Inefficiency of K-FAC for Large Batch Size Training

Linjian Ma*, Gabe Montague*, Jiayu Ye*, Zhewei Yao, Amir Gholami, Kurt Keutzer, Michael W. Mahoney

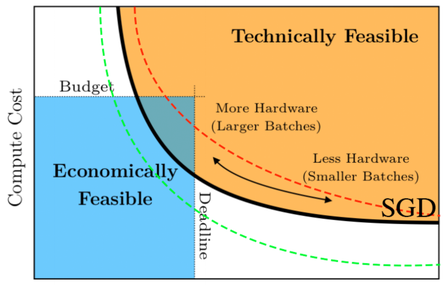
EECS Department, University of California at Berkeley

The Question

- What is the scalability behavior of K-FAC [2], and how does it compare with that of SGD?
- How does increasing batch size affect the hyperparameter sensitivity of K-FAC?

Motivation

Modern development of AI technology is limited by the time and expense required to train deep neural networks. In recent years, optimization researchers have proposed that second-order training methods may help to alleviate the burden of training. Second-order methods are based on higher-order derivatives and may help to significantly accelerate the process.



Is Second Order

BETTER OR WORSE

	Small Batch	Large Batch
SGD	high accuracy	low accuracy
K-FAC	high accuracy	???

Contributions

Our key observations comparing K-FAC and SGD for large batch training are as follows:

- **Performance.** Even with extensive hyperparameter tuning, K-FAC has comparable, but not superior, train/test performance to SGD.
- **Speedup.** Both K-FAC and SGD have diminishing returns with the increase of batch size. Increasing batch size for K-FAC yields lower, i.e., less prominent, speedup, as compared with SGD, when measured in terms of iterations.
- **Hyperparameter Sensitivity.** K-FAC hyperparameter sensitivity depends on both batch size and epochs/iterations. For fixed epochs, i.e., running the same number of epochs, larger batch sizes result in greater hyperparameter sensitivity and smaller regions of hyperparameter space which result in "good convergence." For fixed iterations, i.e., running the same number of iterations, larger batch sizes result in less sensitivity and larger regions of hyperparameter space which result in "good convergence".

SGD vs K-FAC

SGD steps have the form:

$$\theta_{t+1} = \theta_t - \eta_t \frac{1}{|B|} \sum_{(x,y)\in B} \nabla_\theta l(x, y, \theta_t),$$

K-FAC steps have the form:

$$\theta_{t+1} = \theta_t - \eta_t F^{-1} \frac{1}{|B|} \sum_{(x,y)\in B} \nabla_\theta l(x, y, \theta_t),$$

where F is the Fisher Information Matrix (FIM):

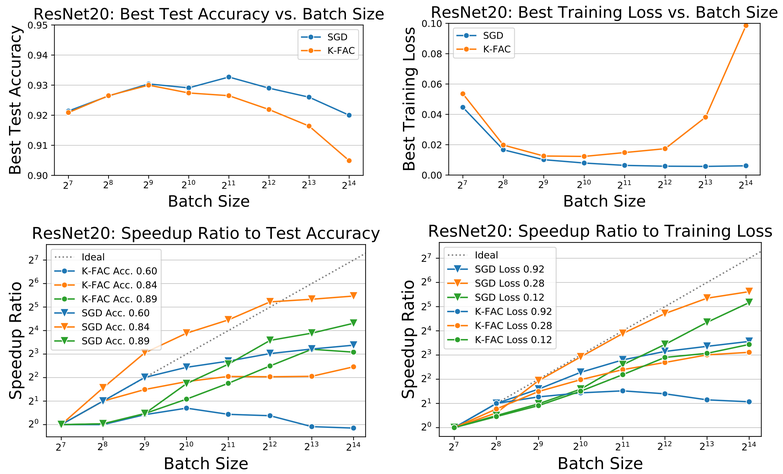
$$F = \mathbb{E}[\nabla_\theta \log p(y|x, \theta) \nabla_\theta \log p(y|x, \theta)^\top].$$

Experimental Setup

We investigate the performances of both K-FAC and SGD on CIFAR-10 with ResNet20 and ResNet32, and on SVHN with AlexNet.

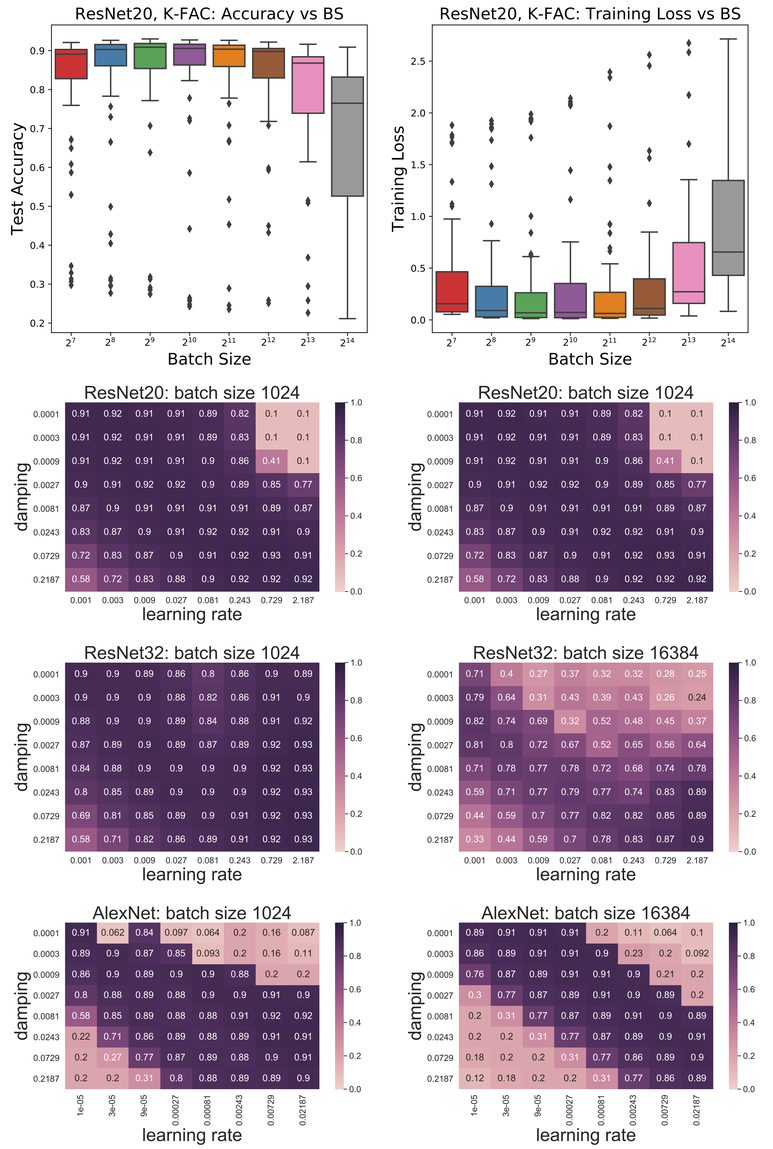
- *Training Budget and Learning Rate Schedule*: for our training of K-FAC and SGD, we use the adjusted epoch budget, in which the epoch limit of training is extended proportionally to the log of the batch size. Specifically, we use the rule: for CIFAR-10, number of training epochs equals(log2(batch size/128) + 1)×100; for SVHN, it equals(log2(batch size/128) + 1)×20. This schedule allows larger-batch training runs more of a chance to converge.
- *Hyperparameter Tuning*: for K-FAC, we conduct a log-space grid search over 64 configurations with learning rates and dampings. we employ a similarly extensive hyperparameter tuning process for SGD, tuning learning rates and momentum.

K-FAC and SGD Performance Comparison



- Compared with K-FAC, SGD made much more progress per-iteration in reducing loss.
- For both K-FAC and SGD, diminishing return effects are present. In all examined cases, K-FAC deviates from ideal scaling (dotted lines) to a greater extent than SGD,as batch size increase.

K-FAC Hyperparameter Sensitivity



We observe that there's a shrinking of the high-accuracy region with increasing batch size. In addition, there's a positive correlation between damping and learning rate.

References

[1] L. Ma, G. Montague, J. Ye, Z. Yao, A. Gholami, K. Keutzer, M. Mahoney, *Inefficiency of K-FAC for Large Batch Size Training*, AAAI'20.

[2] J. Martens and R. B. Grosse, *Optimizing neural networks with Kronecker-factored Approximate Curvature*, CoRR,vol. abs/1503.05671.

[3] S. McCandlish, J. Kaplan, D. Amodei, and OpenAI Dota Team. *An empirical model of large-batch training*, arXiv:1812.06162.