High-Accuracy Low-Precision Training

Chris De Sa*, **Megan Leszczynski**, Jian Zhang,
Chris Aberger, Matt Feldman, Alana Marzoev*,
Kunle Olukotun, Chris Ré

Stanford University
*Cornell University

STANFORD DAWN

“Neural network **predictions** often don't require the precision of floating point calculations with 32-bit or even 16-bit numbers. With some effort, you may be able to use **8-bit integers** to calculate a neural network **prediction** and still maintain the appropriate level of accuracy."

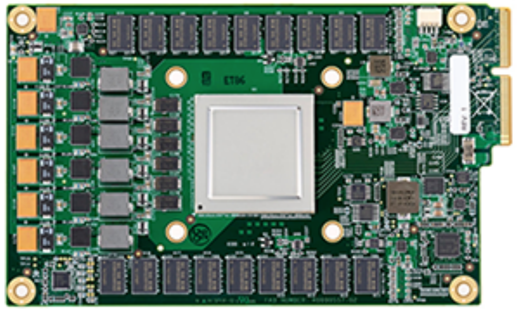
Kaz Sato, Cliff Young, and David Patterson, “An in-depth look at Google’s first Tensor Processing Unit”, Google Cloud Big Data and Machine Learning Blog

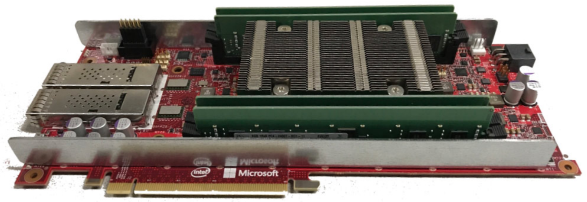
Low Precision: The best thing since sliced bread

Energy

Memory

Throughput





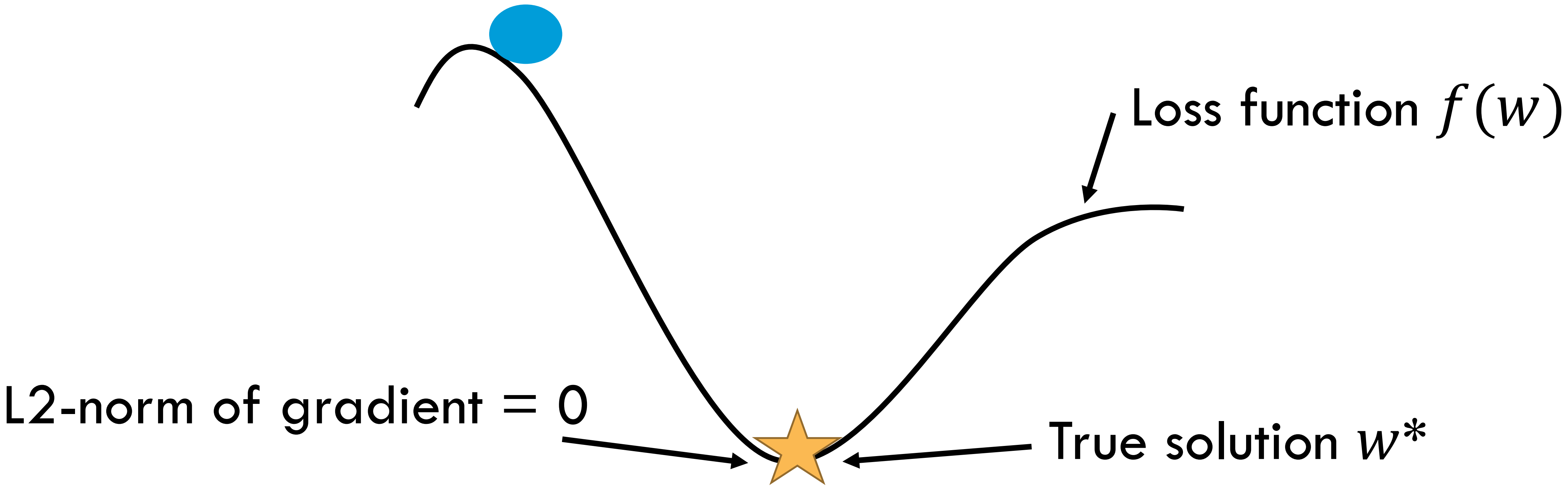
But there's no free lunch…

Accuracy

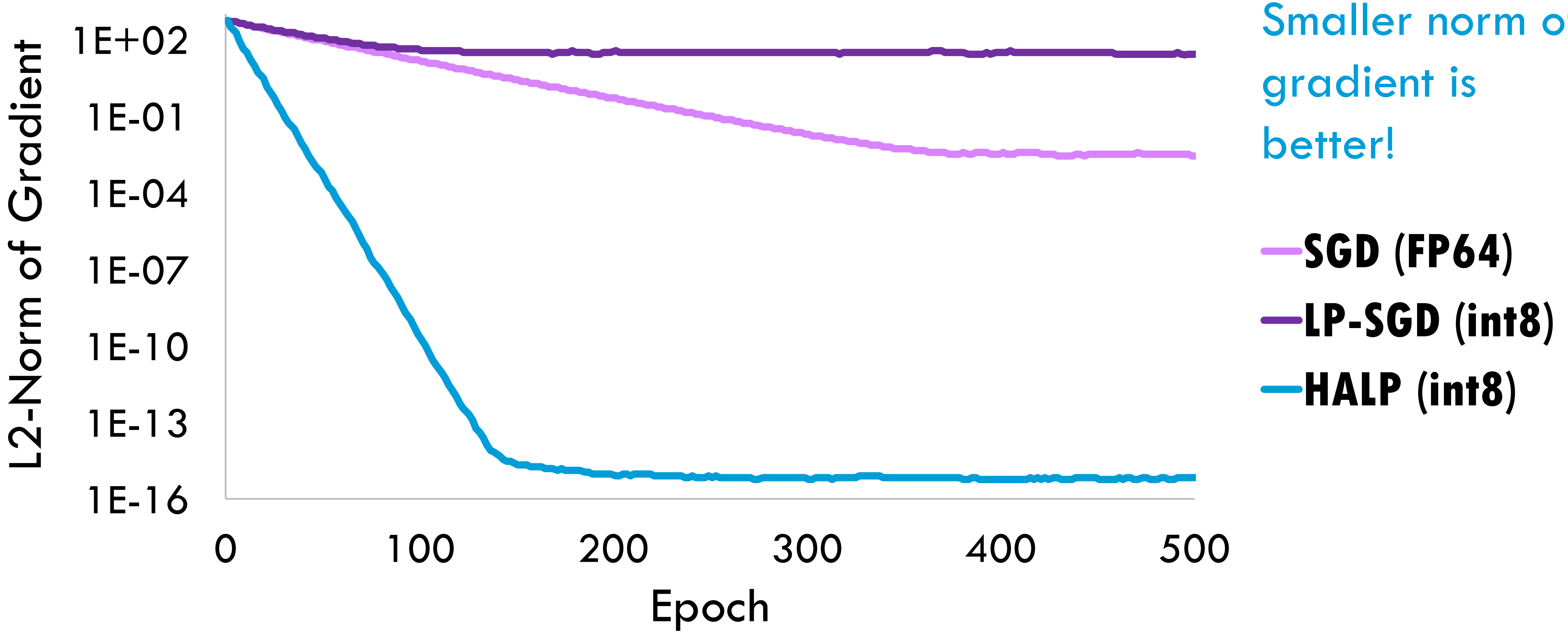
Training usually requires at least 16 bits.

Optimization

$$\text{minimize } f(w) = \frac{1}{N}\sum_{i=1}^{N} f_i(w) \text{ over } w \in \mathbb{R}^d$$



But there's no free lunch…



Smaller norm of gradient is better!

Or is there? HALP outperforms SGD and LP-SGD.

Update Steps

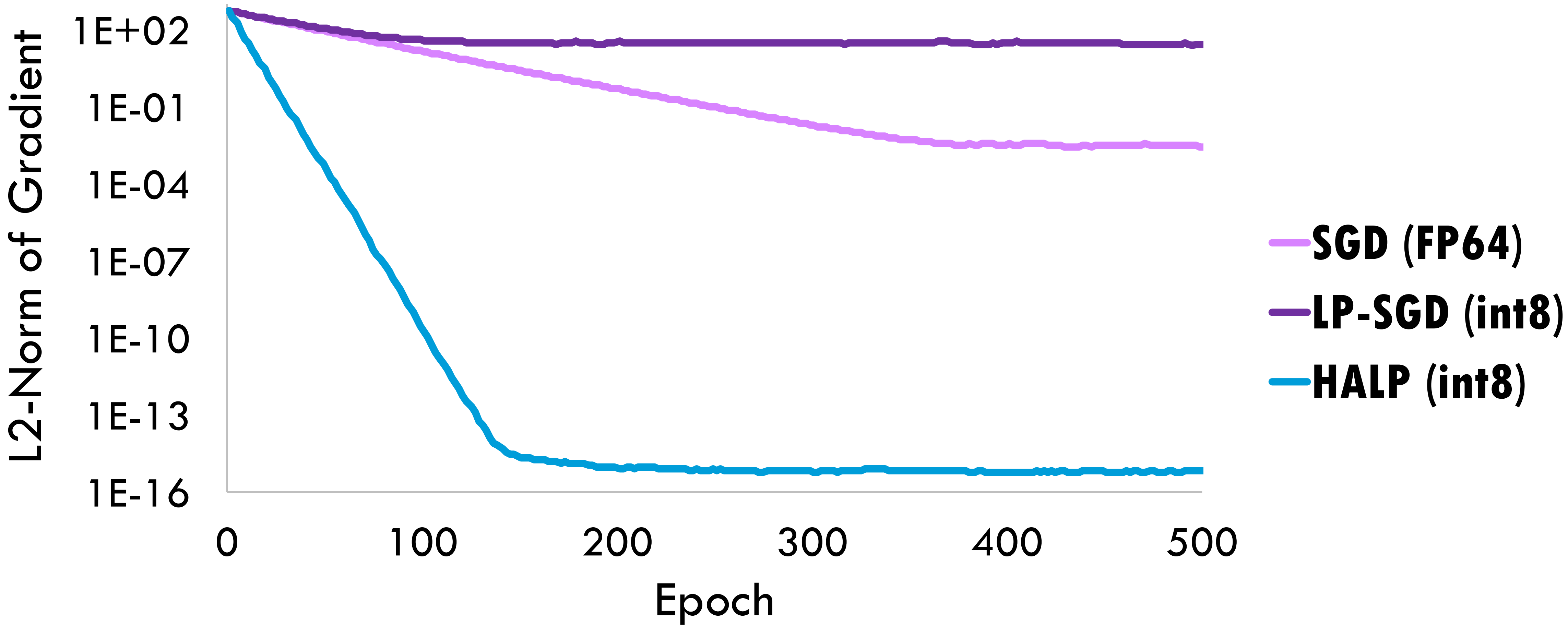
1. SGD
2. LP-SGD
3.
4.
5.

$$w_{t+1} = w_t - \alpha \nabla f_{i_t}(w_t)$$

$$w_{t+1} = w_t - \alpha \nabla f_{i_t}(w_t)$$

= Low-precision

Where do SGD algorithms fall short?



(1) SGD algorithms converge slower

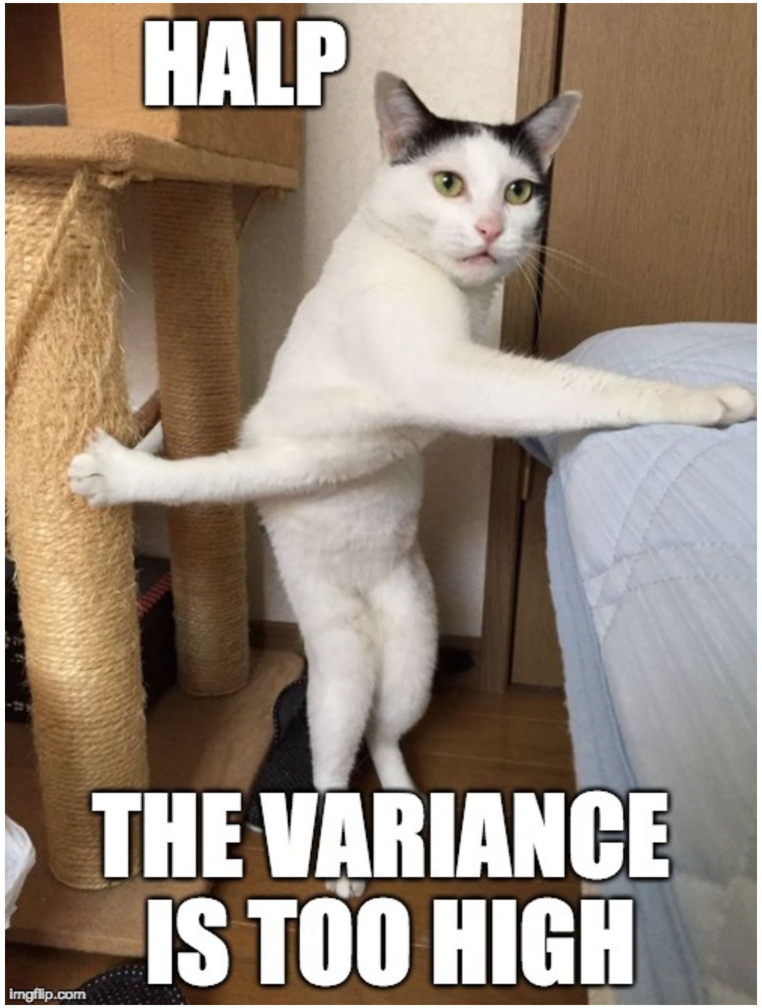
(2) SGD algorithms converge to a higher noise floor

Challenge 1: High Gradient Variance

SGD's small batch size:

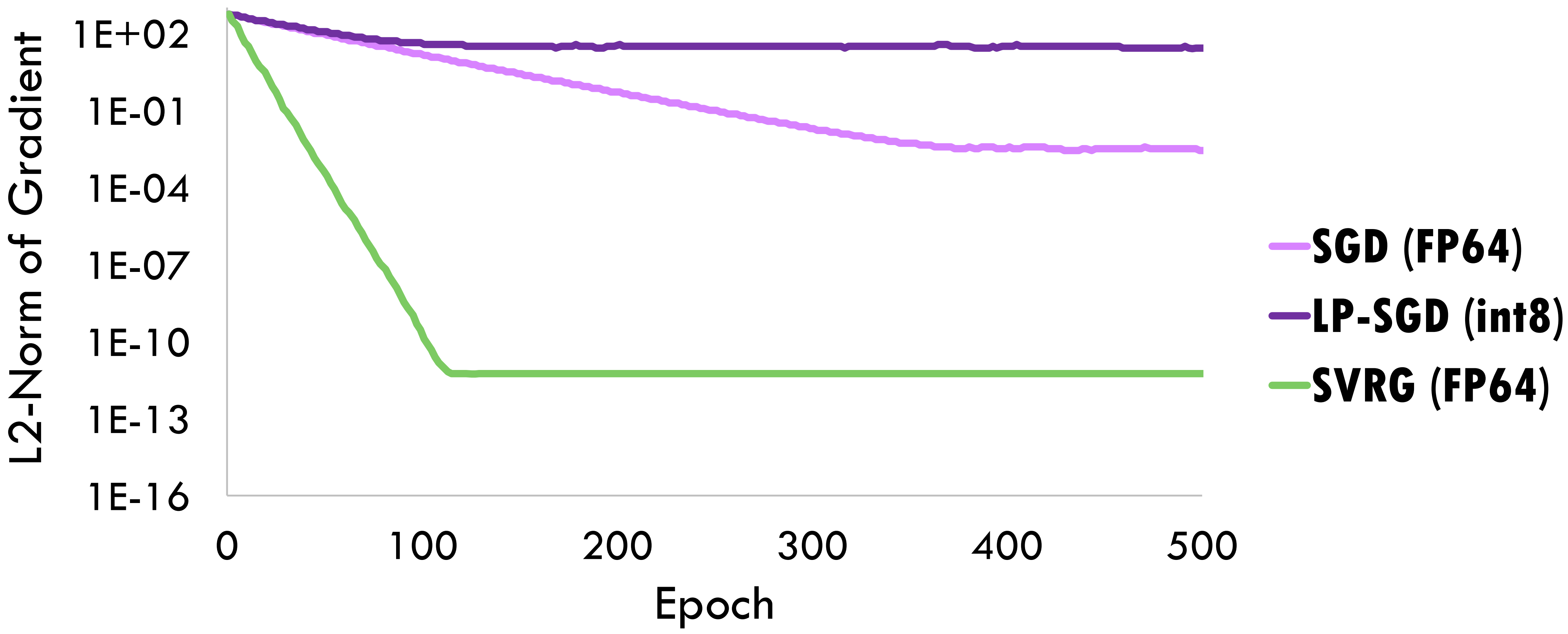
- ↓ Computational cost
- ↑ Gradient variance
- ↓ Convergence rate
- ↑ Noise floor

Low precision introduces even more variance!



Idea 1: Use Stochastic Variance Reduced Gradient (SVRG).

SVRG converges faster than SGD



SVRG is proven to converge at a linear rate due to reduced variance.

Stochastic Variance Reduced Gradient (SVRG)

for k = 1 to K:

$\widetilde{w} = w$

$\tilde{g}$ = full gradient over the dataset

for t = 1 to T:

do SVRG update step

SVRG works really well on many applications with large variance!

$$w_{t+1} = w_t - \alpha(\nabla f_{i_t}(w_t) - \nabla f_{i_t}(\tilde{w}_t) + \tilde{g})$$

Tradeoff: How often do you take the full gradient?

Conventionally, T = 2N to 5N where N = dataset size.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pp. 315–323, 2013.

Update Steps

$\bigcirc$ = Low-precision

1. SGD

$$w_{t+1} = w_t - \alpha \nabla f_{i_t}(w_t)$$

2. LP-SGD

$$w_{t+1} = w_t - \alpha \nabla f_{i_t}(w_t)$$

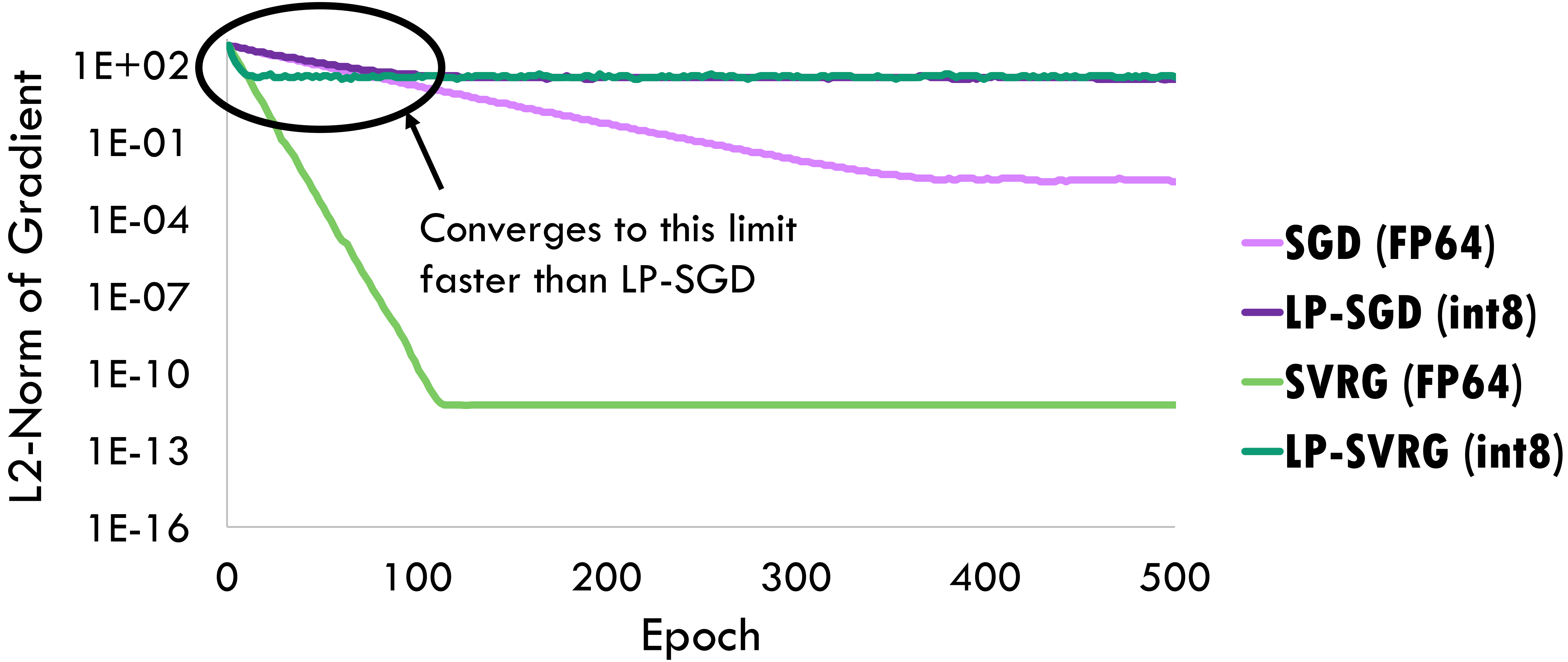
3. SVRG

$$w_{t+1} = w_t - \alpha(\nabla f_{i_t}(w_t) - \nabla f_{i_t}(\tilde{w}_t) + \tilde{g})$$

4. LP-SVRG

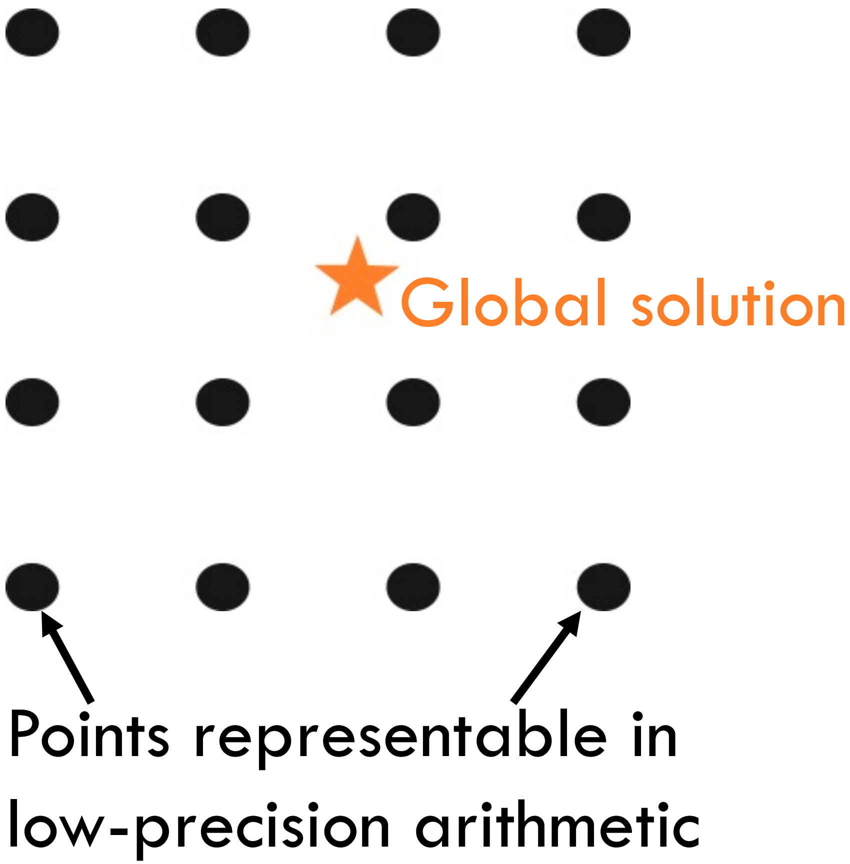
$$w_{t+1} = w_t - \alpha(\nabla f_{i_t}(w_t) - \nabla f_{i_t}(\tilde{w}_t) + \tilde{g})$$

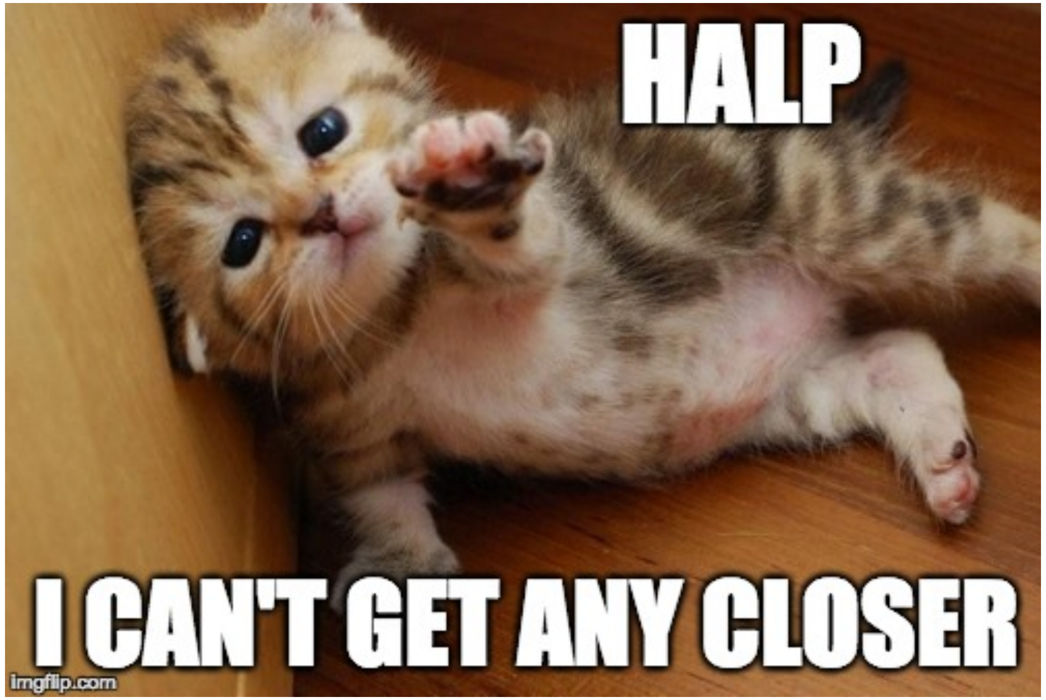
LP-SVRG hits an accuracy limit



Simply making the weights low-precision as in LP-SGD and LP-SVRG is not enough.

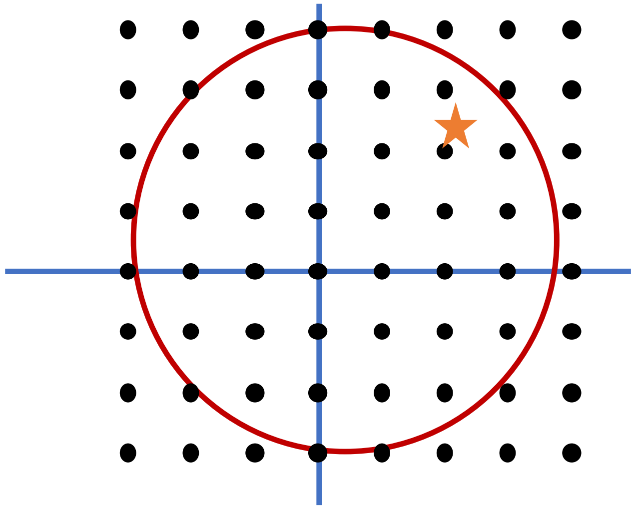
Challenge 2: Static Representable Numbers

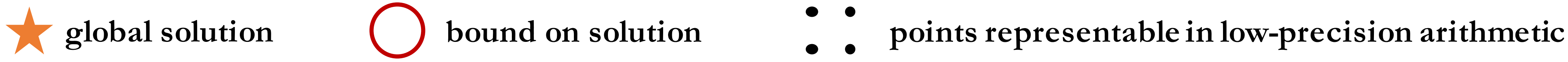




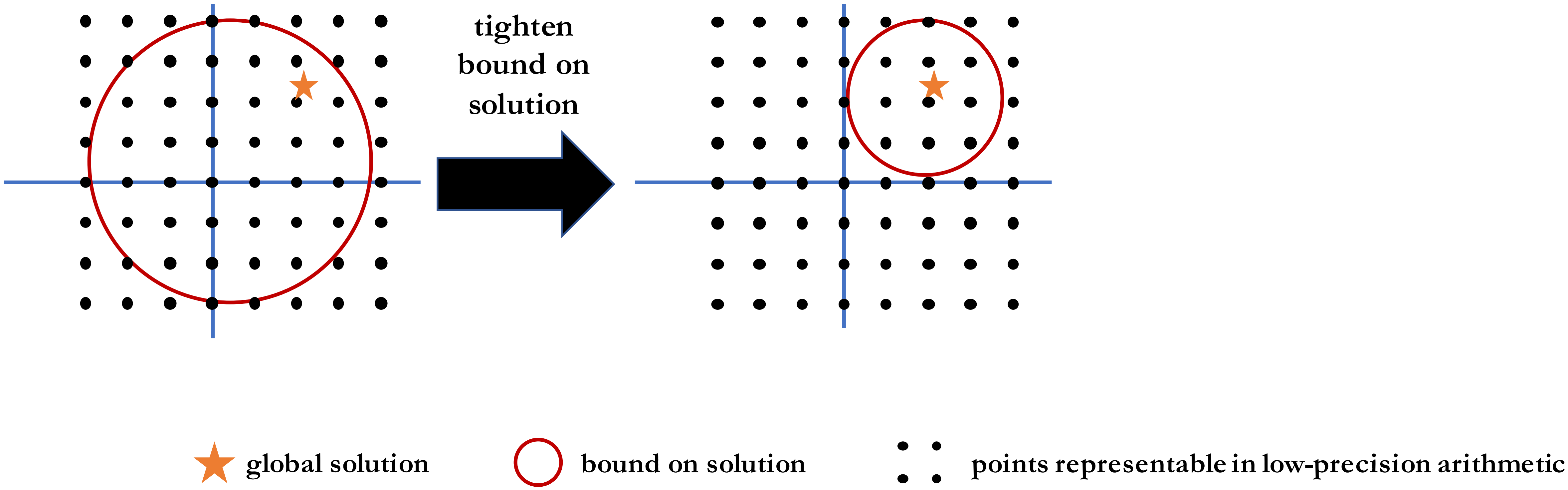
Idea 2: Use Bit Centering.

Bit Centering

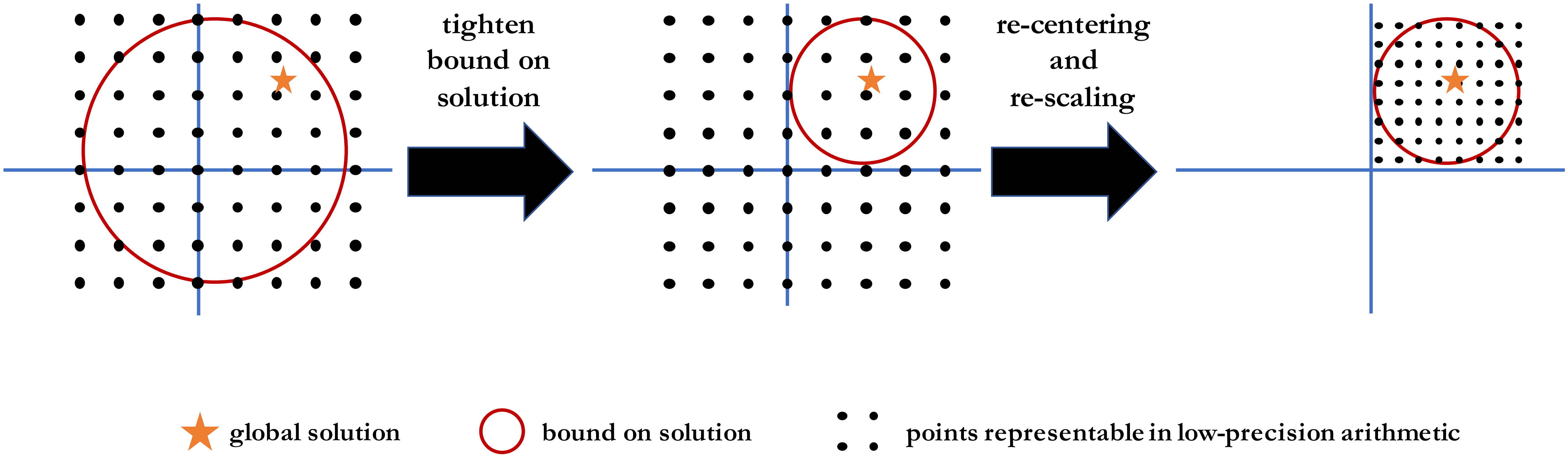




Bit Centering



Bit Centering



Bit Centering

For strongly convex objectives with strong convexity constant μ, bound the location of the optimum with

$$\boxed{\underbrace{\|w - w^*\|}_{z^* \text{ represents this offset}} \leq \frac{1}{\mu}\|\nabla f(w)\|}$$

z^* represents this offset

μ becomes a hyperparameter for non-strongly convex objectives.

HALP = SVRG + Bit Centering

for k = 1 to K:

$\tilde{g}$ = full gradient over the dataset

$\widetilde{w} = \widetilde{w} + z$ (bit center)

for t = 1 to T:

do HALP update step

- Up to 4x faster than full-precision SVRG on convex problems with a C++ implementation using AVX2
- **Can** converge at a linear rate using low precision

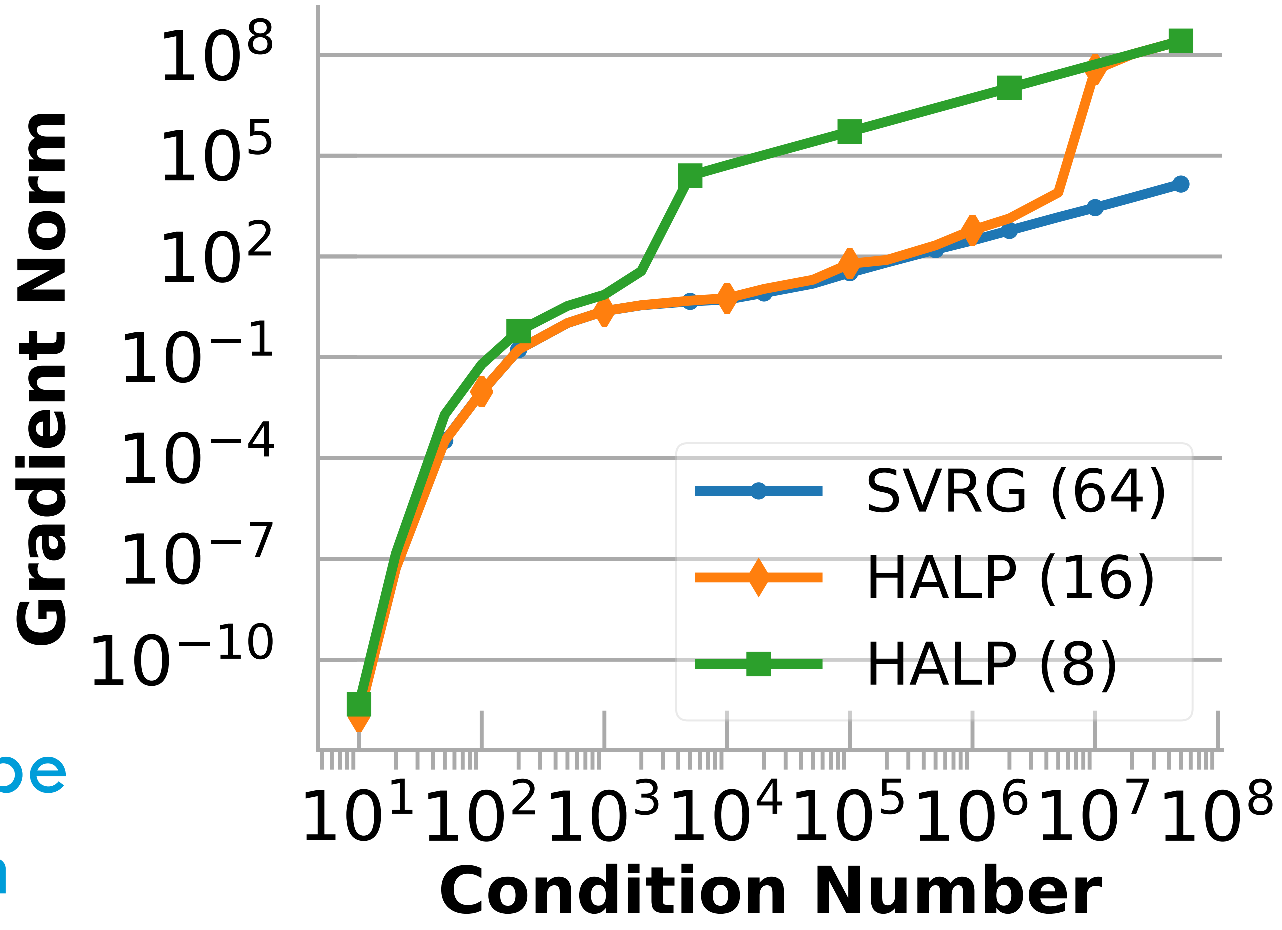
Store $z = w - \widetilde{w}$ instead of w

z is the dynamically changing low-precision representation

$$z_{t+1} = z_t - \alpha(\nabla f_{i_t}(\tilde{w}_t + z_t) - \nabla f_{i_t}(\tilde{w}_t) + \tilde{g})$$

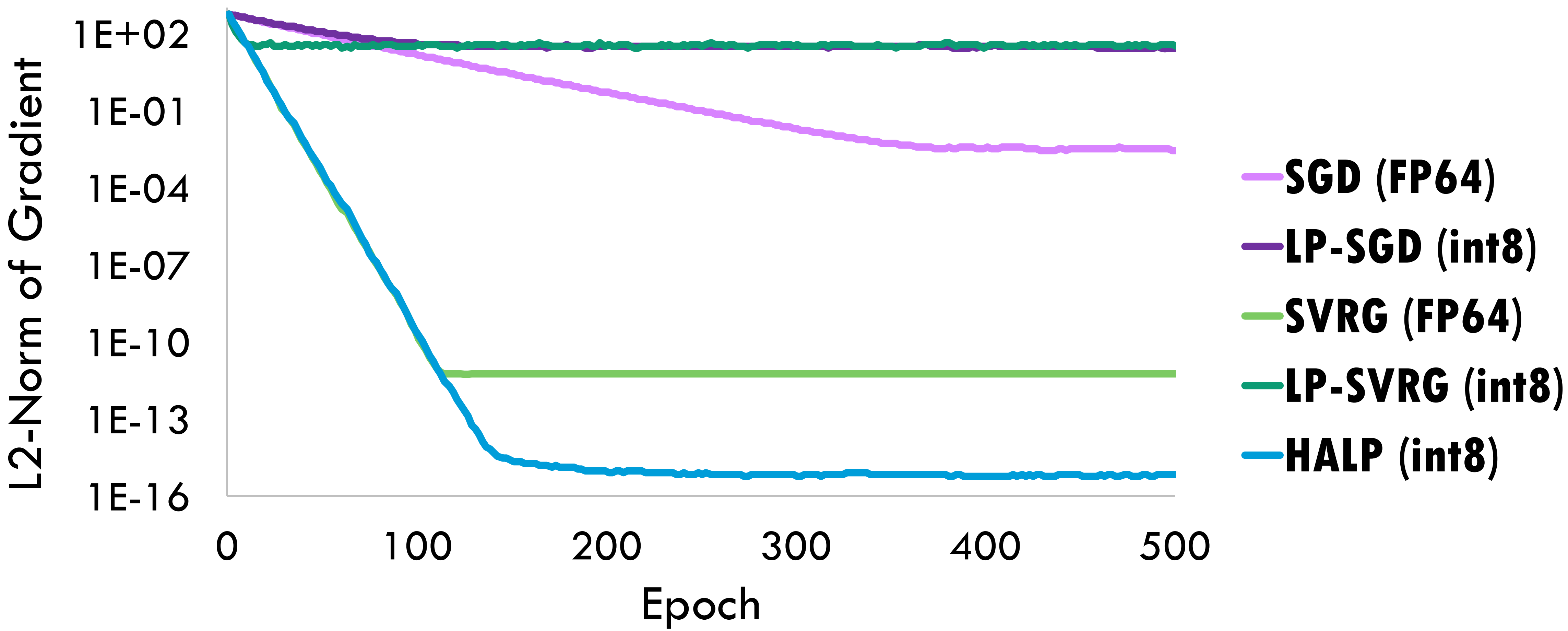
HALP is dependent on conditioning

Tradeoff: The **number of bits needed** for HALP's linear convergence **depends on the condition number.**



Future work: should preconditioning techniques be combined with low-precision training?

HALP surpasses the accuracy limitation



HALP **can** provably converge at a linear rate.

Update Steps

= Low-precision

1. SGD

$$w_{t+1} = w_t - \alpha \nabla f_{i_t}(w_t)$$

2. LP-SGD

$$w_{t+1} = w_t - \alpha \nabla f_{i_t}(w_t)$$

3. SVRG

$$w_{t+1} = w_t - \alpha(\nabla f_{i_t}(w_t) - \nabla f_{i_t}(\tilde{w}_t) + \tilde{g})$$

4. LP-SVRG

$$w_{t+1} = w_t - \alpha(\nabla f_{i_t}(w_t) - \nabla f_{i_t}(\tilde{w}_t) + \tilde{g})$$

5. HALP

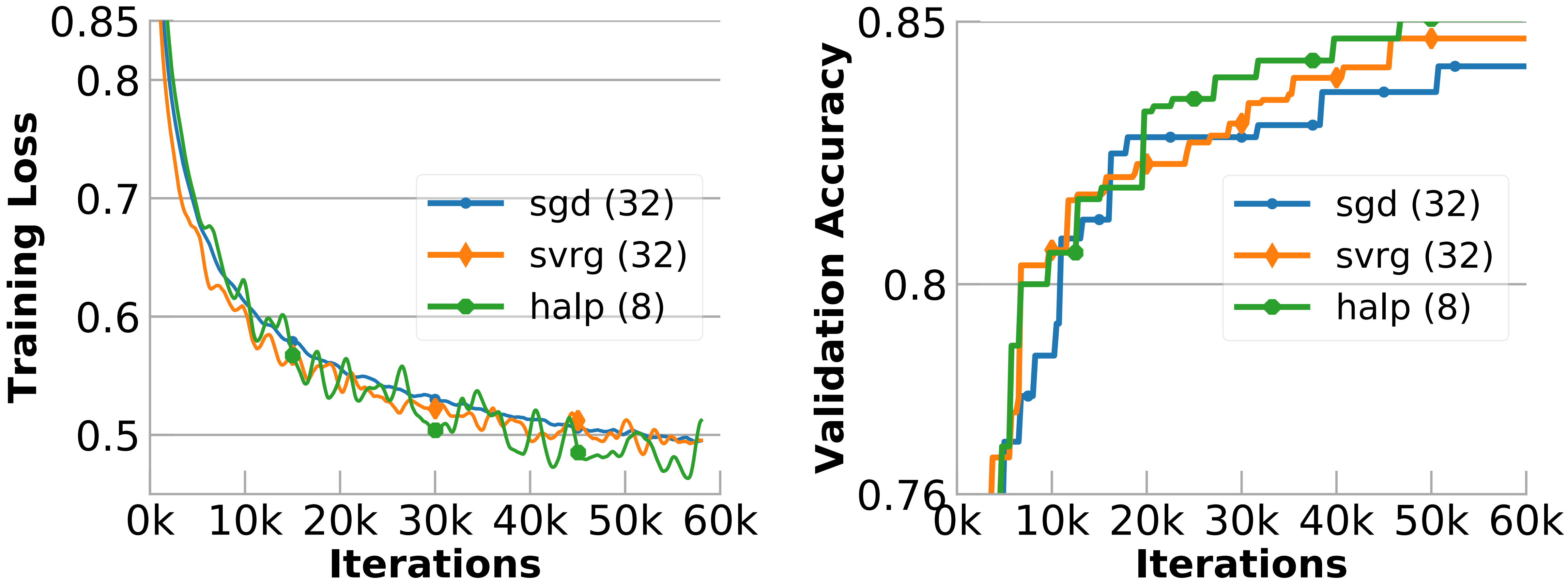
$$z_{t+1} = z_t - \alpha(\nabla f_{i_t}(\tilde{w}_t + z_t) - \nabla f_{i_t}(\tilde{w}_t) + \tilde{g})$$

where $z_t = w_t - \tilde{w}_t$

Deep Learning Results

CNN: HALP versus Full-Precision Algorithms

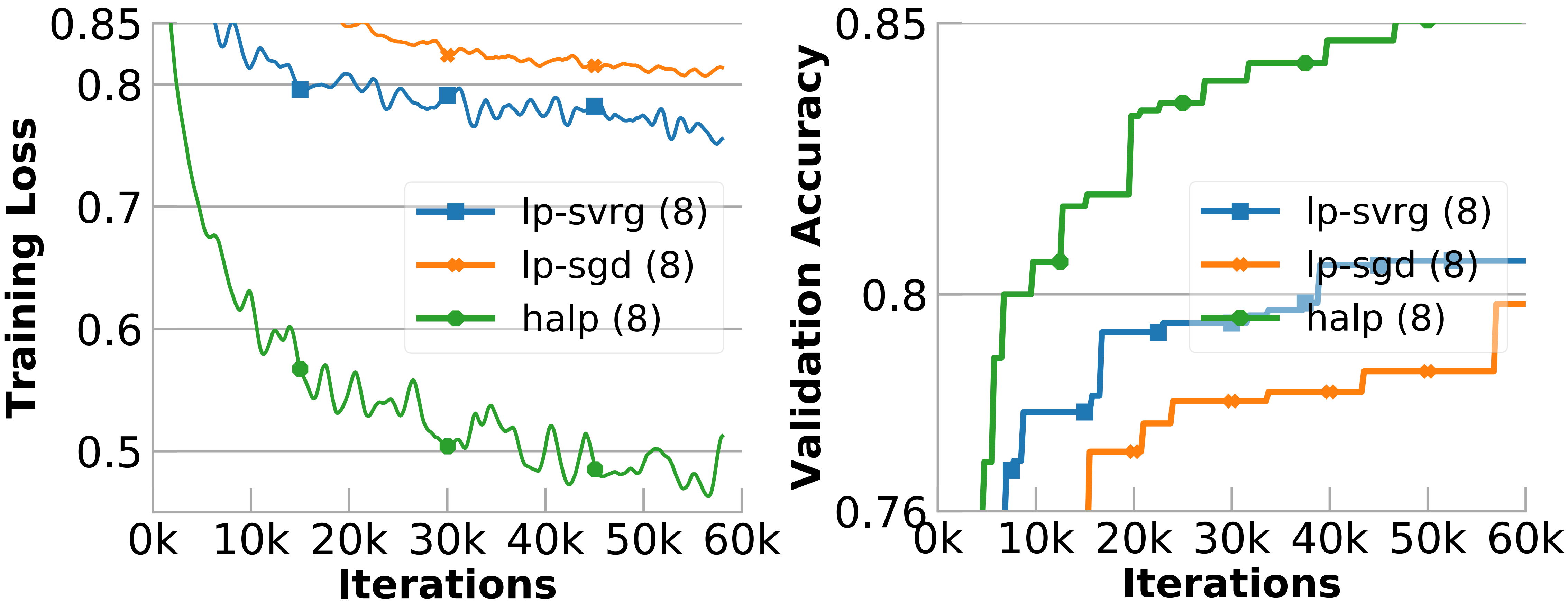
14-layer ResNet on CIFAR10



HALP matches the training loss and validation accuracy of SGD and SVRG.

CNN: HALP versus Low-Precision Algorithms

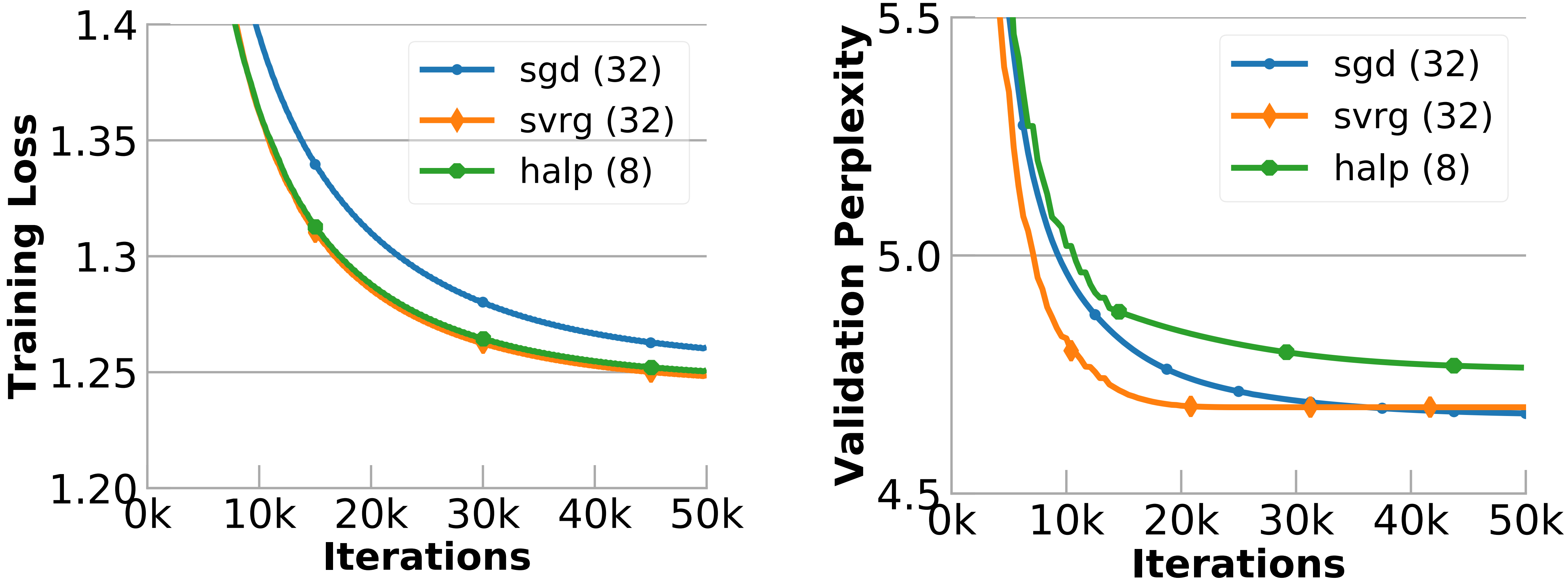
14-layer ResNet on CIFAR10



HALP exceeds the training loss and validation accuracy of LP-SGD and LP-SVRG.

LSTM: HALP versus Full-Precision Algorithms

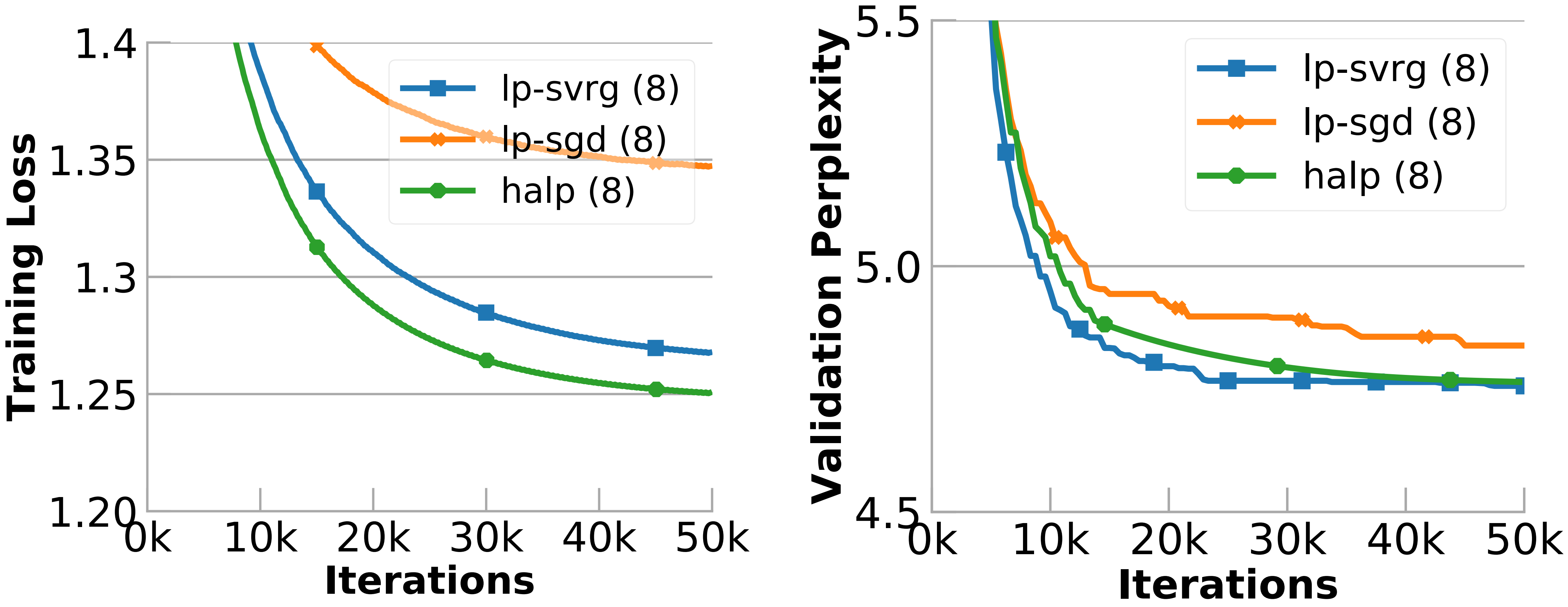
2-layer LSTM on TinyShakespeare



HALP matches SVRG and outperforms SGD in training loss, but reaches a larger (worse) validation perplexity than both.

LSTM: HALP versus Low-Precision Algorithms

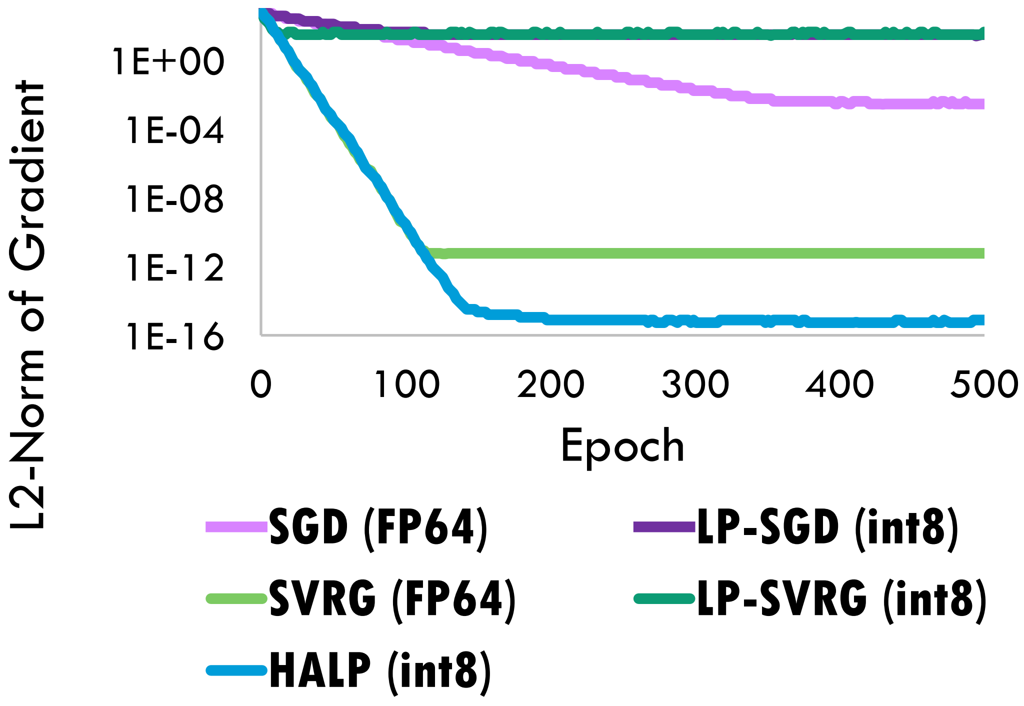
2-layer LSTM on TinyShakespeare



HALP outperforms LP-SVRG and LP-SGD in training loss, while matching LP-SVRG and outperforming LP-SGD in validation perplexity.

Wrap-Up

- HALP = SVRG + Bit Centering
- For convex problems, HALP can converge at a linear rate while using low precision
- Promising results on deep learning
- Future work: More deep learning simulation & FPGA results coming soon



Learn more!

Blog: http://dawn.cs.stanford.edu/2018/03/09/low-precision/

Paper: https://arxiv.org/abs/1803.03383

Contact: mleszczy@stanford.edu

Megan Leszczynski