Large-Scale Distributed Second-Order Optimization Using Kronecker-Factored Approximate Curvature for Deep Convolutional Neural Networks

Kazuki Osawa[1], Yohei Tsuji[1,3], Yuichiro Ueno[1], Akira Naruse[2], Rio Yokota[1,3], and Satoshi Matsuoka[1,4]
[1]Tokyo Tech, [2]NVIDIA, [3]AIST-Tokyo Tech RWBC-OIL, AIST, [4]RIKEN CCS

Tokyo Tech NVIDIA CVPR LONG BEACH CALIFORNIA June 16-20, 2019

Image Classification

Set of image & class label: $S = \{(\boldsymbol{x}, y)\}$

Goal of optimization: $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^m} \mathcal{L}(\boldsymbol{\theta})$

Loss function: $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{|S|} \sum_{(\boldsymbol{x}, y) \in S} -\log p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$

Optimization in Deep Learning

Stochastic learning with mini-batch:

$B \subset S \qquad \mathcal{L}_B(\boldsymbol{\theta}) = \frac{1}{|B|} \sum_{(\boldsymbol{x}, y) \in B} -\log p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$

- **S**tochastic **G**radient **D**escent

  $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \nabla \mathcal{L}_B(\boldsymbol{\theta}^{(t)})$

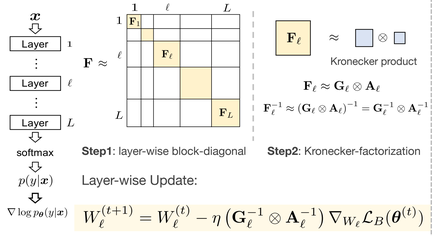
- **N**atural **G**radient **D**escent (S. Amari, 1998)

  $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \mathbf{F}(\boldsymbol{\theta}^{(t)})^{-1} \nabla \mathcal{L}_B(\boldsymbol{\theta}^{(t)})$

  (empirical) Fisher information matrix

  $\mathbf{F}(\boldsymbol{\theta}) = \frac{1}{|B|} \sum_{(\boldsymbol{x}, y) \in B} \nabla \log p_{\boldsymbol{\theta}}(y|\boldsymbol{x}) \nabla \log p_{\boldsymbol{\theta}}(y|\boldsymbol{x})^{\mathrm{T}} \in \mathbb{R}^{m \times m}$

K-FAC (J. Martens+, 2015) accelerates NGD



Step1: layer-wise block-diagonal

Step2: Kronecker-factorization

Layer-wise Update:

$W_\ell^{(t+1)} = W_\ell^{(t)} - \eta \left(\mathbf{G}_\ell^{-1} \otimes \mathbf{A}_\ell^{-1}\right) \nabla_{W_\ell} \mathcal{L}_B(\boldsymbol{\theta}^{(t)})$

Which Optimizer is "Fast" ?

training time = time / iteration × # iterations

SGD	Good	Bad
NGD	Bad	Good
K-FAC	Not Bad	Good

(at least for a small DNN)

Large-Scale Distributed Deep Learning

Large mini-batch training is essential

- The goal of large-scale distributed deep learning is accelerating training by using many GPUs at the same time.
- Larger mini-batch allows us to utilize more GPUs and to converges in fewer iterations.

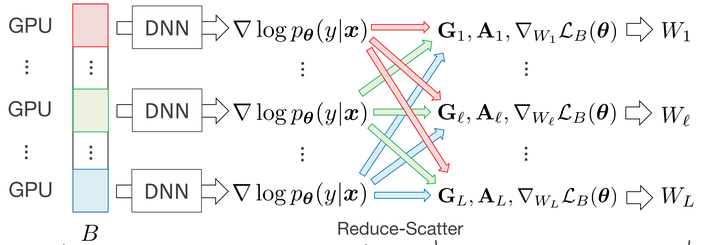
But, large mini-batch training is difficult …

- SGD with large mini-batch (> 8K) suffers from the *generalization gap* (P. Goyal+, 2018).
- Previous approaches tried to solve this problem by *ad-hoc* modification of SGD.
- K-FAC with large mini-batch (<= 2K) was said to be able to generalize well (J. Ba+, 2017), but the comparison to SGD wasn't done with well-trained models.
- No one has shown the clear advantage of K-FAC over SGD since K-FAC requires extra (heavy) computation/communication that need to be distributed among many GPUs.

K-FAC can handle *extremely* large mini-batch (Our Contribution)

- We implemented Distributed K-FAC which allowed us to scale K-FAC up to 1024 GPUs.
- We trained ResNet-50 on ImageNet with extremely large mini-batch size (131K).
- We show the advantage of K-FAC over SGD for large mini-batch for the first time.

Our Approach | Distributed K-FAC



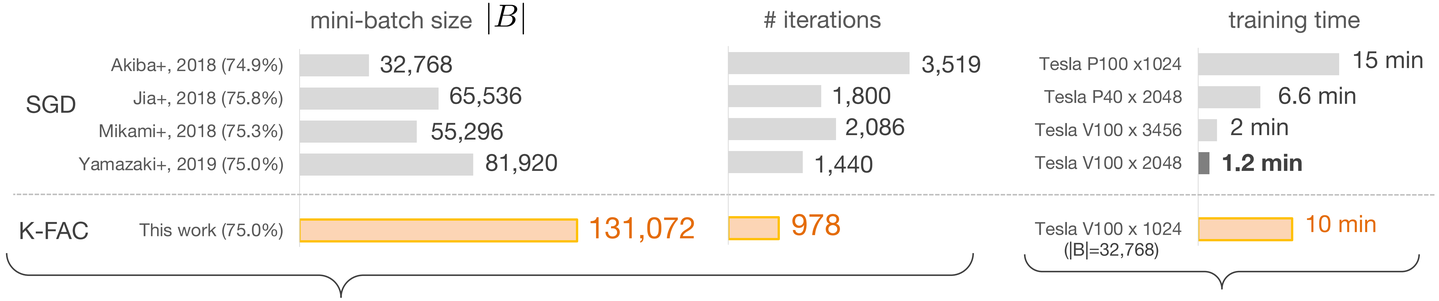
Data parallelism

Each GPU handles different parts of mini-batch B

Layers parallelism

Each GPU applies K-FAC to the weights of different layers (the weights will be synchronized.)

Benchmark | Training ResNet-50 on ImageNet in 978 iterations / 10 minutes with K-FAC



K-FAC converges faster than SGD (SOTA results)

- Our Distributed K-FAC enabled tuning large mini-batch K-FAC on ImageNet *in realistic time*.
- We tuned K-FAC with extremely large mini-batch size {4096, 8192, 16384, 32768, 65536, 131072}.
- We achieved validation accuracy >75% in {10948, 5434, 2737, 1760, 1173, 978} iterations.
- We showed the advantage of a second-order method (K-FAC) over a first-order method (SGD) in ImageNet classification for the first time.

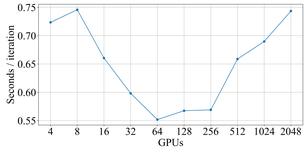
Competitive training time

- Data and layers hybrid parallelism introduced in our design allowed us to train on 1024 GPUs
- We achieved 74.9% in 10 min by using K-FAC with the stale Fisher information matrix for mini-batch size 32,768.
- (32images/GPU x 1024GPUs = 32,768images)

Discussion | Kazuki's-FAQ

Q1. How much does *Distributed K-FAC* improve the performance?

A1. Here is a plot which shows that larger # GPUs improves the performance while # GPUs < # layers (ResNet-50 has 107 layers in total (FC, Conv, BN)).



Q2. Why do you want to accelerate training ResNet-50 on ImageNet?

A2. We chose training ResNet-50 on ImageNet as a benchmark where highly optimized SGD results are available as references. What we really want to accelerate are even larger problems.

Q3. What is the benefit of using K-FAC? It's slower than SGD!

A3. Since K-FAC converges in fewer iterations than SGD, improving the per iteration performance will directly accelerate the training without additional tuning. Even if we end up having similar performance to the best known first-order methods, at least we will have a better understanding of why it works by starting from second-order methods.

Q4. What's the Next?

A4. There is still room for improvement in our distributed design to overcome the bottleneck of computation/communication for K-FAC – the Fisher information matrix can be approximated more aggressively without loss of accuracy.

Q5. Codes available?

A5. Our Chainer implementation is available on GitHub! (PyTorch version is coming soon)

https://github.com/tyohei/chainerkfac

Acknowledgements Computational resource of AI Bridging Cloud Infrastructure (ABCI) was awarded by "ABCI Grand Challenge" Program, National Institute of Advanced Industrial Science and Technology (AIST). This work is supported by JST CREST Grant Number JPMJCR19F5, Japan. This work was supported by JSPS KAKENHI Grant Number JP18H03248. (Part of) This work is conducted as research activities of AIST - Tokyo Tech Real World Big-Data Computation Open Innovation Laboratory (RWBC-OIL). This work is supported by "Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures" in Japan (Project ID: jh180012-NAHI).

Kazuki Osawa, Ph.D. candidate at Tokyo Tech
(oosawa.k.ad at m.titech.ac.jp)
supervised by **Rio Yokota**