Metric Analysis and Performance Optimization in TensorFlow

Tong Yu, Ph.D.
AI and HPC Application R&D
yutong01@inspur.com

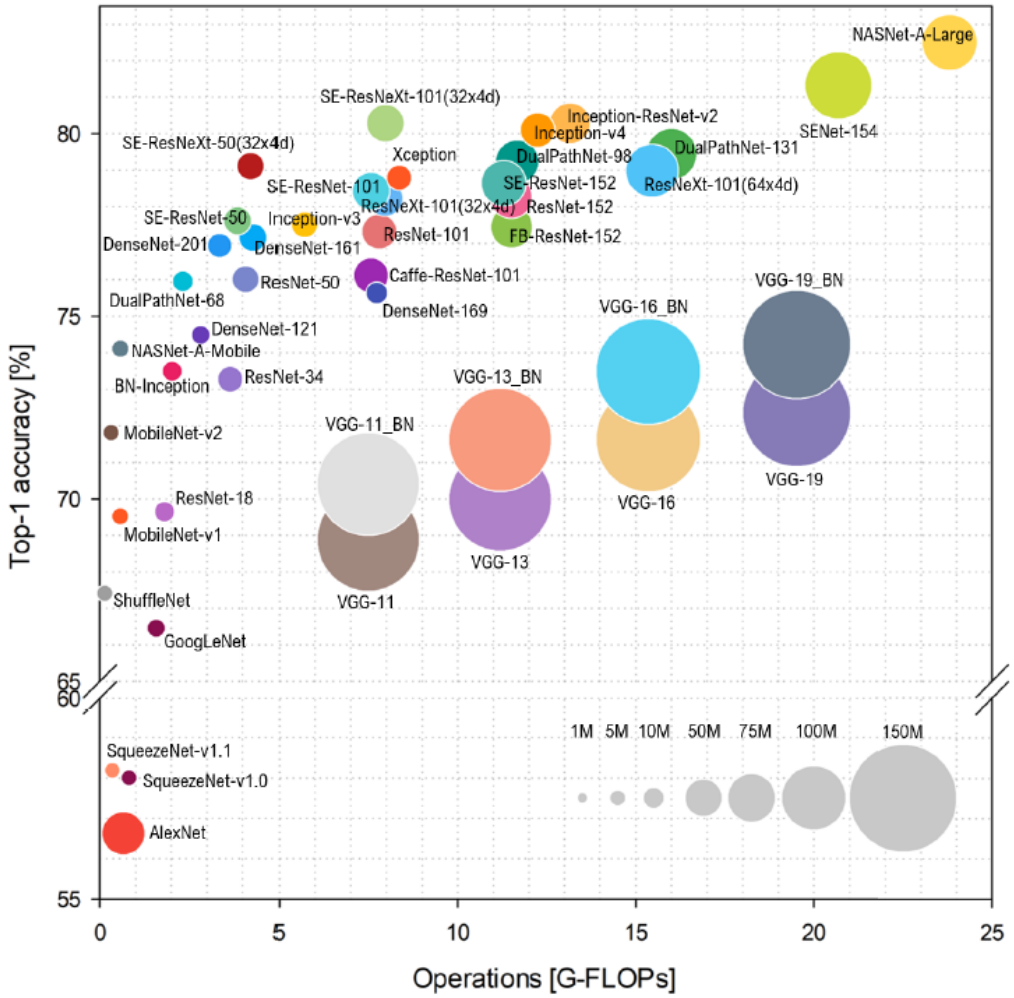
Overview

- The overall goals of this study are to provide methods to improve the performance of deep learning networks in parallel computing environment, quantitatively identify the major bottleneck in performance, and investigate the influence of our measurements on the accuracy and efficiency in deep learning frameworks.
- To address the goals, a tool for performance monitoring has been developed for bottleneck analysis. The effects of our tool is evaluated by FaceNet. Experiments show that on a single GPU, we get a performance improvement of 17%~135% and a near linear scalability on multi GPUs.

Outline

- Introduction and Background
- Part I: Teye: A Tool for Monitoring and Managing Execution Units
- Part II: Analysis of the Bottlenecks for Performance Optimization
- Part III: Performance Optimization on a Single GPU and Communication Optimization Multi-GPUs : A Case Study on FaceNet
- Conclusion

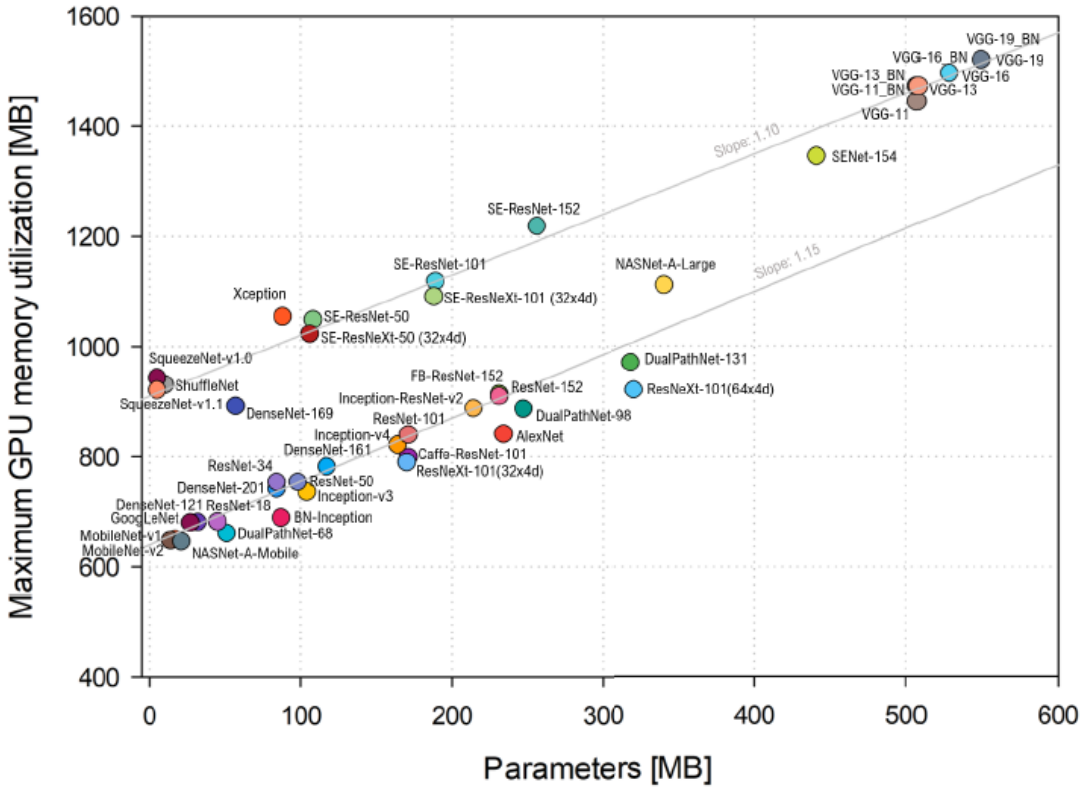
Challenge for deep learning training



- Increasing size of datasets
- Complicated structure of DNN
- Amount of parameters (millions to billions)
- Wide range of hyper-parameters

→ Time consumption

→ Energy consumption



(Figure copyright: Bianco, S., Cadene, R., Celona, L., & Napoletano, P. (2018). Benchmark Analysis of Representative Deep Neural Network Architectures. IEEE Access, 6, 64270-64277.)

inspur

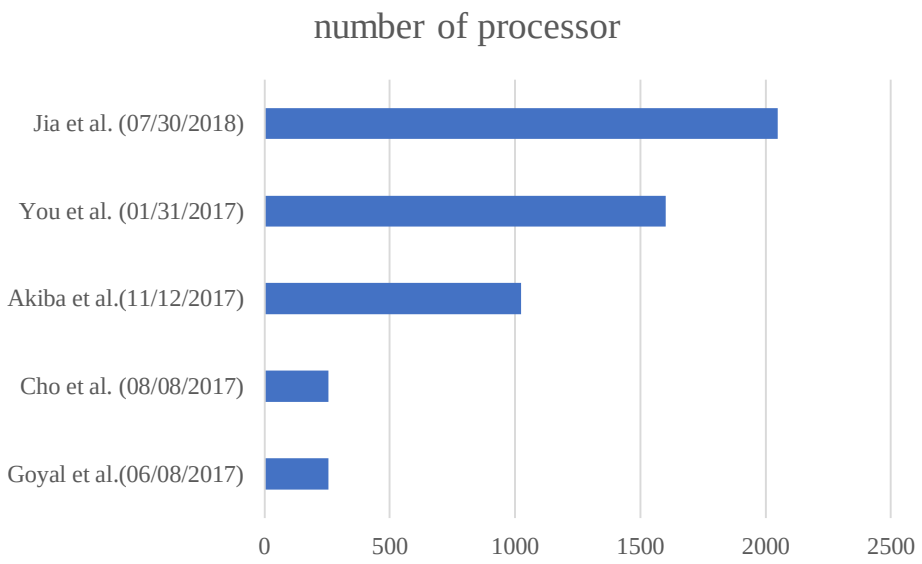
Challenge for deep learning training

Table 1 : Training time and top-1 1-crop validation accuracy with ImageNet/ResNet-50

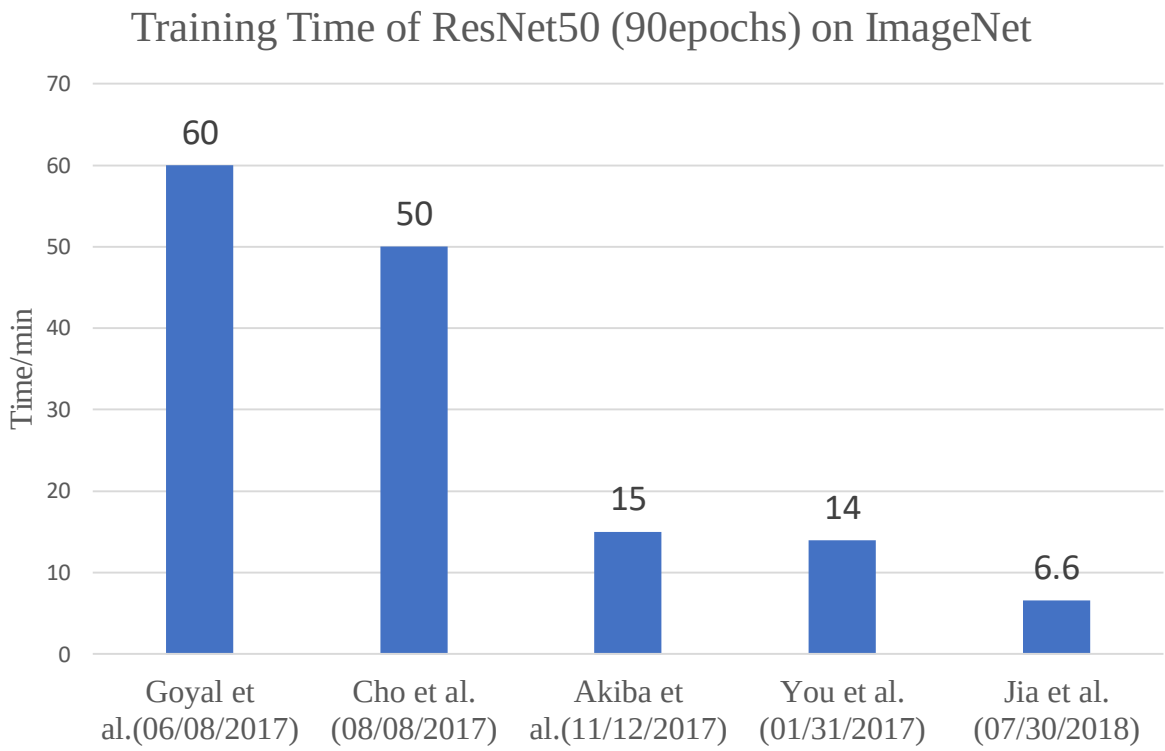
	Batch Size	Processor	DL Library	Time	Accuracy
He et al. **[7]**	256	Tesla P100 x8	Caffe	29 hours	75.3%
Goyal et al. **[1]**	8K	Tesla P100 x256	Caffe2	1 hour	76.3%
Smith et al. **[4]**	8K→16K	full TPU Pod	TensorFlow	30 mins	76.1%
Akiba et al. **[5]**	32K	Tesla P100 x1024	Chainer	15 mins	74.9%
Jia et al. **[6]**	64K	Tesla P40 x2048	TensorFlow	6.6 mins	75.8%
This work	**34K→68K**	**Tesla V100 x2176**	**NNL**	**224 secs**	**75.03%**

Table 2 : GPU scaling efficiency with ImageNet/ResNet-50 training

	Processor	Interconnect	GPU scaling efficiency
Goyal et al. [1]	Tesla P100 x256	50Gbit Ethernet	~90%
Akiba et al. [5]	Tesla P100 x1024	Infiniband FDR	80%
Jia et al. [6]	Tesla P40 x2048	100Gbit Ethernet	87.9%
This work	**Tesla V100 x1088**	**Infiniband EDR x2**	**91.62%**



(Source: Mikami, H., Suganuma, H., Tanaka, Y., & Kageyama, Y. (2018). ImageNet/ResNet-50 Training in 224 Seconds. arXiv preprint arXiv:1811.05233..)



A gap between state-of-the arts publications and common users…

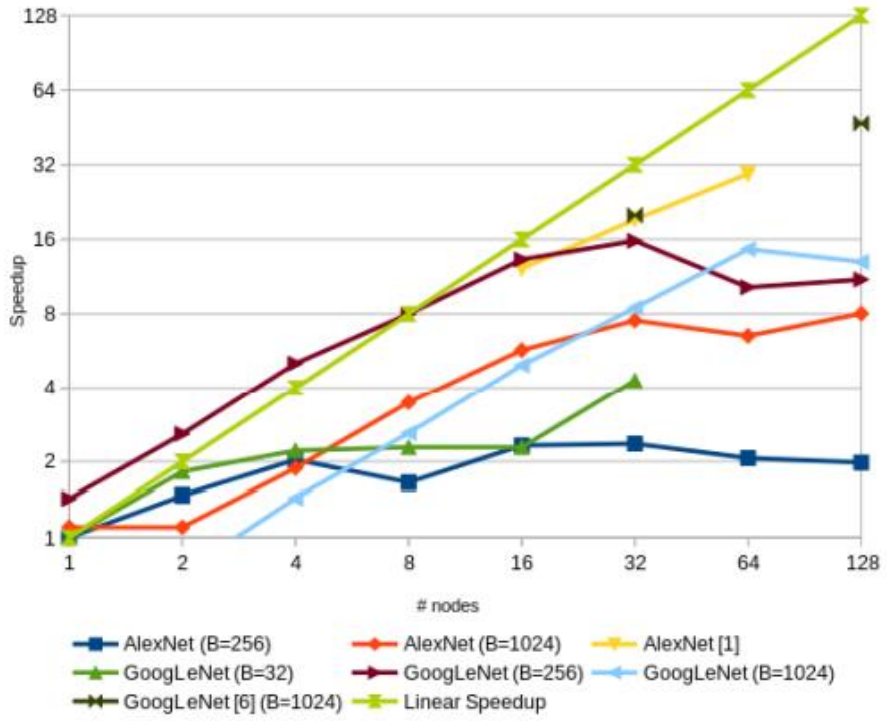
→ Most users do not have computational resources in such a scale

→ Universal methods for users with limited resources are in needed

inspur

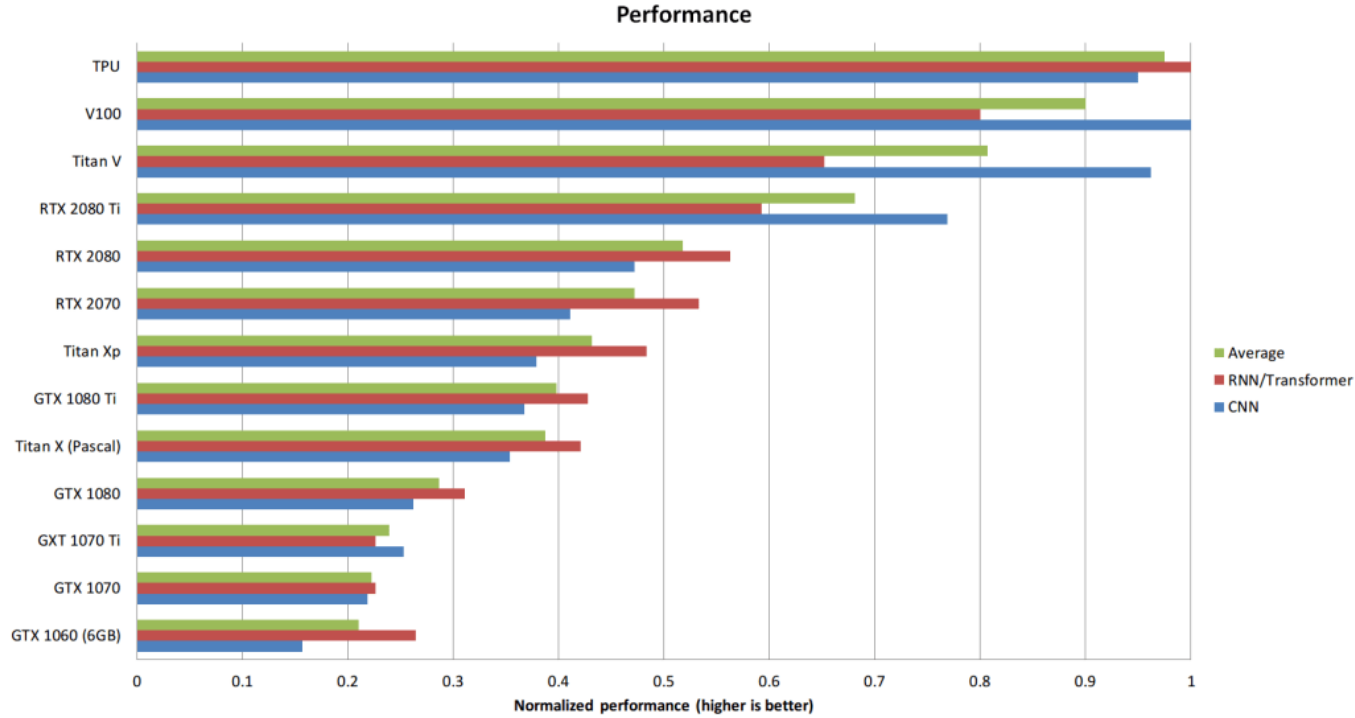
Problems for common users

Bottleneck 1：Limited Scalability



(Keuper, J., & Preundt, F. J. (2016, November). Distributed training of deep neural networks: Theoretical and practical limits of parallel scalability. In Proceedings of the Workshop on Machine Learning in High Performance Computing Environments (pp. 19-26). IEEE Press.

Bottleneck 2：Limited Hardware Utilization



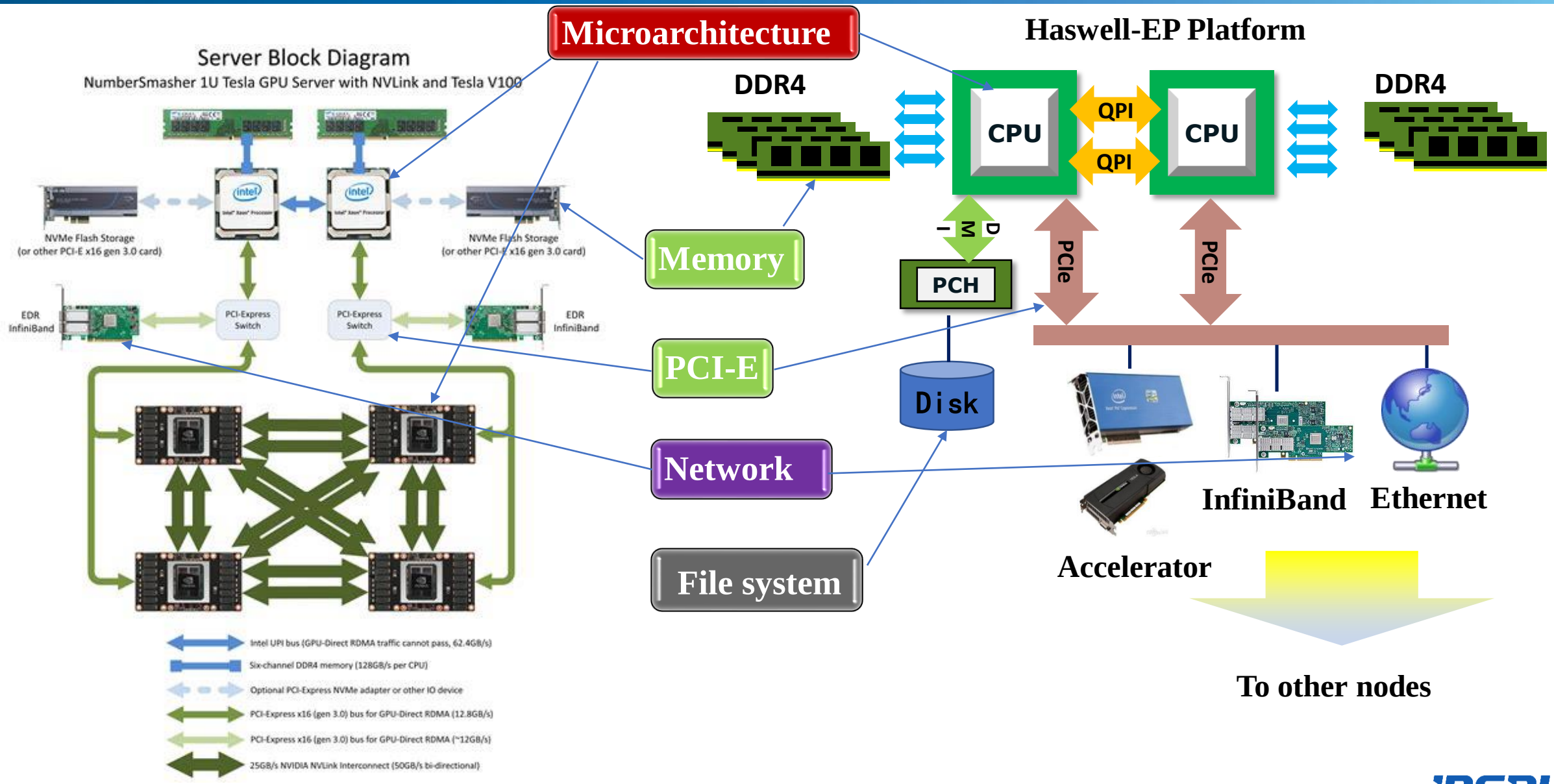
(http://timdettmers.com/2018/11/05/which-gpu-for-deep-learning/)

inspur

Outline

- Introduction and Background
- Part I: Teye: A Tool for Profiling and Performance Tuning for CPU and GPU
- Part II: Analysis of the Bottlenecks for Performance Optimization
- Part III: Performance Optimization on a Single GPU and Communication Optimization Multi-GPUs : A Case Study on FaceNet
- Conclusion

Monitor with Teye



Microarchitecture

- Utilization: usr%, sys%, idle%, iowait%
- Floating Point operations: X87 GFLOPS, SP/DP SSE scalar/packed GFLOPS, SP/DP AVX scalar/packed GFLOPS
- Vectorization Ratio: SP/DP SSE VEC, SP/DP AVX VEC
- Efficiency：CPI

Memory and PCI

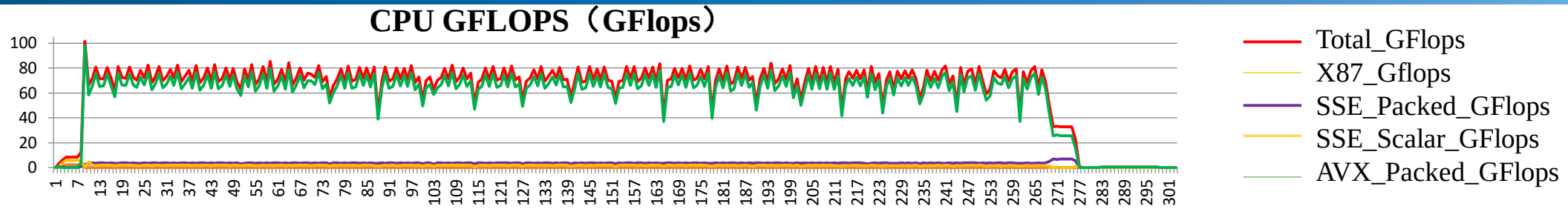
- Memory Usage：Total，used, cached, buffered
- Memory Access：Memory Read Bandwidth、Memory Write Bandwidth
- PCI-Express Access：PCI-E Read Bandwidth、PCI-E Write Bandwidth

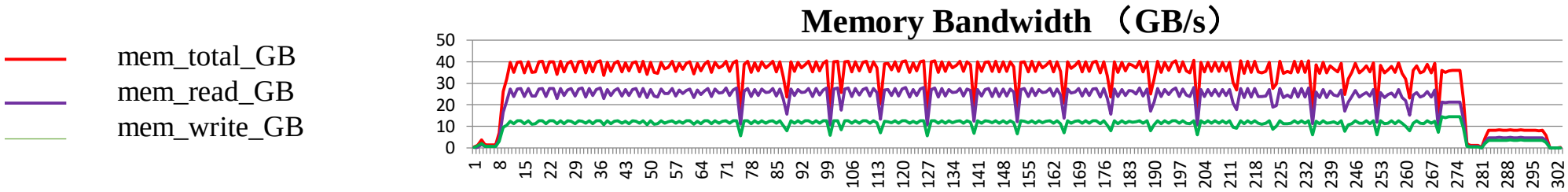
Network

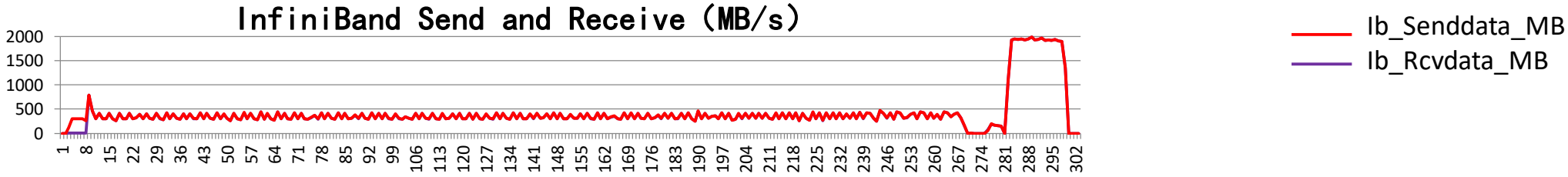
- Internet Communication Standard：Gigabit Ethernet, InfiniBand
- Protocol Support：TCP/IP, UDP, RDMA, IPoIB
- Network Traffic Monitoring：EthX_send, EthX_receive, IB_send, IB_receive
- Network Packet Monitoring：EthX/IB_send/rev_Pkt_size/data

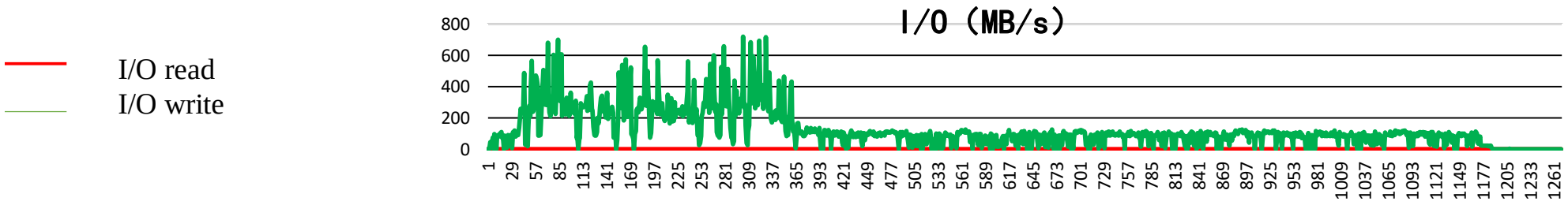
File system

- Local Disk：Local disk Read/ Write、size of data block_Read/ Write
- NFS File System：nfs_customer_read/write、nfs_server_read/write

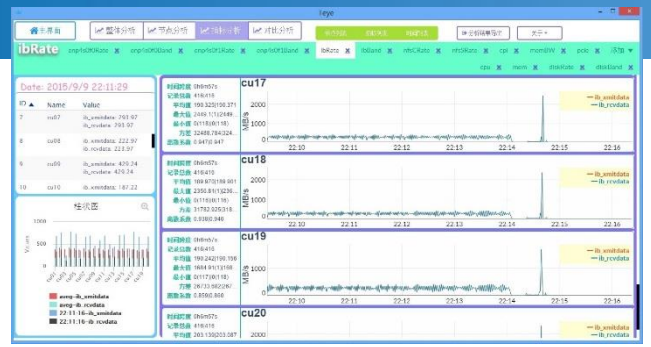

CPU GFLOPS（GFlops）
Total_GFlops
X87_Gflops
SSE_Packed_GFlops
SSE_Scalar_GFlops
AVX_Packed_GFlops


Memory Bandwidth （GB/s）
mem_total_GB
mem_read_GB
mem_write_GB


InfiniBand Send and Receive（MB/s）
Ib_Senddata_MB
Ib_Rcvdata_MB

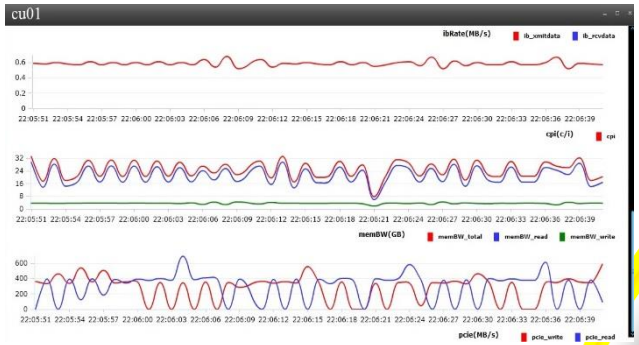

I/O（MB/s）
I/O read
I/O write

Application of Teye



Reveal performance bottleneck during training routines

Provide assistance to administrator in quick reviewing, locating and alerting problems



Bottlene ck detecti

Cluster Evaluation

天眼 TEYE

Visible, Comprehensive and Straightforward

Performance Assessment

Neural Network

Profile and analyze the operating feature and evaluate the performance of DNNs

Suggest potential points for improvement

inspur

Outline

- Introduction and Background
- Part I: Teye: A Tool for Profiling and Performance Tuning for CPU and GPU
- Part II: Analysis of the Bottlenecks for Performance Optimizations
- Part III: Performance Optimization on a Single GPU and Communication Optimization Multi-GPUs : A Case Study on FaceNet
- Conclusion

Here is a Battle for your Face…

Both recognition and detection:

Seetatech

OpenCV

dlib C++ Library

OpenFace ★ Star 11,638

Free and open source face recognition with deep neural networks.

MultiComp Lab

For Face detection or Recognition

MTCNN_face_detection_alignment

Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Neural Networks

PyramidBox NormFace *SphereFace:*

InsightFace: 2D and 3D Face Analysis Project

Deep Face Recognition with Caffe Implementation

MobileID: Face Model Compression

inspur

FaceNet (2015) By Google Inc.

Google claims its 'FaceNet' system has almost perfected recognising human faces - and is accurate 99.96% of the time

- Facebook's rival DeepFace uses technology from Israeli firm face.com
- DeepFace finds a matching face with 97.25% accuracy
- Google researchers call their system the most-accurate technology
- System could be used to automatically recognise photos on Google+

By MARK PRIGG FOR DAILYMAIL.COM
PUBLISHED: 20:46 GMT, 19 March 2015 | **UPDATED:** 07:44 GMT, 20 March 2015

Share 134 shares 8 View comments

FaceNet: A Unified Embedding for Face Recognition and Clustering

Florian Schroff
fschroff@google.com
Google Inc.

Dmitry Kalenichenko
dkalenichenko@google.com
Google Inc.

James Philbin
jphilbin@google.com
Google Inc.

Achievement:
99.63% for Labeled Faces in the Wild
95.12% on YouTube Faces DB

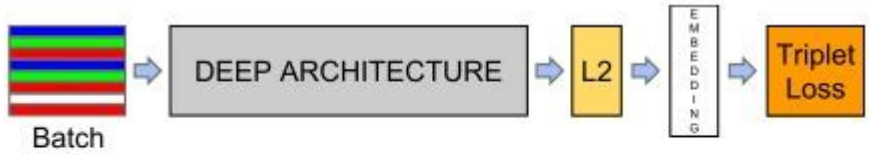


Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

Main points:
1. Face recognition and clustering
2. Purely data driven method: Learning a Euclidean with DNNs →face similarity
3. Test with both ZF net (2013) and Inception (2014)
4. Triplet loss based on LMNN and Softmax

inspur

Based on TensorFlow

	Growth Rate （%）
TensorFlow	80.0
Caffe	17.7
MXNet	21.9
CNTK	19.9

TensorFlow™

- Backed by big community
- Most commonly used deep learning framework
- Large amount of ready-to-use documentation

ebay

Coca-Cola

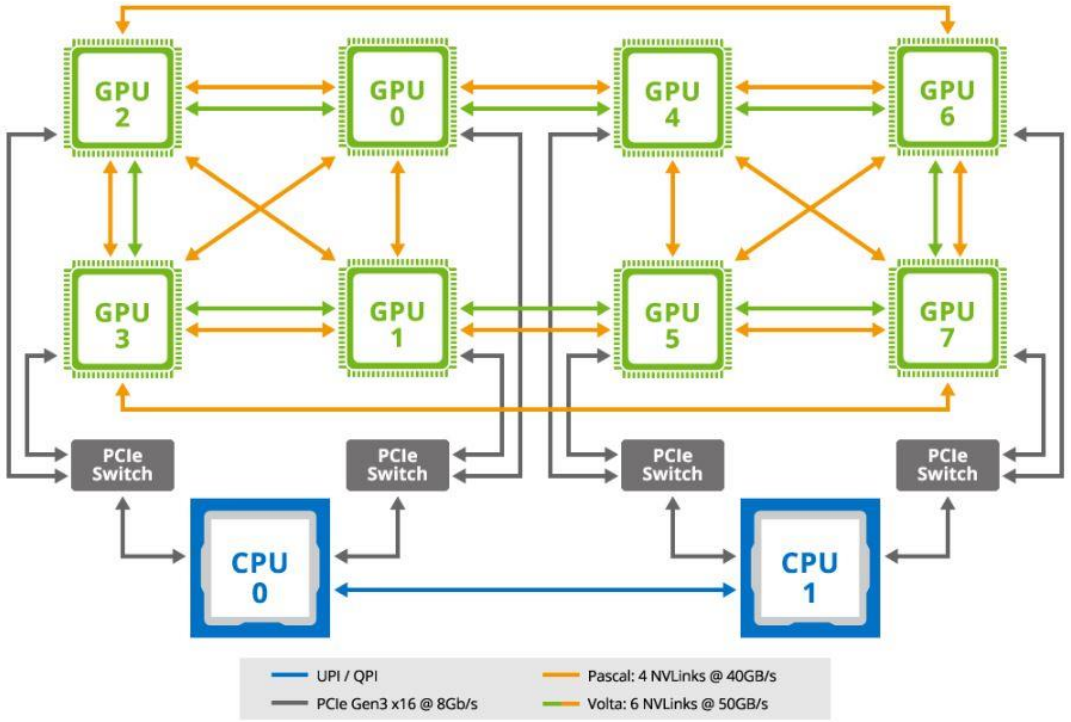
UBER

inspur

Training FaceNet : Hardware

Hardware:
DGX-1

- CPU: Intel® Xeon® CPU E5-2698 v4 @ 2.20 GHz
- GPU: 8×Tesla P100 connected with Nvlink



4 × V100

- CPU: Intel® Xeon® Gold 6132 CPU @ 2.6 GHz
- 4×Tesla V100 connected with PCI-E

inspur

FaceNet Optimization on a Single GPU

Key motivation: Utilization of GPU

1. GFLOPS

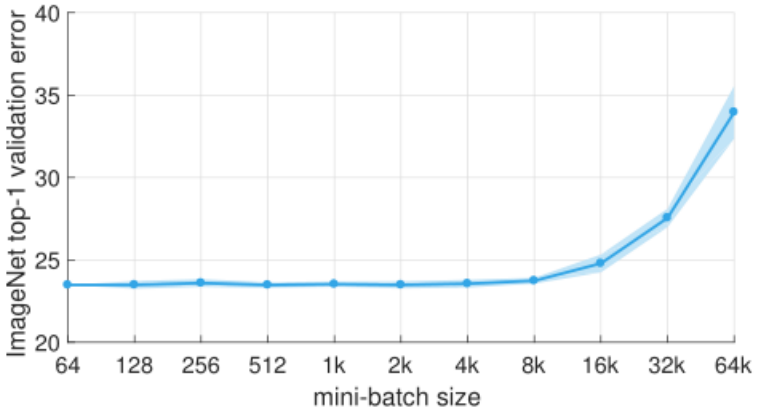
2.3TFlops << Theoretical value of P100

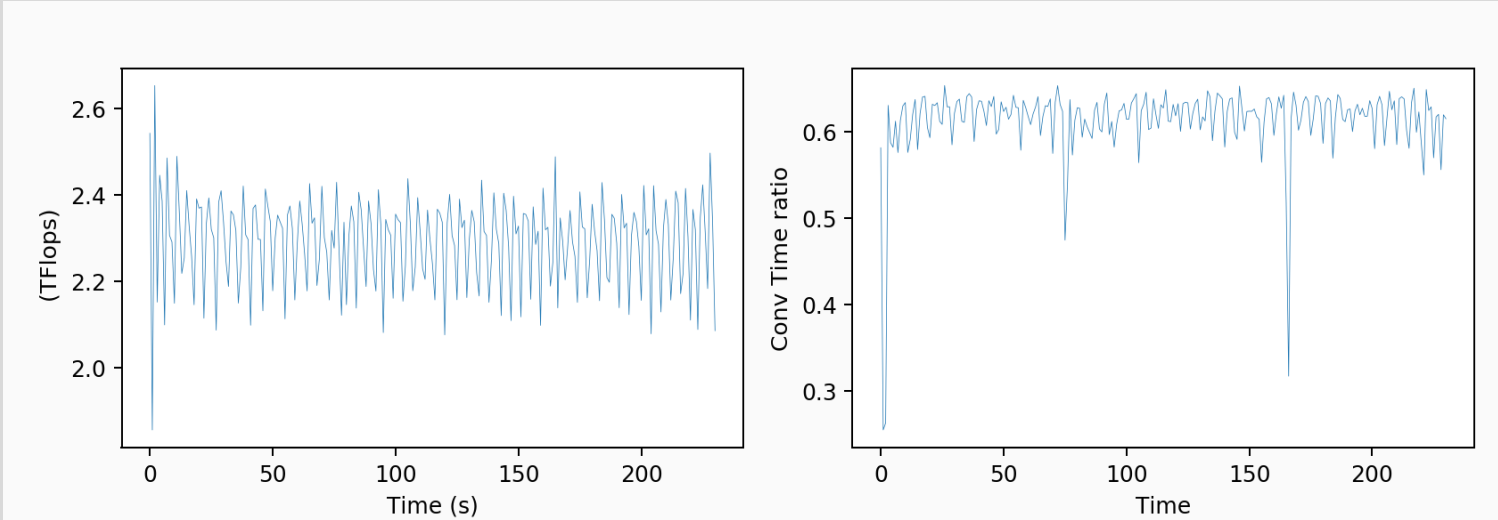
2. Utilization over time

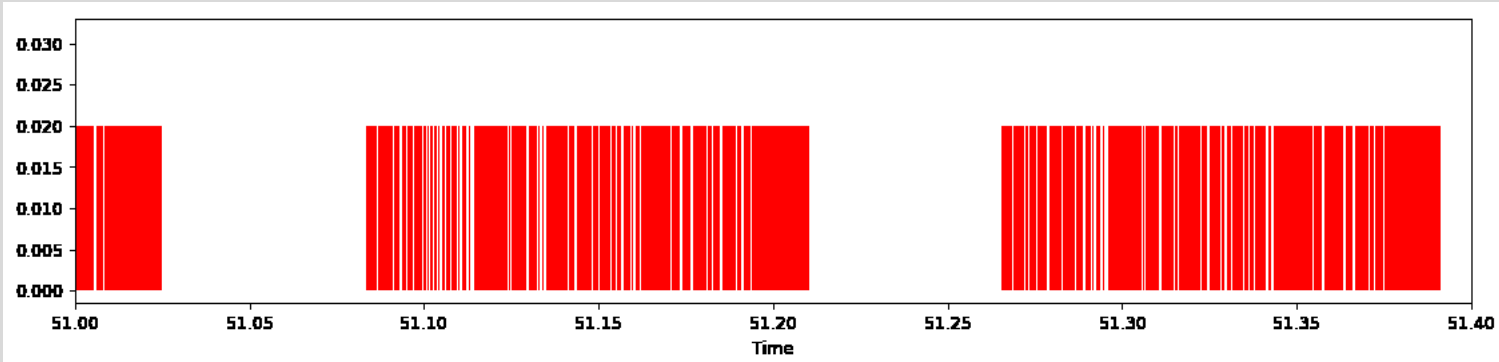
60% work for convolution, and 40% for idle

3. A gap before every iteration

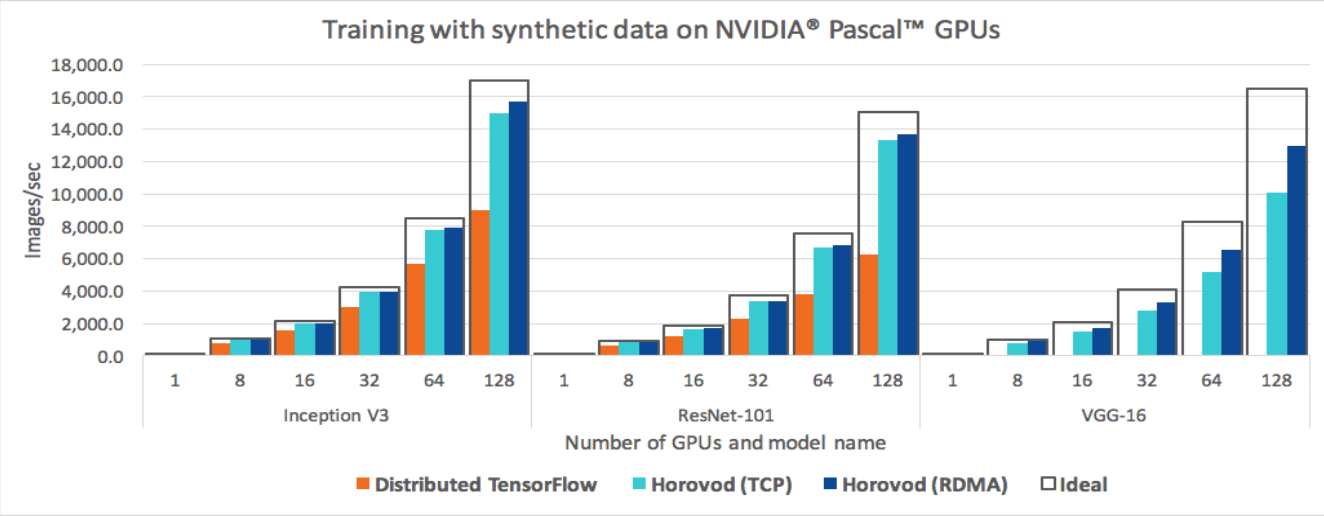
→ for data preprocessing?



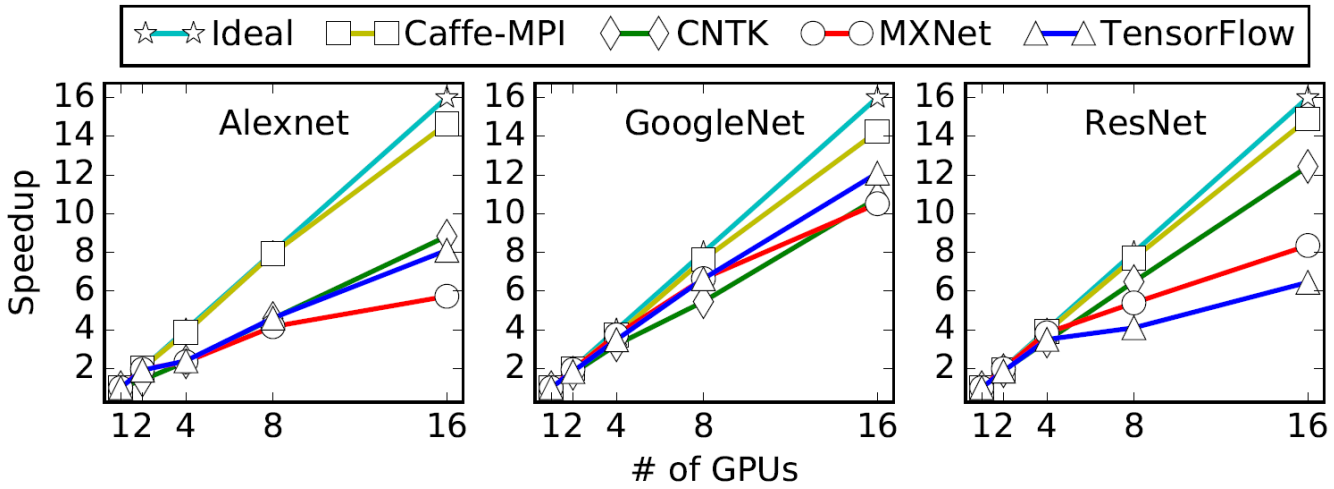




FaceNet Optimization on Multi GPU



(Horovod, 2018)

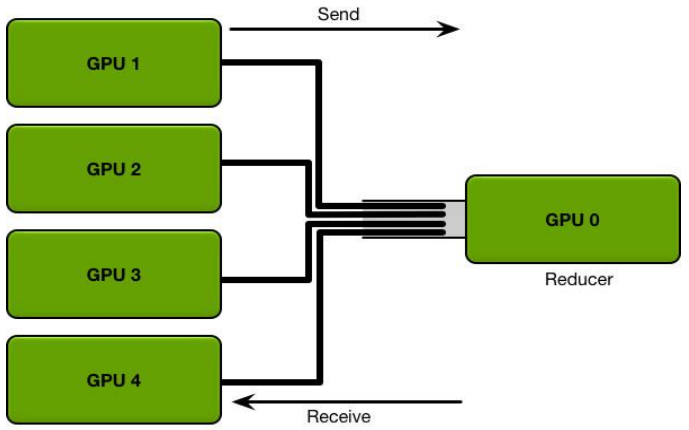


(Shi and Chu, 2017)

Key motivation: FaceNet was developed based on TensorFlow

→ Overcome the shortcomings inherited from Tensorflow

→ Comparatively lower scalability for multi executive units (parameter server) (Time-consuming when start a job)



PS-worker

inspur

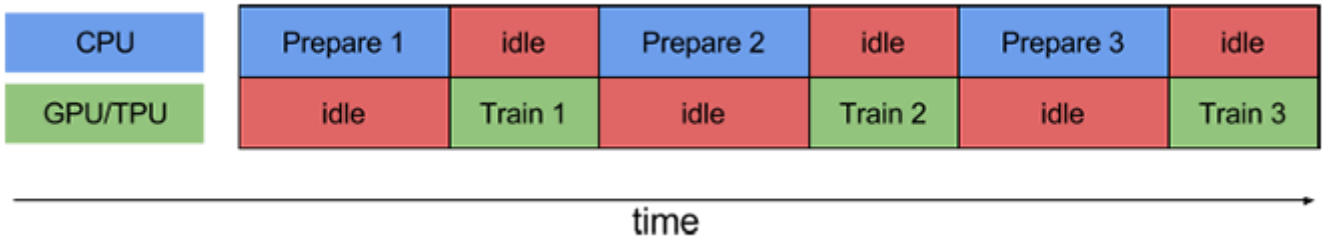
Outline

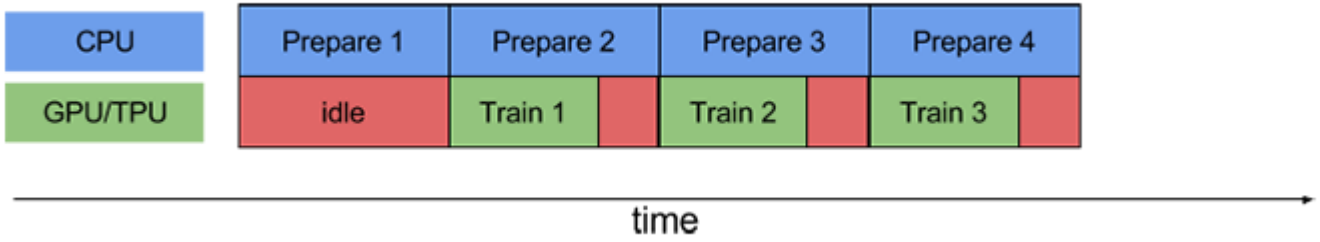
- Introduction and Background
- Part I: Teye: A Tool for Profiling and Performance Tuning for CPU and GPU
- Part II: Analysis of the Bottlenecks for Performance Optimizations
- Part III: Performance Optimization on a Single GPU and Communication

  Optimization Multi-GPUs : A Case Study on FaceNet
- Conclusion

Trouble-shooting: on a Single GPU

Software Pipeline





Problem:

Gap before each iteration

Hypothesis:

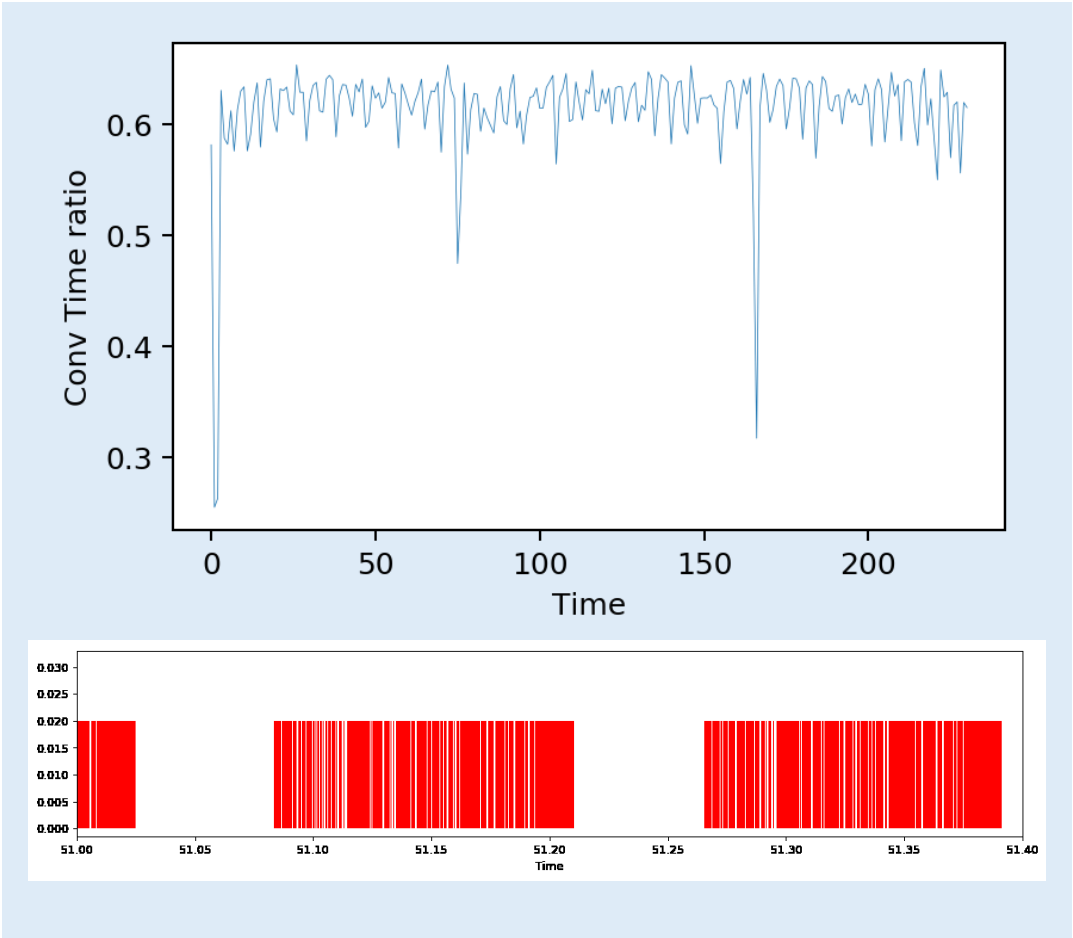
Data preprocessing and training are individual processes. spontaneously with CPU and GPU.

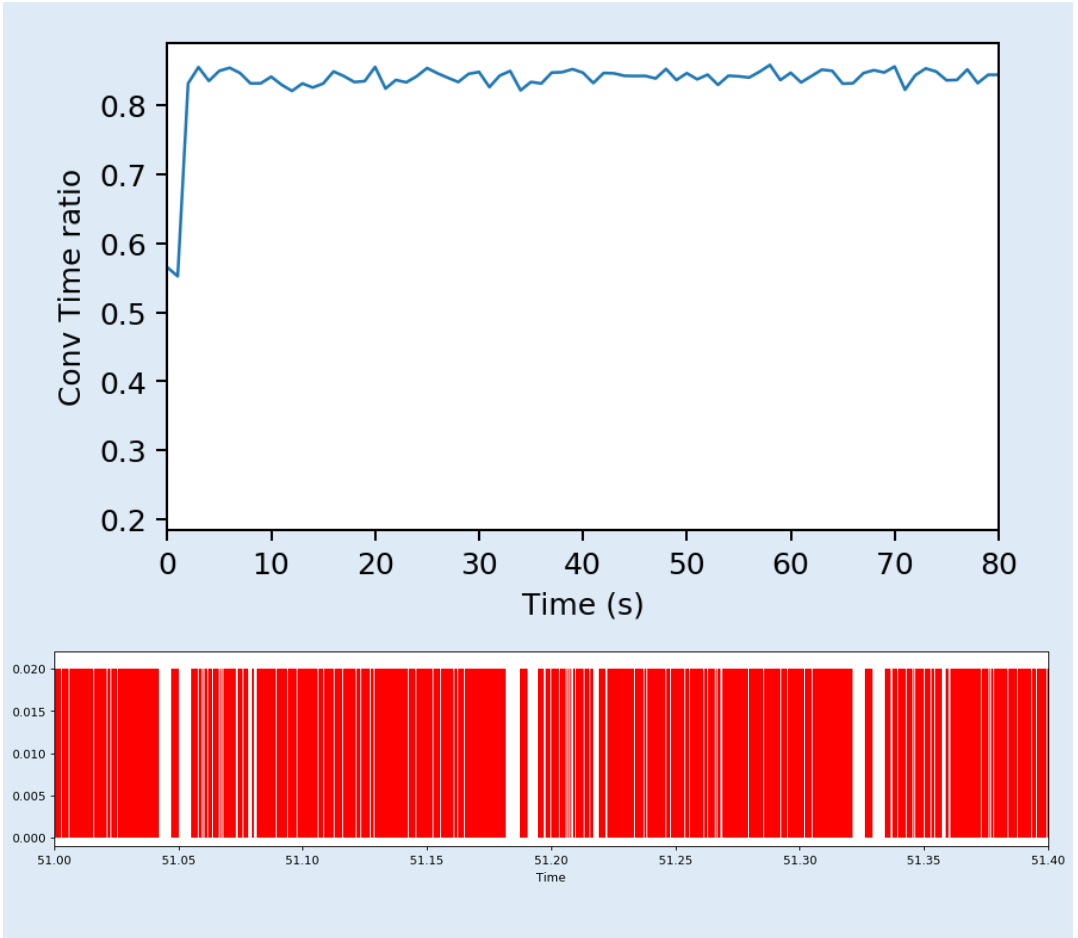
Solution:

tf.data

inspur

Results: on a Single GPU

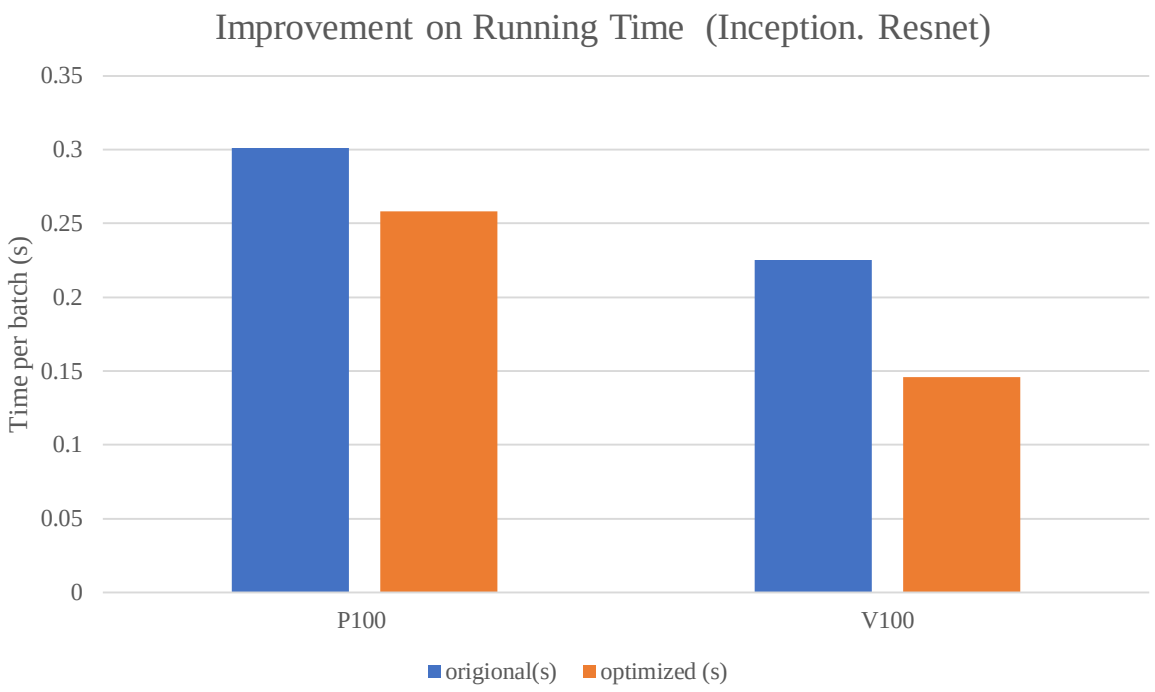


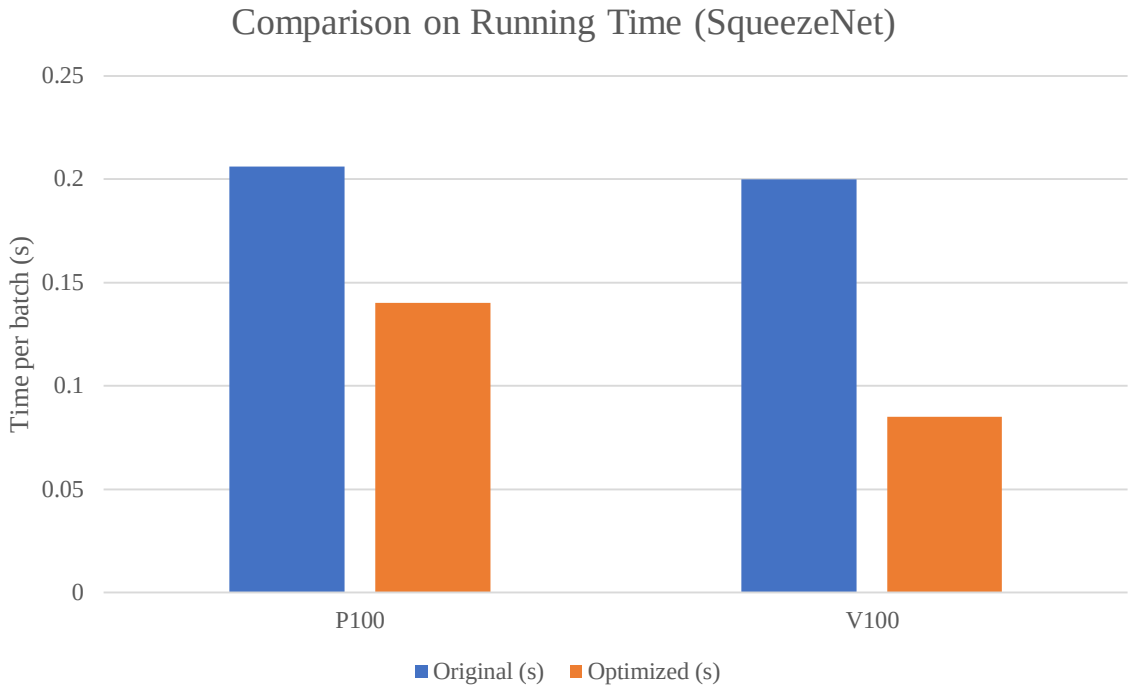


GPU Utilization over time: from 60% to 90%
Almost diminish the influence of data preprocessing

inspur

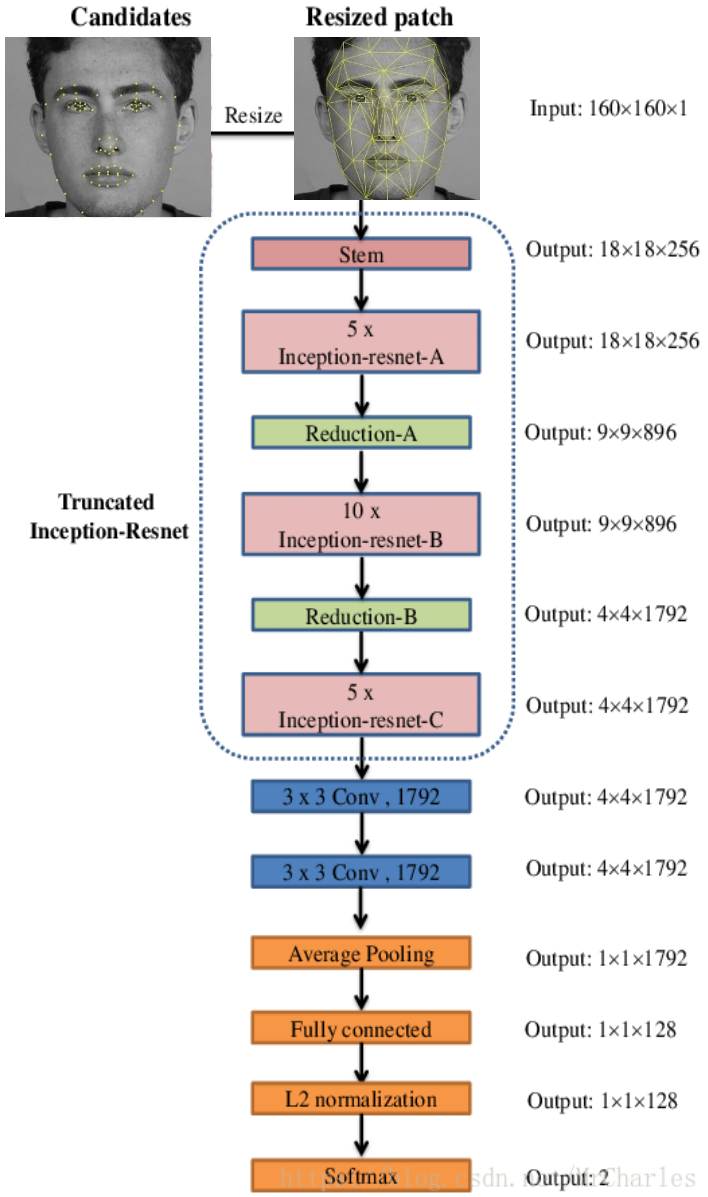
Results: on a Single GPU





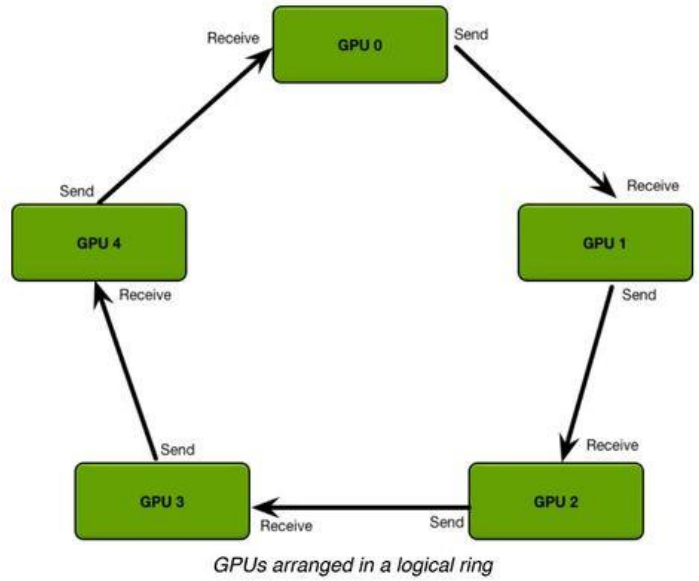
BS=64	Original(s)	Optimized(s)	Improvement over optimization	BS=90	Original(s)	Optimized(s)	Improvement over optimization
P100 (inception.resnet)	0.301	0.258	17%	P100 (squeezenet)	0.206	0.140	47%
V100 (inception.resnet)	0.225	0.146	54%	V100 (squeezenet)	0.200	0.085	135%
Improvement by hardware	34%	77%		Improvement by hardware	3%	65%	

Trouble-shooting: on Multi GPUs



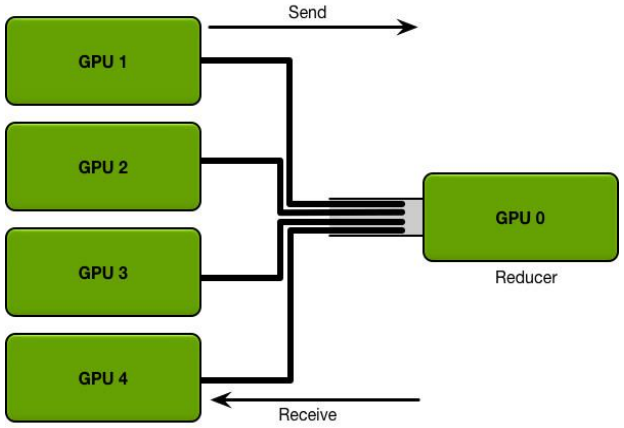
From single GPU to multi GPUs :
Involve Horovod into distributed FaceNet

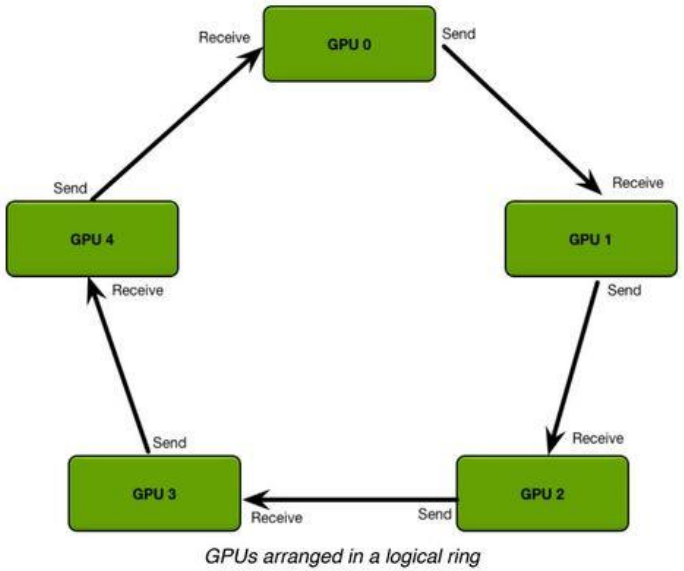
- Adjusted the learning rate by
$$\mathrm{lr} \times \sqrt{\mathrm{hvd.size()}}$$
- Applied ring all-reduce as communication method
  - Communication per node
  $$2(N-1)/N$$
  - Almost irrelated to number of nodes



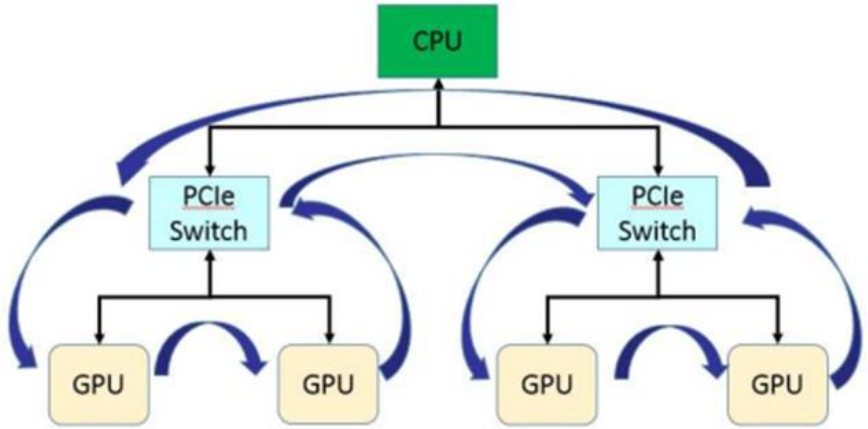
inspur

Trouble-shooting: on Multi GPUs

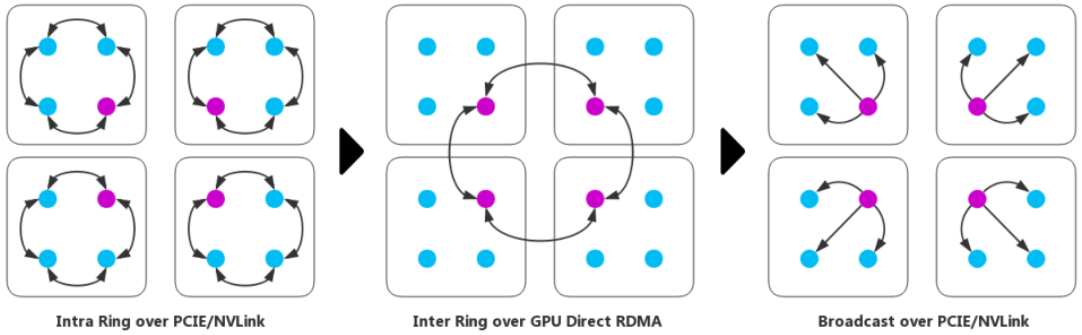




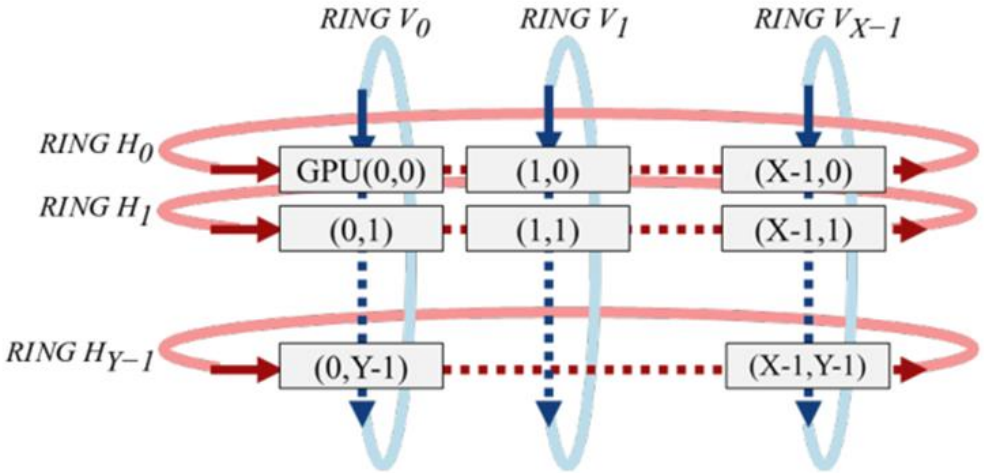
GPUs arranged in a logical ring



(multi-ring communication, 2017 by IBM)



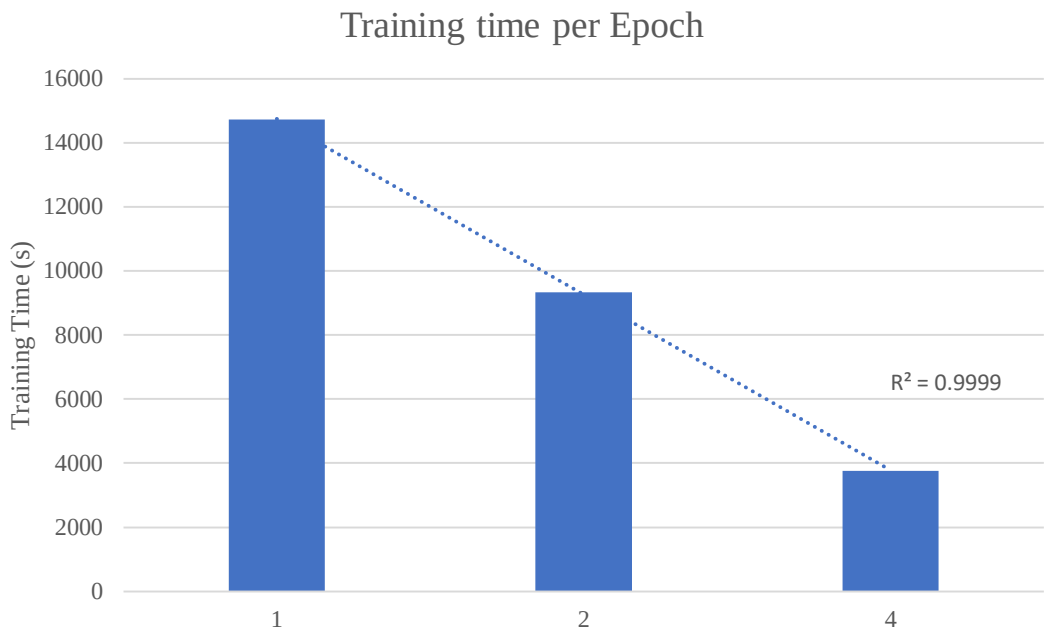
(Three-phase all-reduce, 2018 by Tencent)

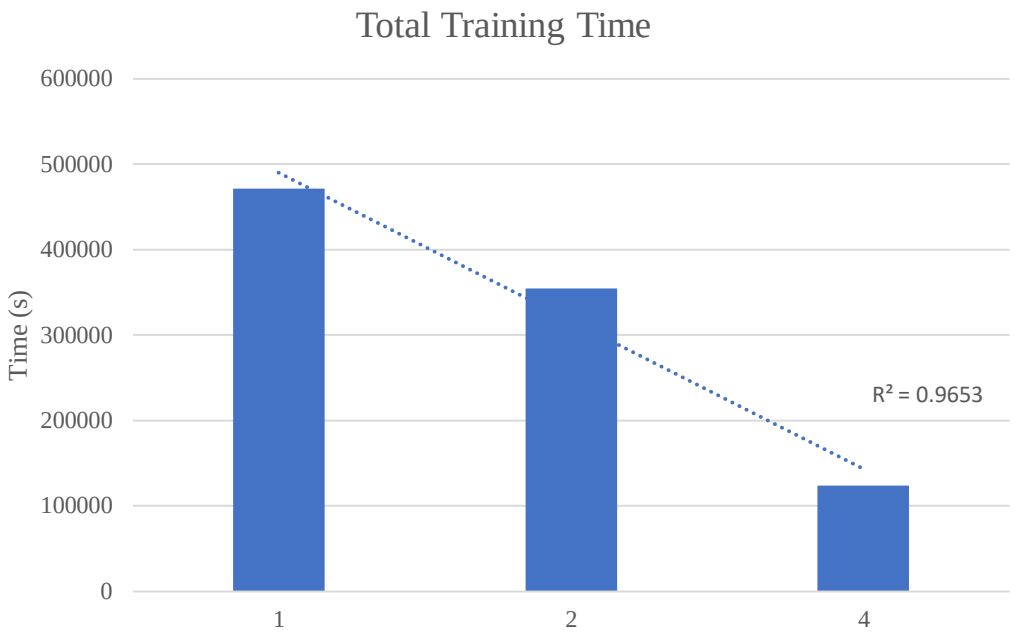


(2D-T all-reduce, 2018 by Tencent)

inspur

Results on Multi GPUs: Near Linear Scalability



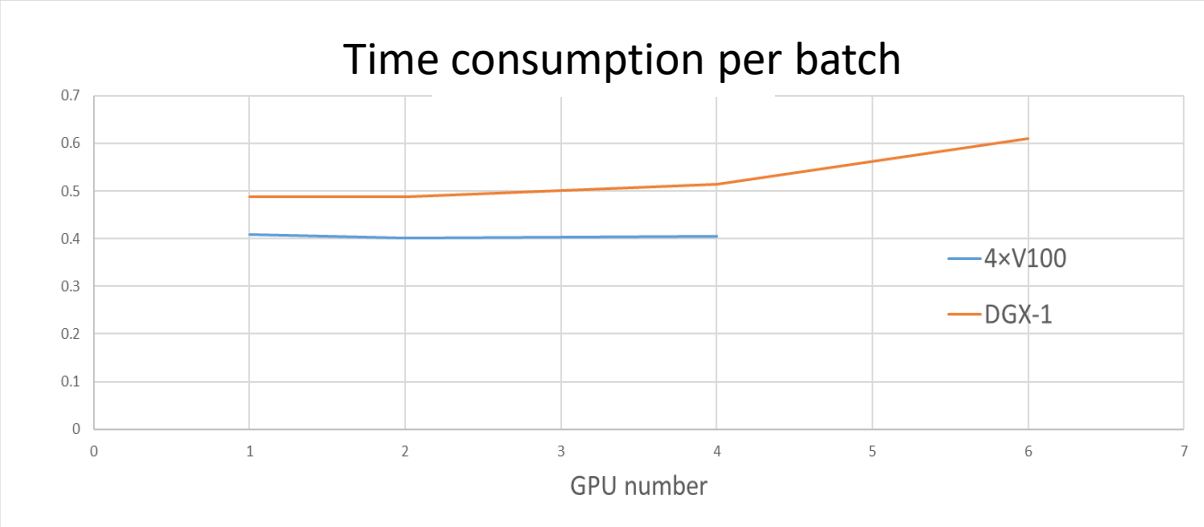


# of nodes	Iteration	Training time per Epoch(s)
1	32000	14720
2	19000	9327.878
4	8250	3753.75

# of nodes	Iteration*Epoch	Total Training time (s)
1	1024000	471040
2	722000	354459.4
4	272250	123873.8

inspur

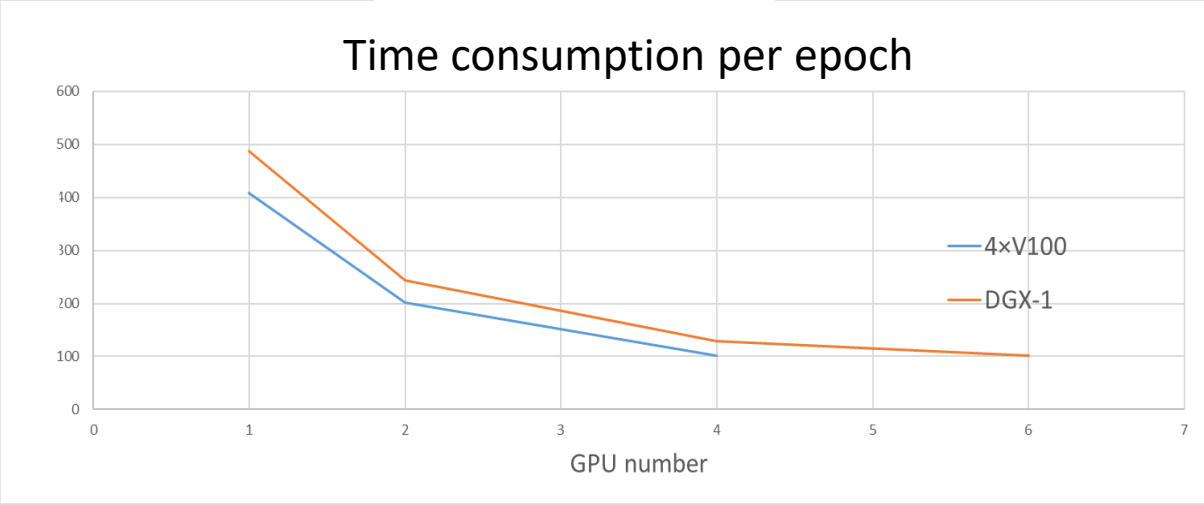
Results on Multi GPUs: P100 vs V100

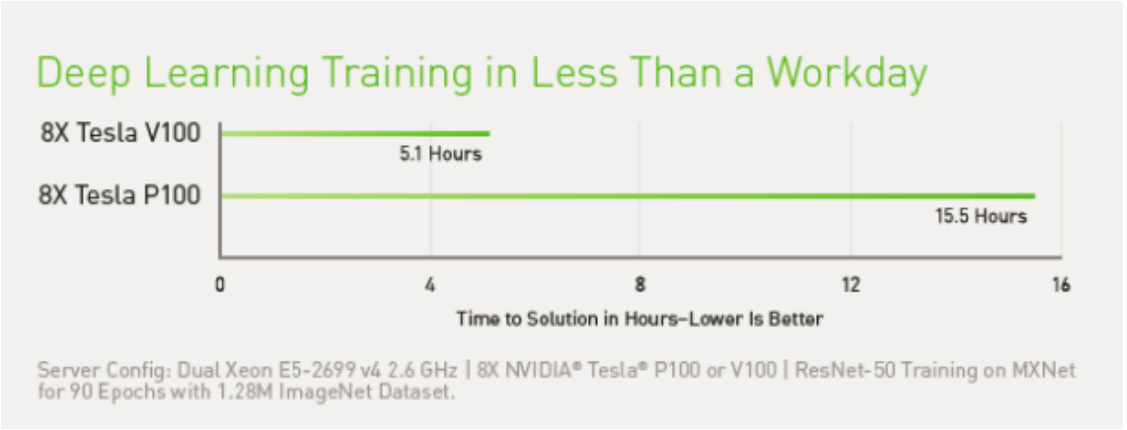


Upper:
Almost the same for 2 and 4 GPUs, but increase for 6 GPU → Communication between GPUs?

Lower:
1.25 times per epoch

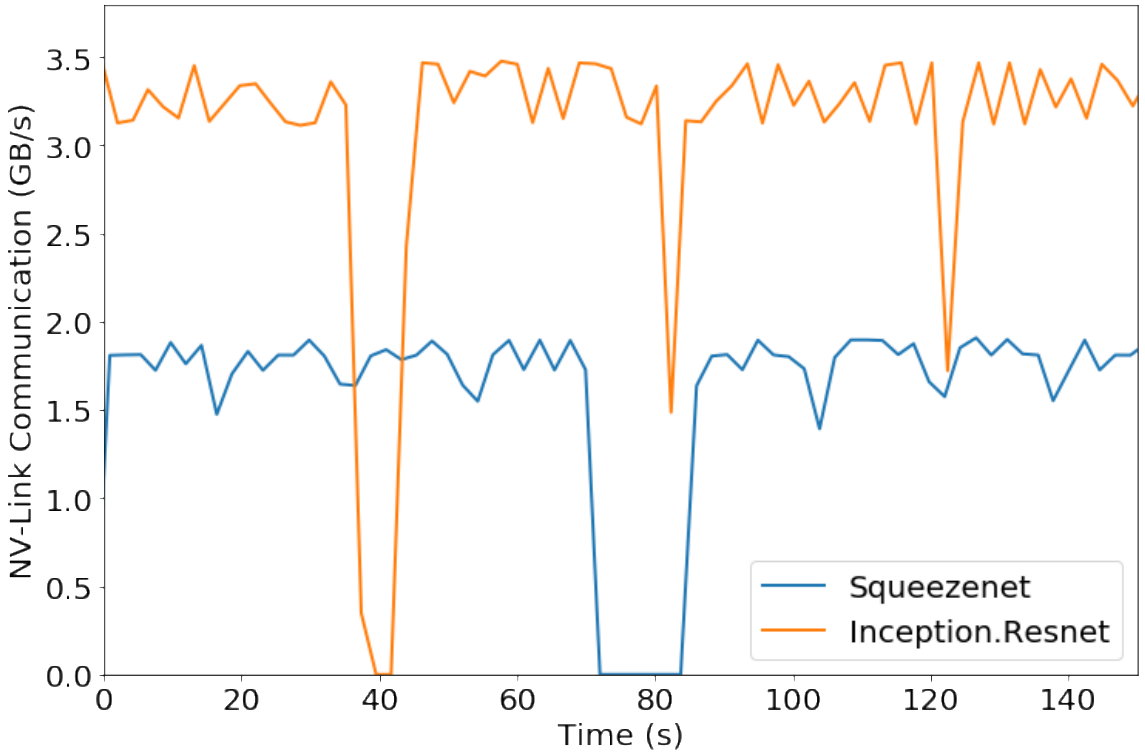
→Clarify the influence of Nvlink





inspur

Results on Multi GPUs: Communication



Nv-link communication between each two GPUs.
（distributed training with 4 P100）

- Not a problem for Nv-link, with its bandwidth of >100GB/s。
- For communication between nodes， may be a burden for 25Gbps Ethernet
- For Ethernet and cloud computing environment, we need to improve communication methods

→ Gradient Fusion communication
Size of fused gradients: Memory size
Number of gradients

→ Half precision communication
full to half precision before All-reduce;
half to full precision before applied

Outline

- Introduction and Background
- Part I: Teye: A Tool for Profiling and Performance Tuning for CPU and GPU
- Part II: Analysis of the Bottlenecks for Performance Optimizations
- Part III: Performance Optimization on a Single GPU and Communication Optimization Multi-GPUs : A Case Study on FaceNet
- Conclusion

Conclusion

- Teye, as part of AI station, is effective in monitoring computational consumption, reflecting performance in clusters, and indicating methods to optimize parallel computations.
- With the assistance of Teye, we found the bottleneck in optimizing the performance of FaceNet.
- On a single GPU, the influence of data preprocessing was almost diminished. We got A 17%~54% performance improvements for Inception.Resnet, and a 47%~135% performance improvement for SqueezeNet.
- On multi GPUs, we got near linear scalability.

Thank You!