SONY

ABCIを活用した大規模分散学習への取り組み

Yoshiki TANAKA

第41回AIセミナー「ABCIグランドチャレンジ2019成果報告会」

2020.02.21

Sony Corporation R&D センター

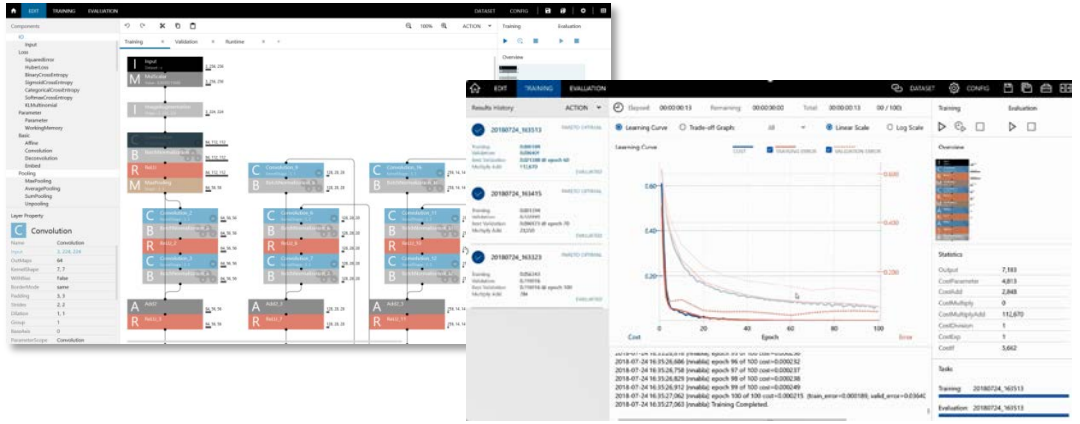
Sony's Product

Deep Learning is being utilized in many application domains.

Sony's deep learning software

Neural Network Console

dl.sony.com



GUI based deep learning IDE

Windows Desktop (free) & Cloud (paid)

Neural Network Libraries

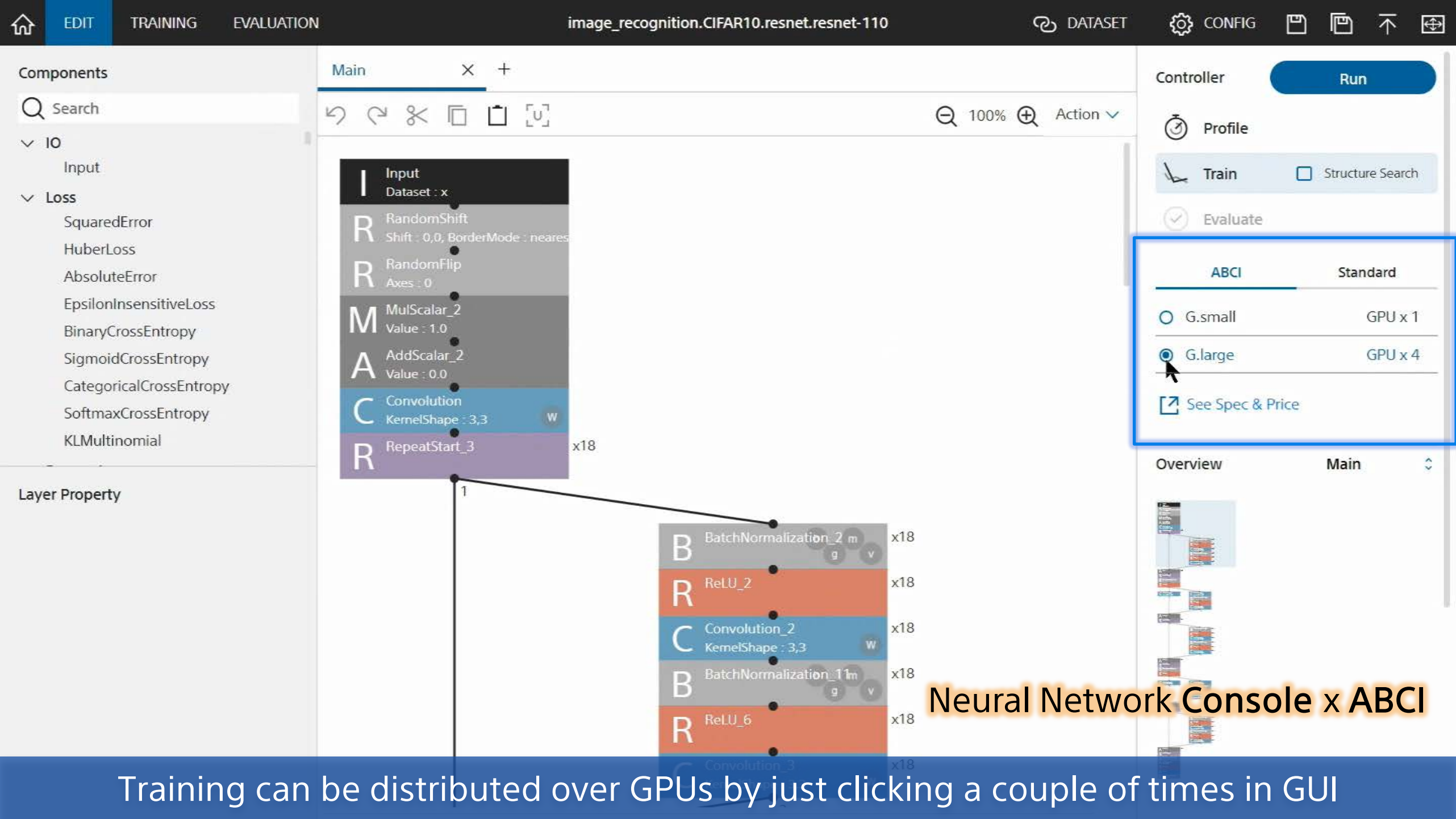
nnabla.org


```
x = nnabla.Variable((batch_size, 1, 28, 28))
c1 = PF.convolution(x, 16, (5, 5), name='c1')
c1 = F.relu(F.max_pooling(c1, (2, 2)))
c2 = PF.convolution(c1, 16, (5, 5), name='c2')
c2 = F.relu(F.max_pooling(c2, (2, 2)))
f3 = F.relu(PF.affine(c2, 50, name='f3'))
y = PF.affine(f3, 50, name='f4')
```

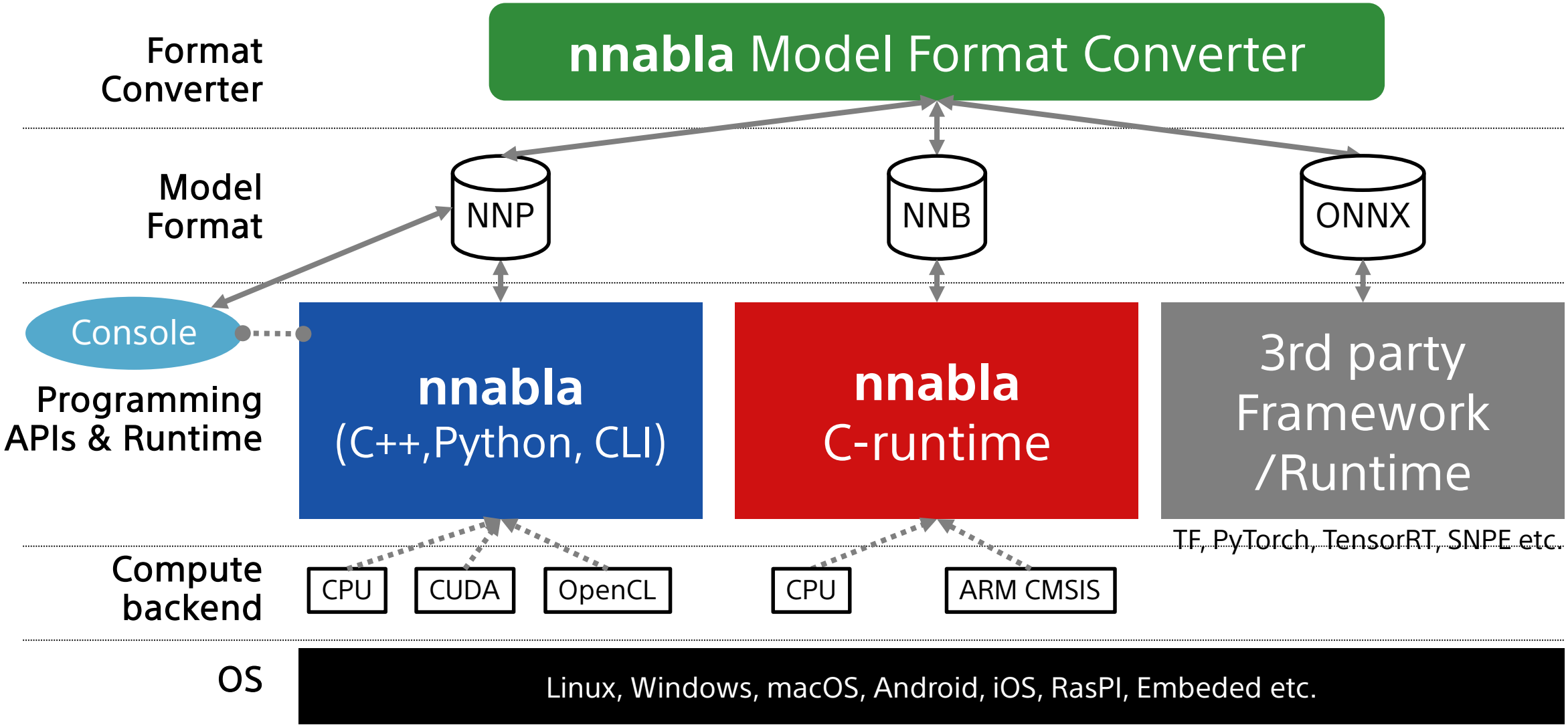
Deep learning framework with Python API

Open source (Apache 2.0 license)

Intuitive, fast and easy to deploy


EDIT
TRAINING
EVALUATION
image_recognition.CIFAR10.resnet.resnet-110
DATASET
CONFIG
Components
Search
IO
Input
Loss
SquaredError
HuberLoss
AbsoluteError
EpsilonInsensitiveLoss
BinaryCrossEntropy
SigmoidCrossEntropy
CategoricalCrossEntropy
SoftmaxCrossEntropy
KLMultinomial
Layer Property
Main
100%
Action
Input
Dataset : x
RandomShift
Shift : 0,0, BorderMode : neares
RandomFlip
Axes : 0
MulScalar_2
Value : 1.0
AddScalar_2
Value : 0.0
Convolution
KernelShape : 3,3
RepeatStart_3
x18
1
BatchNormalization_2
ReLU_2
Convolution_2
KernelShape : 3,3
BatchNormalization_11
ReLU_6
Controller
Run
Profile
Train
Structure Search
Evaluate
ABCI
Standard
G.small
GPU x 1
G.large
GPU x 4
See Spec & Price
Overview
Main
Neural Network Console x ABCI
Training can be distributed over GPUs by just clicking a couple of times in GUI

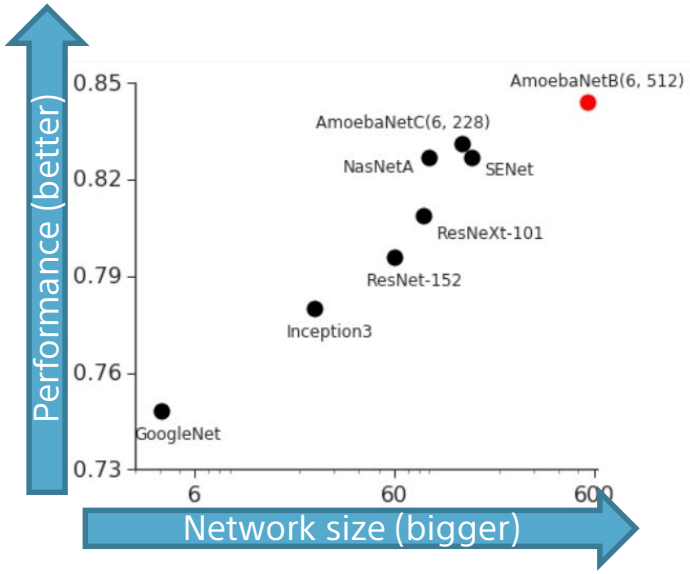
Neural Network Libraries



Why large scale training is important? 1/2

Bigger models are better in performance (accuracy)

Recognition Cite: GPipe



Biggest models win in recognition tasks

Generation Cite: BigGAN

Class conditional image generator

Image generation models has also become bigger (requires 512 cores TPUv3 pod to train)

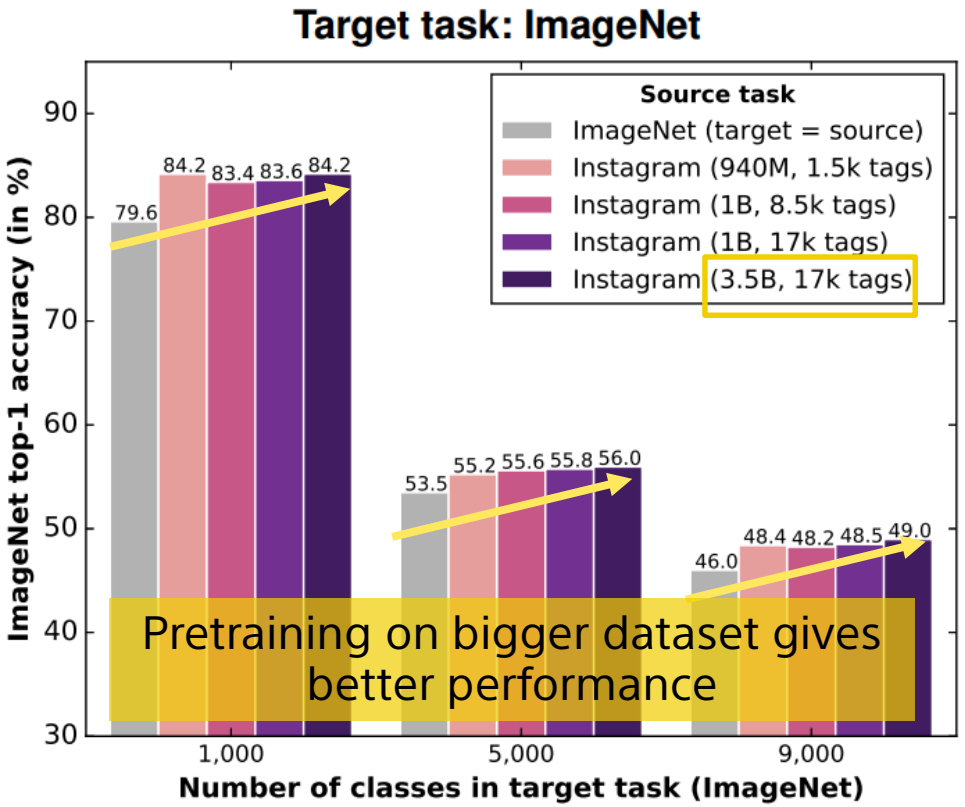
Batch	Ch.	Param (M)	Shared	Skip-z	Ortho.	Itr $\times 10^3$	FID	IS
256	64	81.5	SA-GAN Baseline			1000	18.65	52.52
512	64		✗	✗	✗	1000	15.	(±1.18)
1024	64		✗	✗	✗	1000	14	(±1.42)
2048	64		✗	✗	✗	732	12	(±3.83)
2048	96	173.5	✗	✗	✗	295(±18)	9.54(±0.62)	92.98(±4.27)
2048	96	160.6	✓	✗	✗	185(±11)	9.18(±0.13)	94.94(±1.32)
2048	96	158.3	✓	✓	✗	152(±7)	8.73(±0.45)	98.76(±2.84)
2048	96	158.3	✓	✓	✓	165(±13)	8.51(±0.32)	99.31(±2.10)
2048	64	71.3	✓	✓	✓	371(±7)	10.48(±0.10)	86.90(±0.61)

Bigger

Better

Why large scale training is important? 2/2

Large dataset gives better performance



https://research.fb.com/publications/exploring-the-limits-of-weakly-supervised-pretraining/

Data size becomes bigger

https://twitter.com/goodfellow_ian/status/1084973596236144640?s=20

Bigger model, dataset, and data size impose large memory and training time

ソニーにおける分散DNN学習の目的

Deep Learning is being utilized in many application domains.

ソニーにおける分散DNN学習の目的

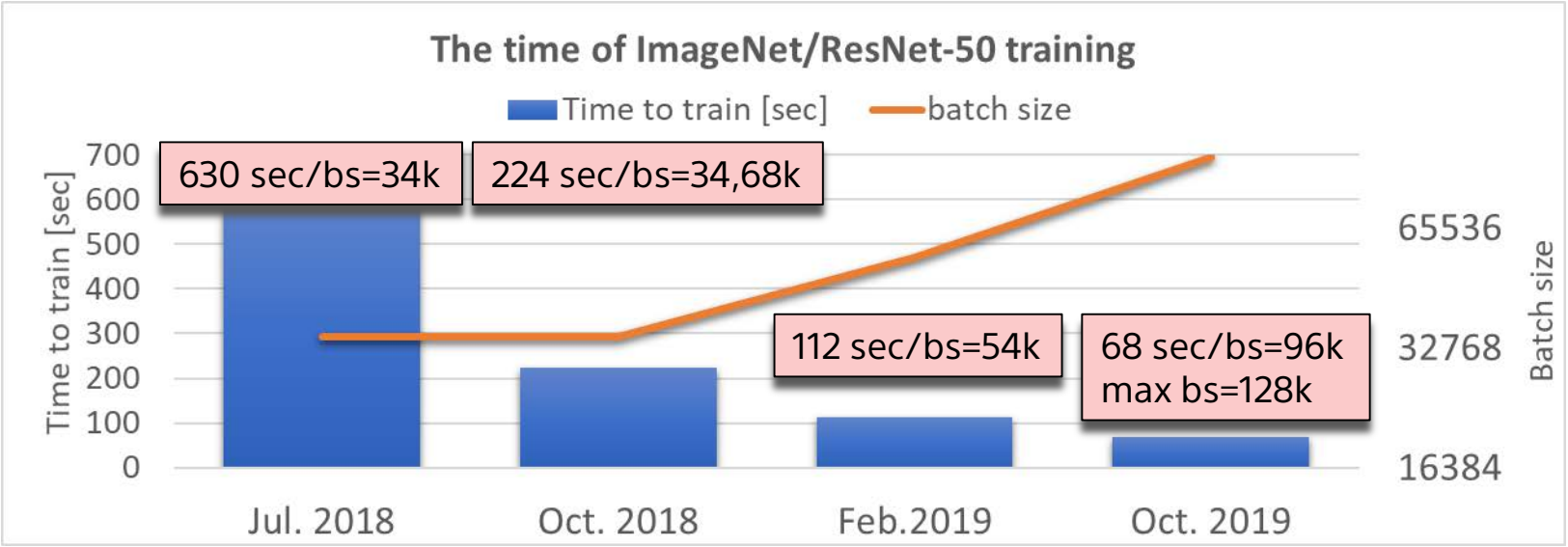
DNN学習時間の短TAT化による製品開発の加速化

- H/W の進化
- 学習アルゴの進化
- **分散学習**
  - Large Scale：より多くのGPUの利用
  - Large Batch：多種多様なタスクに対して Large Batch で学習可能にすること

Deep Learning is being utilized in many applications and services

ABCI Grand ChallengeでImageNet/ResNet-50による分散学習の挑戦を通じて知見獲得

ImageNet/ResNet-50 with ABCI

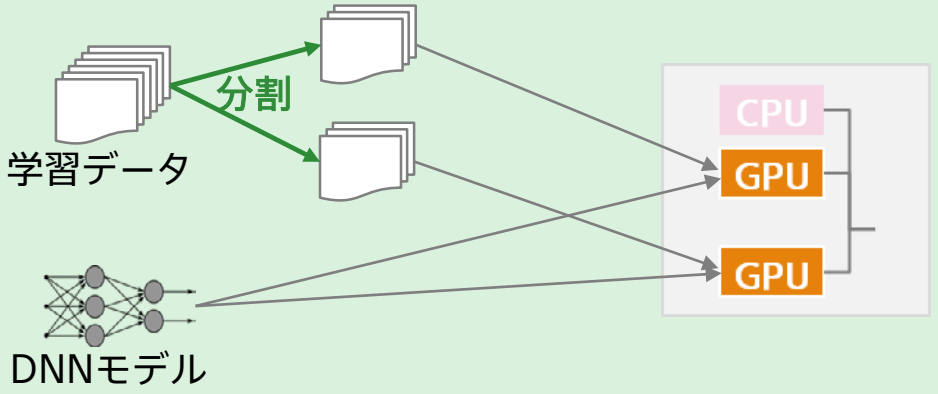


- We are studying distributed training to train DNN models in shorter time.

- And to benchmark our framework, we have published the training results of ImageNet/ResNet-50 that contain the time of training, the final accuracy and techniques for large batch training.

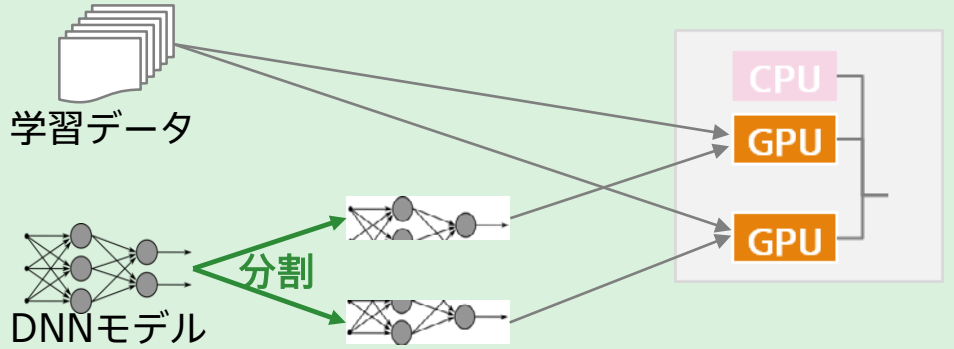
分散学習：データパラレルとモデルパラレル

データパラレル



入力となる学習データを分割させて学習する方式

モデルパラレル



学習モデル自体を分割させて学習する方式

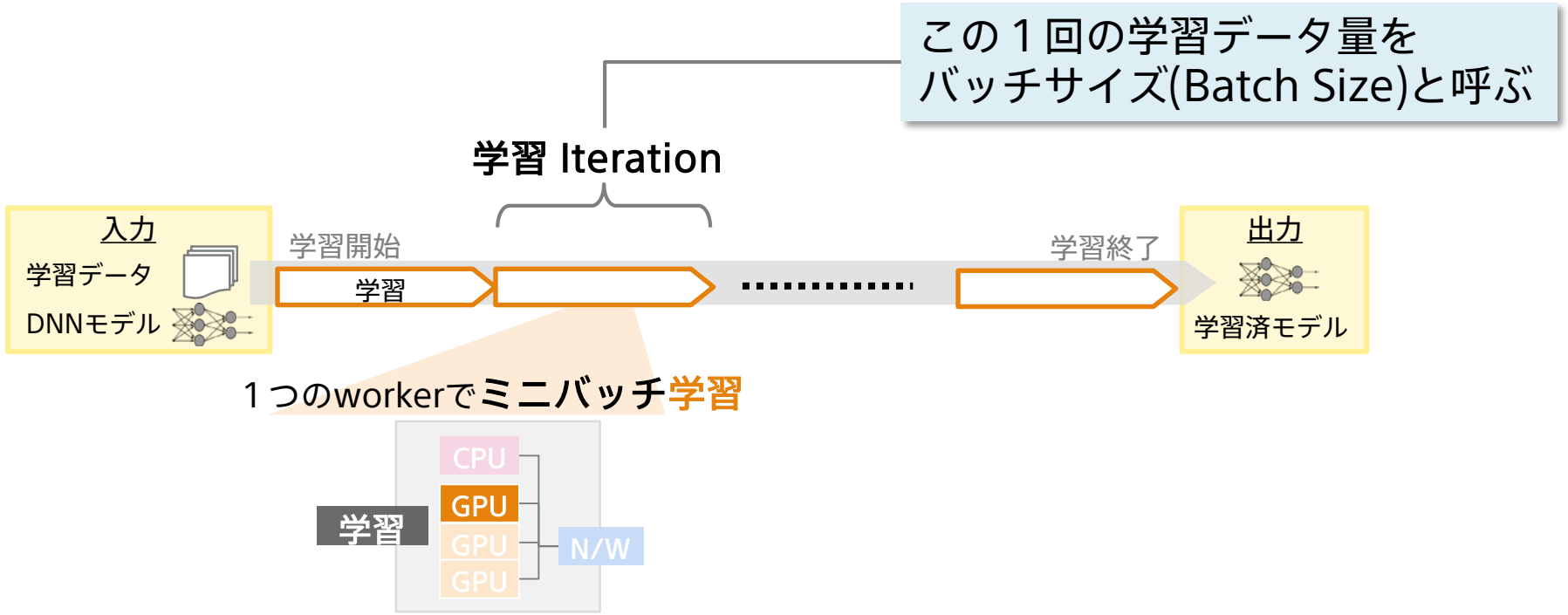
	データパラレル	モデルパラレル
Pros	負荷を均等に分割することが容易で処理のばらつきが少なく、シンプルなシステム構成で実現できる。	各 worker で必要とするメモリを少なく抑えることができる。バッチサイズ増加を抑制する効果も。
Cons	バッチサイズ増加による収束速度が劣化する。	各 worker の処理効率を高めるのが難しい。

DNN Training on single GPU.

ミニバッチ学習

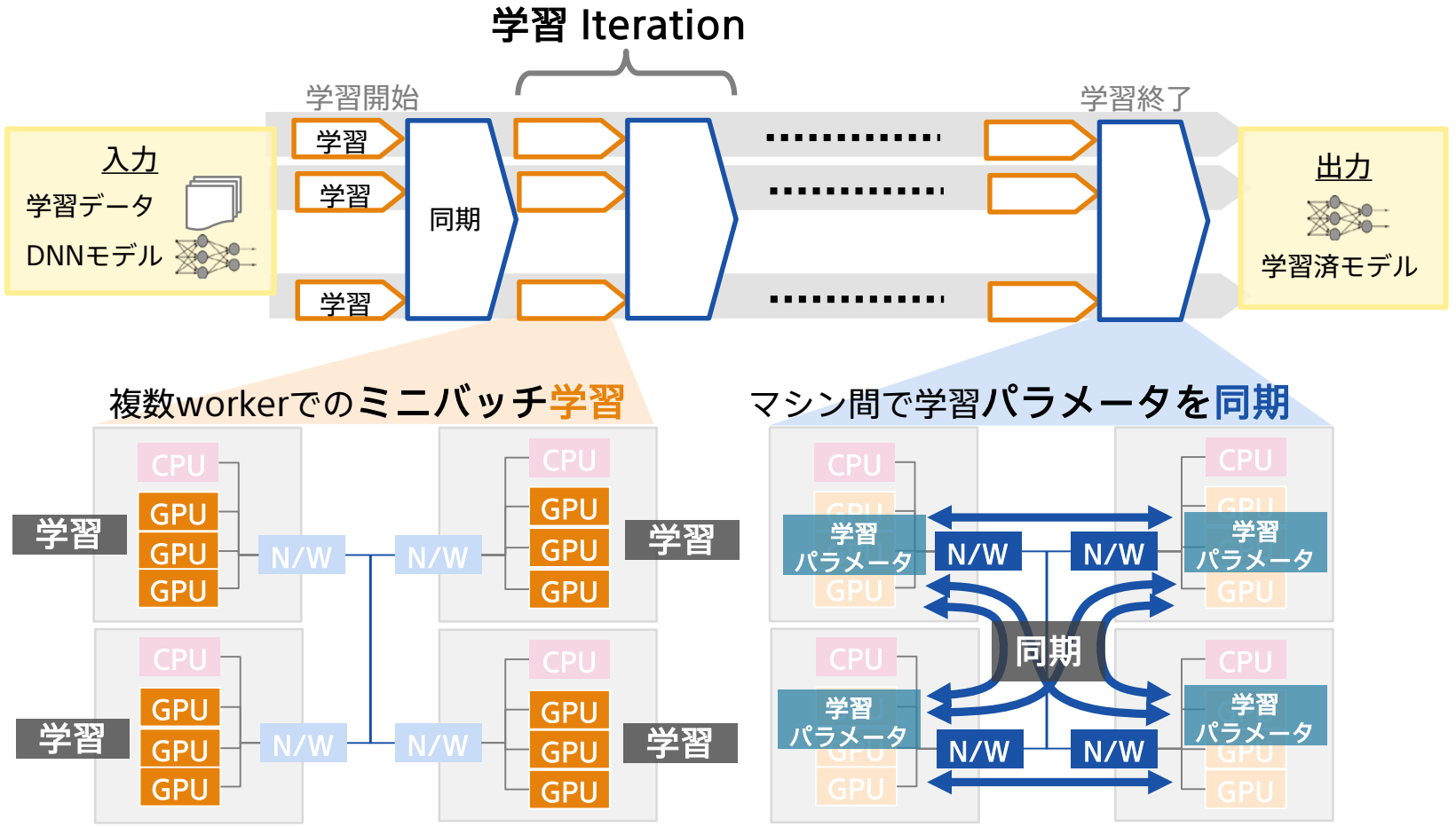
学習データセットを適度なサイズの “ミニバッチ” に分割し

学習を繰り返し(学習 Iteration) ながらパラメータ(重み)を更新していく



Data-parallel distributed DNN Training

複数workerでミニバッチ学習：**学習**後にパラメータ**同期**が必要



Problems of distributed large-batch training

1. To achieve convergence with very small number of update steps
   - The larger the mini-batch size is, the more number of GPUs we can use.
   - However, larger mini-batch size also makes it more difficult for convergence.

   Mini-batch size will be N-times size, if using N GPUs.
   The point is developing the way to train even with large mini-batch size.

2. To reduce gradient synchronization time
   - Ring topology becomes too slow, if using more than hundreds of GPUs .

   The point is developing a topology (for collective communication) which allow for an efficient synchronization even with the thousands of GPUs.

Large Batch 学習で精度劣化する要因

重みの更新回数が減少し
最適解に近づくのが難しくなる

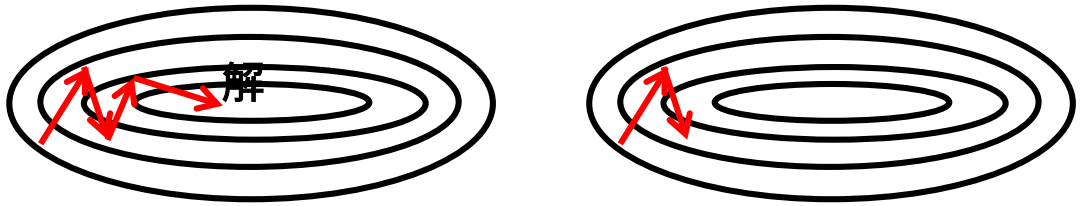
1 worker

$$w^{t+1} = w^t - \frac{\eta}{|B_1|} \sum_{x_i \in B_1} \nabla l(x_i, w^t)$$

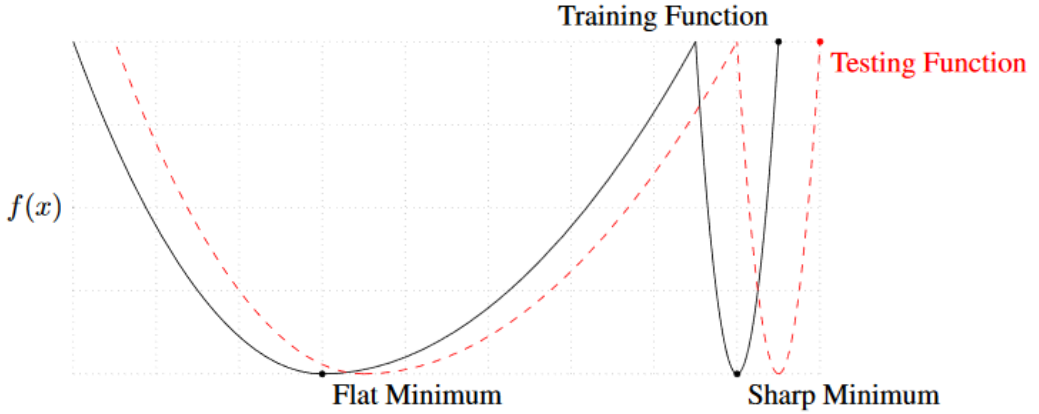
2 workers

$$w^{t+1} = w^t - \frac{\eta}{2|B_1|} \sum_{x_i \in B_2} \nabla l(x_i, w^t)$$

w ：重み
$|B_1|$：バッチサイズ
η ：学習率($Learning\ rate$)

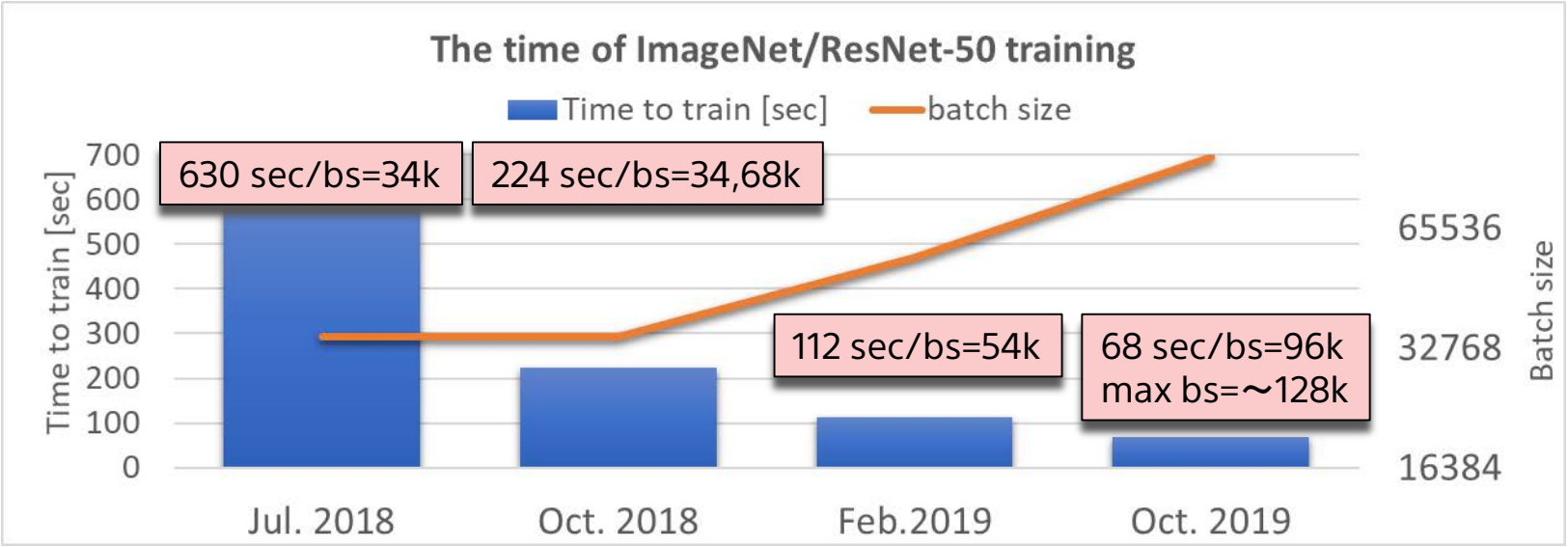


データが持つノイズが薄まるため
Sharp Minima に陥りやすくなる



※ N. S. Keskar et al. “On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima” ICLR 2017

ImageNet/ResNet-50 with ABCI



- We are studying distributed training to train DNN models in shorter time.
- And to benchmark, we have published the results of ImageNet/ResNet-50 training time of training software and techniques for large batch training.

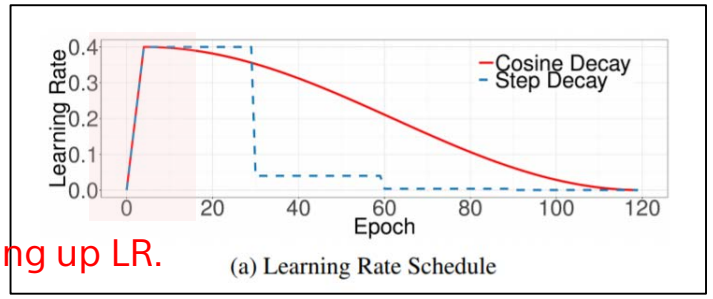
Batch-size Control

STiLL optimizer

Overview of common techniques for large batch training

Learning rate warmup*

- Linearly scaling up LR in several epochs.



(a) Learning Rate Schedule

Warming up LR.

LARS optimizer**

- Layer-wise Adaptive LR Scaling with the ratio of weights and grads.

$$\lambda^l = \eta \times \frac{||w^l||}{||\nabla L(w^l)|| + \beta * ||w^l||}$$

Label Smoothing***

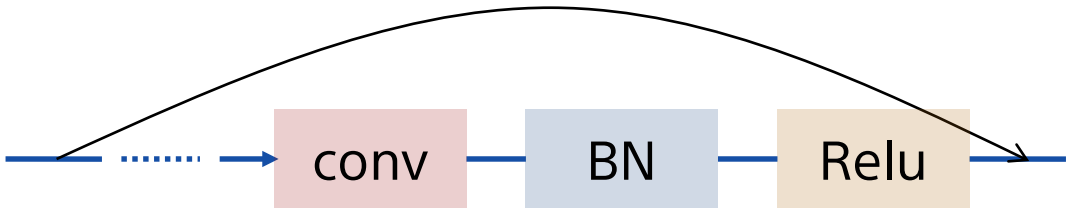
- Smoothing i-th label of Softmax output **q**
  (0 < ε < 1, K is the number of classes.)

$$q_i = \begin{cases} 1-\varepsilon & \text{if } i = y, \\ \varepsilon/(K-1) & \text{otherwise,} \end{cases} \quad (4)$$

*** C. Szegedy et al Rethinking the Inception Architecture for Computer Vision (arxiv.org/abs/1512.00567)

Zero-γ-init**

- Initilizing γ of batchnorm in residual path to 0.



↑ Init γ = 0 here.

* Y. you et al. Large Batch Training of Convolutional Networks (arxiv.org/abs/1708.03888)
** P.Goyal et al. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour (arxiv.org/abs/1706.02677)

Up to 64K mini-batch these are enough to train.
However we need another technique to increase batch-size more and more !

STiLL : Smoothly Transition from LAMB* to LARS.

- We proposed the new optimizer “STiLL” that uses two optimizer, LAMB and LARS at the same time.

- STiLL starts from LAMB:LARS=100:0 to update the weights, then it linearly changes the ratio of them during warmup duration.

- After the warmup duration, it uses only LARS to update weights.

$$w(t+1) = w(t) - f(w,g,t)$$

$$f(w,g,t) = \begin{cases} \theta_{lars}(t) * lr_{lars}(t) * LARS(w,g,t) + \theta_{lamb}(t) * lr_{lamb} * LAMB(w,g,t), & t < T_{warmup} \\ lr_{lars}(t) * LARS(w,g,t), & t \geq T_{warmup} \end{cases}$$

$$\theta_{lars}(t) = \left\{ \frac{lr_{lars}(0)}{lr_{lars}(T_{warmup})} + \frac{t}{T_{warmup}} \left(1 - \frac{lr_{lars}(0)}{lr_{lars}(T_{warmup})} \right) \middle| \theta_{lars}(t) + \theta_{lamb}(t) = 1, t < T_{warmup} \right\}$$

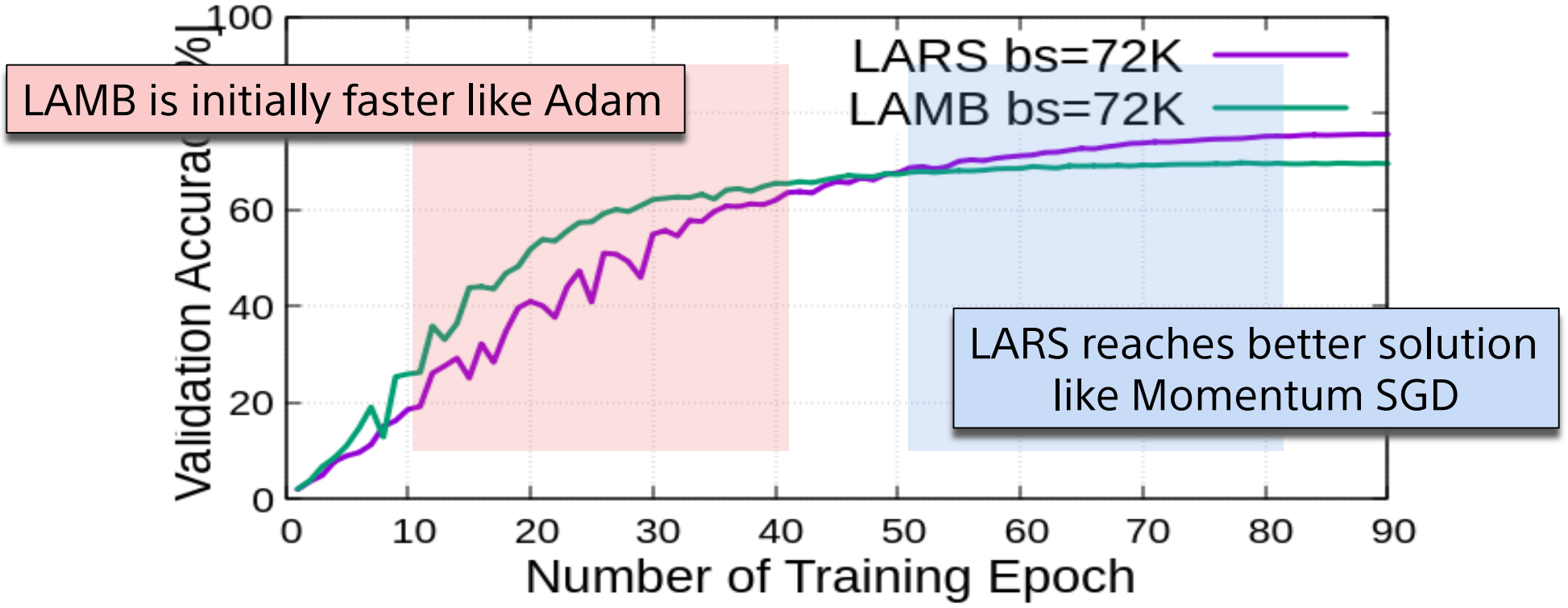
$$\begin{cases} f(w,g,t) \equiv update\ function \\ w \equiv weight \\ g \equiv gradient \\ t \equiv current\ step \\ lr \equiv learning\ rate \\ T_{warmup} \equiv duration\ of\ lr\ warmup \\ \theta_{lars}(t) \equiv ratio\ of\ LARS\ update \\ \theta_{lamb}(t) \equiv ratio\ of\ LAMB\ update \end{cases}$$

*Y.You et al, Large Batch Optimization for Deep Learning: Training BERT in 76 minutes https://arxiv.org/abs/1904.00962

Why we mixed them?

- We expected that LAMB accelerates convergence in the beginning of training and LARS helps to get better generalization performance by combining them.

↓ batch training with LARS and LAMB (these cases did not converge.)

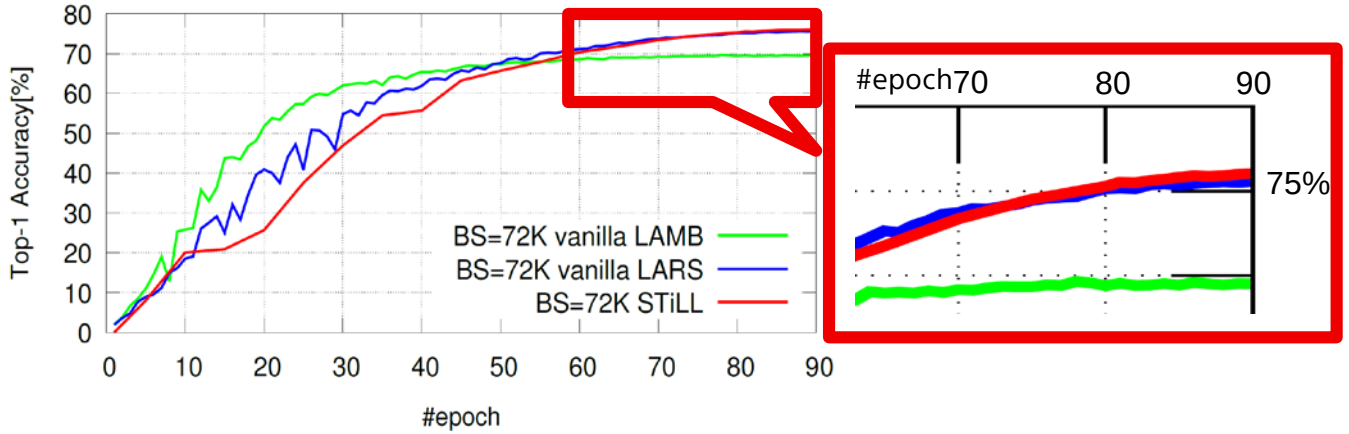


Experiments(1): Performance comparison with LARS, LAMB and STiLL

- Firstly, we conducted ImageNet/ResNet-50 training while changing optimizers, “LAMB”, “LARS” and “STiLL”, and increasing batch-size from 72K to 96K.

- STiLL outperforms “LAMB” and “LARS” in the final validation accuracy.

Training curve of LAMB, LARS and STiLL



Top-1 accuracy of ImageNet/ResNet-50 model after training for 90 epochs

Mini-Batch Size	LAMB	LARS	STiLL
72K	70.9%	75.5%	**76.1%**
80K	N/A	75.1%	**75.9%**
96K	N/A	N/A	**75.5%**

Experiments(2): Training longer epochs could give better performance

- Then we made extra experiments to see if the model get better generalization performance by training longer epochs.
- In the result, the model could reach good solution even over 100K batch training.

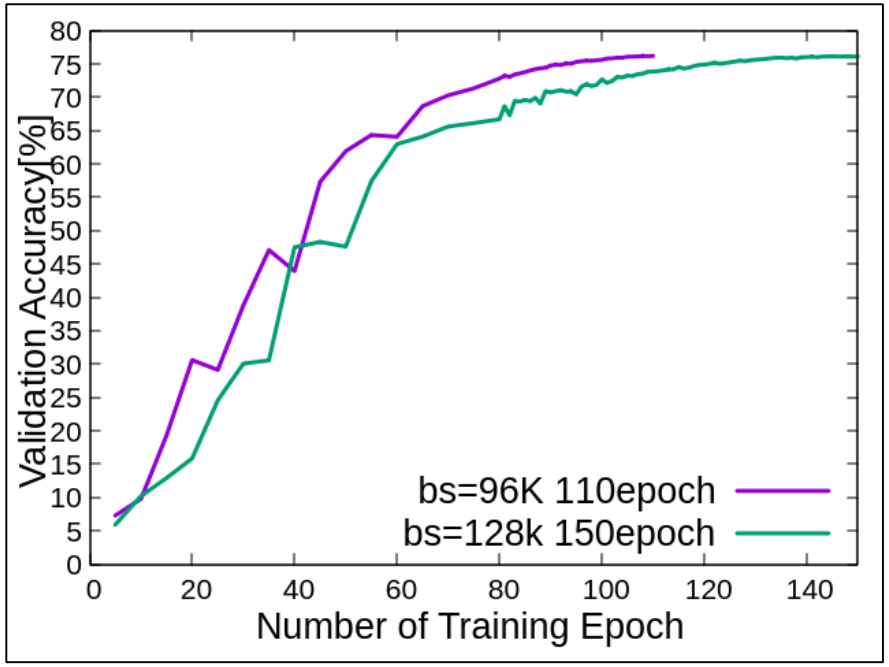
Settings

- Use the same settings as the previous exp.

Results

BS	epoch	#GPU	Acc
96K	110	3072	76.19%
128K	150	4096	76.12 %

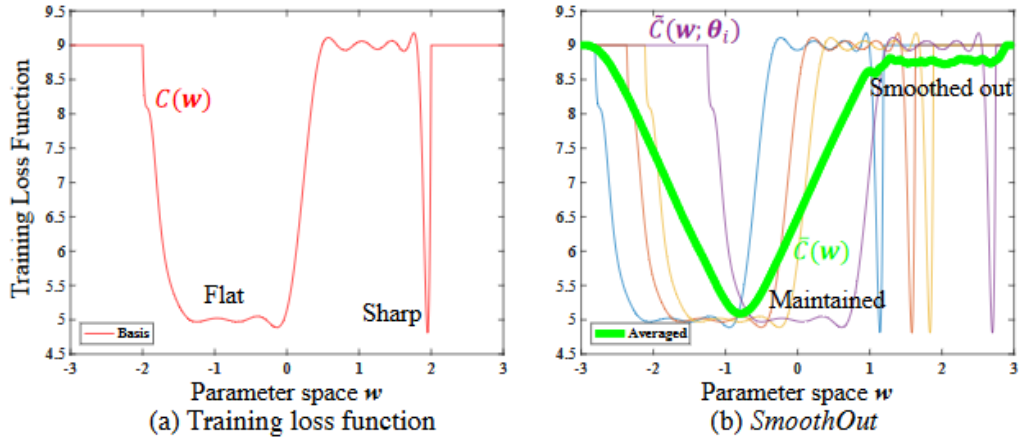
Training curve



Other Technique – Smoothout, that strengthens regularization.

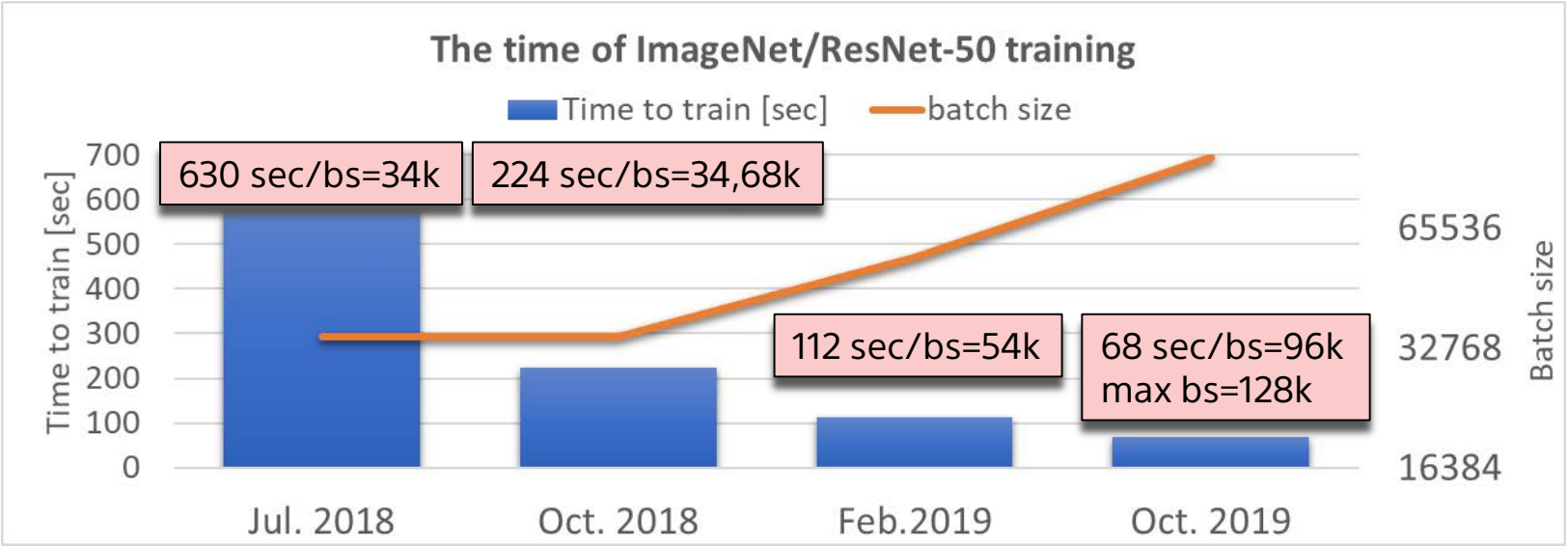
- Smoothout* is one of the regularization technique proposed by W.Wen.
  1. Add noise from Uniform dist. to the weights before forward computation.
  2. Do forward and backward computation.
  3. Sub. noise from the weights.
  4. Update weights.
- “Smoothout” can avoid sharp-minima, that is more important in large-batch training.

  We use this over 64K batch training.



*W.Wen et al. SmoothOut: Smoothing Out Sharp Minima to Improve Generalization in Deep Learning https://arxiv.org/abs/1805.07898

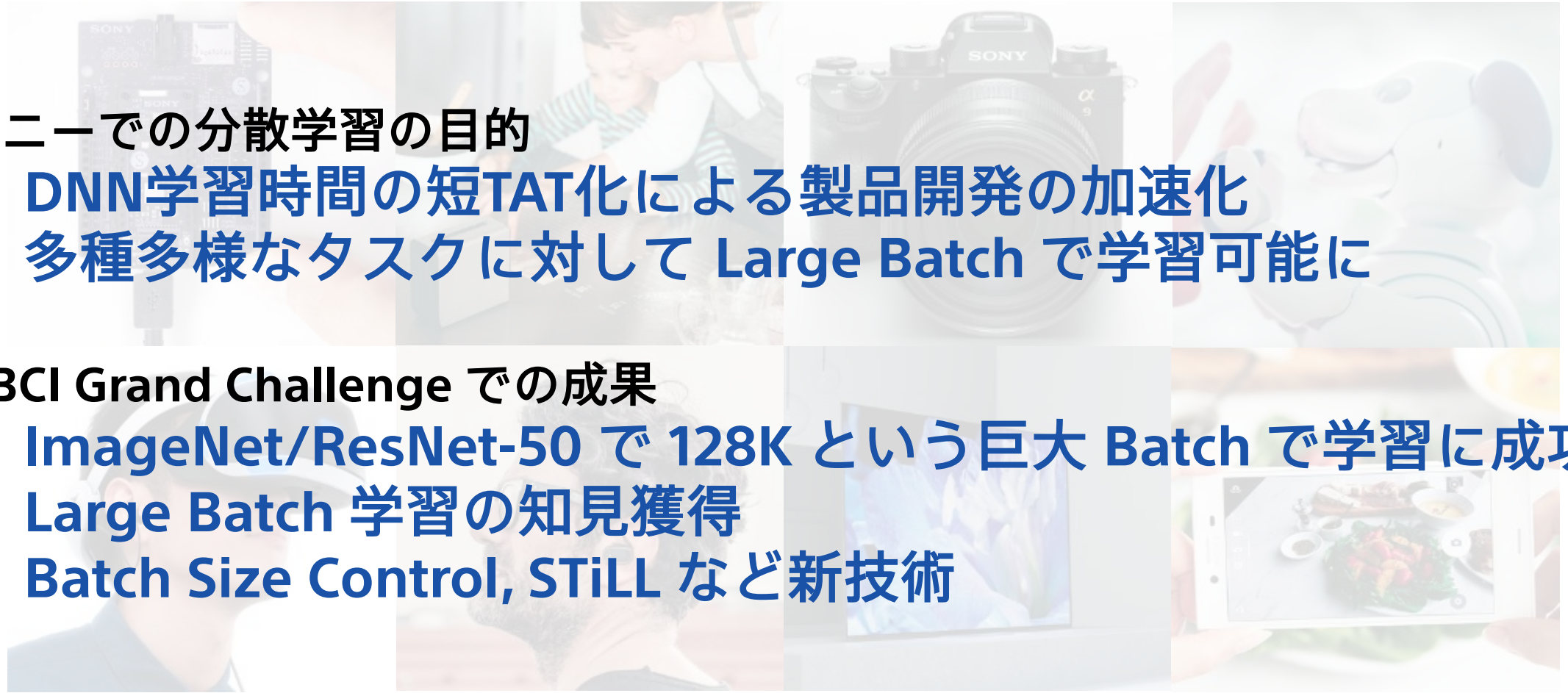
ImageNet/ResNet-50 with ABCI



- We are studying distributed training to train DNN models in shorter time.

- And to benchmark our framework, we have published the training results of ImageNet/ResNet-50 that contain the time of training, the final accuracy and techniques for large batch training.

まとめ



ソニーでの分散学習の目的

- DNN学習時間の短TAT化による製品開発の加速化
- 多種多様なタスクに対して Large Batch で学習可能に

ABCI Grand Challenge での成果

- ImageNet/ResNet-50 で 128K という巨大 Batch で学習に成功
- Large Batch 学習の知見獲得
  Batch Size Control, STiLL など新技術

Deep Learning is being utilized in many application domains.

Question?