ACSL

TECHNION Israel Institute of Technology

THE ANDREW & ERNA VITERBI FACULTY OF ELECTRICAL ENGINEERING

Faster Neural Network Training with Approximate Tensor Operations

Menachem Adelman
Intel & Technion
adelman.menachem@gmail.com

Kfir Y. Levy
Technion
kfirylevy@technion.ac.il

Ido Hakimi
Technion
idohakimi@gmail.com

Mark Silberstein
Technion
mark@ee.technion.ac.il

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

DNN Acceleration Approaches

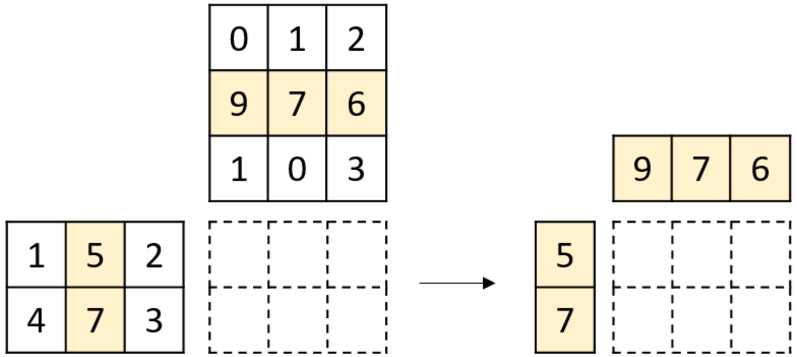
Quantization and low precision (Hubara et al., 2016; Micikevicius et al., 2017; Seide et al., 2014; Wen et al., 2017)	Enforcing low-rank structures (Mamalet & Garcia, 2012; Kuchaiev & Ginsburg, 2017)	Weight extrapolations (Kamarthi & Pittner, 1999)
Channel gating/pruning (Hua et al., 2019; Gao et al., 2018)	Asynchronous gradient updates (Recht et al., 2011; Strom, 2015)	Selective sparsification and locality-sensitive hashing (Kitaev et al., 2020)
Model compression (Denton et al., 2014; Jaderberg et al., 2014; Lebedev et al., 2014; Osawa et al., 2017; Gong et al., 2014; Han et al., 2015)	Partial gradient updates (Sun et al., 2017a)	Low-rank approximation (Choromanski et al., 2020; Wang et al., 2020)

- All these approaches can be interpreted as **approximations**
- Can we extend approximations to the matrix/tensor operation level?

Approximate Matrix Multiplication

- There is a rich literature on approximate matrix multiplication
- In this work, we focus on column-row sampling (CRS) (Drineas & Kannan, 2001; Drineas et al., 2006)
  - Computationally light-weight
  - Sampled matrices can be multiplied using dense HW and libraries

$$A^\top B \approx \sum_{t=1}^{k} \frac{1}{k p_{i_t}} A^{\top (i_t)} B_{(i_t)}$$



$$p_i = \frac{|A_{(i)}||B_{(i)}|}{\sum_{j=1}^{n} |A_{(j)}||B_{(j)}|}$$

Can we train neural networks with approximate matrix multiplication?
What are the relations between exact and approximate training?

Approximate Linear Regression

- Plugging-in CRS in linear regression SGD training leads to biased gradient estimates
- We develop *Bernoulli-CRS* sampling algorithm which samples column/row pairs independently
- Applied to linear regression, training with Bernoulli-CRS is equivalent to minimizing the original loss with dynamically-scaled L_2 weight regularization:

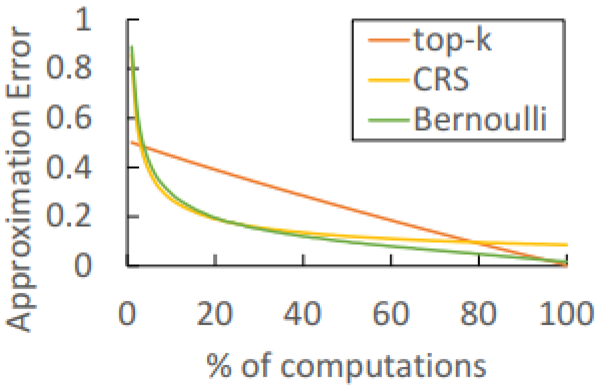
$$\mathcal{R}(w) = \mathbf{E}\left[\sum_{j=1}^{n} \frac{1-p_j}{p_j} x_j^2 w_j^2\right]$$

Non-linear Deep Networks

- Hard to provide general guarantees for approximate training due to non-linear activations
- However, if approximations are limited to the backward pass then under certain conditions the approximated gradients are unbiased with bounded second moments
- This implies the same SGD convergence properties of the original problem! (See e.g Ge et al., 2015)

Top-*k* - Selection Without Scaling

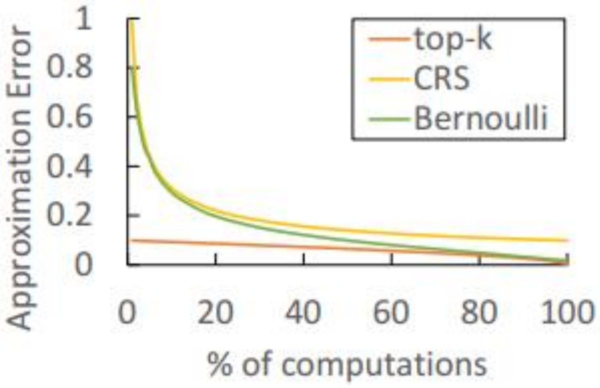
- Both CRS and Bernoulli-CRS required scaling factors to be unbiased
- Under certain conditions, selecting the column-row pairs with the highest sampling probabilities and without scaling provide the MMSE estimator minimizing the approximation error



(a) Matrix product: both matrix entries drawn from $\mathcal{N}(1, 1)$

Bernoulli > CRS for large k

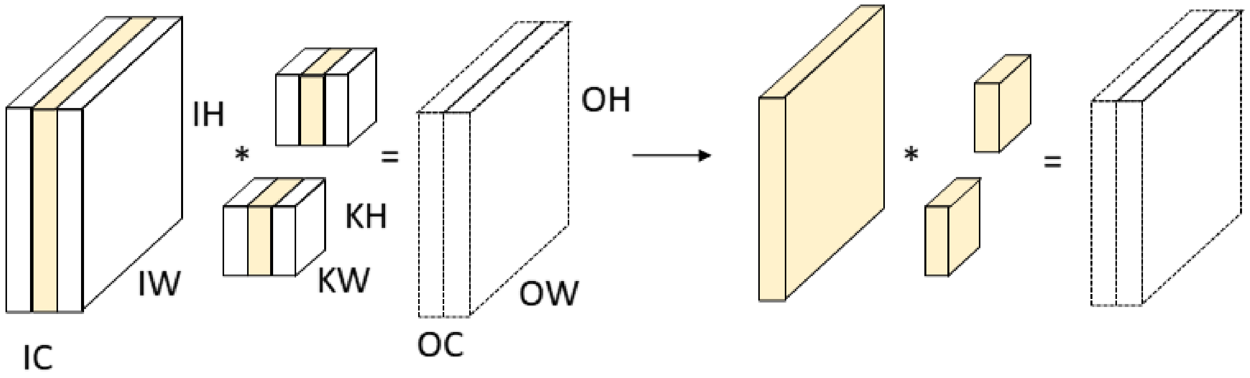
Top-*k* > Bernoulli > CRS for zero-mean matrix



(b) Matrix product: one matrix entries drawn from $\mathcal{N}(0, 1)$, the other from $\mathcal{N}(1, 1)$

Approximating Convolutions for CNNs

- Extending CRS to convolution by sampling across the input channels:



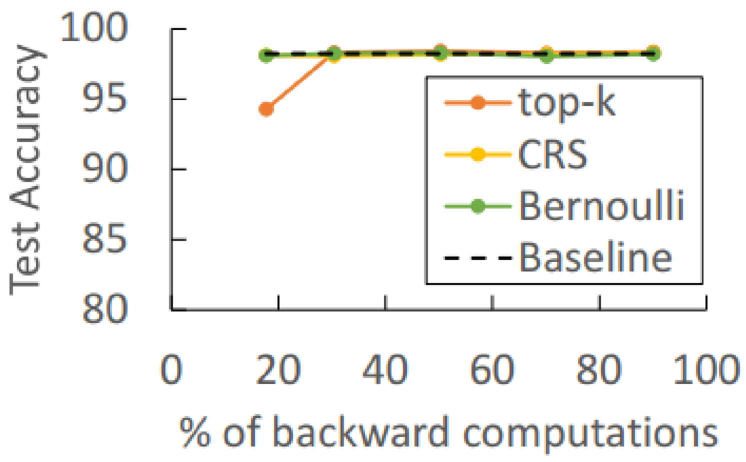
- We prove the approximation is unbiased and derive optimal sampling probabilities
- Bernoulli and Top-*k* can be derived for convolutions as well

Experimental Results

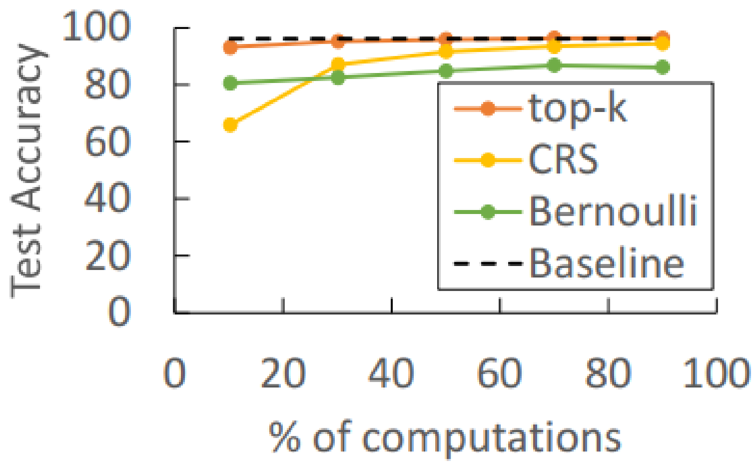
- We implement CRS, Bernoulli sampling and Top-*k* in PyTorch
- No change in hyper-parameters
- Evaluating on MLP and CNN for MNIST, Wide Resnet-28-10 for CIFAR-10 and ResNet-50 and ResNet-152 for ImageNet
- Training on Nvidia V100

Forward vs Backward Sampling

- Backward-only sampling worked well on MNIST but provided worse results on CIFAR-10
- In CIFAR-10 with forward sampling, Top-*k* performed the best



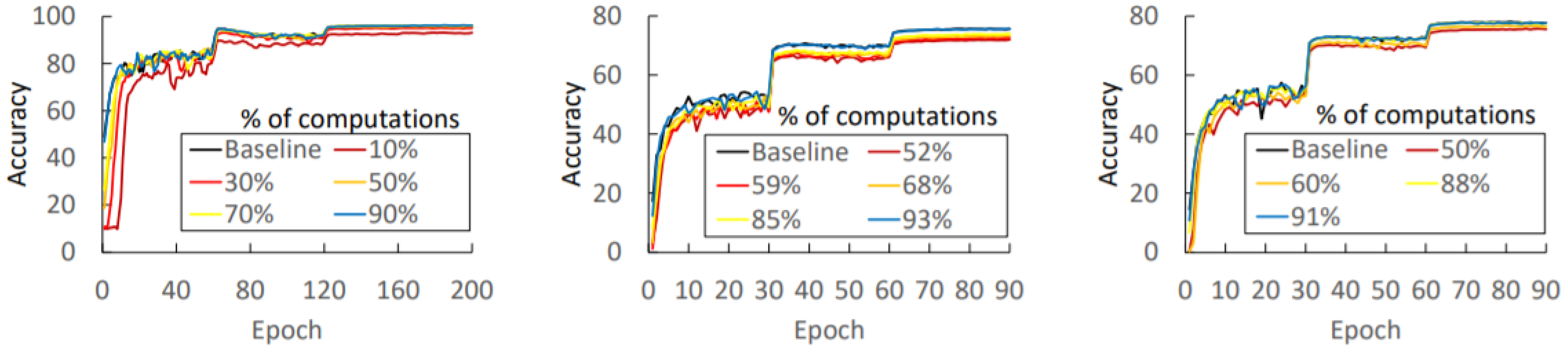
3-layer MLP on MNIST
(exact forward, approximate backward)



WRN-28-10 on CIFAR-10
(approximate forward and backward)

Learning Curves

- Learning curves of approximate training follow the accurate baseline



(a) WRN-28-10 CIFAR-10 (b) ResNet-50 Imagenet (top-1) (c) ResNet-152 Imagenet (top-1)

Figure 6: Learning curves for validation accuracy under different top-k sampling ratios

Experimental Results – Top-*k*

- Results for Top-*k* forward sampling:

Network		Compute Reduction	Accuracy (Baseline)	Training Speedup
MLP (MNIST)		50%	98.22% (98.22%)	-
CNN (MNIST)		66%	99.25% (99.35%)	-
WRN-28-10 (CIFAR-10)		50%	95.89% (96.17%)	1.33x
ResNet-50 (ImageNet)		6.5%	75.63% (75.6%)	1.04x
ResNet-152 (ImageNet)	Single Node	40%	76.44% (77.65%)	1.16x
		9%	77.66% (77.65%)	1.04x

Approximations provide up to 66% reduction in the amount of computations and 1.3x wall-time speedup

Multi-Node Training

- We develop another flavor of top-*k* selection according to the weight norms only
- Reduce the gradient communication in data-parallel training
- Up to 1.37x training speedup

Network		Compute Reduction	Communication Reduction	Accuracy (Baseline)	Training Speedup
ResNet-152 (ImageNet)	8 Nodes	40%	48%	76.44% (77.65%)	1.37x
		12%	23%	77.48% (77.65%)	1.13x
		9%	13%	77.8% (77.65%)	1.09x

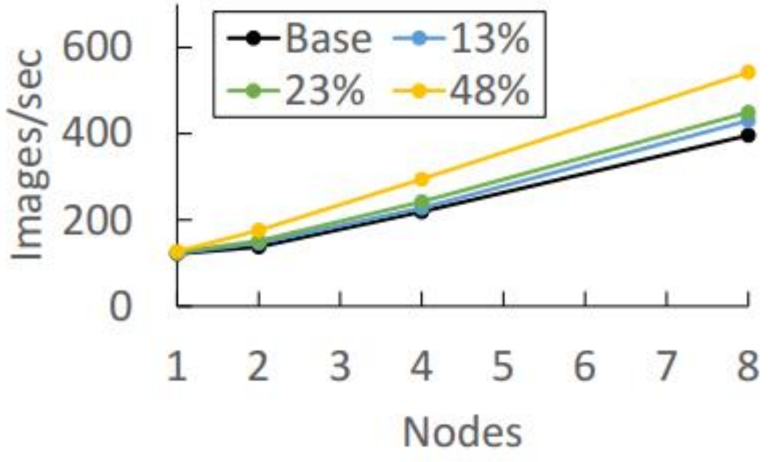


Figure 5. AllReduce with top-k-weights sampling (% fewer gradients sent).

Approximations can reduce communication on top of compute

Conclusion

- We demonstrate the utility of sample-based approximation for neural network training, both theoretically and empirically
- Research opportunities:
  - Further acceleration through dedicated GPU primitives fusing sampling and matrix multiplication/convolution
  - Varying and adaptive sampling rates for different layers and iterations
  - Studying other approximation algorithms
  - Applications in resource-constrained environments
  - Bridging the gaps between our theoretical results and what worked best in practice
- We believe that sample-based approximations and fast approximations in general are valuable additions to the toolbox of techniques for deep learning acceleration