Speeding up ImageNet Training on Supercomputers

Yang You[1], Zhao Zhang[2], Cho-Jui Hsieh[3], James Demmel[1], Kurt Keutzer[1]
UC Berkeley[1], TACC[2], UC Davis[3]

ABSTRACT

In this paper, we showcase supercomputers' capability of speeding up ImageNet training using thousands of processors. Our technical solution is based on the layer-wise adaptive rate scaling (LARS) algorithm. Using the Stampede2 supercomputer, we are able to reduce the 100-epoch AlexNet and ResNet-50 training on the ImageNet 1k-category (ImageNet-1k) dataset from hours to 11 minutes and 20 minutes, respectively. Our solution matches the state-of-the-art top-1 test accuracy in both cases. Particularly for ResNet-50, the top-1 test accuracy converges to the baseline of 74.9% at the 64th epoch, which is 14 minutes from the beginning. Compared to the baseline of a previous study from a group of Facebook researchers, our solution shows a higher top-1 test accuracy on batch sizes that are larger than 16k. The implementation is open source and has been released in the Intel distribution of Caffe v1.0.7.

KEYWORDS

Distributed Machine Learning, Scalable Algorithm

ACM Reference format:
Yang You[1], Zhao Zhang[2], Cho-Jui Hsieh[3], James Demmel[1], Kurt Keutzer[1]. 2018. Speeding up ImageNet Training on Supercomputers. In *Proceedings of SysML conference, Stanford, CA, USA, Feb 2018 (SysML'18),* 2 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Deep neural networks (DNN) are increasingly popular in both scientific research and industry. Traditionally, practitioners train models on one or multiple GPUs, and a single training process can take days to finish. For example, the 90-epoch ResNet-50 training on the ImageNet-1k dataset using a Nvidia M40 GPU takes 14 days. In particular, this DNN training process has $\sim 10^{18}$ single precision operations in total. On the other hand, the world's current fastest supercomputer can finish 2×10^{17} single precision operations per second. If the above training can make full use of the computing capability of the fastest supercomputer, it should be able to finish in five seconds.

Over the last two years, researchers have focused on closing this significant performance gap by scaling DNN training to larger numbers of processors. Most successful approaches to scaling ImageNet training have used the synchronous stochastic gradient descent (synchronous SGD). One of the challenges using synchronous SGD at scale is to maintain the machine efficiency, where it requires sufficient amount of work for each processor. Thus, the focus on scaling DNN training has translated into a focus on developing training algorithms that enable large batch size in data-parallel synchronous SGD without losing test accuracy over a fixed number of epochs.

SysML'18, Feb 2018, Stanford, CA, USA
2018. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

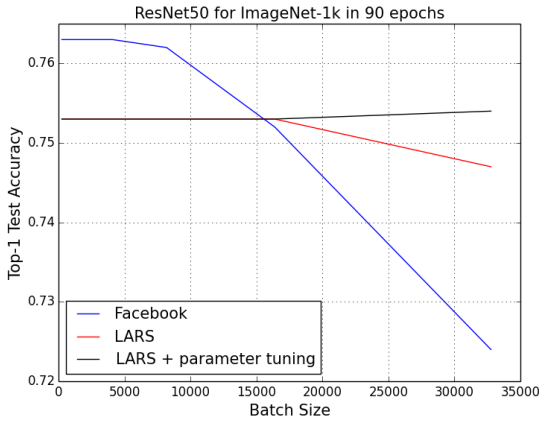

Figure 1: Top-1 Test Accuracy Comparison. Our baseline's accuracy is slightly lower than Facebook's version (76.2% vs 75.3%), as we use weaker data augmentation. However, at large batch sizes larger than 16k, our accuracy is significantly higher than Facebook's accuracy [1].

Table 1: Compare to state-of-the-art ImageNet training speed with ResNet-50.

Work	Batch Size	Hardware	Test Accuracy	Time
Akiba et al.	32K	1024 P100 GPUs	74.9%	15 mins
Our version	32K	2048 Intel KNLs	74.9%	14 mins

As a result, we have seen the batch size and number of processors increase from 1k batch size on 128 processors [3] to 8k batch size on 256 processors [1] over the last two years. In collaboration with researchers at NVIDIA, we proposed LARS algorithm [6], which increased batch size further to 32k for some DNN models.

Following up on this work, we want to evaluate the effectiveness of the LARS algorithm on scales of thousands of processors and explore supercomputer's capability in reducing DNN training time. In this paper we present the results of this investigation: we efficiently used 2,048 Intel Xeon Platinum 8160 processors to finish the 100-epoch AlexNet training in 11 minutes with 58.6% accuracy, and we used 2,048 Intel Xeon Phi 7250 processors to finish the 90-epoch ResNet-50 training in 20 minutes while converging to the state-of-the-art top-1 test accuracy of 74.9% from the 64th epoch, which is 14 minutes from the beginning. Furthermore, our solution shows test higher accuracy on batch sizes that are larger than 16k 1, compared to previous results from Facebook researchers. Our code has been released in the Intel distribution of Caffe v1.0.7.

2 STATE-OF-THE-ART RESULTS

For large batch DNN training, we need to ensure that the large batch achieves similar top-1 test accuracy with the small batch by running the same number of epochs. Here we fix the number

of epochs because: statistically, one epoch means the algorithm touches the entire dataset once; and computationally, fixing the number of epochs means fixing the total number of floating point operations. State-of-the-art approaches for large batch training include two techniques:

(1) **Linear Scaling** [4]: As the batch size increases from B to kB, the learning rate should increase from η to $k\eta$.

(2) **Warmup Scheme** [1]: With a large learning rate (η), the training process should start from a small η and increase it to the large η in the first few epochs.

By using these techniques, researchers can scale the batch size to 8K. Table 2 shows the state-of-the-art results for large batch training. Goyal et al. [1] finished the 90-epoch ResNet-50 training in 65 minutes on 256 P100 GPUs by using a batch size of 8k. However, we observe that state-of-the-art approaches can only scale batch size to 1k for AlexNet and 8k for ResNet-50. If we increase the batch size to 4k for AlexNet, we can only achieve a 53.1% top-1 test accuracy, which is a 4.9% loss to the baseline.

Table 2: State-of-the-art Large Batch Size and Accuracy.

Team	Model	Small Batch / Accuracy	Large Batch / Accuracy
Google [4]	AlexNet	128 / 57.7%	1k / 56.7%
Amazon [5]	ResNet-152	256 / 77.8%	5k / 77.8%
Facebook [1]	ResNet-50	256 / 76.40%	8k / 76.26%

3 LAYER-WISE ADAPTIVE RATE SCALING

To improve the accuracy for large batch training, a new rule of learning rate (LR) schedule was developed. As mentioned before, we use $w = w - \eta \nabla w$ to update the weights. Each layer has its own weight w and gradient ∇w. Standard SGD algorithm uses the same LR (η) for all the layers. However, from our experiments, we observe that different layers may need different LRs. The reason is that the ratio between $||w||_2$ and $||\nabla w||_2$ varies significantly for different layers. From example, we observe that $||w||_2/||\nabla w||_2$ is only 20 for the conv1.1 layer but 3,690 for the fc6.1 layer, as shown in Table 3.

To speedup the convergence, the users need to use a large LR for the fc6.1 layer. However, this large LR may lead to divergence in the conv1.1 layer. We think this is an important reason of the optimization difficulty in large batch training. Goyal et al [1] proposed the warmup scheme to solve this problem. The warmup scheme works well for ResNet-50 training (batch size $\leq$ 8k). However, only using this recipe does not work for AlexNet with batch size $>$ 1k and ResNet-50 with batch size $>$ 8k.

Table 3: The Ratios between $||w||_2$ and $||\nabla w||_2$ on the fc6.1 and conv1.1 Layer of AlexNet

Layers	$\|\|w\|\|_2$	$\|\|\nabla w\|\|_2$	$\|\|w\|\|_2/\|\|\nabla w\|\|_2$
fc6.1	6.400	0.001734	3690
conv1.1	0.098	0.004909	20

The Layer-wise Adaptive Rate Scaling (LARS) scheme [6] was proposed to improve large-batch's accuracy. The base LR rule is defined in Equation (1).

Table 4: We use the same data augmentation with the original ResNet-50 model [2].

Batch Size	epochs	Peak Top-1 Accuracy	hardware	time
256	90	75.3%	16 KNLs	45h
16384	90	75.3%	1024 CPUs	52m
16000	90	75.3%	1600 CPUs	31m
32768	90	75.4%	2048 KNLs	20m
32768	64	74.9%	2048 KNLs	14m

$$\eta = l \times \gamma \times \frac{||w||_2}{||\nabla w||_2} \quad (1)$$

l is the scaling factor, which we set as 0.001 for AlexNet and ResNet training. γ is a tuning parameter for users. Usually γ can be chosen by linear scaling. In this formulation, different layers can have different LRs. In practice, we add momentum (denoted as μ) and weight decay (denoted as β) to SGD, and use the following sequence for LARS:

(1) get the local LR for each learnable parameter by $\alpha = l \times ||w||_2/(||\nabla w||_2 + \beta||\nabla w||_2)$;

(2) get the true LR for each layer by $\eta = \gamma \times \alpha$;

(3) update the gradients by $\nabla w = \nabla w + \beta w$;

(4) update acceleration term a by $a = \mu a + \eta \nabla w$;

(4) update the weights by $w = w - a$.

4 EXPERIMENTAL RESULTS

We examine the effectiveness of our solution which is based on the LARS algorithm, the linear scaling technique, and the warmup scheme on the Stampede2 supercomputer. Enabled by the large batch size of 32k, the 100-epoch AlexNet training finsihed in 11 mins on 2,048 Intel Xeon Phi processors, and the 90-epoch ResNet-50 training finished in 20 mins on 2,048 Intel Platinum 8160 processors without losing top-1 test accuracy. Other evaluation results is presented in Table
reftab:resnet50$_s peed_c ost$.

5 ACKNOWLEDGEMENT

The Layer-wise Adaptive Rate Scaling (LARS) algorithm was developed by Y. You, B. Ginsburg, and I. Gitman when Y. You was an intern at NVIDIA [6]. The work presented in this paper was supported by the National Science Foundation, through the Stampede 2 (OAC-1540931) award.

REFERENCES

[1] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. *arXiv preprint arXiv:1706.02677* (2017).

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

[3] Forrest N Iandola, Matthew W Moskewicz, Khalid Ashraf, and Kurt Keutzer. 2016. FireCaffe: near-linear acceleration of deep neural network training on compute clusters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2592–2600.

[4] Alex Krizhevsky. 2014. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997* (2014).

[5] Mu Li. 2017. *Scaling Distributed Machine Learning with System and Algorithm Co-design*. Ph.D. Dissertation. Intel.

[6] Yang You, Igor Gitman, and Boris Ginsburg. 2017. Scaling SGD Batch Size to 32K for ImageNet Training. (2017).