DDLBench: Towards a Scalable Benchmarking Infrastructure for Distributed Deep Learning

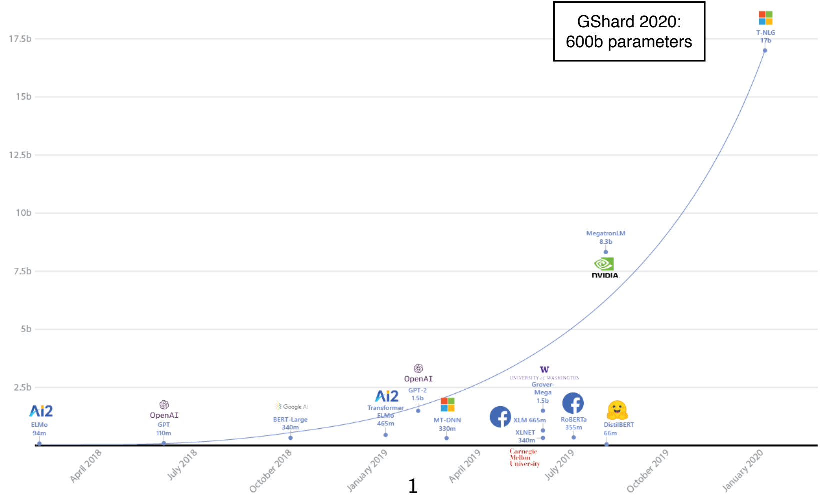
Matthijs Jansen [1, 2] Valeriu Codreanu [1] Ana-Lucia Varbanescu [2]

[1]SURFsara

[2]University of Amsterdam

November 11, 2020

Model size explosion



[1]

[1] C Rosset. "Turing-nlg: A 17-billion-parameter language model by microsoft". In: *Microsoft Blog* (2019)

Distributed deep learning

TABLE I
TRAINING TIME AND TOP-1 VALIDATION ACCURACY WITH RESNET-50 ON IMAGENET

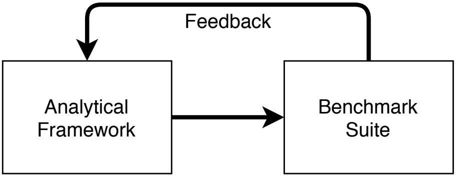
	Batch Size	Processor	DL Library	Time	Accuracy
He et al. [1]	256	Tesla P100 $\times$ 8	Caffe	29 hours	75.3 %
Goyal et al. [2]	8,192	Tesla P100 $\times$ 256	Caffe2	1 hour	76.3 %
Smith et al. [3]	8,192 $\rightarrow$ 16,384	full TPU Pod	TensorFlow	30 mins	76.1 %
Akiba et al. [4]	32,768	Tesla P100 $\times$ 1,024	Chainer	15 mins	74.9 %
Jia et al. [5]	65,536	Tesla P40 $\times$ 2,048	TensorFlow	6.6 mins	75.8 %
Ying et al. [6]	65,536	TPU v3 $\times$ 1,024	TensorFlow	1.8 mins	75.2 %
Mikami et al. [7]	55,296	Tesla V100 $\times$ 3,456	NNL	2.0 mins	75.29 %
This work	**81,920**	**Tesla V100 $\times$ 2,048**	**MXNet**	**1.2 mins**	**75.08%**

[2]

[2] Masafumi Yamazaki et al. "Yet Another Accelerated SGD: ResNet-50 Training on ImageNet in 74.7 seconds". In: *CoRR* abs/1903.12650 (2019). arXiv: 1903.12650. URL: http://arxiv.org/abs/1903.12650

DDLBench

- A generalizable, ready-to-use benchmark suite for distributed deep learning
- Accompanied by an analytical framework

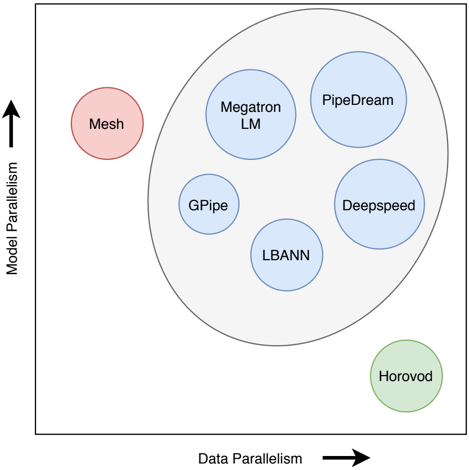


Benchmark suite

Datasets: MNIST, CIFAR-10, ImageNet, Highres
Neural networks: ResNet, VGG, MobileNet v2

Name	*#classes*	*#images*	*Color profile*	*Resolution*
MNIST	10	70000	Grayscale	28 x 28
CIFAR-10	10	60000	RGB	32 x 32
ImageNet	1000	1280000	RGB	224 x 224
Highres	1000	60000	RGB	512 x 512

Distribution models



Frameworks and distribution models

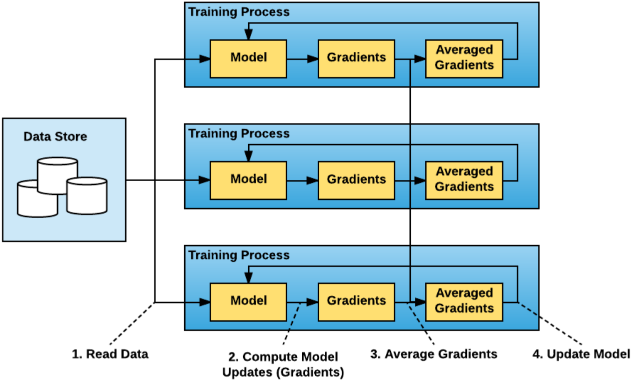
Table: A comparison of different distribution models for machine learning.

Model	*TF*	*PyTorch*	*CPU*	*GPU*	*Data*	*Model*	*Pipeline*
tf.distribute	X		X	X	X		
tf.Mesh	X		X	X	X	X	
PipeDream		X		X		X	X
(torch)GPipe	X	X		X		X	X
Horovod	X	X	X	X	X	X	
torch.distributed		X	X	X	X	X	

Analytical framework

- Data parallelism: Horovod
- Model / Pipeline parallelism: TorchGPipe, PipeDream

Data parallelism



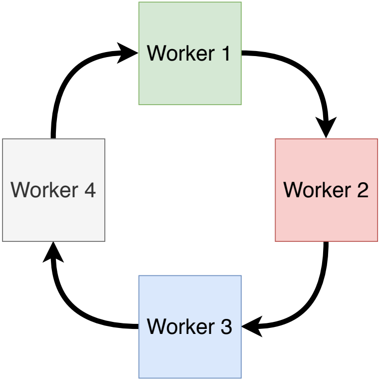
[3]

[3] Alexander Sergeev and Mike Del Balso. "Horovod: fast and easy distributed deep learning in TensorFlow". In: *arXiv preprint arXiv:1802.05799* (2018)

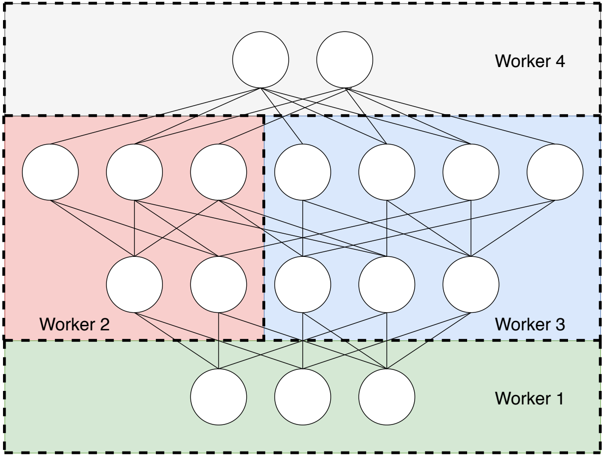
Performance model

$$T_{horovod} = \frac{T_{seq}}{W} + 2(W-1) \cdot \max_{i=1}^{W}(L_{i,i+1} + \frac{\min(G, th)}{W \cdot BW_{i,i+1}}) \quad (1)$$

Symbol	Description
T	Training time
W	#workers
$L_{i,j}$	Latency worker i to j
G	Total gradient size
th	Tensor fusion threshold
$BW_{i,j}$	Bandwidth worker i to j



Model parallelism



Workload

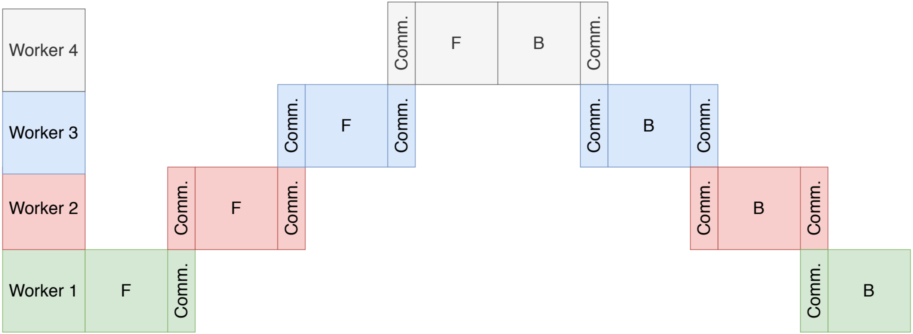


Figure: Execution pipeline of model parallelism.

TorchGPipe

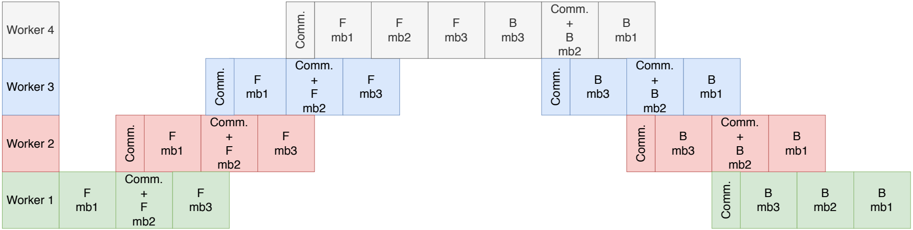
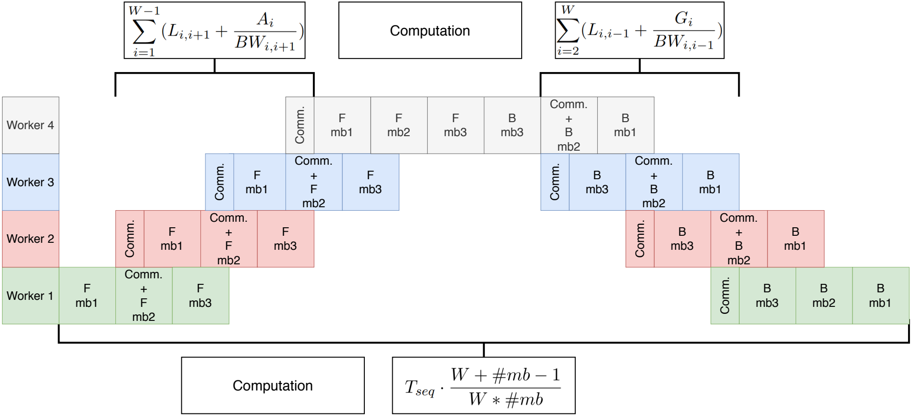


Figure: Execution pipeline of GPipe with 3 micro-batches per batch.

Performance model



PipeDream

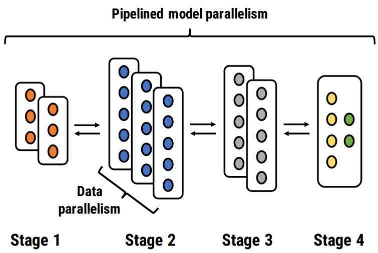


Figure 6: Pipeline Parallel training in PipeDream combines pipelining, model- and data-parallel training.

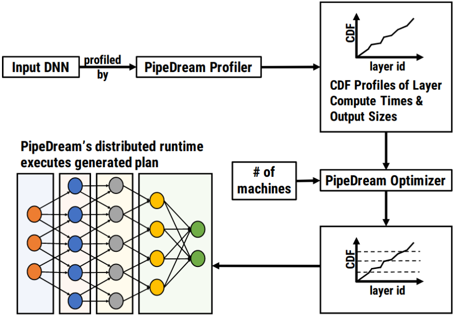


Figure 7: PipeDream's automated mechanism to partition DNN layers into stages. PipeDream first profiles the input DNN, to get estimates for each layer's compute time and output size. Using these estimates, PipeDream's optimizer partitions layers across available machines.

4

[4] Aaron Harlap et al. "Pipedream: Fast and efficient pipeline parallel dnn training". In: *arXiv preprint arXiv:1806.03377* (2018)

Performance model

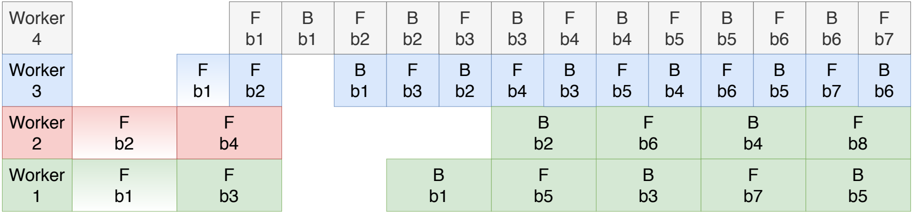


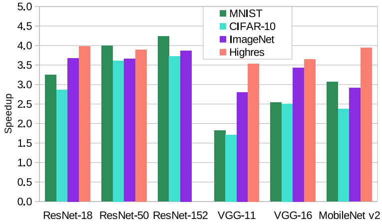
Figure: Execution pipeline of PipeDream with 3 model partitions in a 2-1-1 configuration.

$$T_{pipedream} = \frac{T_{seq}}{W}$$

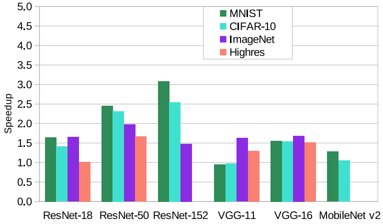
Batch size configuration

Dataset	*PyTorch*	*Horovod*	*GPipe (#mb)*	*PipeDream*
MNIST	128	128	3072 (24)	128
CIFAR-10	64	64	2048 (32)	64
ImageNet-1000	32	32	384 (12)	32
Highres	32	32	48 (12)	32

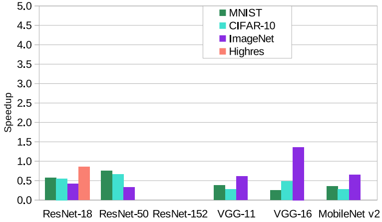
Single-node benchmarks



(a) Horovod

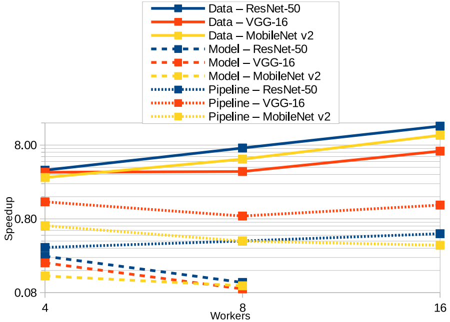


(b) GPipe

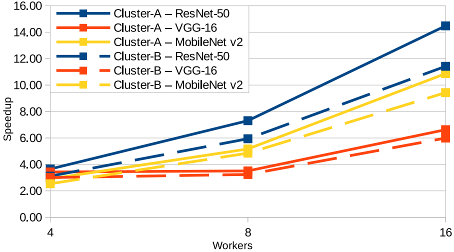


(c) PipeDream

Performance scaling



(a) Data, Model and Pipeline parallelism

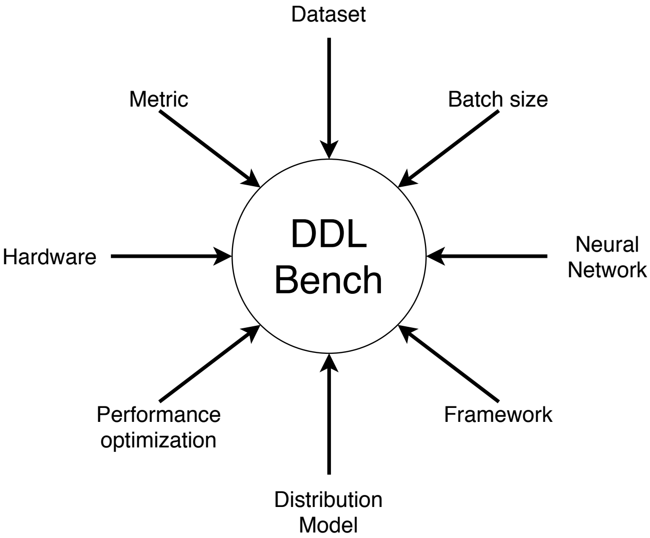


(b) Titan RTX vs 1080 Ti

Conclusion

- v0.1: 4 datasets, 6 neural networks, 3 distribution models
- DDLBench can capture the complex and dynamic behaviour of DDL applications
- Designed with diversity and extensibility in mind

Extending DDLBench



Integration

DEEP500

MLPerf

HPC AI500

Thank you for your attention

https://github.com/sara-nl/DDLBench