . Now we apply exponential smoothing to both term i.e.:

$$\begin{aligned} \hat{h}_k &= \hat{h}_{k-1} \cdot \beta + h_k \cdot (1 - \beta) \\ \hat{s}_k &= \hat{s}_{k-1} \cdot \beta + s_k \cdot (1 - \beta) \end{aligned} \qquad (4)$$

resulting in smoothed values $\hat{h}_k$ and $\hat{s}_k$ representing the average decrease of the loss with our current step size and the average sufficient decrease. Resulting in Eq. 5

$$\hat{h}_k \geq c \cdot \eta_k \cdot \hat{s}_k, \qquad (5)$$

Problem

- Lowering step size has no immediate effect on h since f(w_k + n_k*d_k) is dependend on step size n_k. Impractical to recompute.

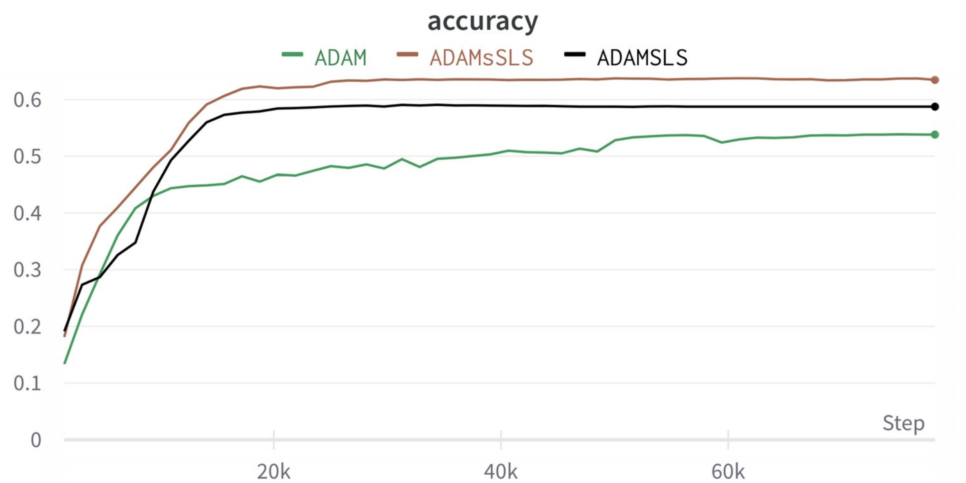
  → but this is actually not a problem, just makes changes in step size take place over a longer period in time

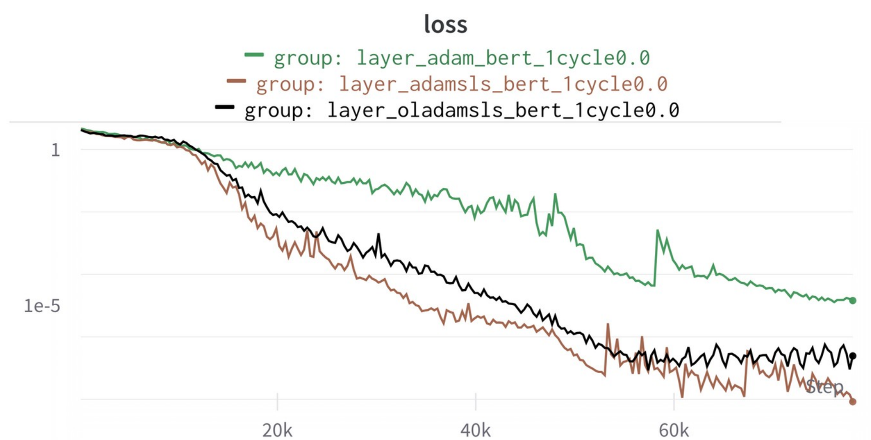
Experiments

- Experiments on a variety of architectures:
- CNN ResNet34
- Transformers BERT
- MLPs 2 hidden layers
- ? something missing ?

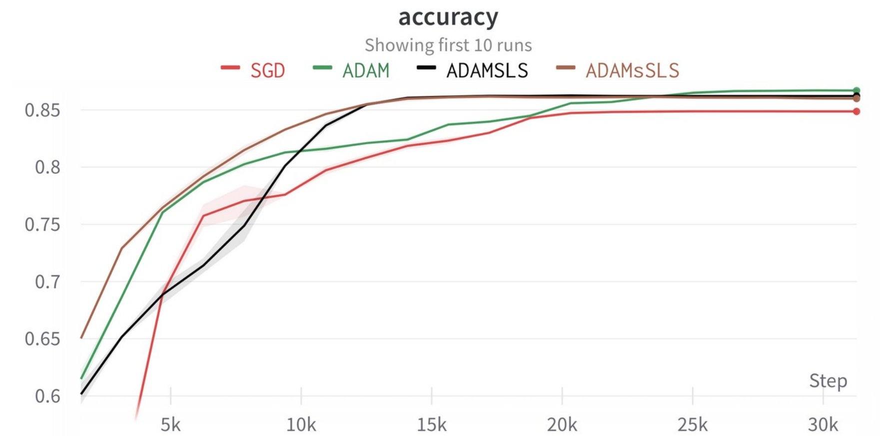
Results CNN

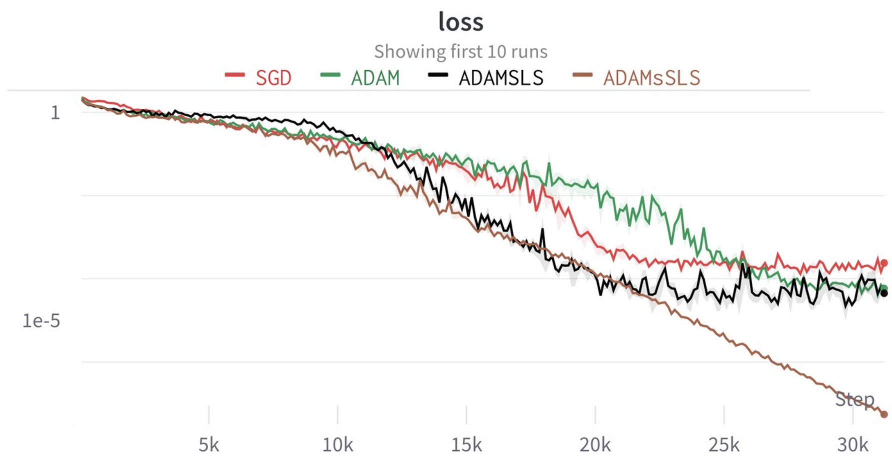
Cifar100





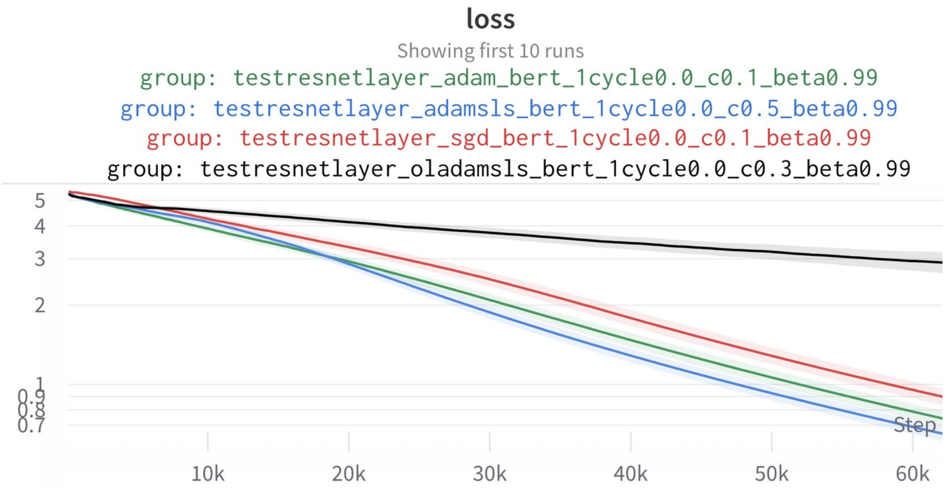
Cifar10

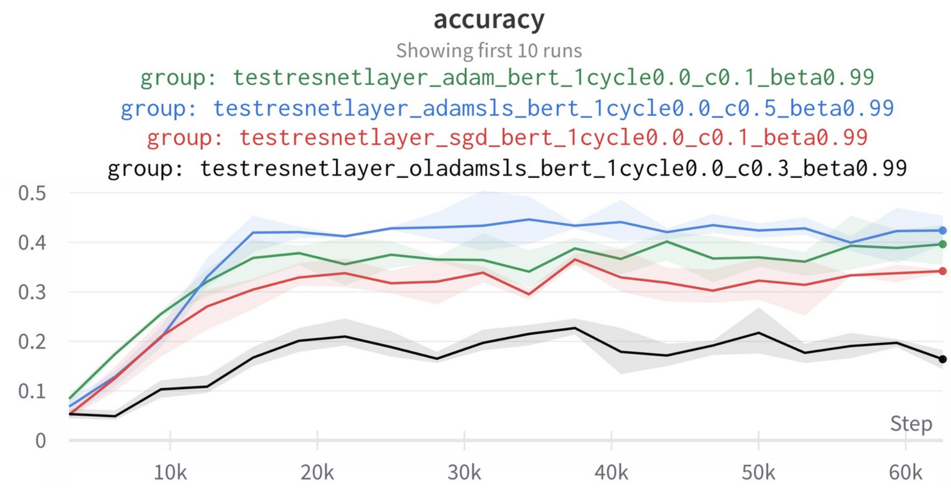


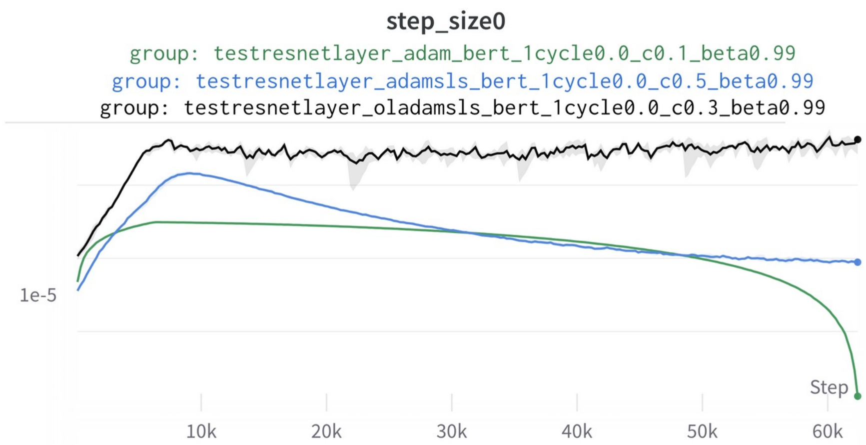


Results CNN

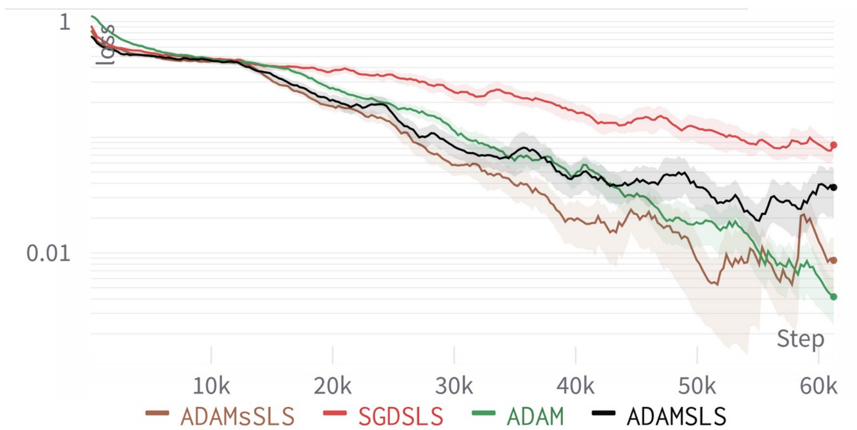
TinyImageNet

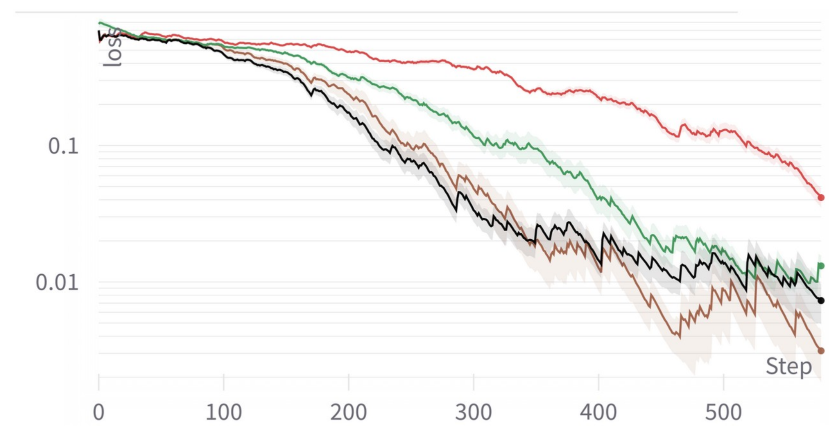


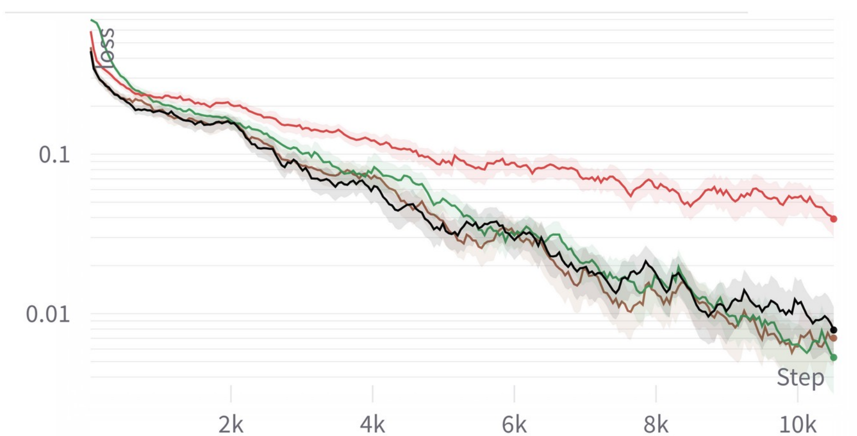


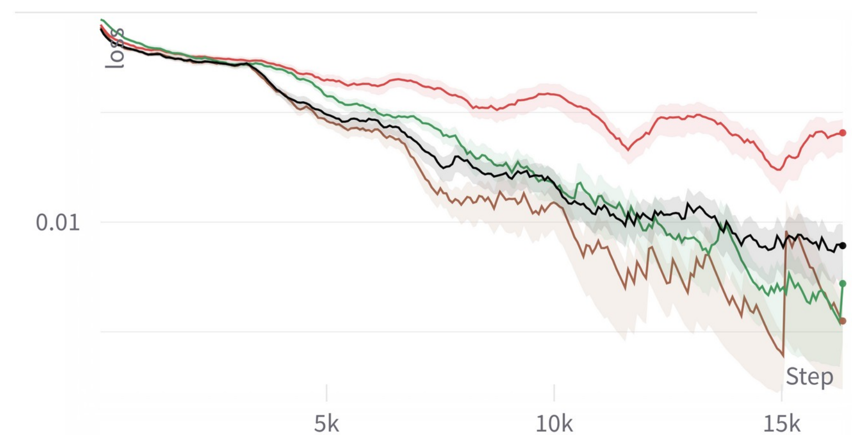


Results Transformer



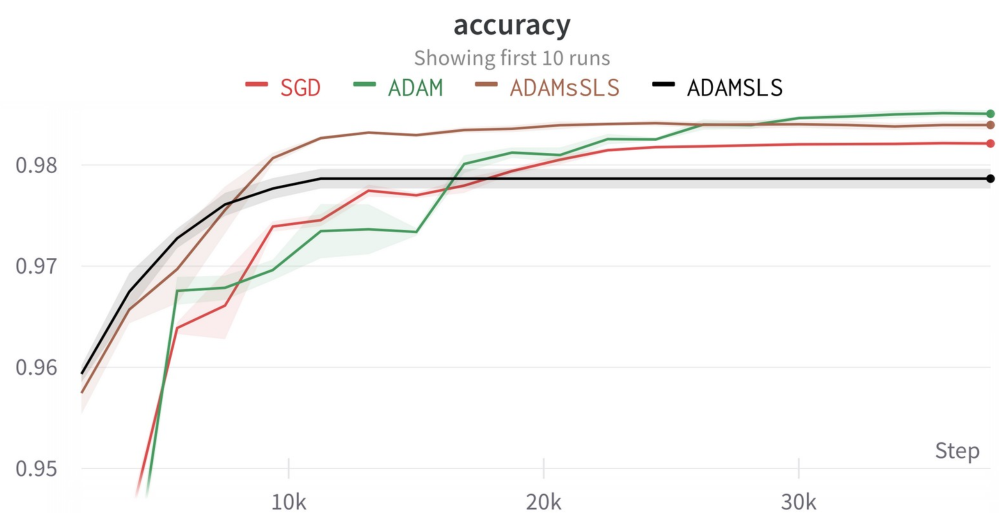


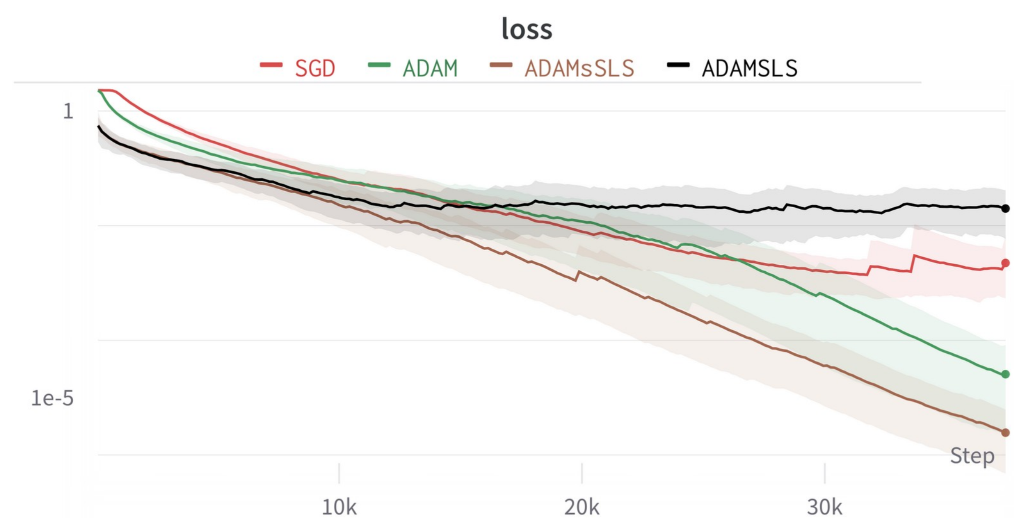




Results MLP

Mnist with 2 hidden layers:





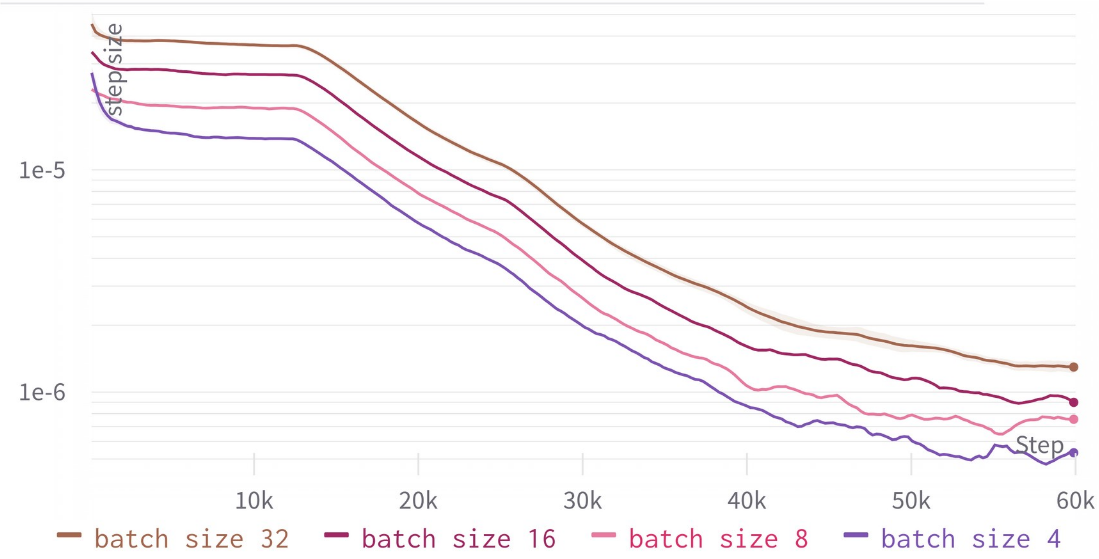
More interesting Ideas & TODOs:

More interesting Ideas & TODOs:

UNIVERSITÄT
BIELEFELD

Theory

- Some theoretical results: learning rate and batch size should be related. We see the same proportionality as predicted for our implementation: lr ~ sqrt(batch_size) for adaptive optimizers



Theory

- Network size to learning rate relation could also easily be looked at.

Graphs are fine but metrics are better?

- Some metric to evaluate optimizers?
- Intuition:
- Should be similar to area under the curve just for loss.
- Should be logarithmic (i.e. halving of loss is valued the same)

$$L = \frac{\sum_t \log(l_t)}{\hat{t}}$$

Theory

- Some theoretical proofs would be nice for example:
- more robustness towards noise induced by the batch size. Will still be limited.
- At the moment all proofs for SLS convergence assume no noise induced by batch size.

Thank you for listening

Questions?