SW/HW Innovations in Emerging DL Training Systems

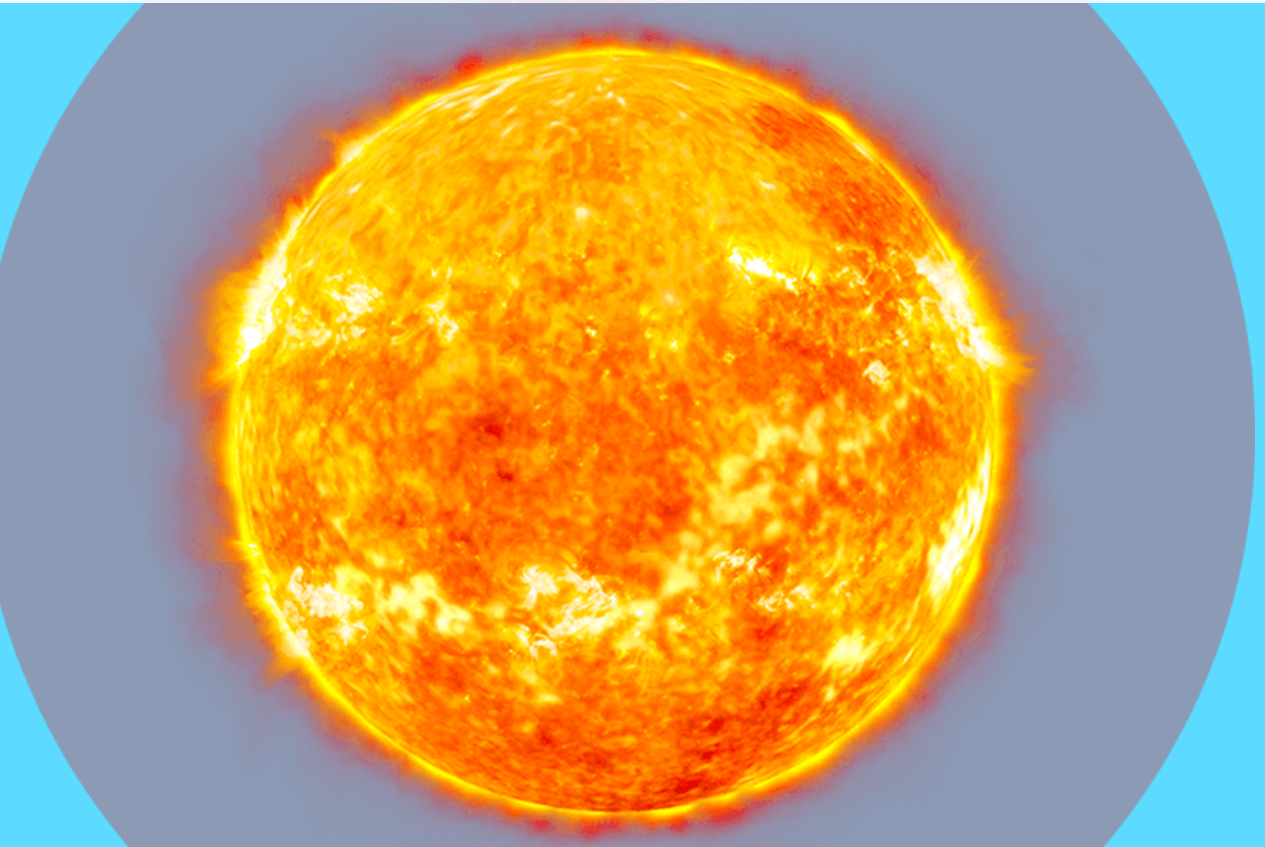
Urmish Thakker
Engineering Manger, NLP Group

SambaNova SYSTEMS

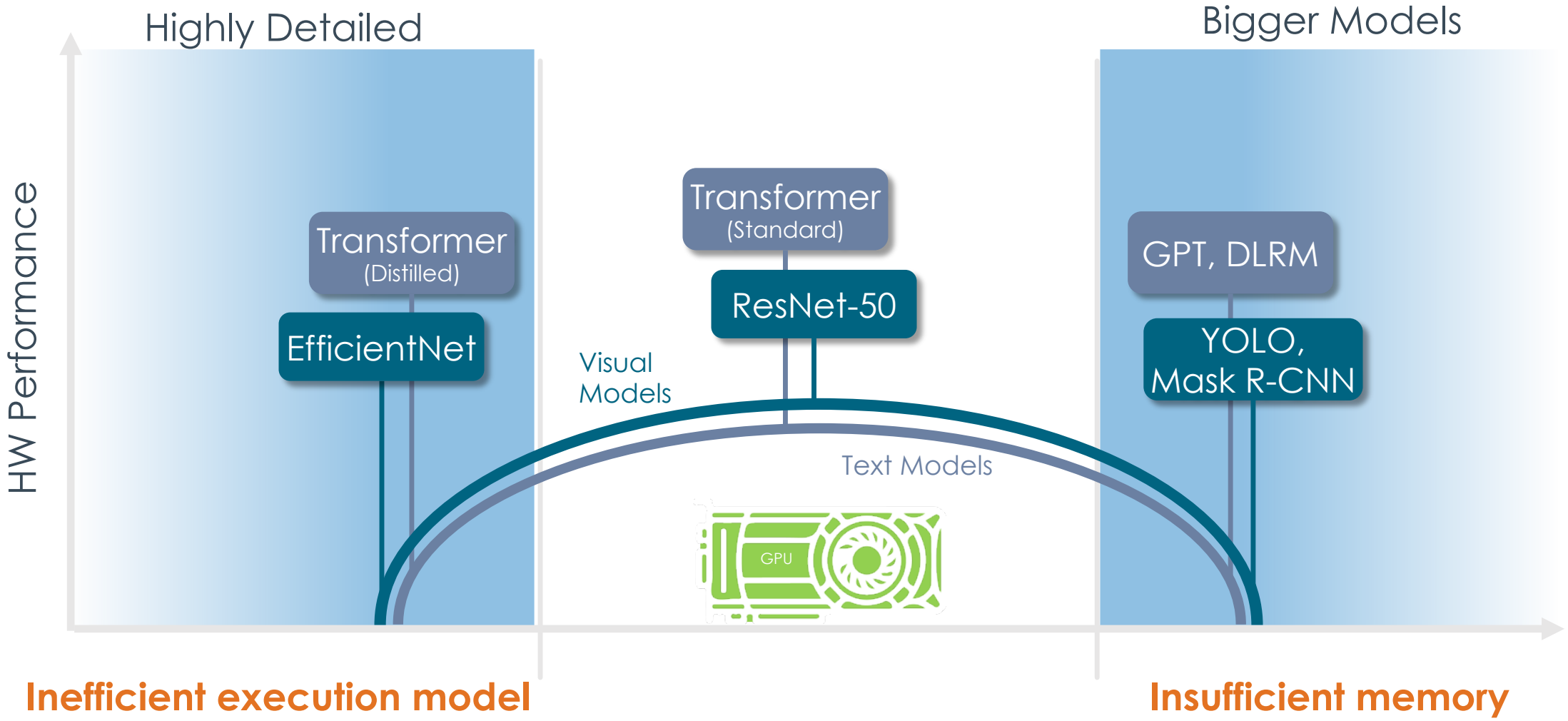
Too Hot

Goldilocks Zone

Too Cold

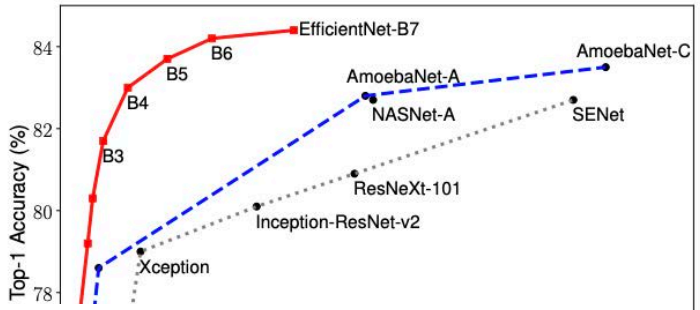


Yesterday's Goldilocks Zone is Constraining Progress



Trend of SOTA Models

Highly detailed



TinyBERT: Distilling BERT for Natural Language Understanding

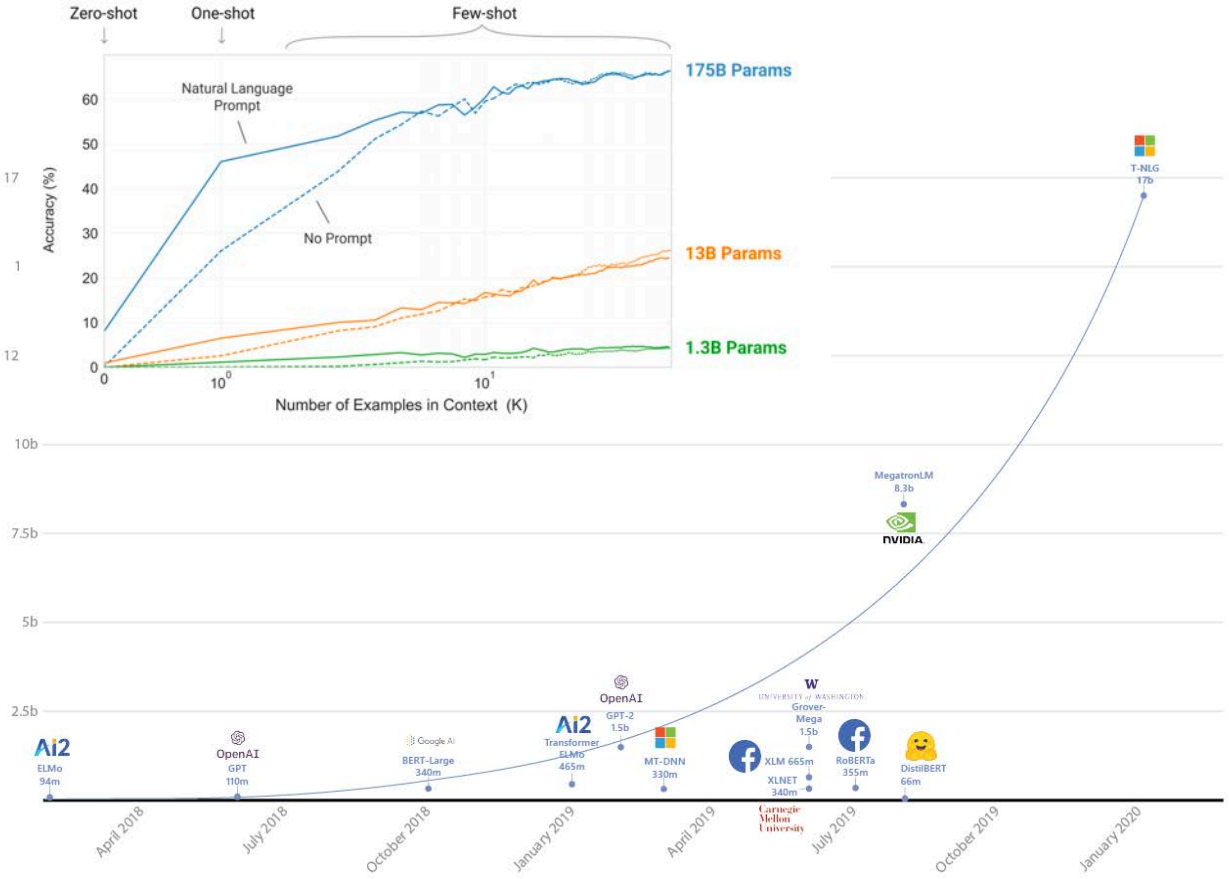
Xiaoqi Jiao[1]*†, Yichun Yin[2]*‡, Lifeng Shang[2]‡, Xin Jiang[2]
Xiao Chen[2], Linlin Li[3], Fang Wang[1]‡ and Qun Liu[2]
[1]Key Laboratory of Information Storage System, Huazhong University of Science and Technology, Wuhan National Laboratory for Optoelectronics
[2]Huawei Noah's Ark Lab

{yin
{che

DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter

Victor SANH, Lysandre DEBUT, Julien CHAUMOND, Thomas WOLF
Hugging Face
{victor,lysandre,julien,thomas}@huggingface.co

Bigger Models



Emerging Hardware and Systems

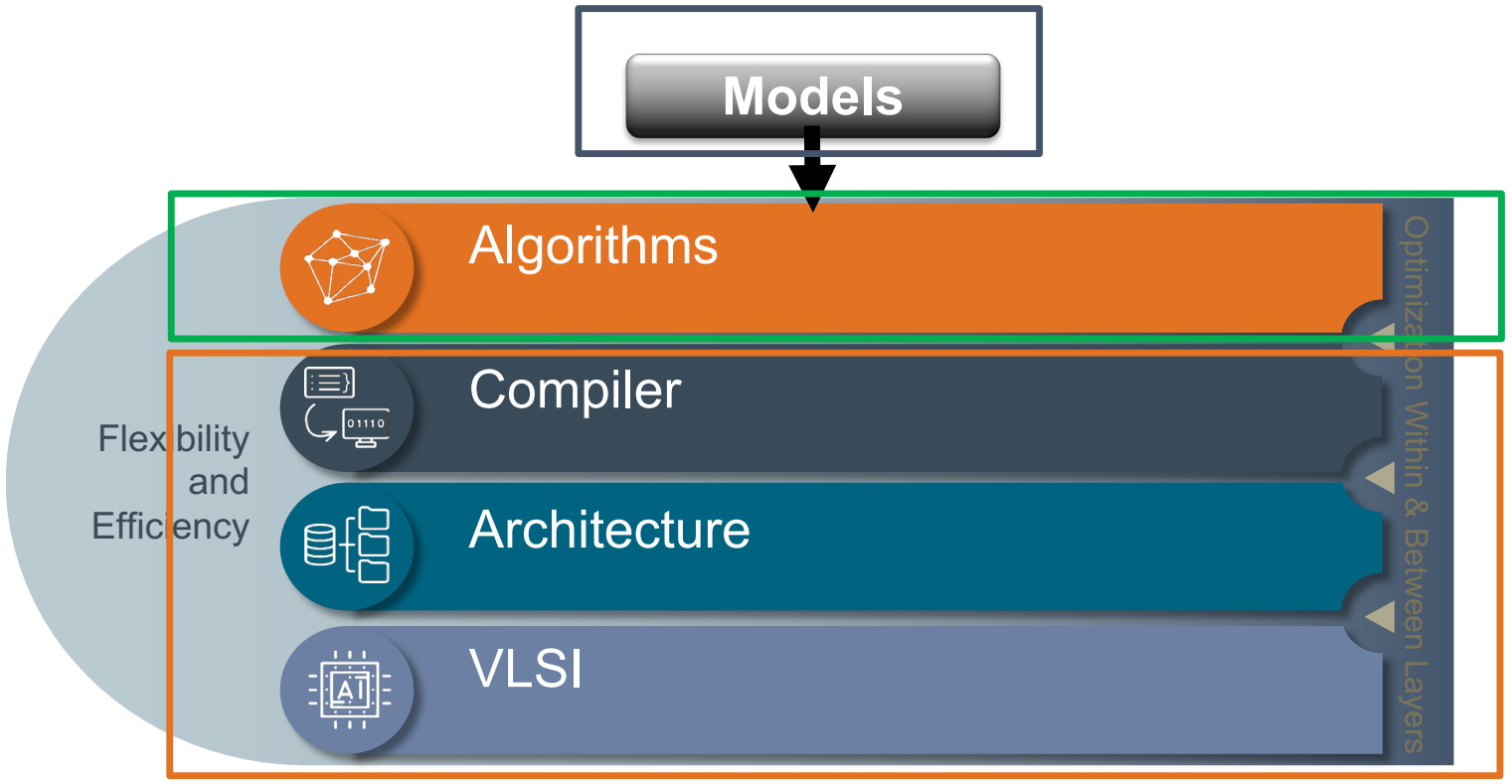
Our Mission

Shaping the next-generation ML / DL computing system to accelerate the full model spectrum

How do we break out of the Godilocks Zone?

Fundamental advances required at all layers of the SW/HW stack.

The SambaNova Systems Advantage



Application innovations

High model accuracy

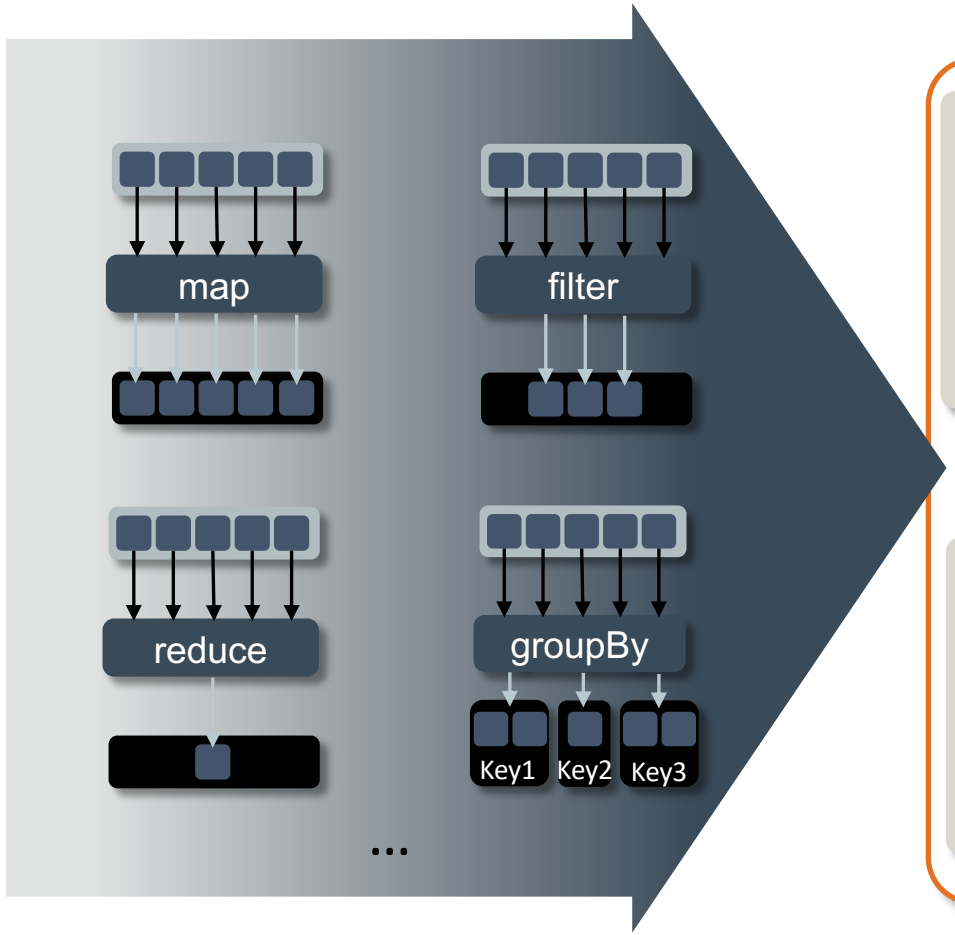
High compute efficiency

Part 1.

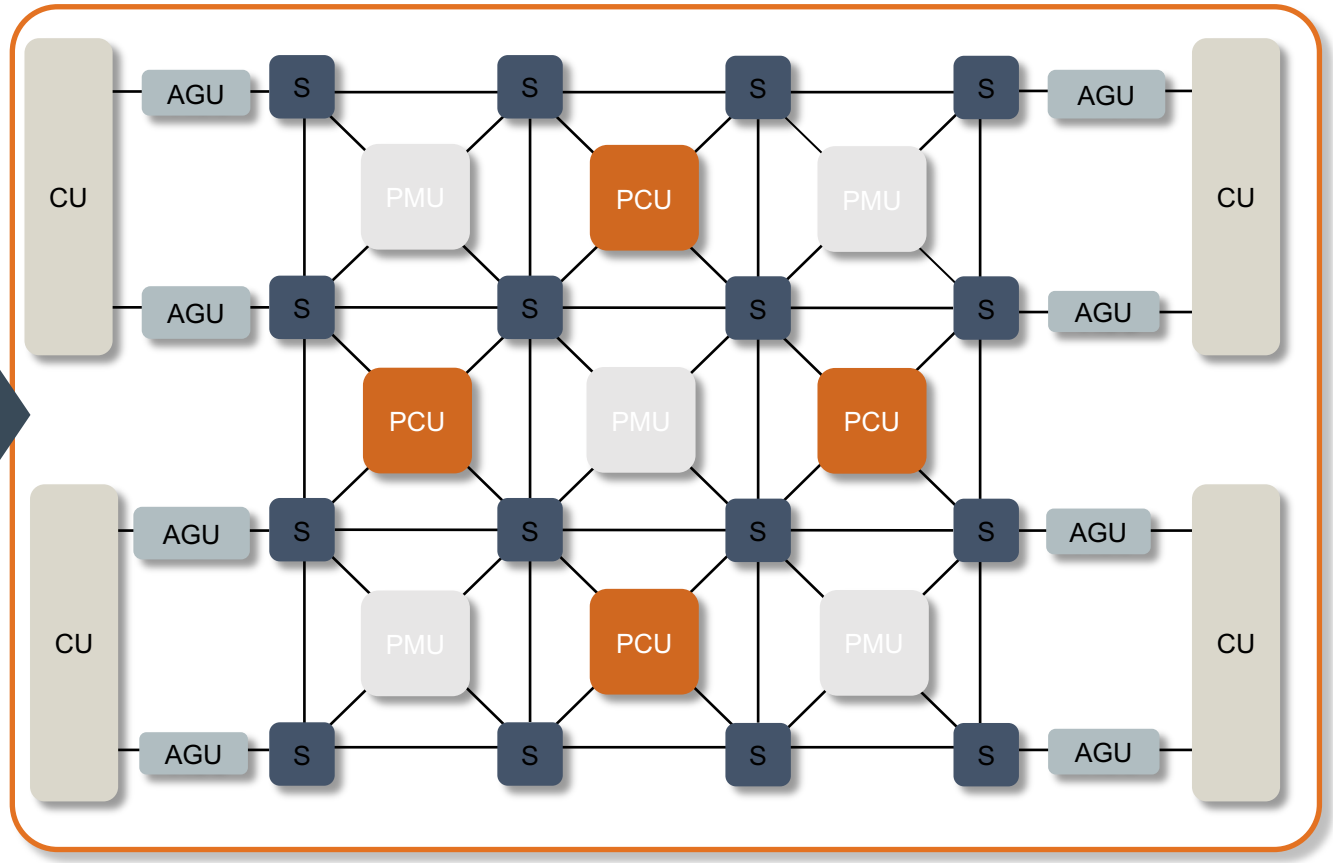
Enabling higher compute efficiency

Architecture: Reconfigurable Dataflow Unit (RDU)

Parallel Patterns

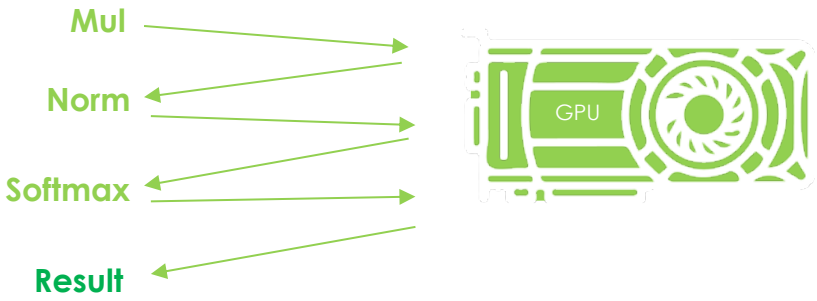


Array of reconfigurable compute, memory and communication

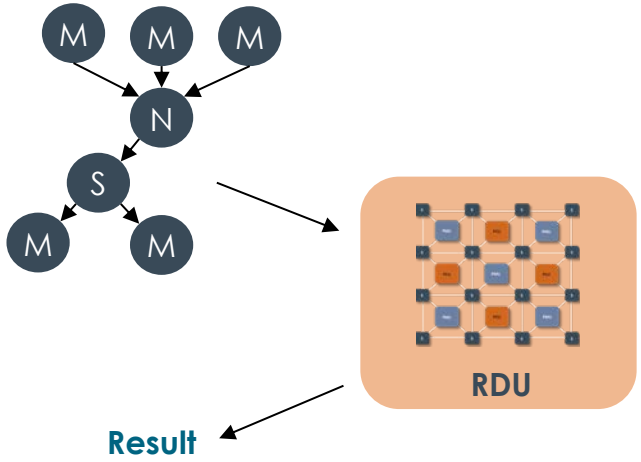


Spatial Dataflow Within an RDU

The old way: kernel-by-kernel

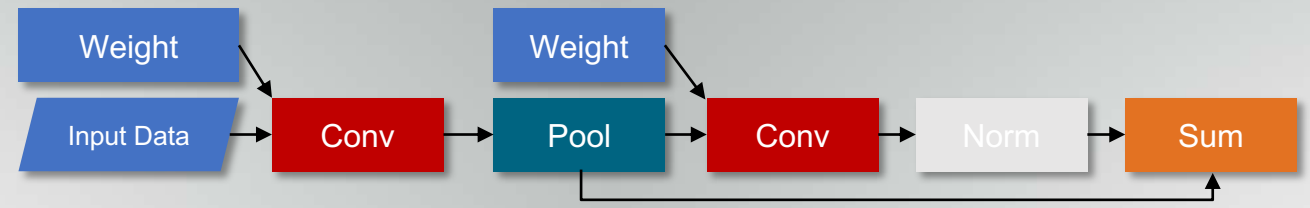


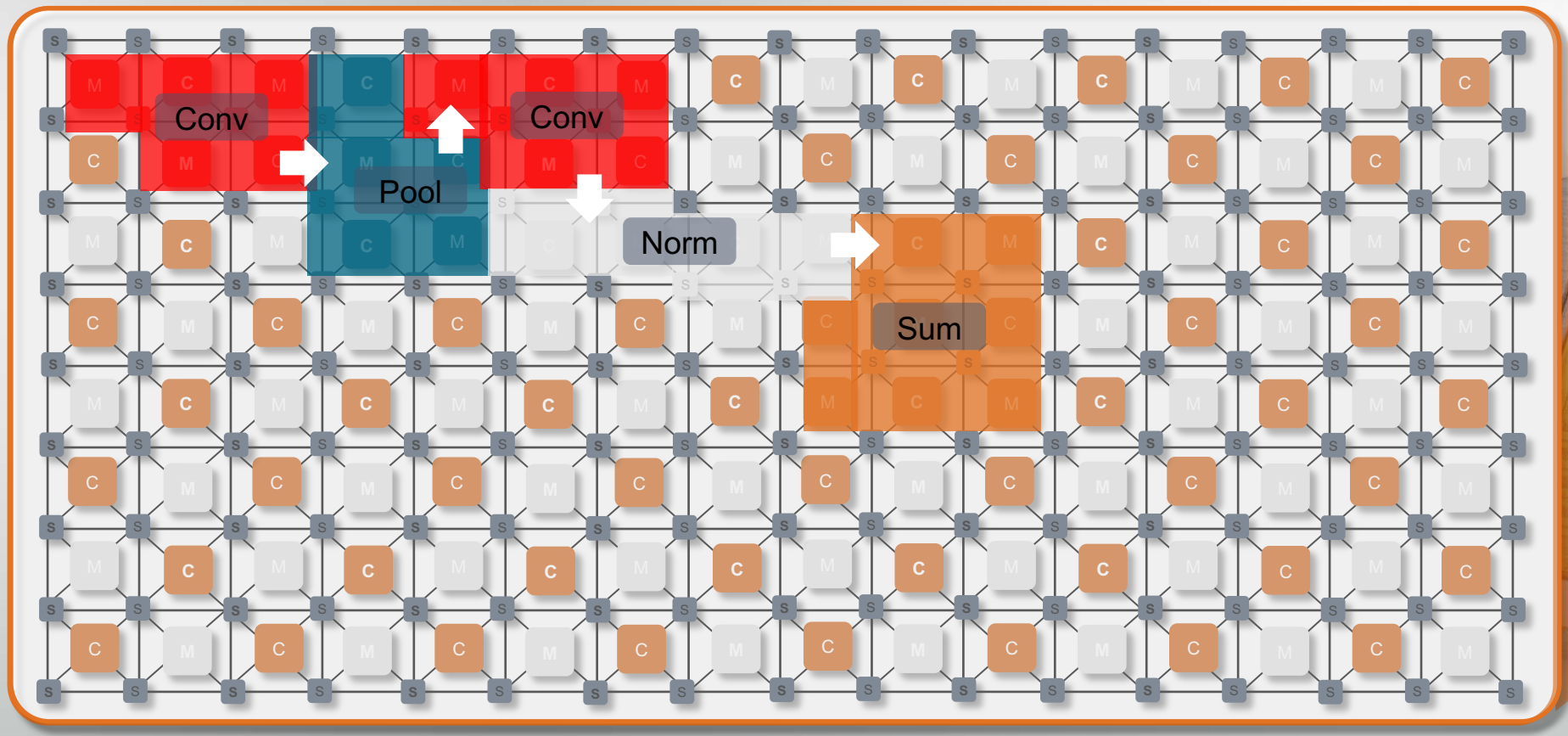
The Dataflow way: spatial

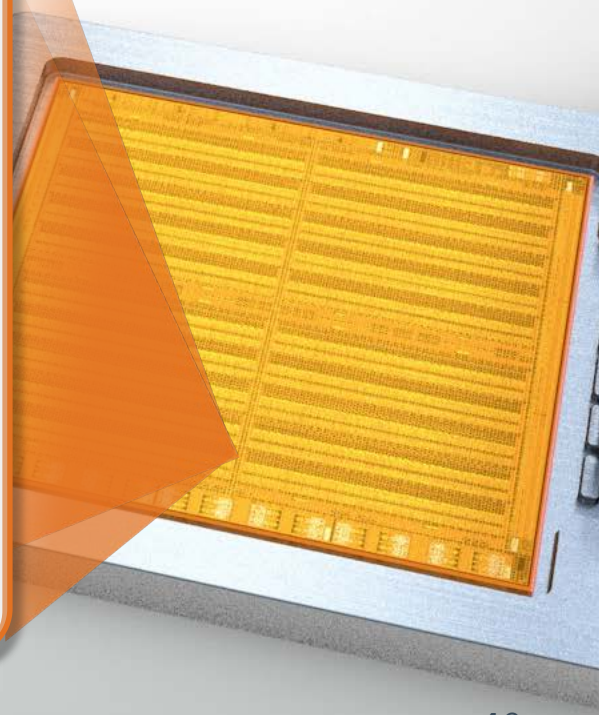


SambaFlow eliminates overhead and maximizes utilization

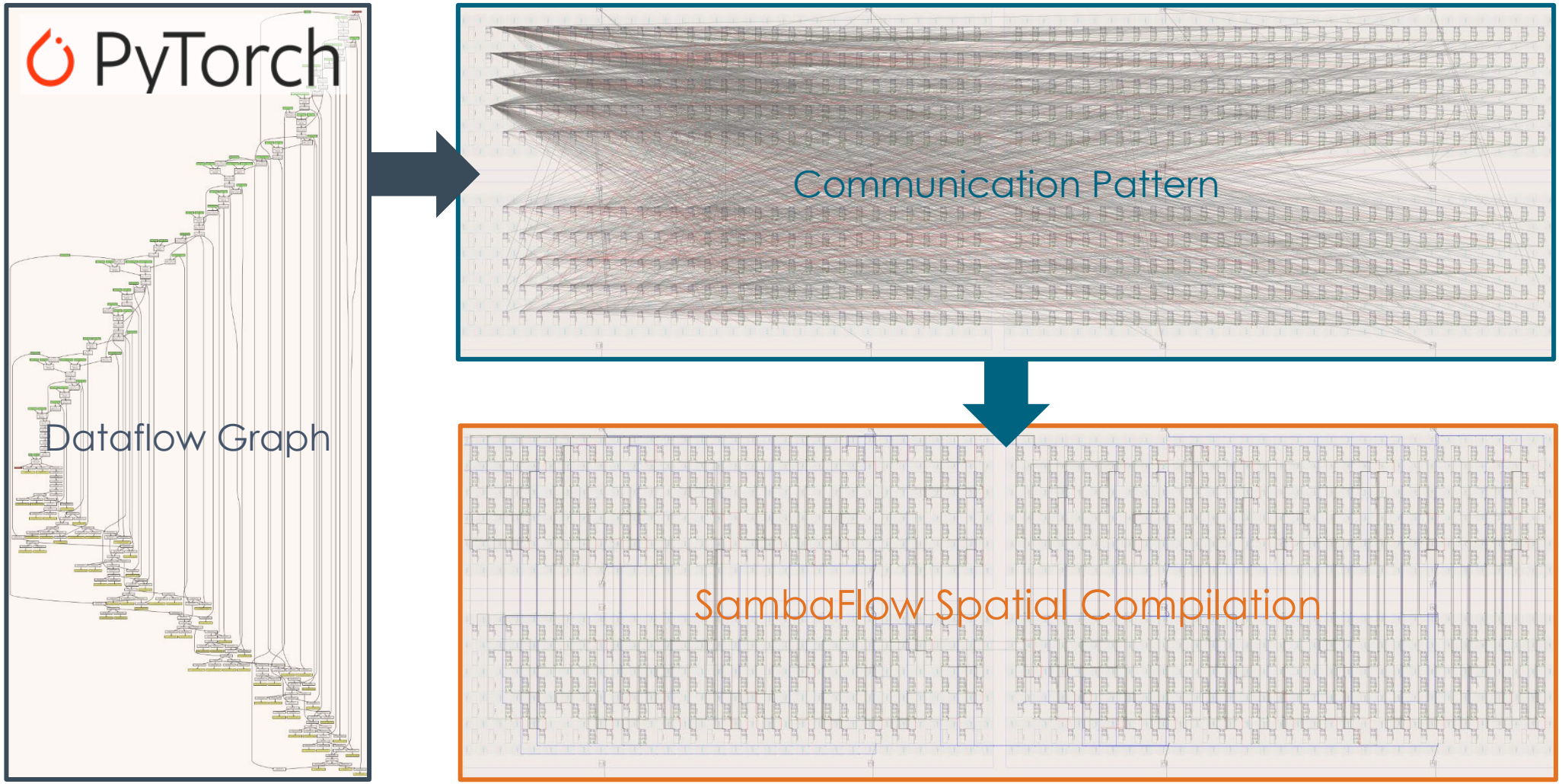
Rapid Dataflow Compilation to RDU





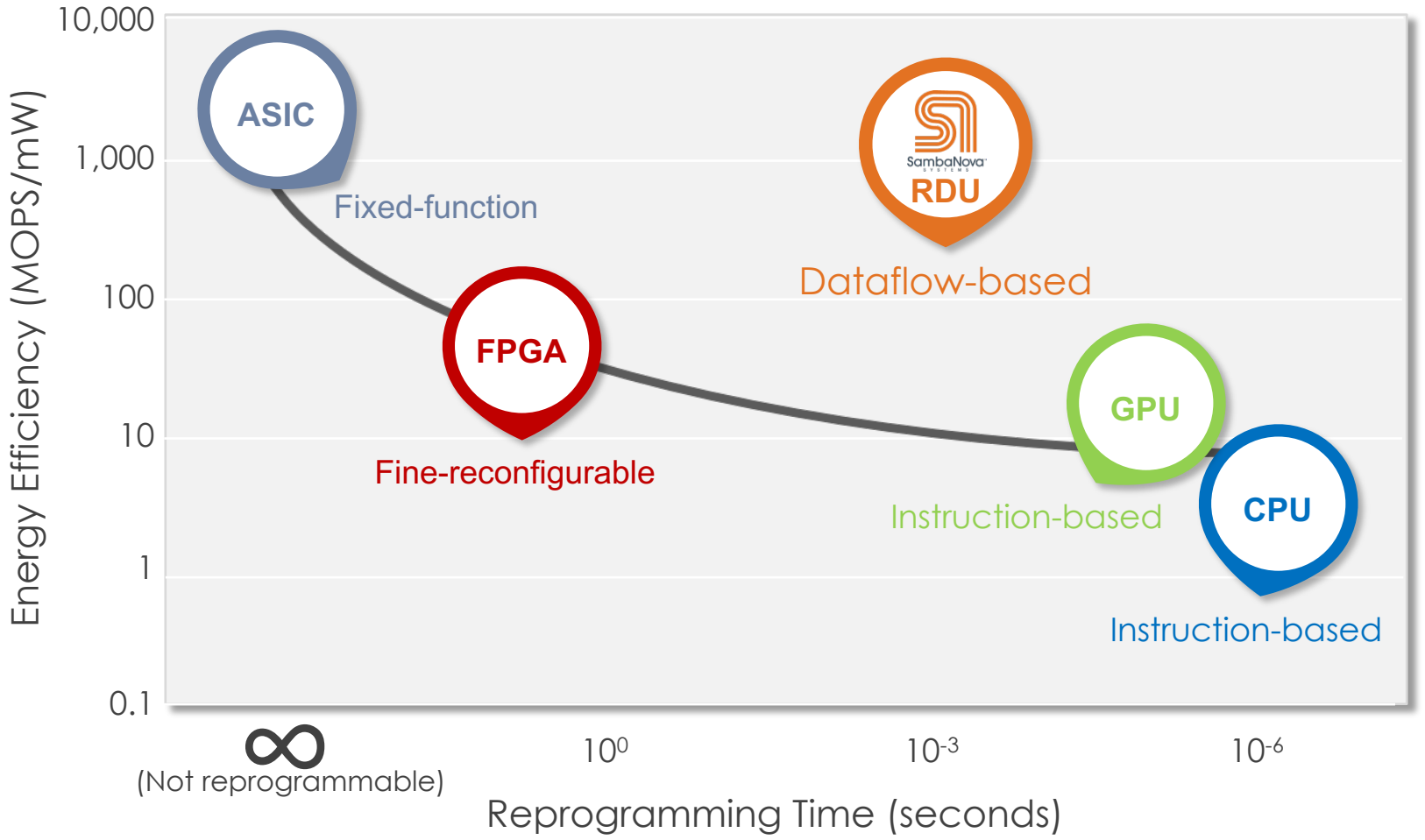


SambaFlow Produces Highly Optimized Spatial Mappings



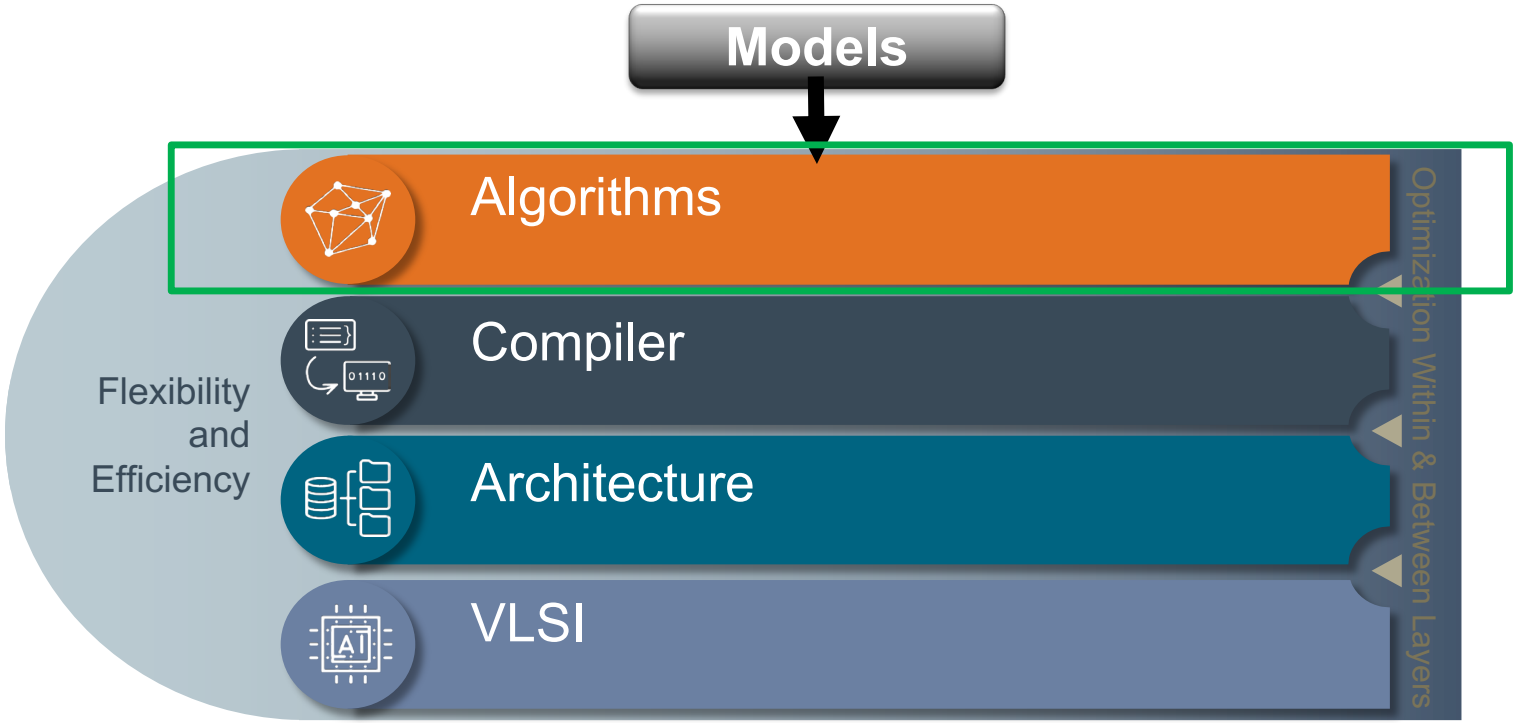
Uncompromised Programmability and Efficiency

Breaking out of the programmability vs. efficiency tradeoff curve



The SambaNova Systems Advantage

Achieve low time-to-accuracy



High model accuracy

Part 2. High model accuracy:

+ Pure 16-bit FPU training

Low Precision (< 32-bit) Training

Binarized Neural Networks: Training Neural Networks with Weights and Activations Constrained to +1 or −1

Matthieu Courbariaux[*1] MATTHIEU.COURBARIAUX@GMAIL.COM
Itay Hubara[*2] ITAYHUBARA@GMAIL.COM
Daniel Soudry[3] DANIEL.SOUDRY@GMAIL.COM
Ran El-Yaniv[2] RANI@CS.TECHNION.AC.IL
Yoshua Bengio[1,4] YOSHUA.UMONTREAL@GMAIL.COM

[1]Université de Montréal
[2]Technion - Israel Institute of Technology
[3]Columbia University
[4]CIFAR Senior Fellow
*Indicates equal contribution. Ordering determined by coin flip.

Recurrent Neural Networks With Limited Numerical Precision

Joachim Ott*, Zhouhan Lin‡,Ying Zhang‡, Shih-Chii Liu*, Yoshua Bengio‡†
*Institute of Neuroinformatics, University of Zurich and ETH Zurich
ottj@ethz.ch, shih@ini.ethz.ch
‡Département d'informatique et de recherche opérationnelle, Université de Montréal
†CIFAR Senior Fellow
{zhouhan.lin, ying.zhang}@umontreal.ca

Training Deep Neural Networks with 8-bit Floating Point Numbers

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen and Kailash Gopalakrishnan
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{nwang, choij, danbrand, cchen, kailash}@us.ibm.com

Higher system efficiency, minimal impact on acc. for **specific models**

Efficiency of Low Precision Floating-point-units (16 vs. 32-bit)[1,2]

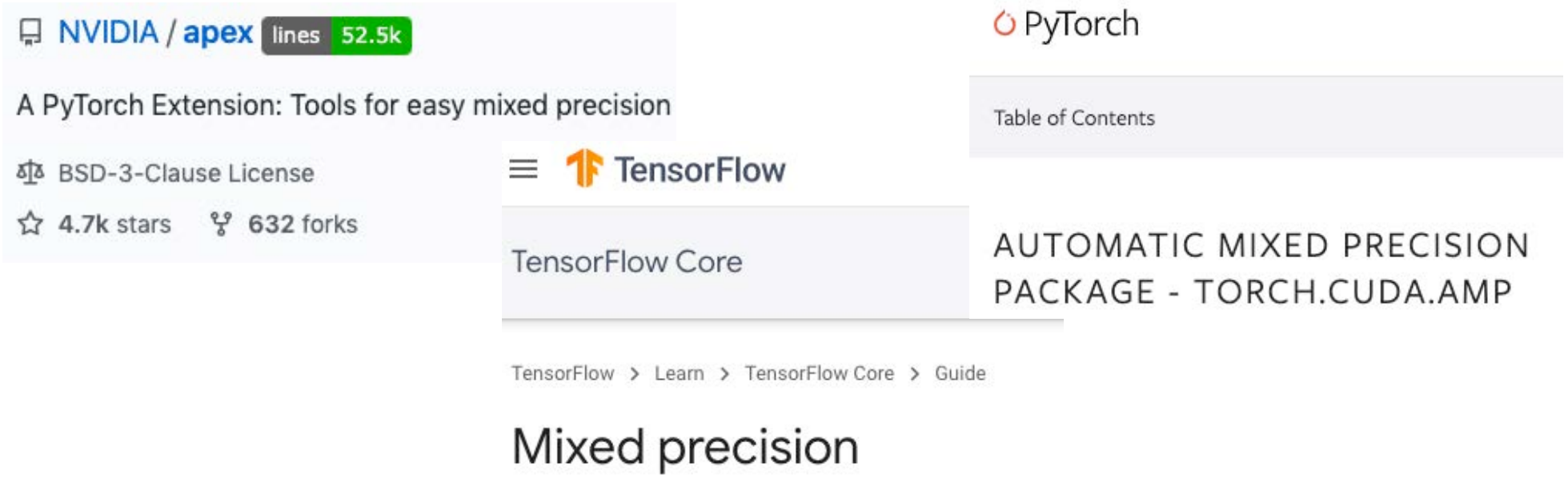
1.5X lower chip area

3X higher energy efficiency

1.5X higher throughput

1. Horowitz. ISSCC 2014
2. Galal et. al. ISCA 2013

Mixed Precision for *Generic* DL Training (16 + 32 bits FPU)



Illusion:

16-bit FPU alone is not enough to maximize model acc.

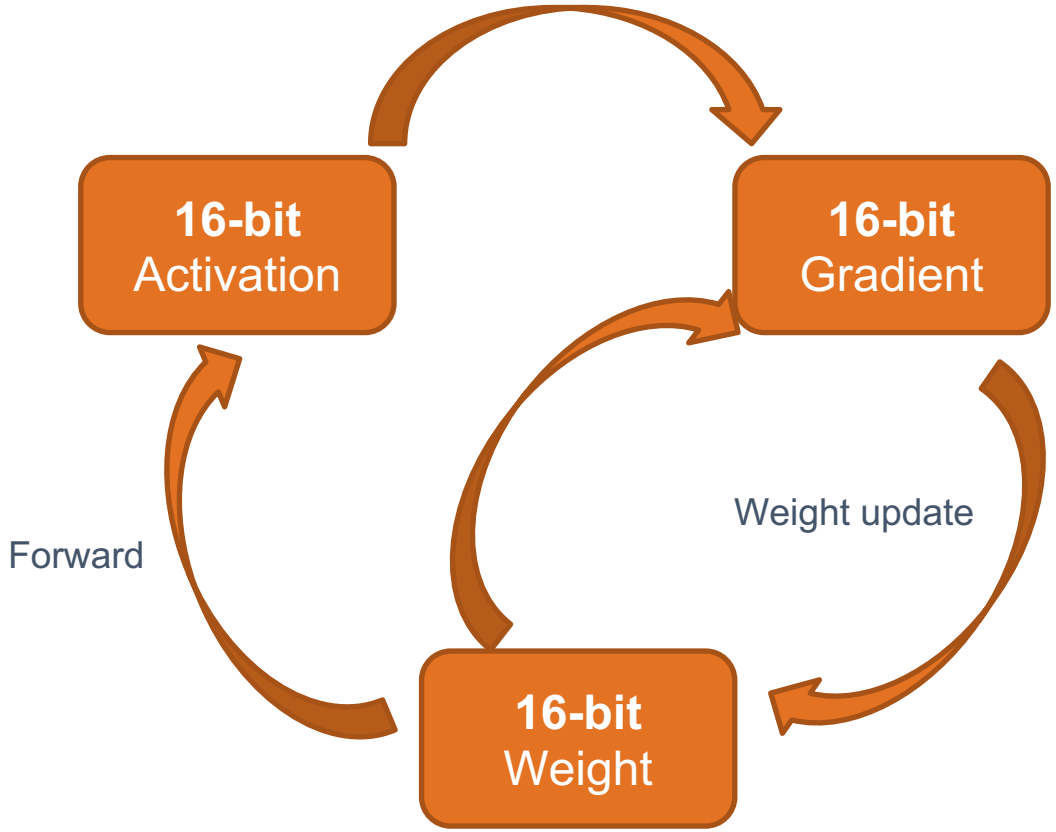
Can we support ***only 16-bit FPU*** *on accelerators*

&

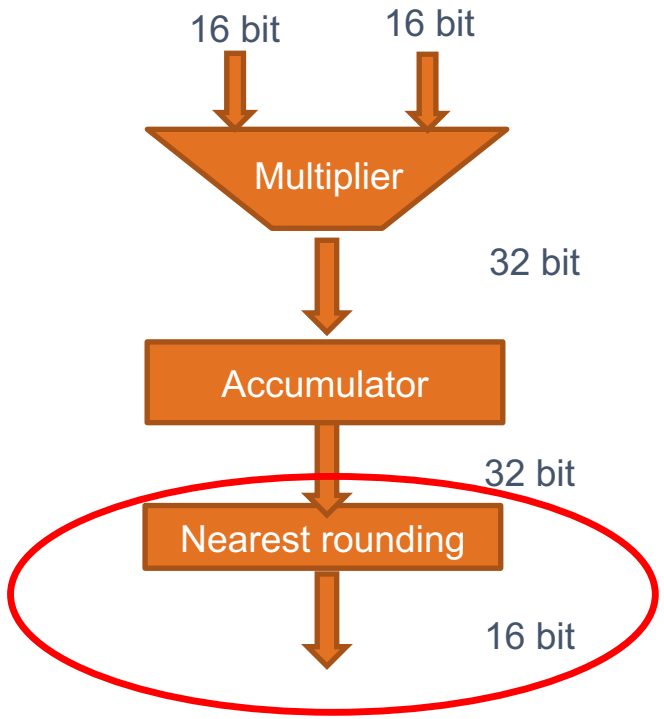
achieve model acc. matching 32-bit training?

Pure 16-bit (BFloat16) FPU Training

Data Flows

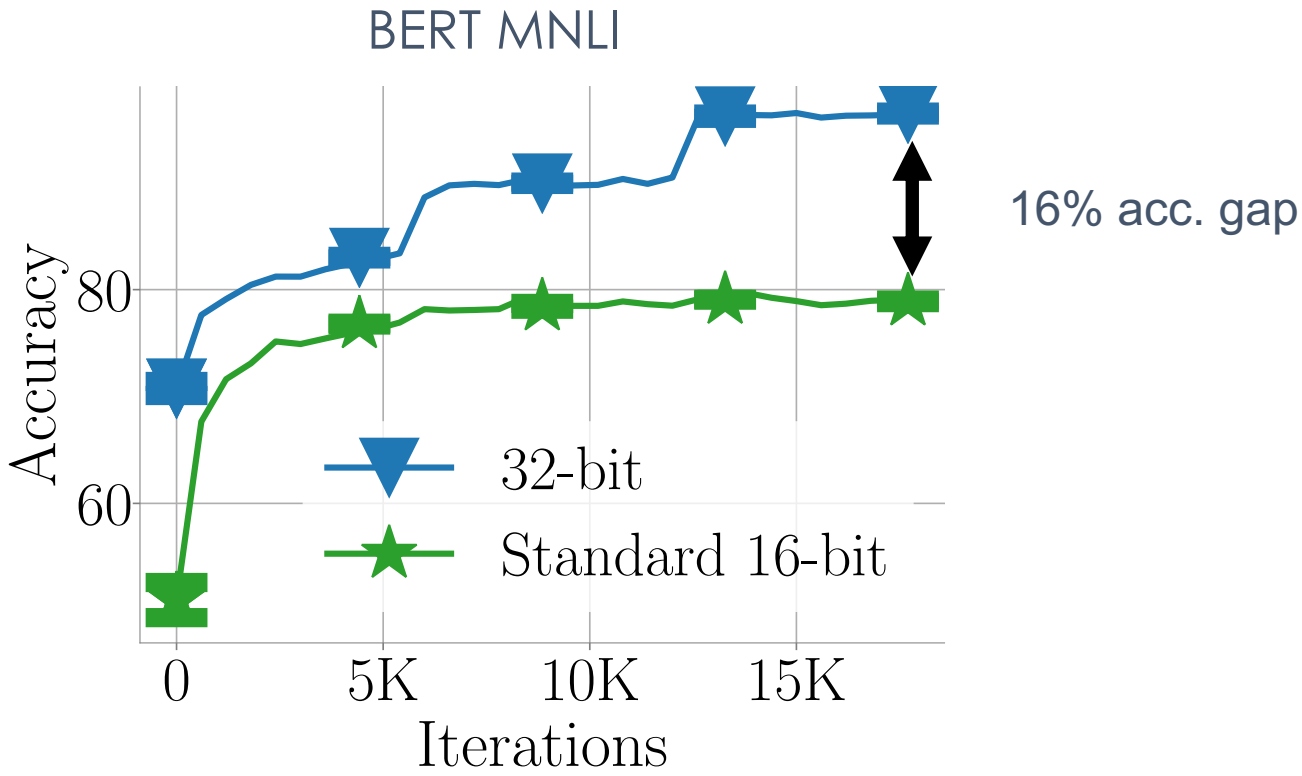


Multiply-Accumulation Units



Primary source of numerical error

The Accuracy Challenge



Standard 16-bit FPU training degrades model accuracy

The Devil: Nearest Rounding(NR) for Model Weight Updates

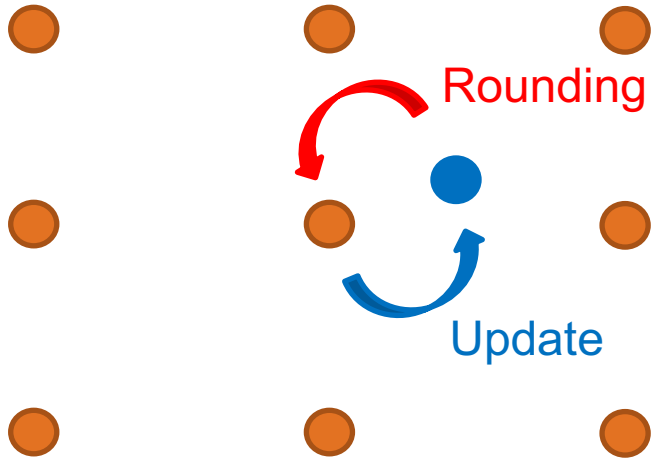
$$\boldsymbol{w}_{t+1} = \boldsymbol{Q}\left(\boldsymbol{w}_t - \alpha \nabla f_{\sigma(t)}(\boldsymbol{w}_t)\right)$$

Model weight

Nearest Rounding

Minibatch

Model weights halt when updates becomes small



The Devil: Nearest Rounding (NR) for Model Weight Updates

Theory sketch for least-squares regression

Machine precision

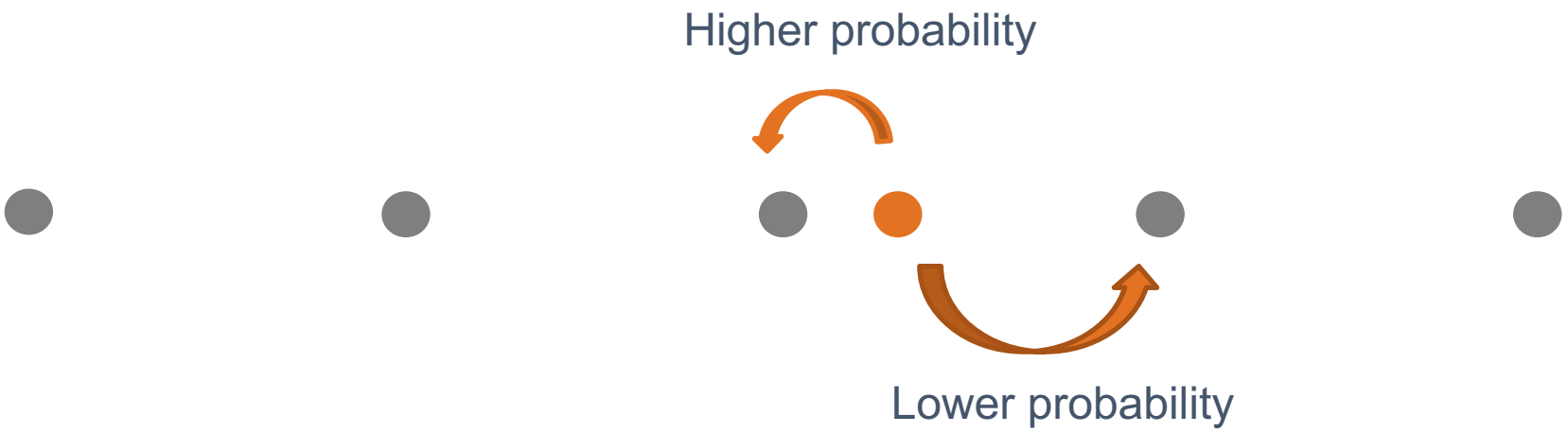
$$\|\boldsymbol{w}_t - \boldsymbol{w}^*\| \geq \mathcal{O}\left(\epsilon \cdot \min_j \left|w_j^*\right|\right)$$

Optimal solution

j-th dim of the optimal solution

Inaccurate weight update fundamentally degrades convergence

Stochastic Rounding to the Rescue

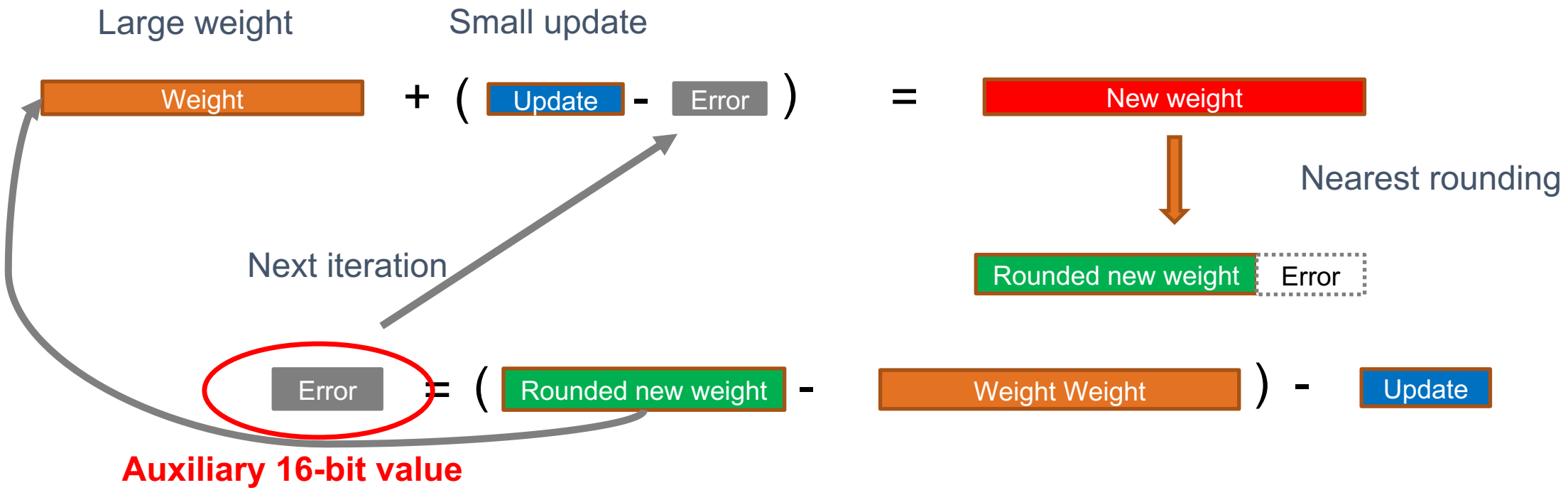


Intuition

The expectation of unbiased estimates is as accurate as weights w/o rounding

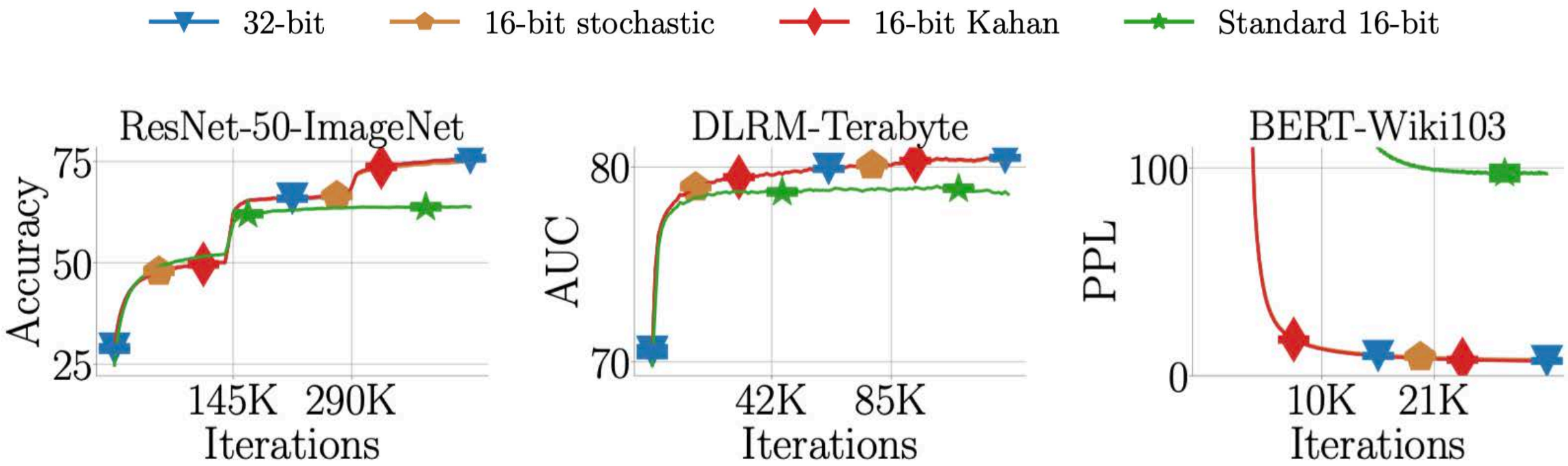
Kahan Summation as Alternative Enhancement

Auxiliary 16-bit values to track and correct weight update errors from NR



Experiment:

Pure 16-bit training can match 32-bit training in model acc.



In submission ‘20

Summary

With support for

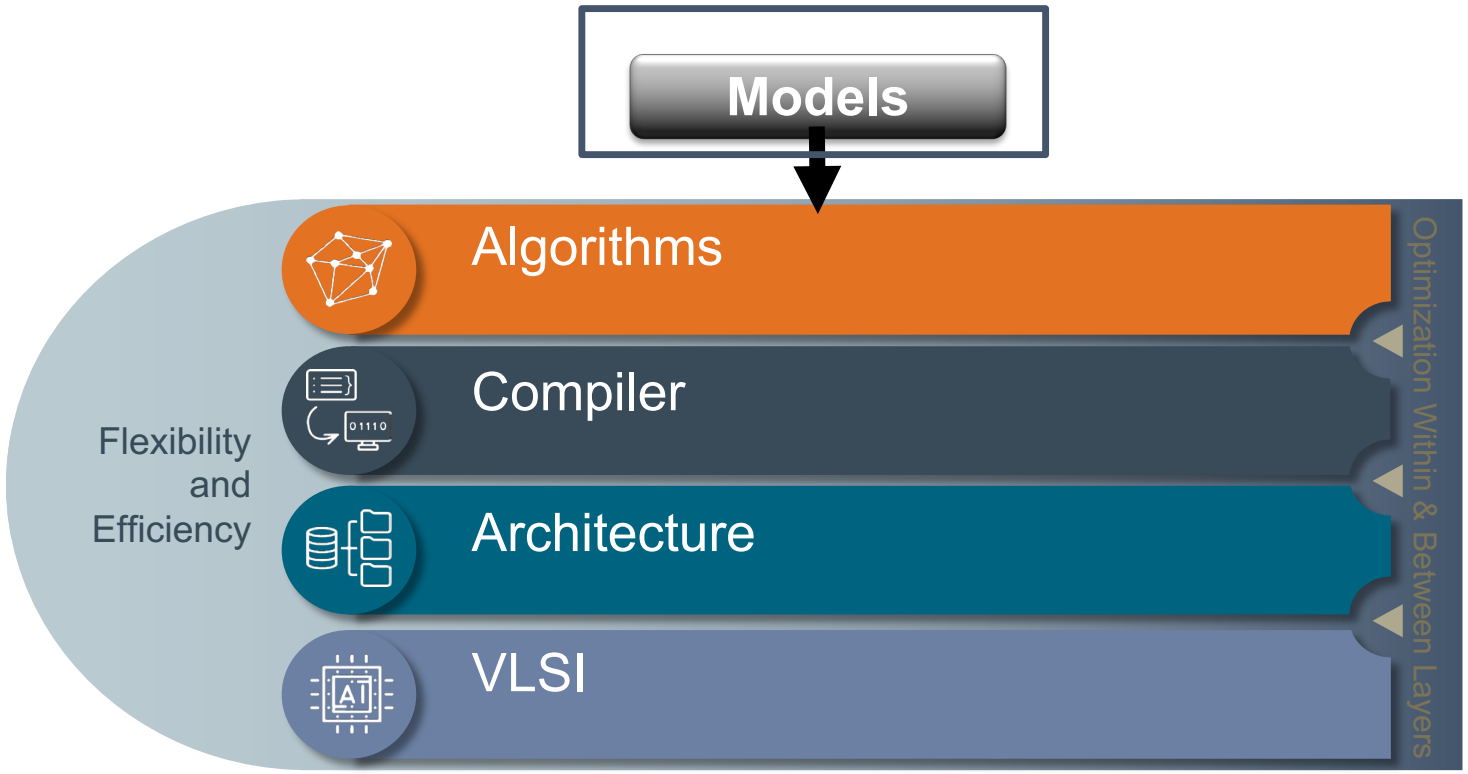
&

Stochastic rounding

Kahan summation

Accelerators with only 16-bit compute units can match acc. of 32-bit training

The SambaNova Systems Advantage



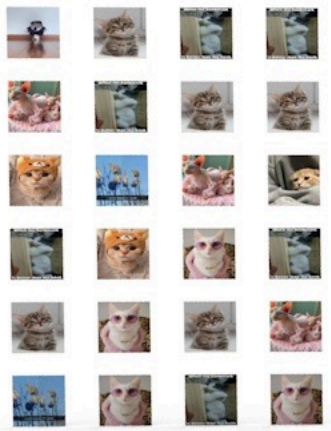
Application innovations

Part 3. Model Innovations:

Powered by our architecture and algorithm

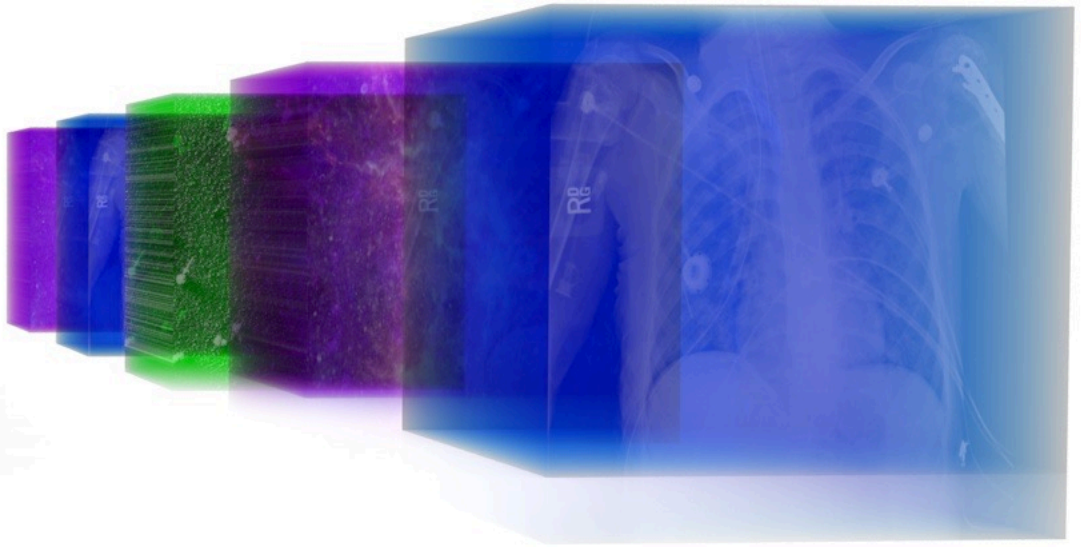
Computer Vison

Evolution of high-resolution Deep Learning



Low-resolution
(e.g. cats)

4k images
(e.g. Autonomous driving)

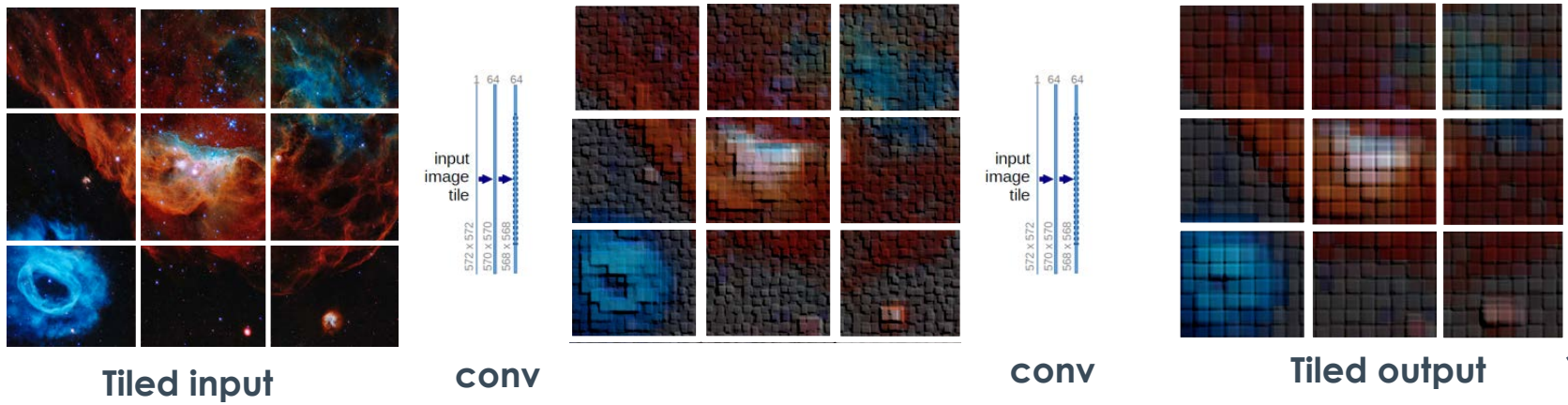


50k x 50k
(e.g. astronomy,
medical imaging, virus, …

No Compromise High-Res Segmentation

Classic: chop image into sub-images

Loses information in output!

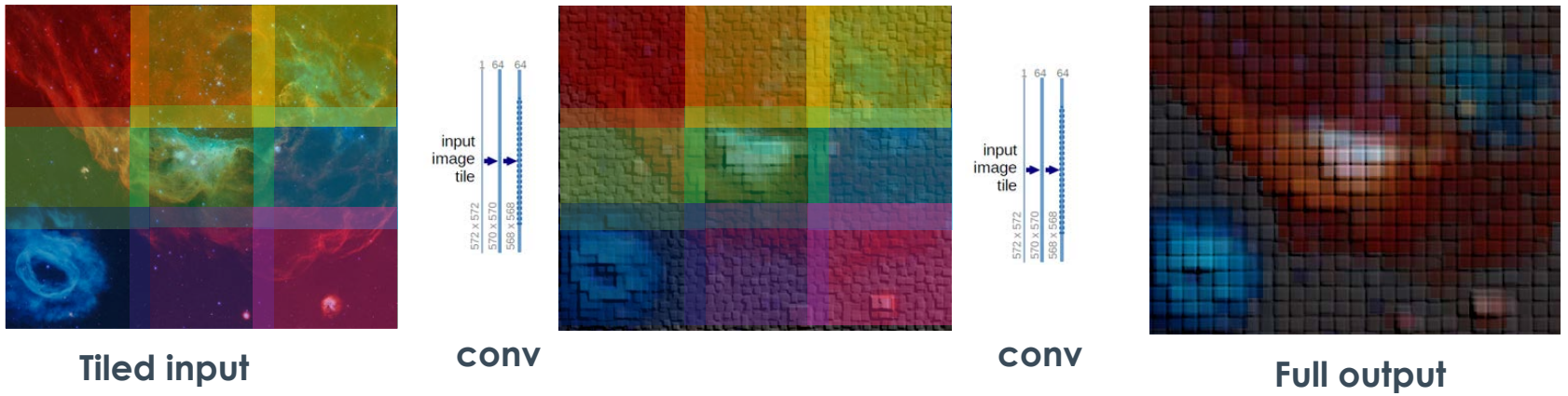


83.1%

SOTA IOU

89.6%

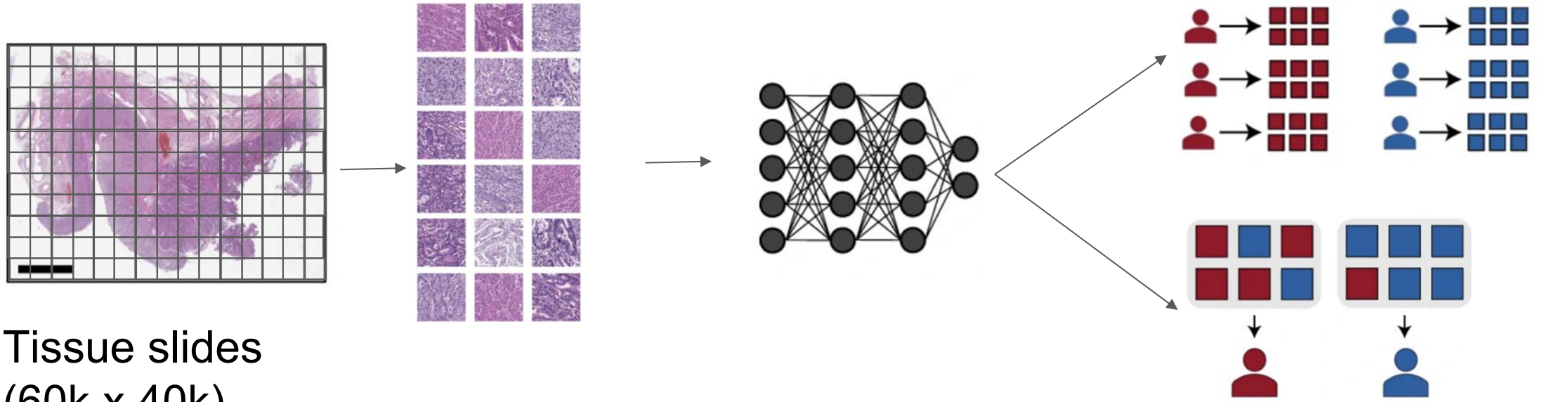
SambaNova: Full image processing



Training w/o information loss from full-image processing

High-Res Pathology with Slide-level Label (TCGA)

Train with Patch label = slide label

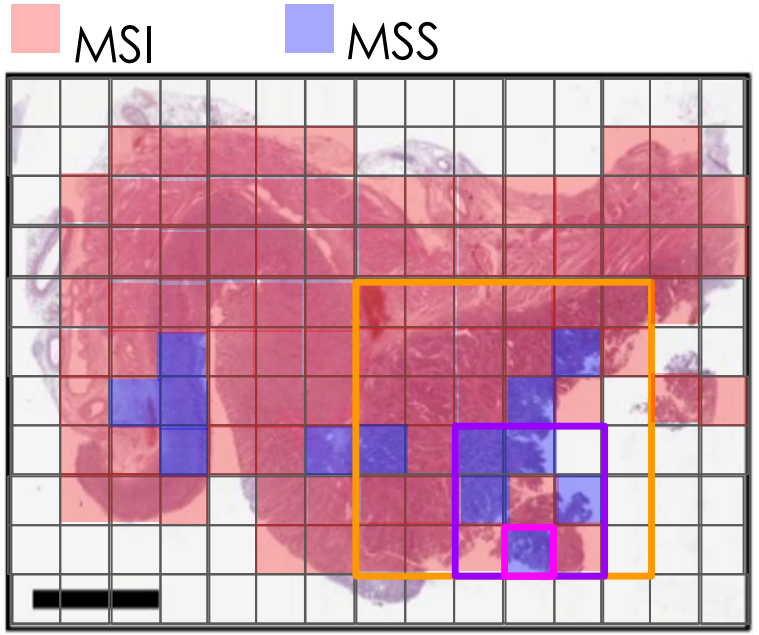


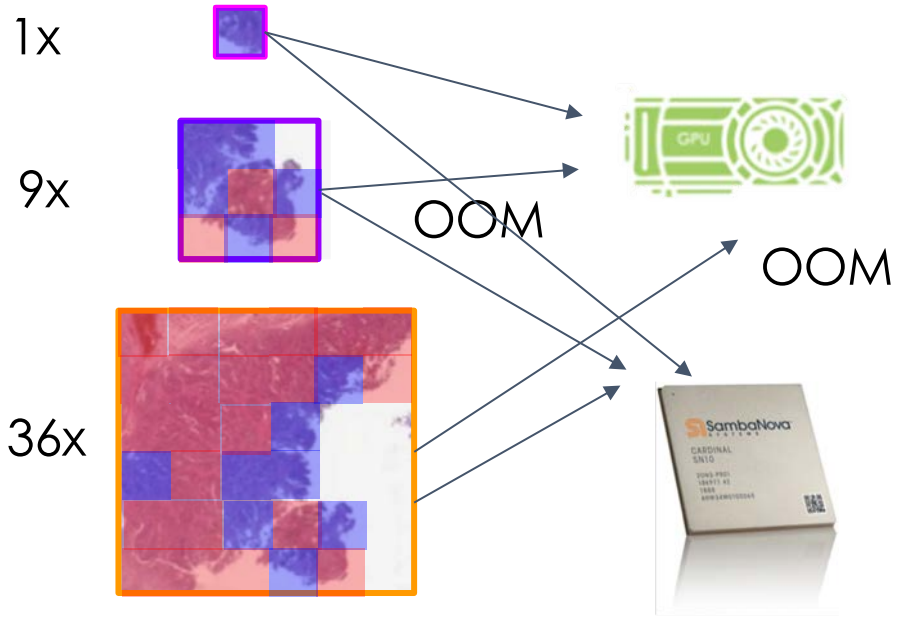
Tissue slides
(60k x 40k)

Voting at slide level

Noisy patches limits model accuracy

High-Res Pathology with Slide-level label (TCGA)

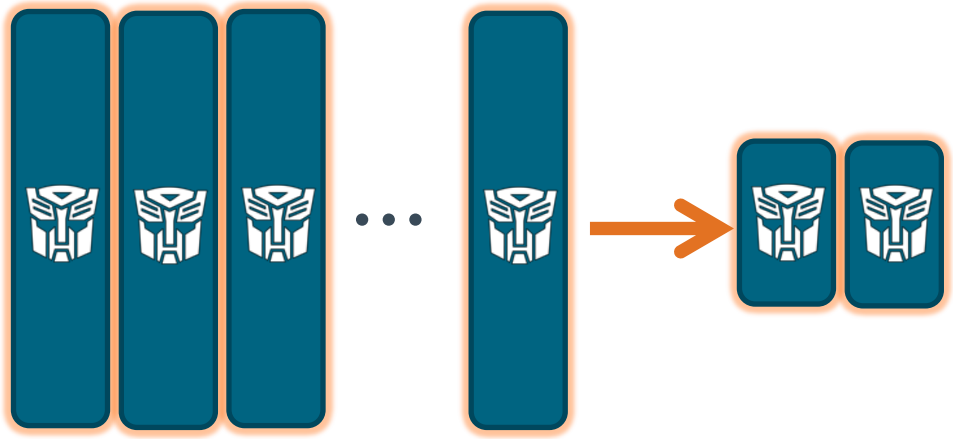




16X larger patches → 6 Pt higher AUC

Natural Language Processing

Breakthrough efficiency in NLP model online deployment



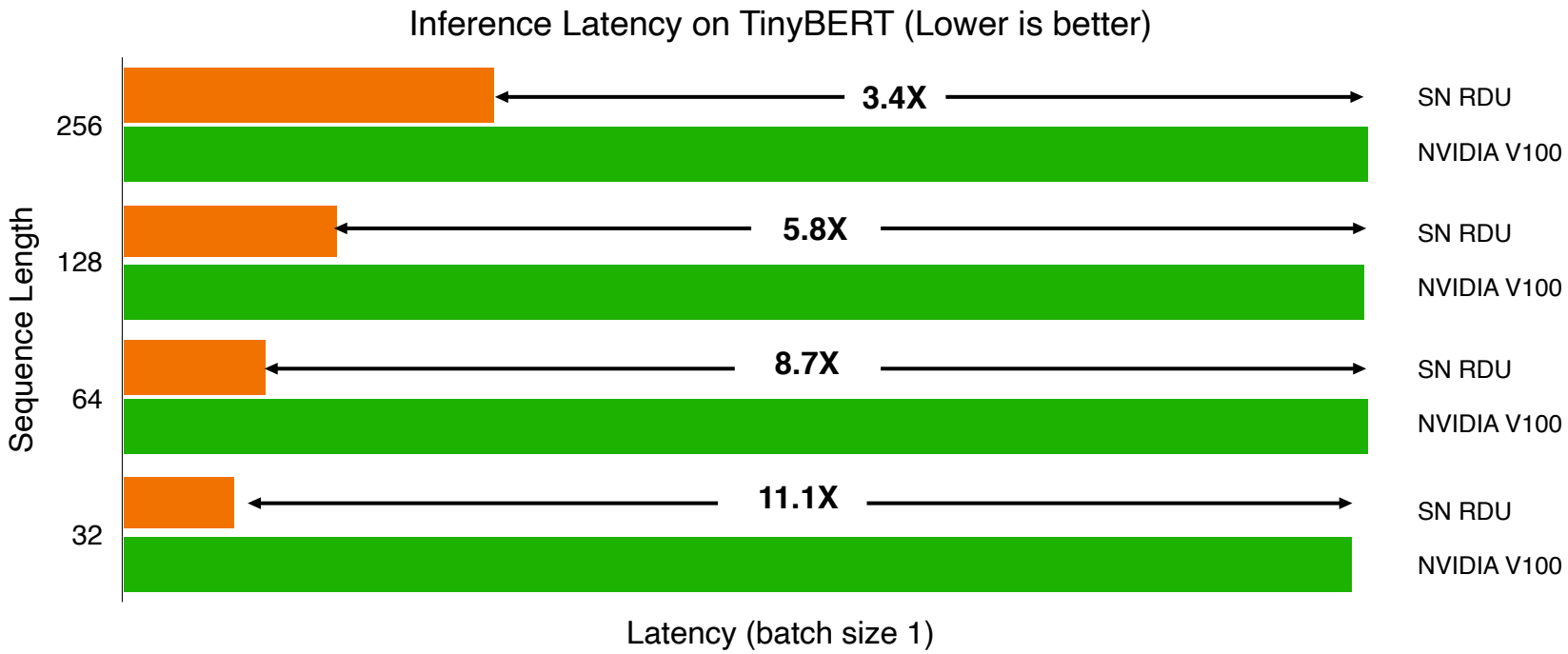
Distilled tiny Bert model

+

Siri, what is the whether in SF?

Short sequence input

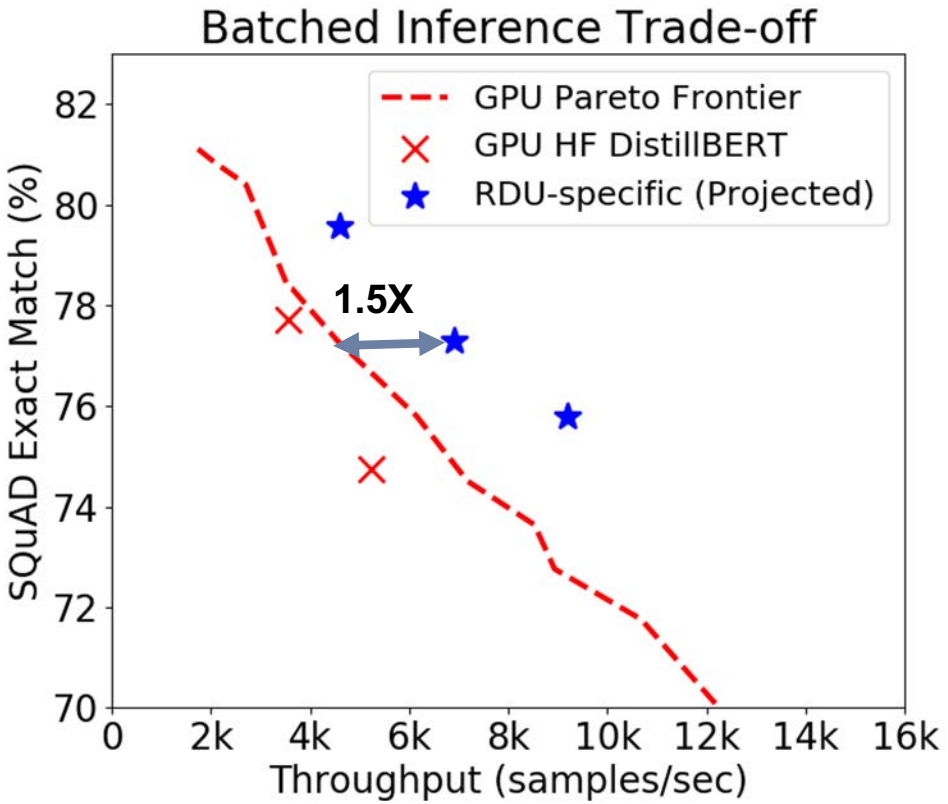
Breakthrough Efficiency in NLP Model Online Deployment

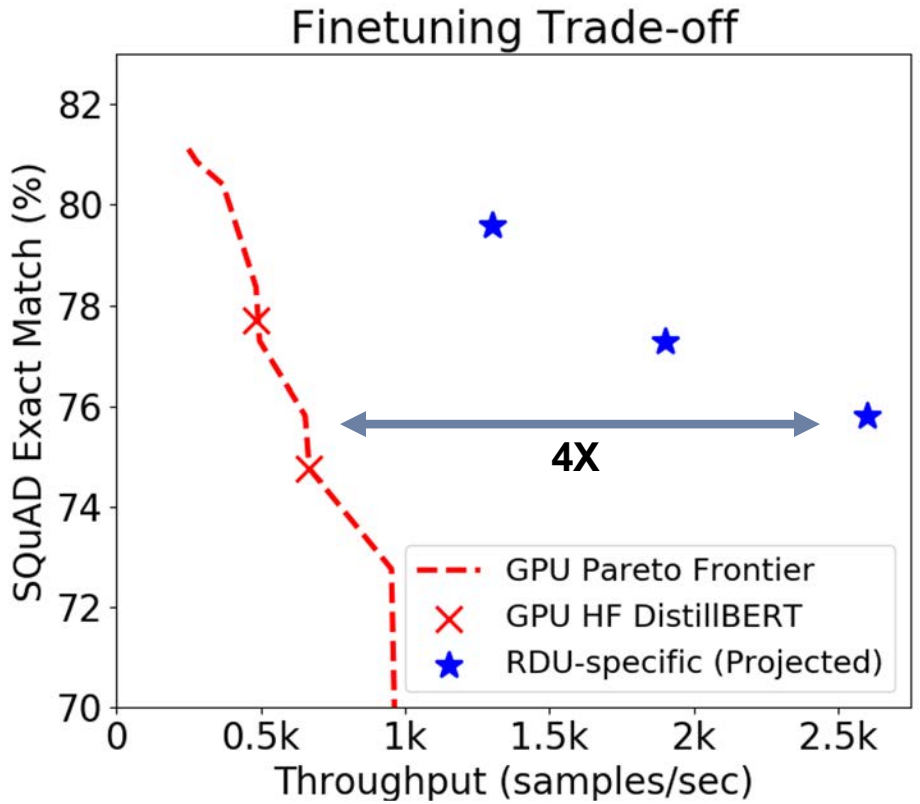


Enable up to 11X speedup for online training and inference

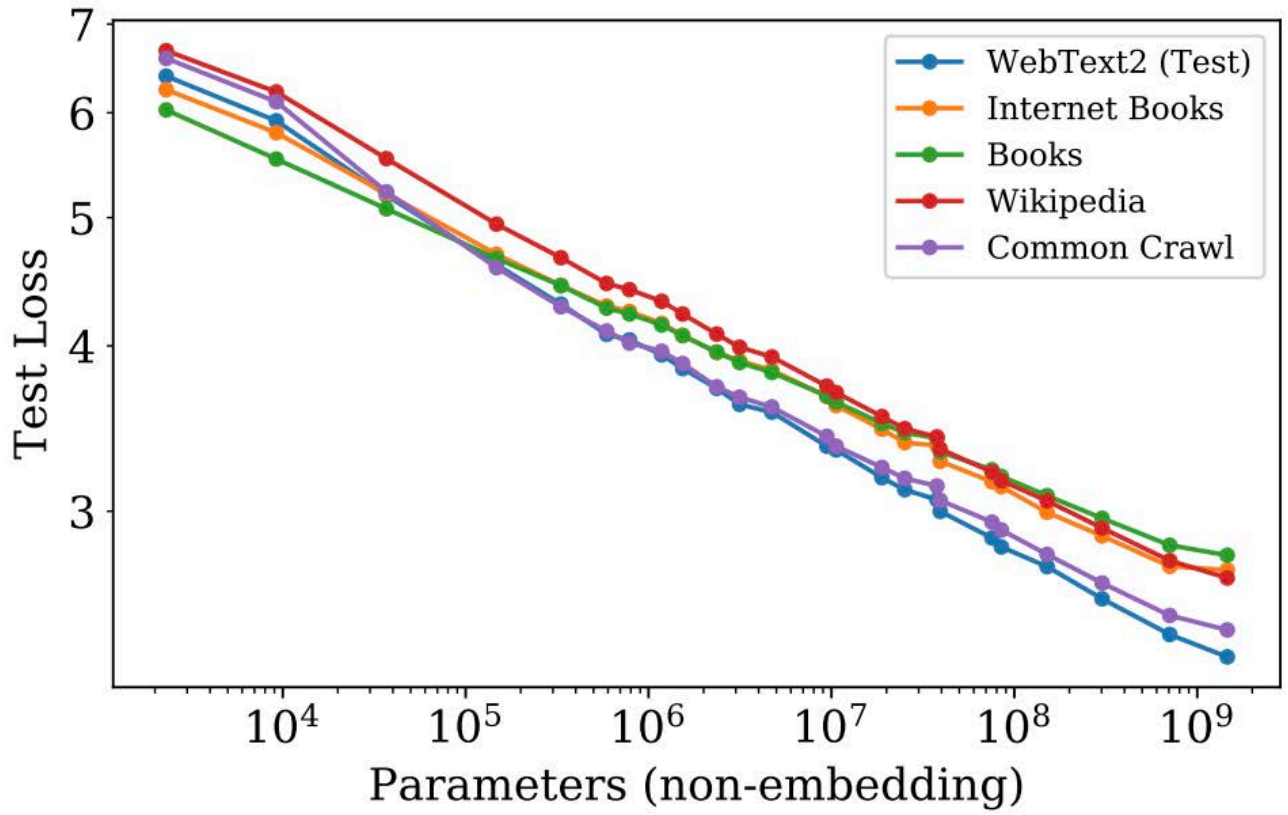
Expand Pareto Frontier beyond GPU

Higher acc. at higher **downstream training** and **inference** throughput **(RDU/CPU…)**



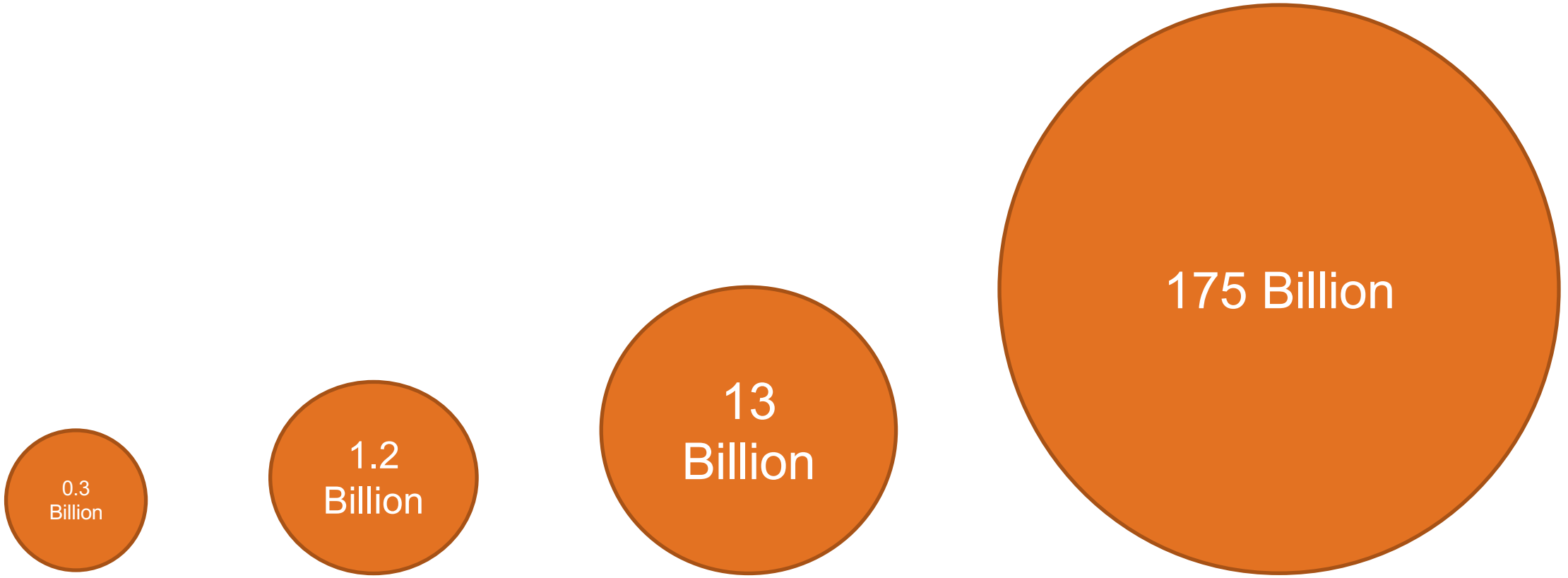


Scaling Laws for Neural Language Models



Application accuracy improves as the size of the language model increases

GPT Family



Train Large Language Models, Without Code Changes

1T parameter NLP training with a small footprint and programming ease

Model size	Attention heads	Hidden size	Number of layers	Number of parameters (billion)	Model-parallel size
1.7B	24	2304	24	1.7	1
3.6B	32	3072	30	3.6	2
7.5B	32	4096	36	7.5	4
18B	48	6144	40	18.4	8
39B	64	8192	48	39.1	16
76B	80	10240	60	76.1	32
145B	96	12288	80	145.6	64
310B	128	16384	96	310.1	128
530B	128	20480	105	529.6	280
1T	160	25600	128	1008.0	512

Large Kernels
- To Hide Communication Costs
- Statistically Nonoptimal

Complex System Engineering To Enable Model Architecture Exploration

PyTorch

Out-of-Box Models
- Huggingface Models
- Write yours in Pytorch

Developer Efficiency
- Focus on ML problems instead of System Engineering

High Accuracy Models
- No Compromise on Model Architecture required to hide System Deficiencies

https://arxiv.org/pdf/2104.04473.pdf

Enabling Large Model Architectures With a Single System

Order of magnitude performance improvement, an order of magnitude fewer systems

64 DGX-2
1,024 V100s
32 TB HBM
16 racks

8 RDU,
12 TB DRAM,
¼ rack

1 DataScale system

"One Model" 1Trillion Params in a Single System: **Same** Programming Model

Conclusion