# Impact of Changes in the Mini-Batch Size on CNN Training Epoch Time

Peter Bryzgalov (peter@stair.center), Toshiyuki Maeda (tosh@stair.center), Yutaro Shigeto (shigeto@stair.center)
Software Technology and Artificial Intelligence Research Laboratory, Chiba Institute of Technology, Japan (https://stair.center)

The goal of this poster is to clarify how and why changes in the mini-batch size affect CNN training epoch time.

*Mini-batch* size is the number of training samples used in one training iteration. *Epoch time* is the time to use all training samples exactly one time for training.

## How does the mini-batch size affect CNN training epoch time?

### Experiment

- Measure training epoch time over a wide range of mini-batch sizes.
  - Use VGG16[1] CNN with CIFAR100 dataset in Chainer[2] for the experiments.
- Multiple GPU types.

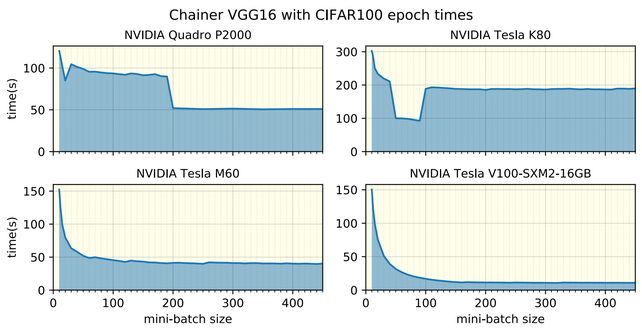



**Fig.1.** Demonstration of high epoch time variability: epoch time of Chainer VGG16 model with CIFAR100 dataset on four types of NVIDIA GPUs over a wide range of mini-batch sizes.

### Observations

- Extremely high variability of epoch time over the range of mini-batch sizes.
  - Epoch time grows exponentially towards smaller mini-batch sizes.
  - On larger mini-batch sizes epoch time gradually stops decreasing (around 200 and larger mini-batch sizes in Fig.1).
- Abrupt changes on machines with Quadro P2000 and Tesla K80 GPUs.

**Table 1.** CPU and GPU specifications

| **CPU** | **GPU** model | Architecture, CUDA capability | | GFLOP/s single prec. |
|---|---|---|---|---|
| Intel Core i5-8400 | NVIDIA Quadro P2000 | Pascal | 6.1 | 3000 |
| Intel Xeon E5-2686 v4 | NVIDIA Tesla K80 | Kepler | 3.7 | 5600-8700 |
| Intel Xeon E5-2686 v4 | NVIDIA Tesla M60 | Maxwell | 5.2 | 7400-9600 |
| Intel Xeon E5-2686 v4 | NVIDIA Tesla V100-SXM2-16GB | Volta | 7.0 | 14900-15700 |

## Changes of convolution algorithms cause abrupt changes in epoch time

### Experiment

- Verify that changes of convolution algorithms cause abrupt changes in epoch time.

### Methods

- Record convolution algorithms used by the Chainer VGG16 sample over a range of mini-batch sizes. The results are shown in the Fig.2.
- Estimate the time of convolutional layers to see if there are abrupt changes. Estimated times are shown in Fig.3.
  - To estimate convolutional layers time, we simulate them with a versatile benchmarking tool DNNMark[4].
- Compare changes in layer times with changes of convolution algorithms.

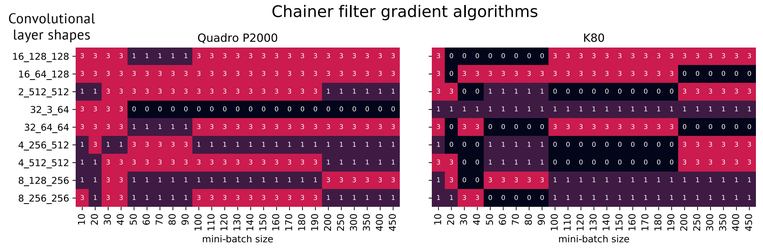
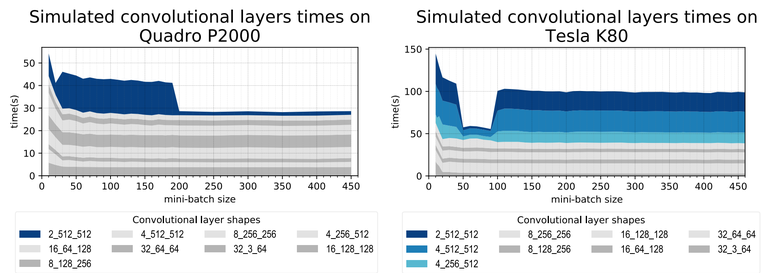



**Fig.2.** Convolution algorithms used by Chainer for VGG16 convolutional layers on machines with Quadro P2000 and Tesla K80 GPUs. Y-axis is labeled by convolutional layer shapes. We can see how algorithms chosen by cuDNN heuristics change depending on the mini-batch size.
**Colours and numbers** denote different algorithms chosen by cuDNN.



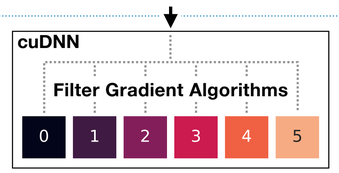

* The results on Tesla M60 and Tesla V100 are omitted here for lack of space.
No abrupt changes in convolutional layers times were observed on Tesla M60 and Tesla V100.

**Fig.3.** The execution time of convolutional layers in VGG16 model per one epoch. Layer 2_512_512 on Quadro P2000 machine and the top 3 layers on K80 have abrupt changes in their execution time.

### Observations

- The shapes of aggregated time of convolutional layers are very similar to the epoch times. There are abrupt changes in convolutional layer times.
- The abrupt changes in the time correlate with the changes of algorithms used for filter gradients calculations.
  - The change of the algorithm from 3 to 1 causes a significant increase in execution time of the 2_512_512 layer on Quadro P2000, and the change of the algorithm from 0 to 1 causes decrease in execution time of 3 layers on K80 (Fig.3).

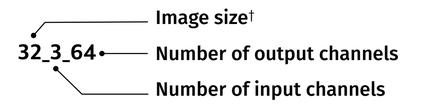
#### ☞ *Convolution algorithms*




For each convolution operation, cuDNN[3] provides several algorithms. There are 6 algorithms for calculating error gradients with respect to filters. cuDNN heuristics choose different algorithms depending on the convolutional layer shape and the mini-batch size.

#### ☞ *Convolutional layer shapes*

A convolutional layer has many parameters, such as dimensions of input and output tensors, the shape of convolution filters and padding. However, for the VGG16 model, there are only three parameters that change between convolutional layers: images size and the number of channels in input and output images.




† For each convolutional layer, input image width and height, output image width and height are all equal and denoted as image size.

## Summary

- We found that minor changes in the mini-batch size may cause abrupt changes in the CNN training epoch time.
- The cause of the abrupt changes is that underlying cuDNN library heuristics chooses different convolution algorithms depending on the mini-batch size.

## Future Work

- Extend our experiments:
  - Consider other machine learning frameworks.
  - Consider cuDNN auto-tuning — algorithm selection policy based on performance benchmarking.
  - Consider other CNN models and datasets.
- Design a performance model for predicting CNN training epoch time on unseen mini-batch sizes and unseen GPU types.
- Consider estimating training time to reach a target accuracy level.
  - There are two main CNN training scenarios: CNN is either trained for a fixed number of epochs or to reach a particular level of prediction accuracy.
  - In this work, we consider the training scenario when the number of epochs in CNN training does not change.

## References

1. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. In: International Conference on Learning Representations (2015)
2. Tokui,S.,Oono,K.,Hido,S.,ClaytonJ.:Chainer: a Next-Generation Open Source Framework for Deep Learning. In: Proceedings of Workshop on Machine Learning Systems in The Twenty-ninth Annual Conference on Neural Information Process- ing Systems (2015)
3. NVIDIA CUDA® Deep Neural Network library (cuDNN). https://developer.nvidia.com/cuDNN (2019)
4. Dong, S., Kaeli, D.: DNNMark: A Deep Neural Network Benchmark Suite for GPUs. In: Proceedings of the General Purpose GPUs. pp. 63–72 (2017)