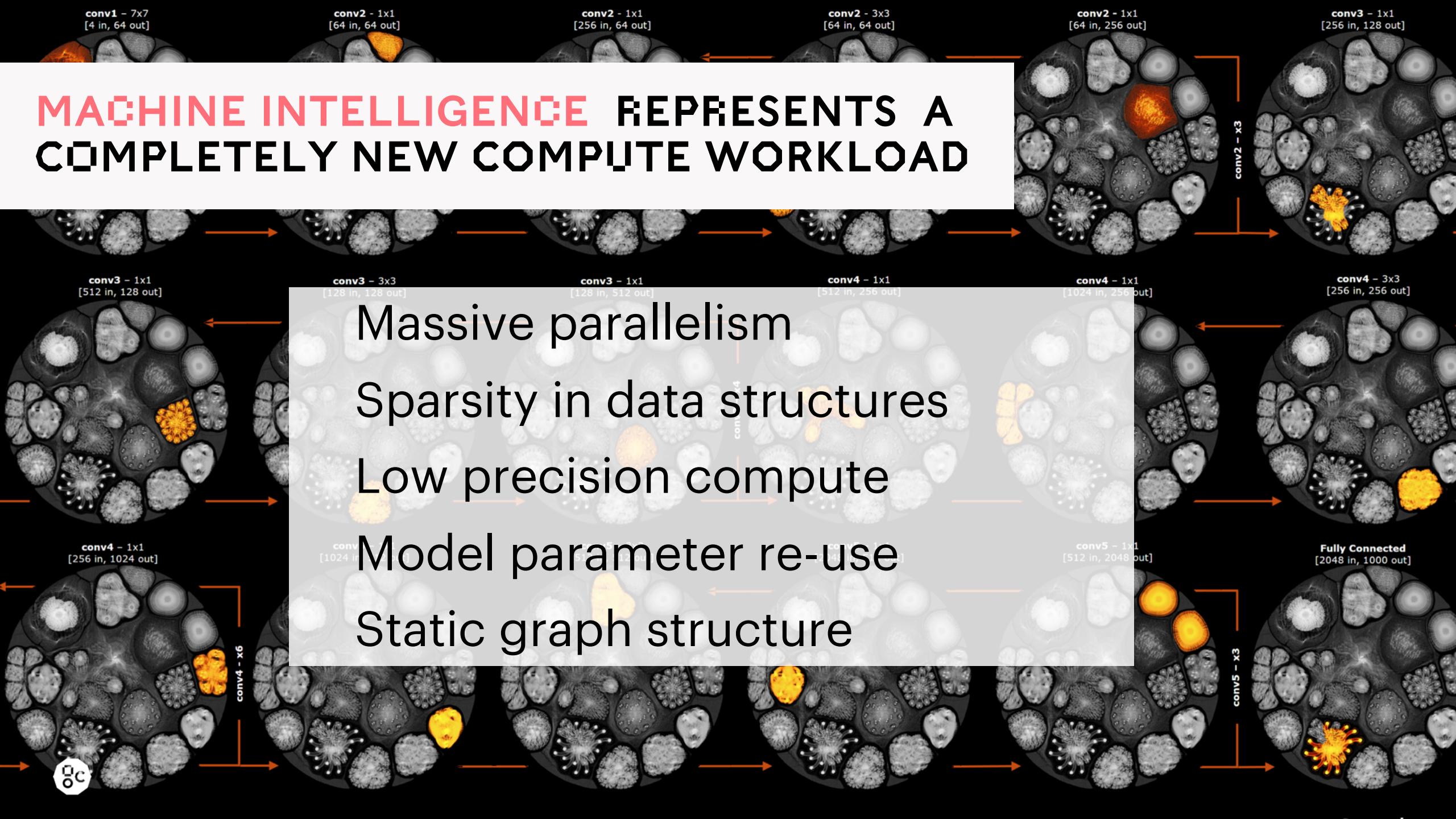
NEW APPROACHES FOR ENABLING MACHINE LEARNING INNOVATION

Alexander Tsyplikhin, Senior AI Engineer

GRAPHCORE

Neural network visualization from POPLAR™

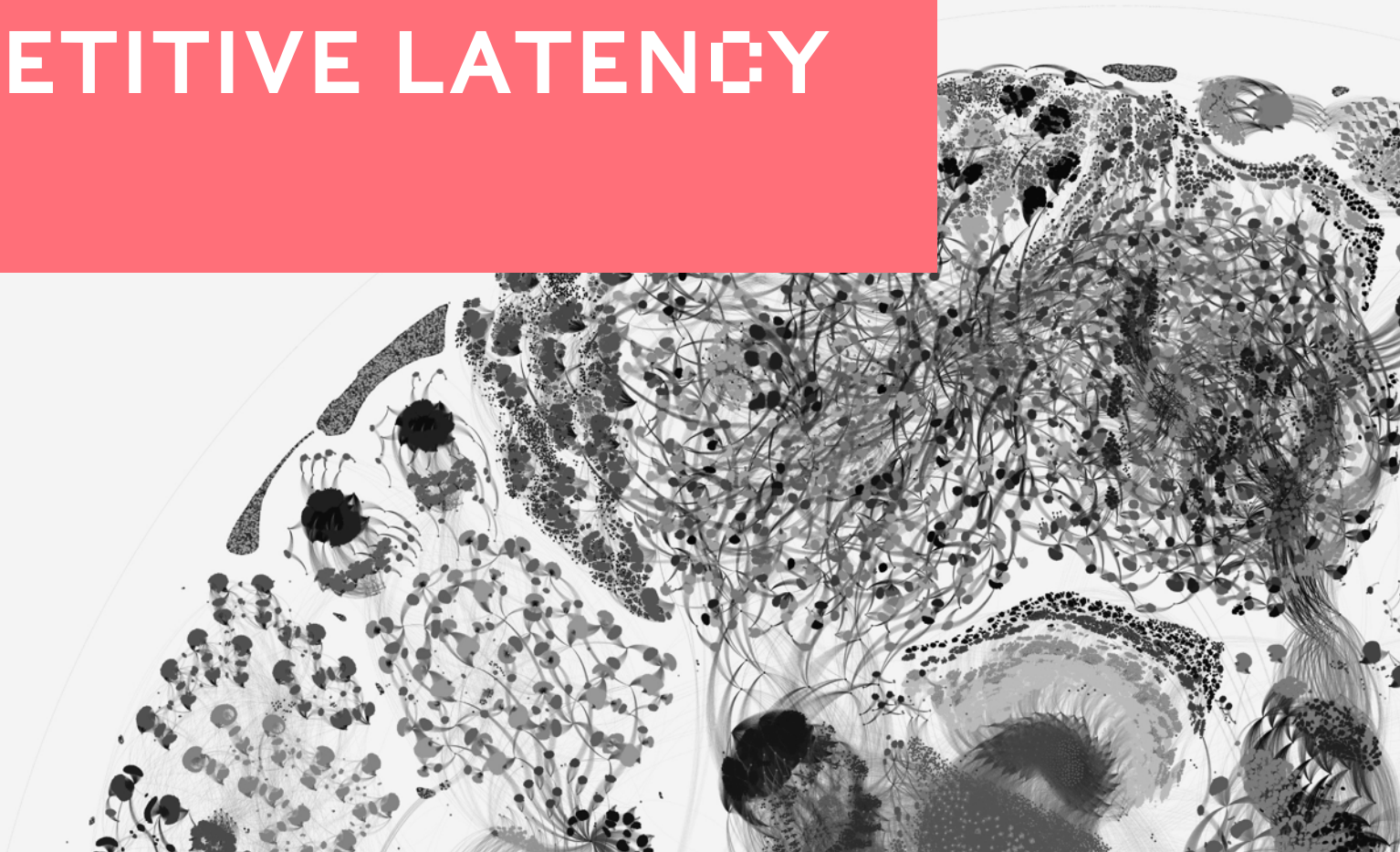


MACHINE INTELLIGENCE REPRESENTS A COMPLETELY NEW COMPUTE WORKLOAD

- Massive parallelism
- Sparsity in data structures
- Low precision compute
- Model parameter re-use
- Static graph structure

IN FINANCIAL MODELLING

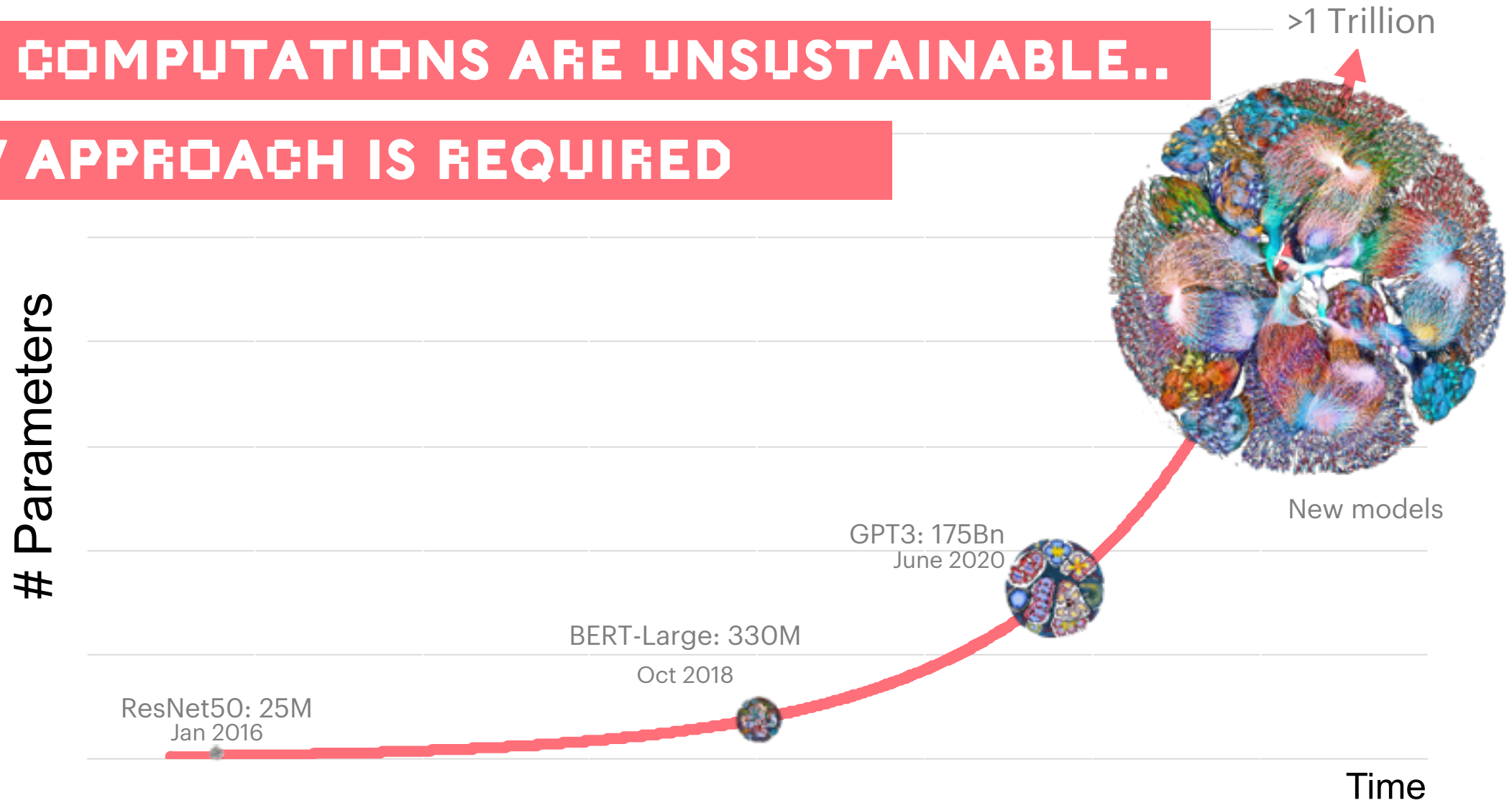
MODELS ARE OFTEN OVERSIMPLIFIED TO ACHIEVE COMPETITIVE LATENCY OR FIGHT NOISE



MODEL SIZE GROWING MASSIVELY

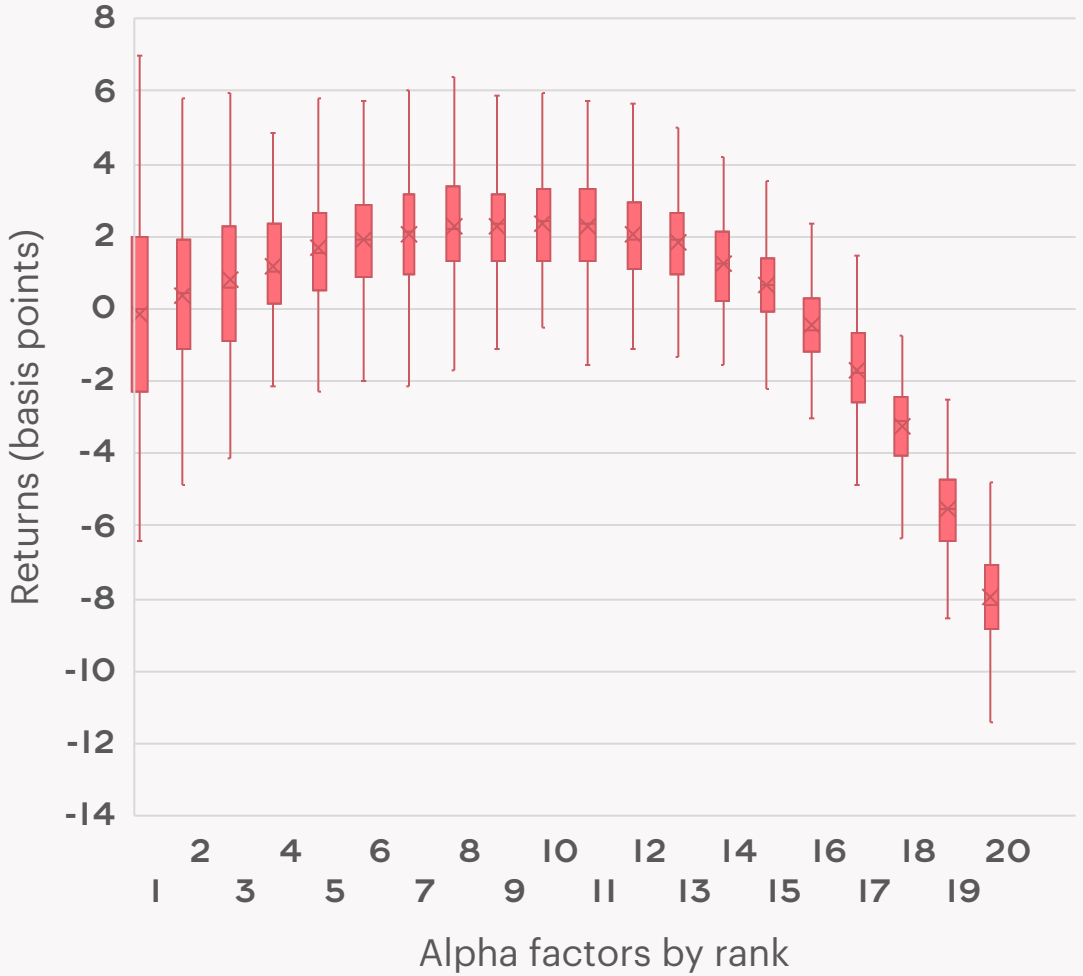
DENSE COMPUTATIONS ARE UNSUSTAINABLE..

A NEW APPROACH IS REQUIRED



LET’S CONSIDER 3 EXAMPLES

ALPHA ESTIMATION



Traditional method: linear regression

- Limited efficiency on real data

Markov Chain Monte Carlo (MCMC):

predictions are distributions,

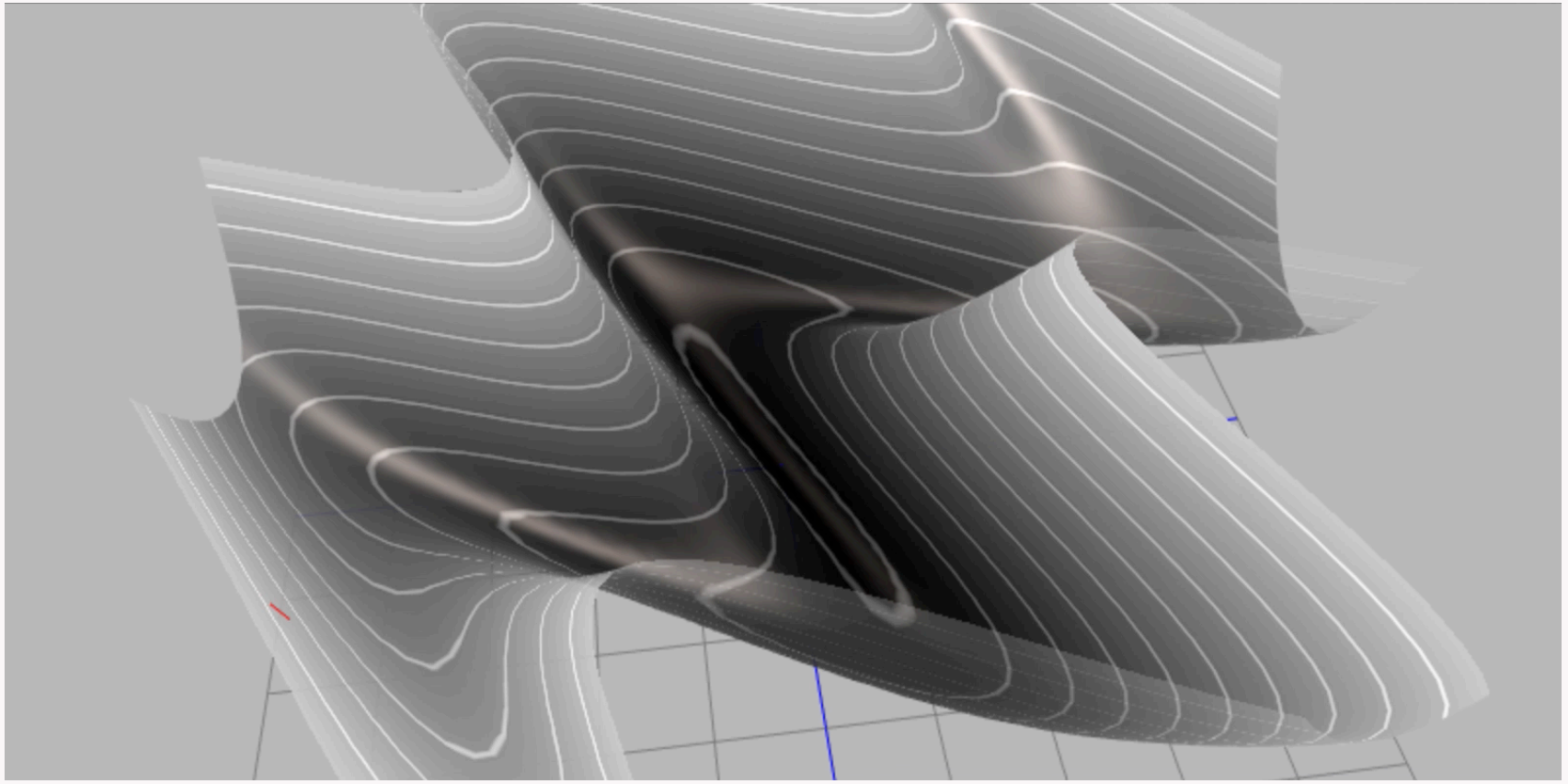
not just point values

This helps to:

- Avoid overfitting
- Deal with noisy data
- Handle non-linearities

HAMILTONIAN MONTE CARLO

A hockey puck sliding over a surface without friction, being stopped at some point in time and then kicked again in a random direction.

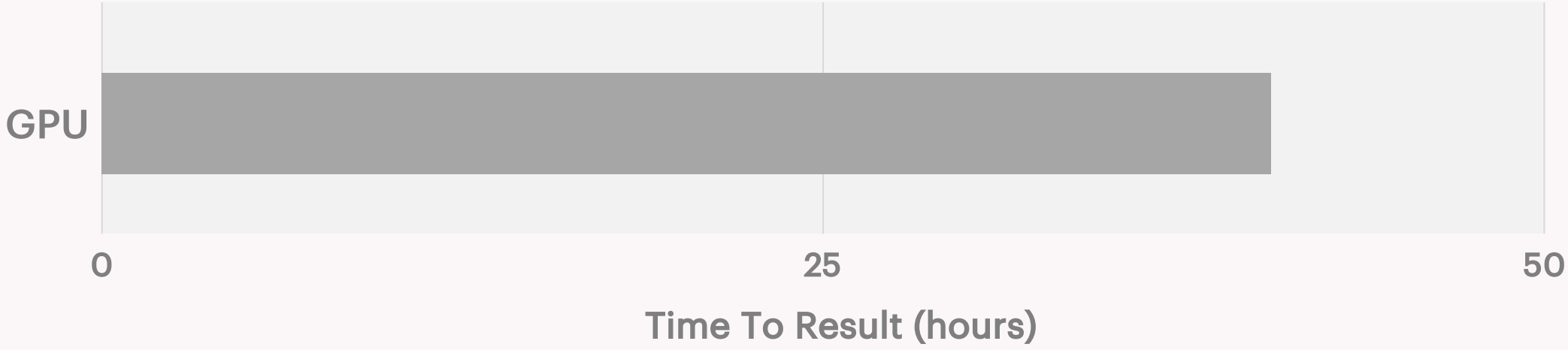


http://arogozhnikov.github.io/2016/12/19/markov_chain_monte_carlo.html

THE PROBLEM WITH MCMC

MCMC has long been considered too computationally expensive.

Runs for hours or days. Not practical using traditional technology.



NOT A STAC BENCHMARK

NOTES:
Markov Chain Monte Carlo – Probabilistic model with TensorFlow Probability, representative of workload used by Carmot Capital
Neural network with 3 fully-connected layers (num units in 1st layer=40, #dimensions in training set =22, #leapfrog steps=1000, calcs in sliding window=200)
GPU (300W TDP) - 800 samples

LSTM FOR TIME SERIES ANALYSIS

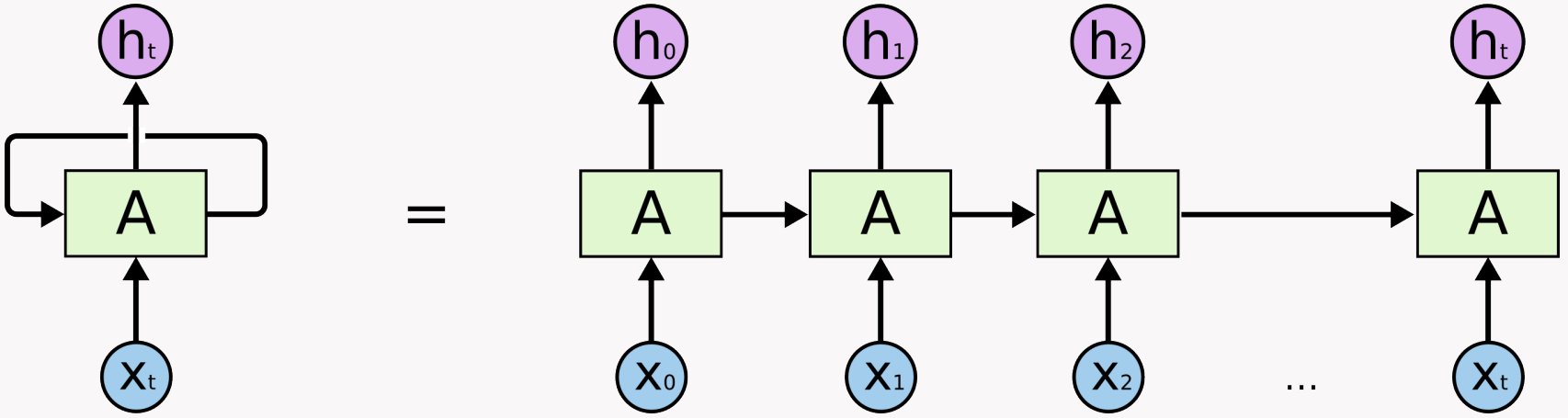
Long short-term memory (LSTM) models:

efficient for sequences of observations

Used for:

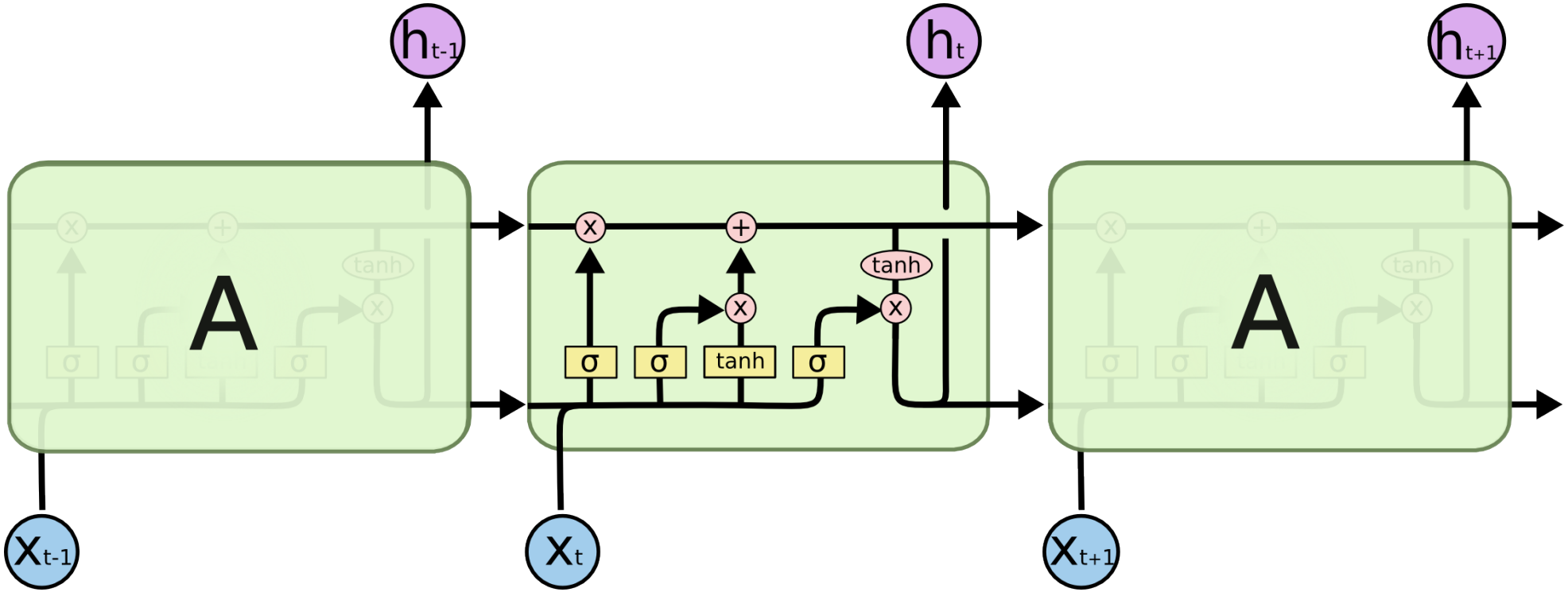
- Financial time series analysis
- Feature extraction
- Alpha estimation
- Fraud detection

RECURRENT NEURAL NETWORK



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM

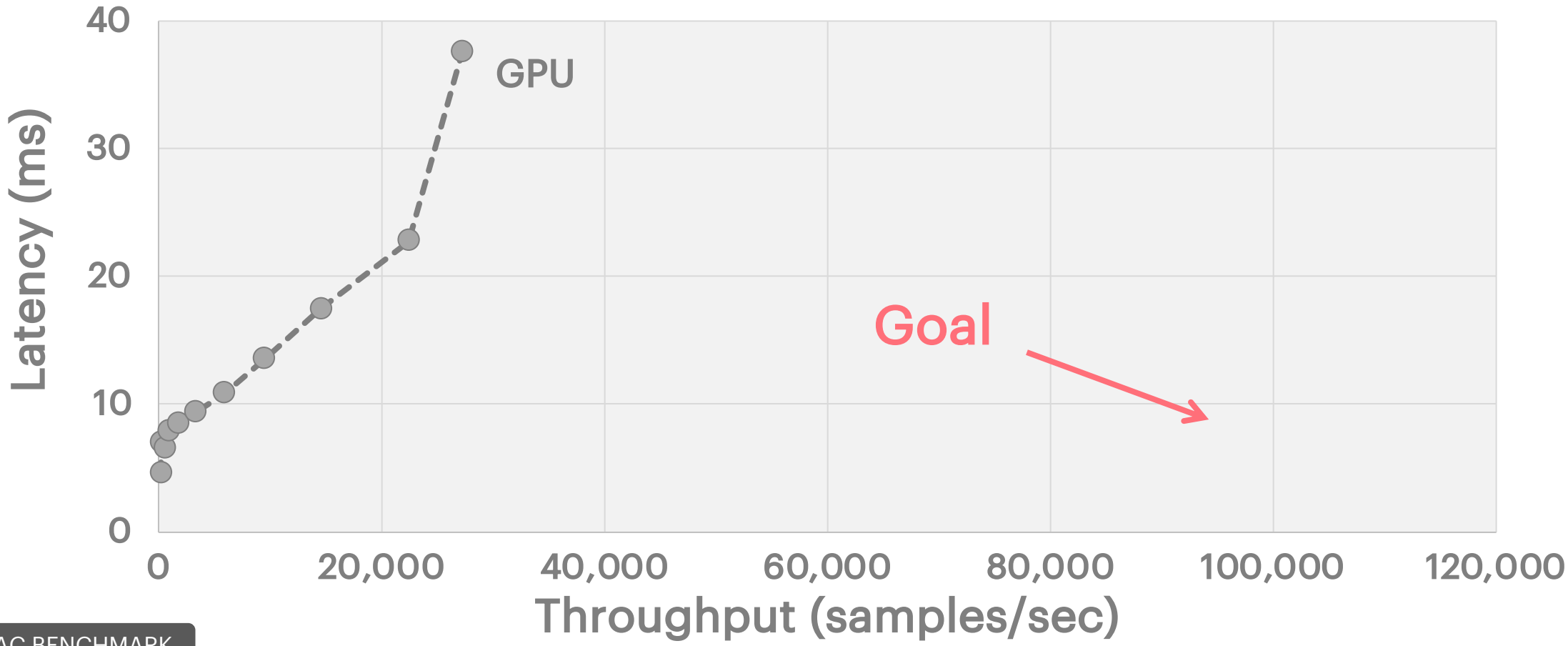


Neural Network Layer
Pointwise Operation
Vector Transfer
Concatenate
Copy

https://colah.github.io/posts/2015-08-Understanding-LSTMs/

THE PROBLEM WITH LSTM

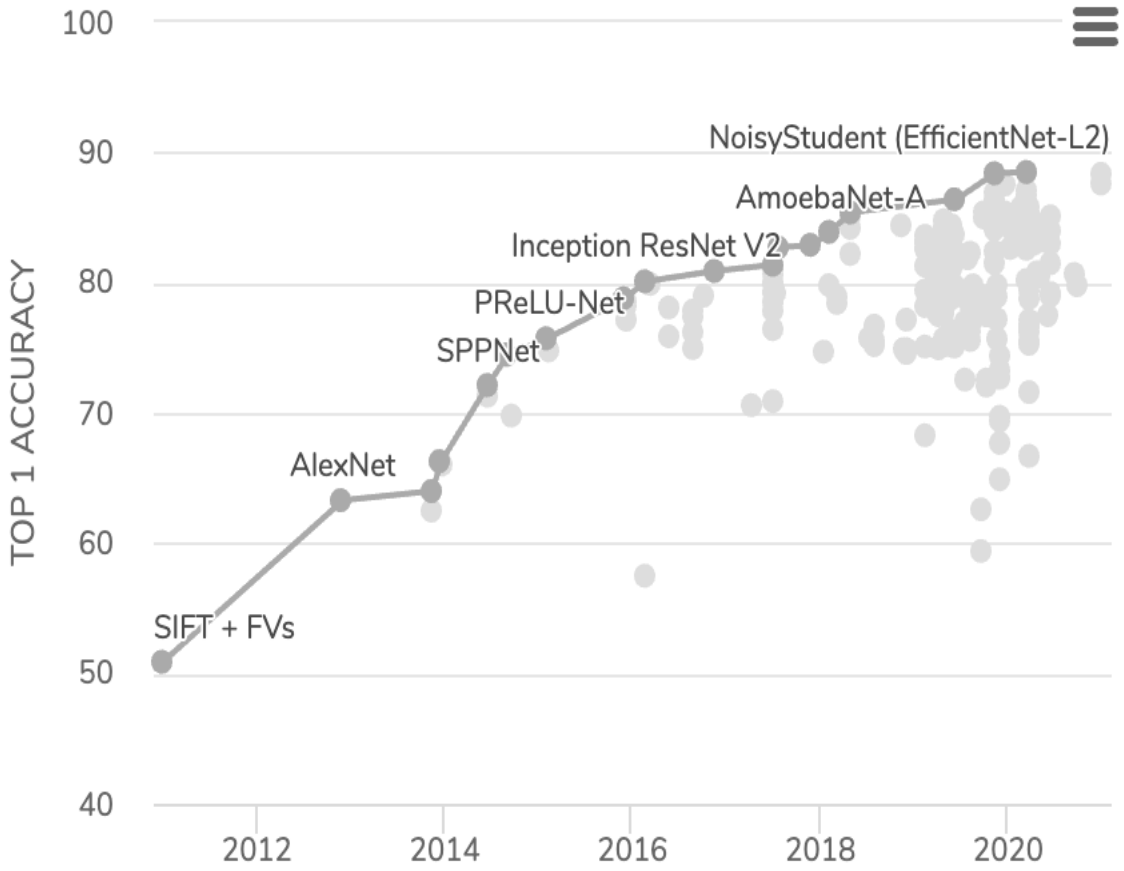
Inference latencies are often too high, or throughput is too low



NOT A STAC BENCHMARK

NOTES:
2 LSTM layers, each with 256 units, 200 time steps, 16 input dimensions, real data, mixed precision
GPU: using TensorFlow with optimizations @ 300W TDP (Batch Sizes upto 1024)

EFFICIENTNET FOR IMAGE ANALYSIS



https://paperswithcode.com/sota/image-classification-on-imagenet

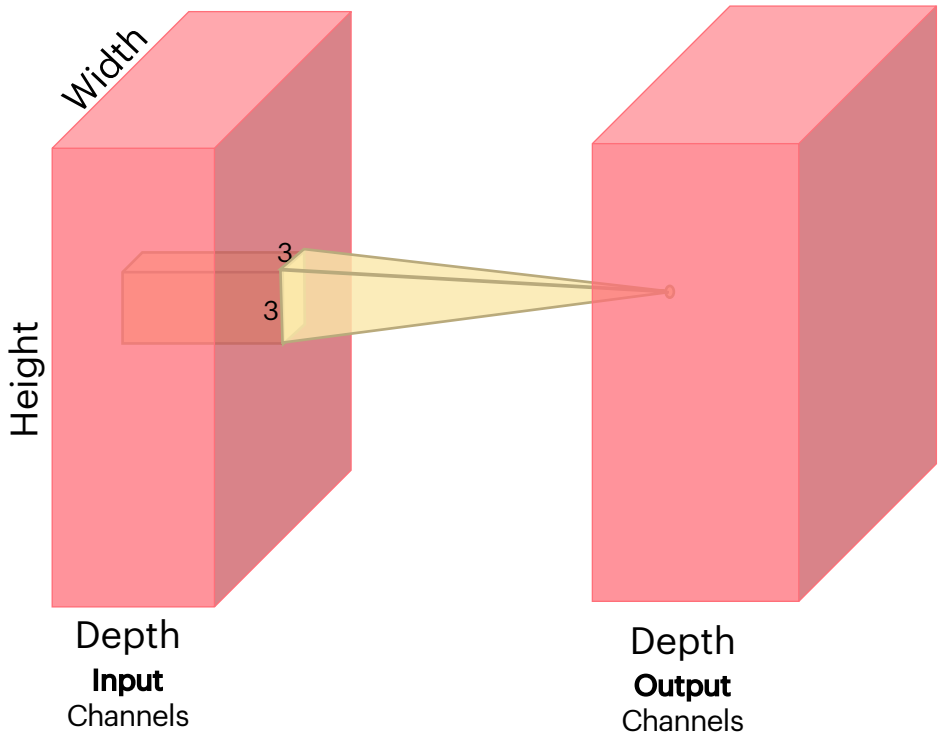
Some hedge funds use alternative data, such as images.

SOTA computer vision models use group or depthwise convolutions:

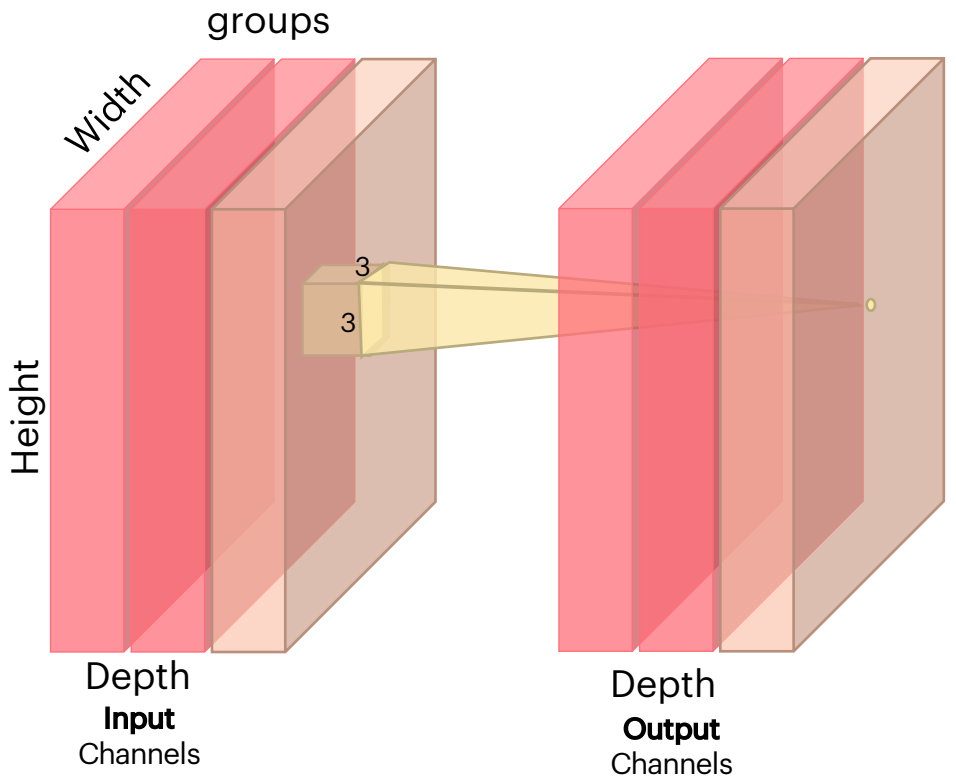
- Out of the top 50 published results on ImageNet, **42 use group or depthwise convolutions**
- ResNeXt and EfficientNet are the leading backbone architectures

GROUP CONVOLUTION

Regular convolution



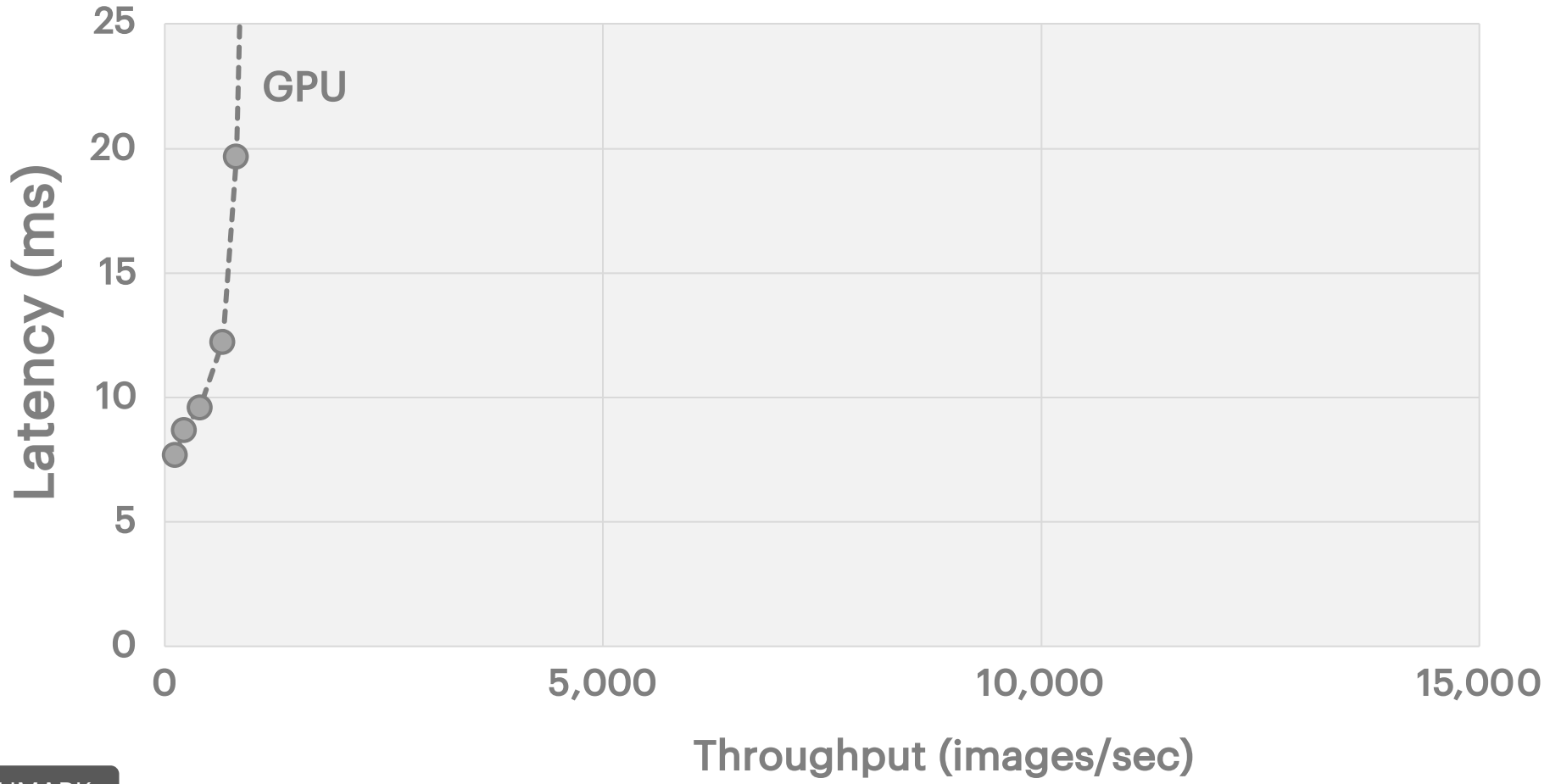
Group convolution



From https://slideplayer.com/slide/16902656/ slides by Joseph E. Gonzalez

THE PROBLEM WITH EFFICIENTNET

Both training and inference for group convs can be slow on traditional technology

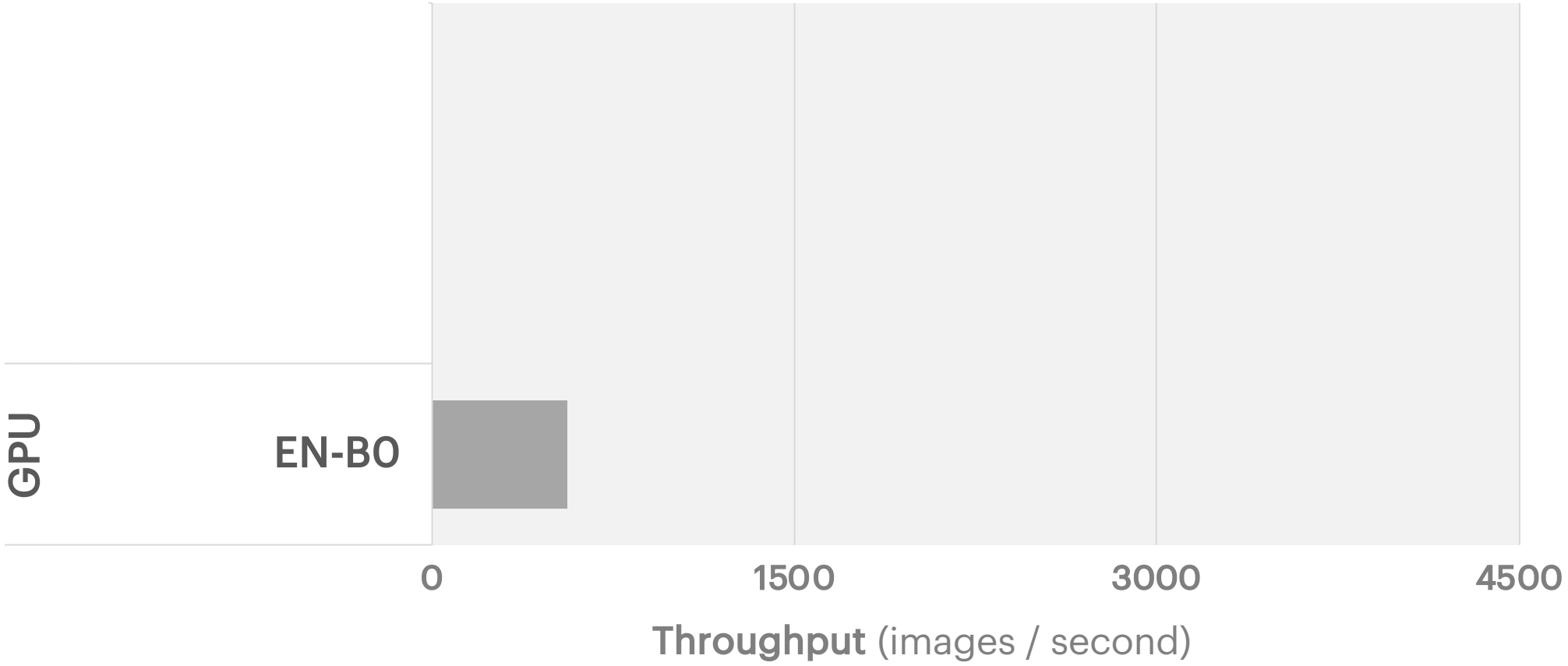


NOT A STAC BENCHMARK

NOTES:
EfficientNet-B0 | Synthetic Data | throughput comparison using highest throughput | latency comparison using lowest latency
GPU: 1x GPU (FP32) using TensorFlow & published Google reference. Batch Size 1-32 @ 300W TDP
GPU results using public Google repo (https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet/)

EFFICIENTNET-B0: TRAINING

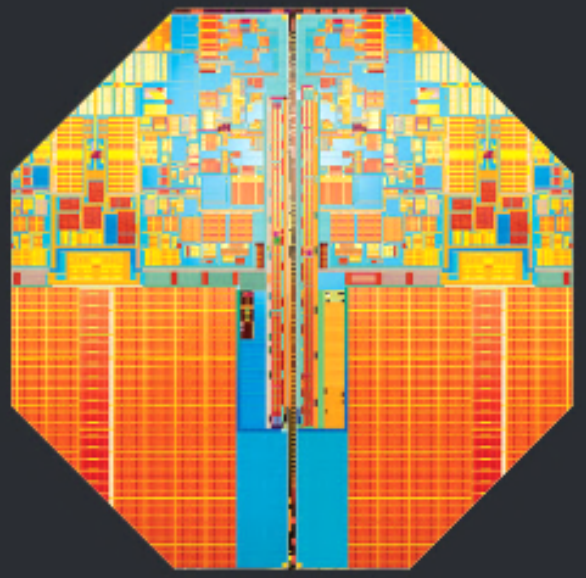
7x higher throughput



NOTES:
EfficientNet-B0 | Real Data (ImageNet)
GPU: 2x GPU (FP32) using TensorFlow @ 600W TDP (note: GPU throughput drops with modified version of EN-B0)
GPU results using public Google repo (https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet/)

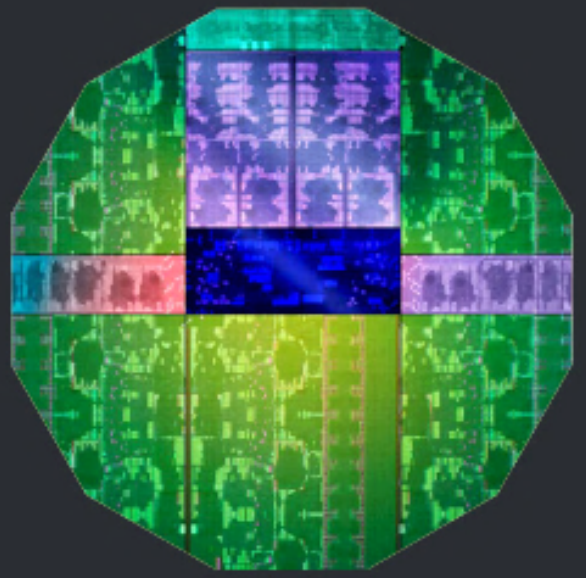
WHAT IS HOLDING THESE MODELING TECHNIQUES BACK?

LEGACY PROCESSOR ARCHITECTURES HAVE BEEN REPURPOSED FOR ML



CPU

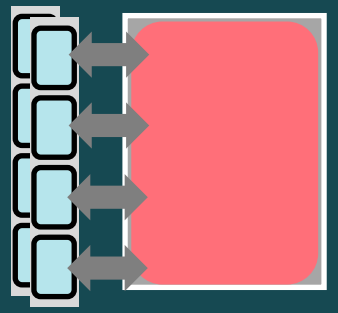
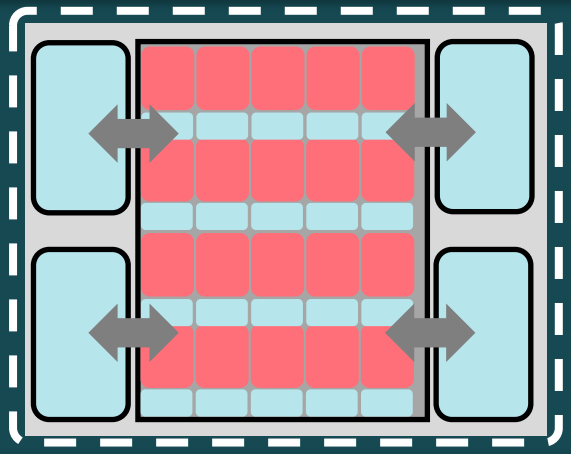
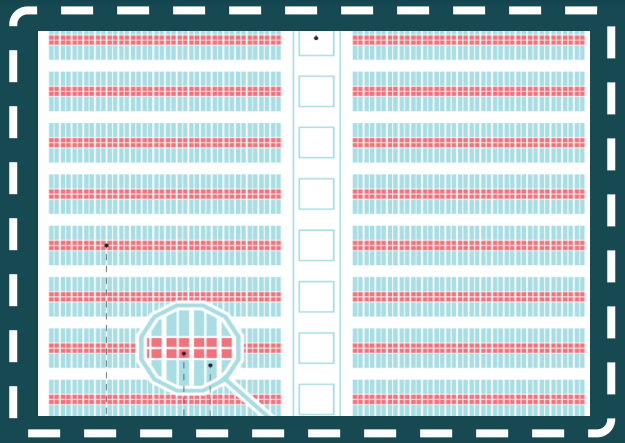
Apps and Web/
Scalar



GPU

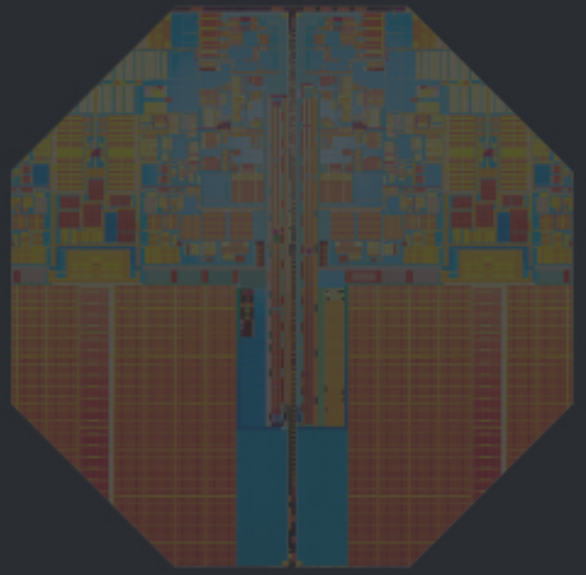
Graphics and HPC/
Vector

MACHINE LEARNING REQUIRES 2 KINDS OF COMPUTE INNOVATION

	CPU	GPU	IPU
Parallelism	Suitable for scalar processes	SIMD/SIMT architecture. Suitable for large blocks of dense contiguous data	**Massively parallel MIMD. High performance/efficiency as ML trends to sparsity & small kernels**
			
Memory Access	Off-chip memory	Model and Data spread across off-chip and small on-chip cache and shared mem.	**Model & Data in tightly coupled large locally distributed SRAM**
	1x	5x – 32x	320x

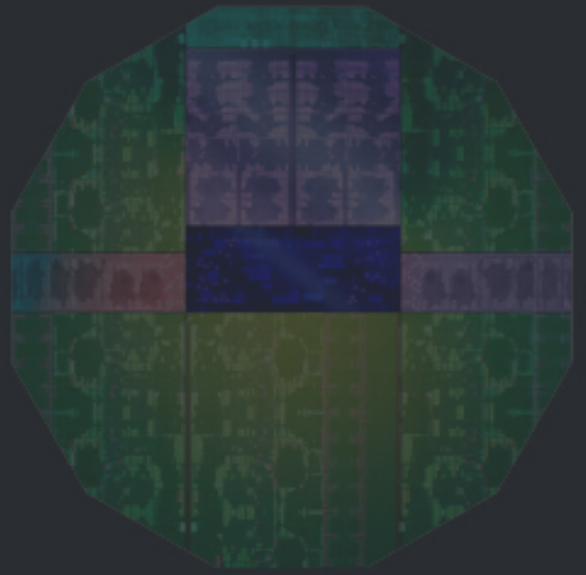
Processor Memory

GRAPHCORE HAVE IMPLEMENTED THOSE INNOVATIONS IN THE IPU



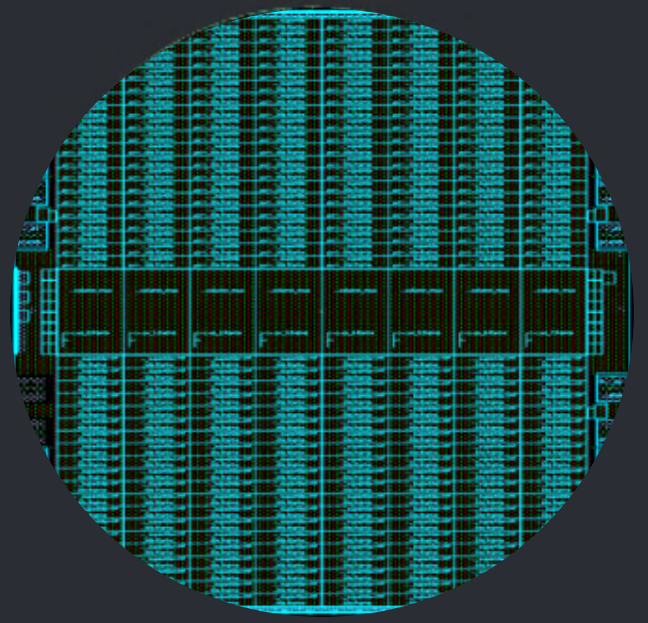
CPU

Apps and Web/
Scalar



GPU

Graphics and HPC/
Vector



IPU

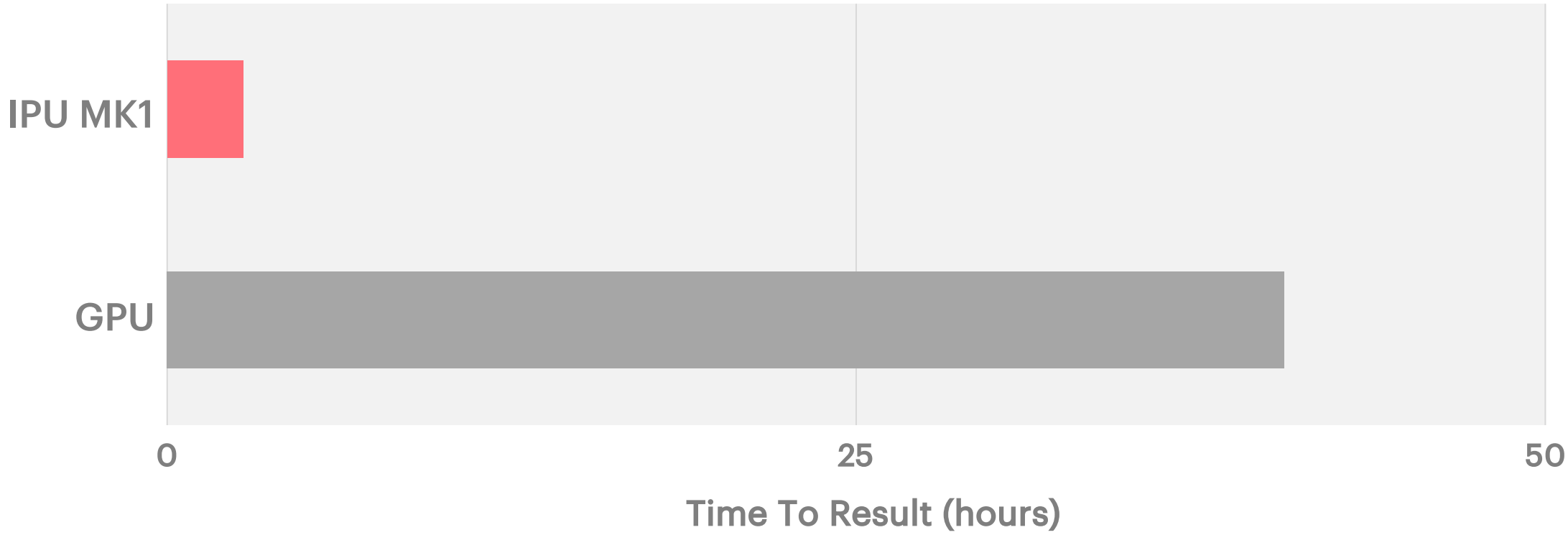
**Artificial Intelligence/
Graph**

LET’S RETURN TO OUR EXAMPLES

MCMC : NOW FEASIBLE

Alpha estimation with TensorFlow Probability

15x faster time to train

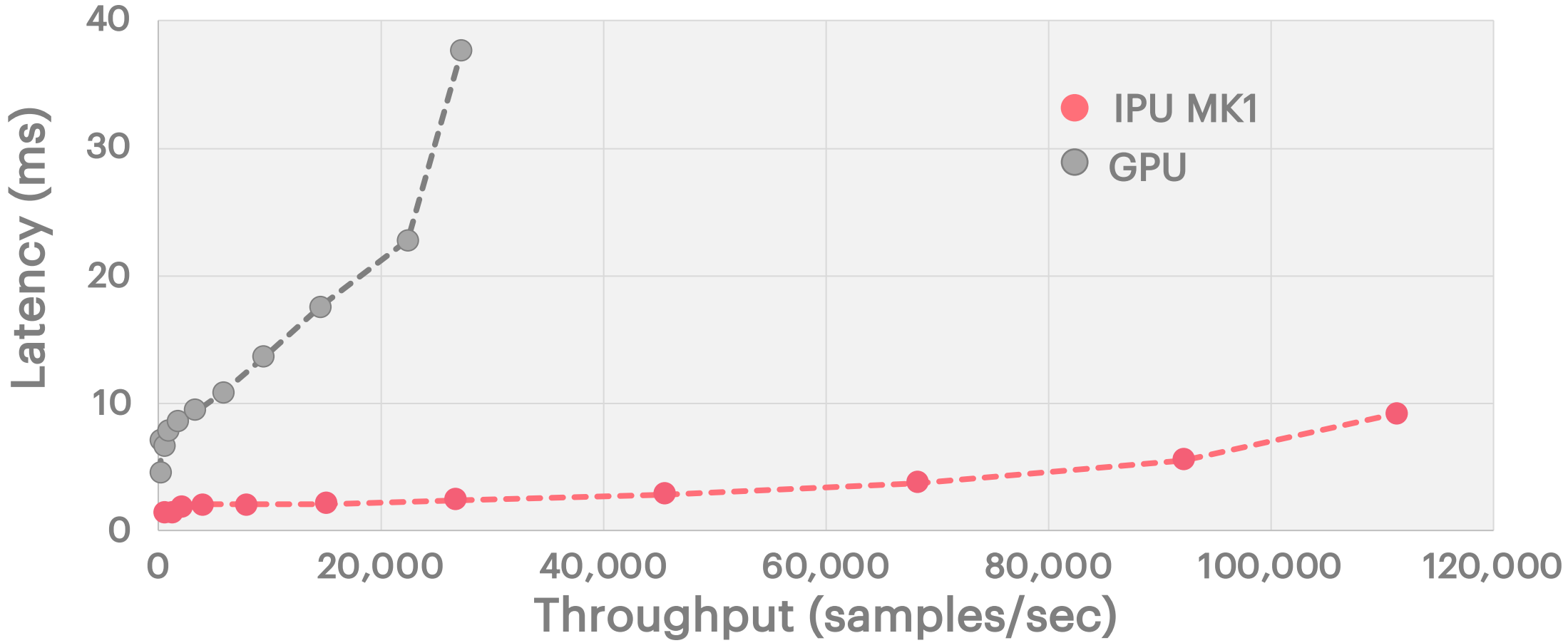


NOT A STAC BENCHMARK

NOTES:
Markov Chain Monte Carlo – Probabilistic model with TensorFlow Probability, representative of workload used by Carmot Capital
Neural network with 3 fully-connected layers (num units in 1st layer=40, #dimensions in training set =22, #leapfrog steps=1000, calcs in sliding window=200)
IPU: C2 card (300W TDP) results (SDK 1.2 GA) – 800 samples
GPU (300W TDP) - 800 samples

LSTM INFERENCE FOR TIME SERIES ANALYSIS

>300x higher throughput at lower latency
for feature generation and alpha estimation

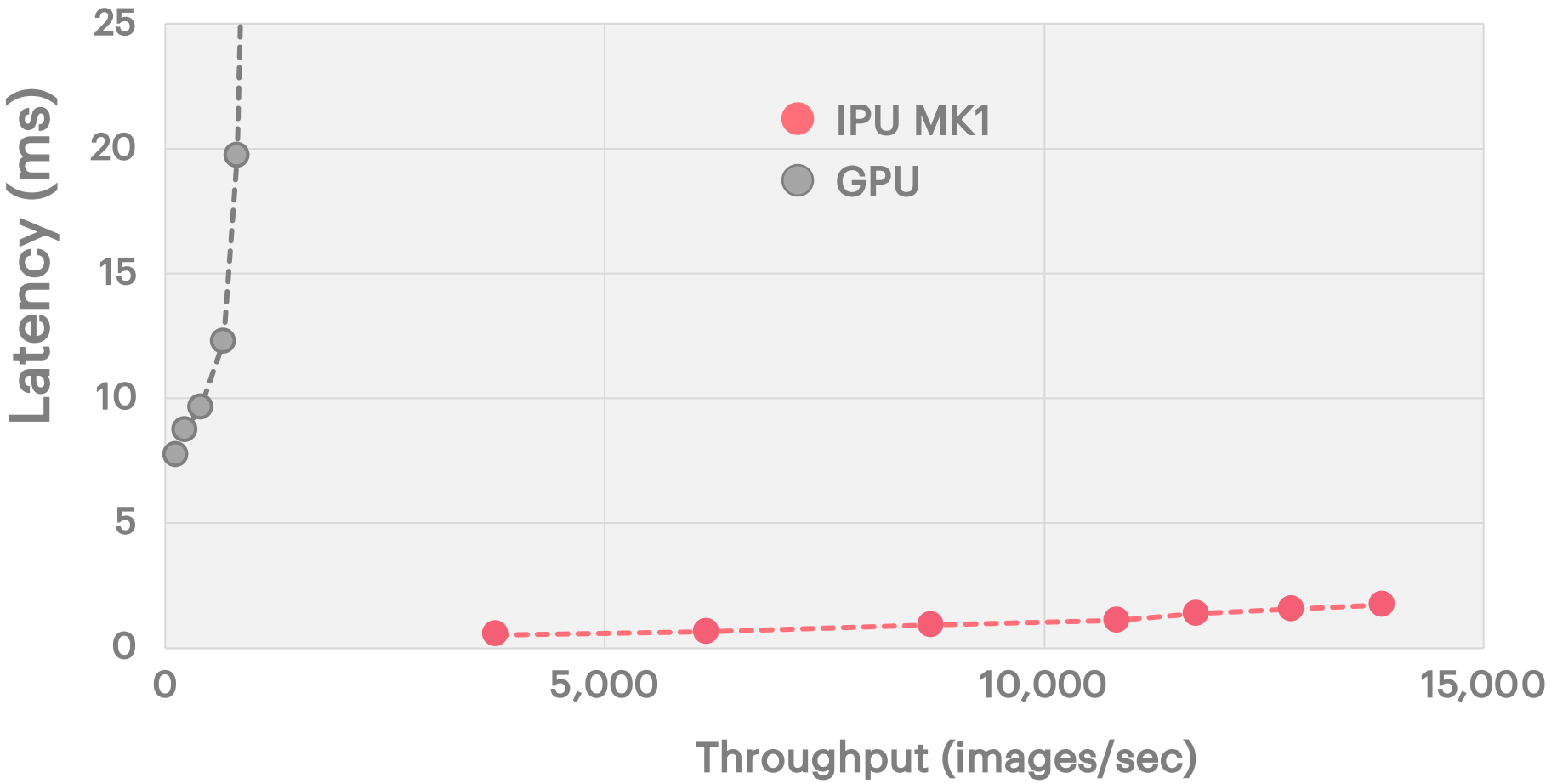


NOT A STAC BENCHMARK

NOTES:
2 LSTM layers, each with 256 units, 200 time steps, 16 input dimensions, real data, mixed precision
IPU: Graphcore C2 using TensorFlow and PopNN (SDK 1.2 GA) @ 300W TDP (Batch Sizes upto 1024)
GPU: using TensorFlow with optimizations @ 300W TDP (Batch Sizes upto 1024)

EFFICIENTNET-B0 INFERENCE

>15x higher throughput | >14x lower latency

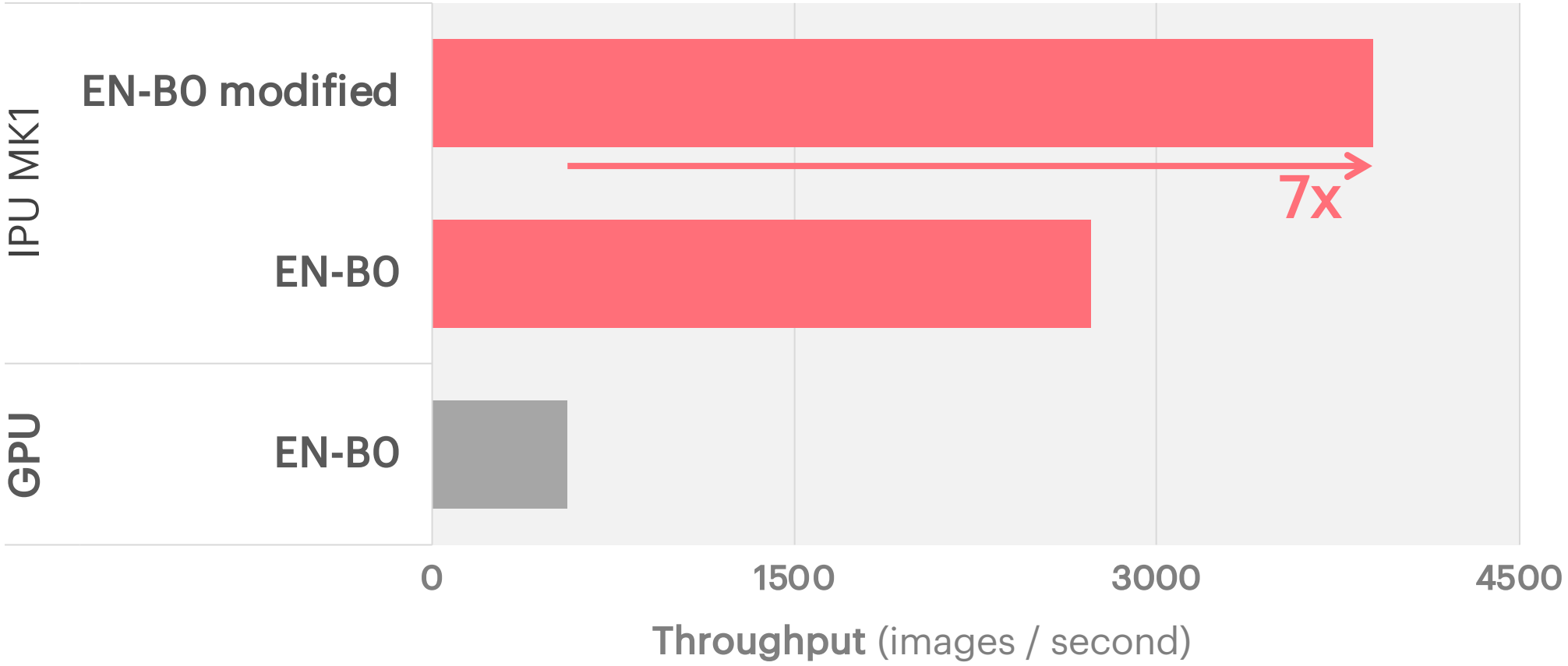


NOT A STAC BENCHMARK

NOTES:
EfficientNet-B0 | Synthetic Data | throughput comparison using highest throughput | latency comparison using lowest latency
IPU: Graphcore C2 (SDK 1.2 GA) mixed-precision using TensorFlow Batch Size 1-12 @ 300W TDP
GPU: 1x GPU (FP32) using TensorFlow & published Google reference. Batch Size 1-32 @ 300W TDP
GPU results using public Google repo (https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet/)

EFFICIENTNET-B0: TRAINING

7x higher throughput



NOT A STAC BENCHMARK

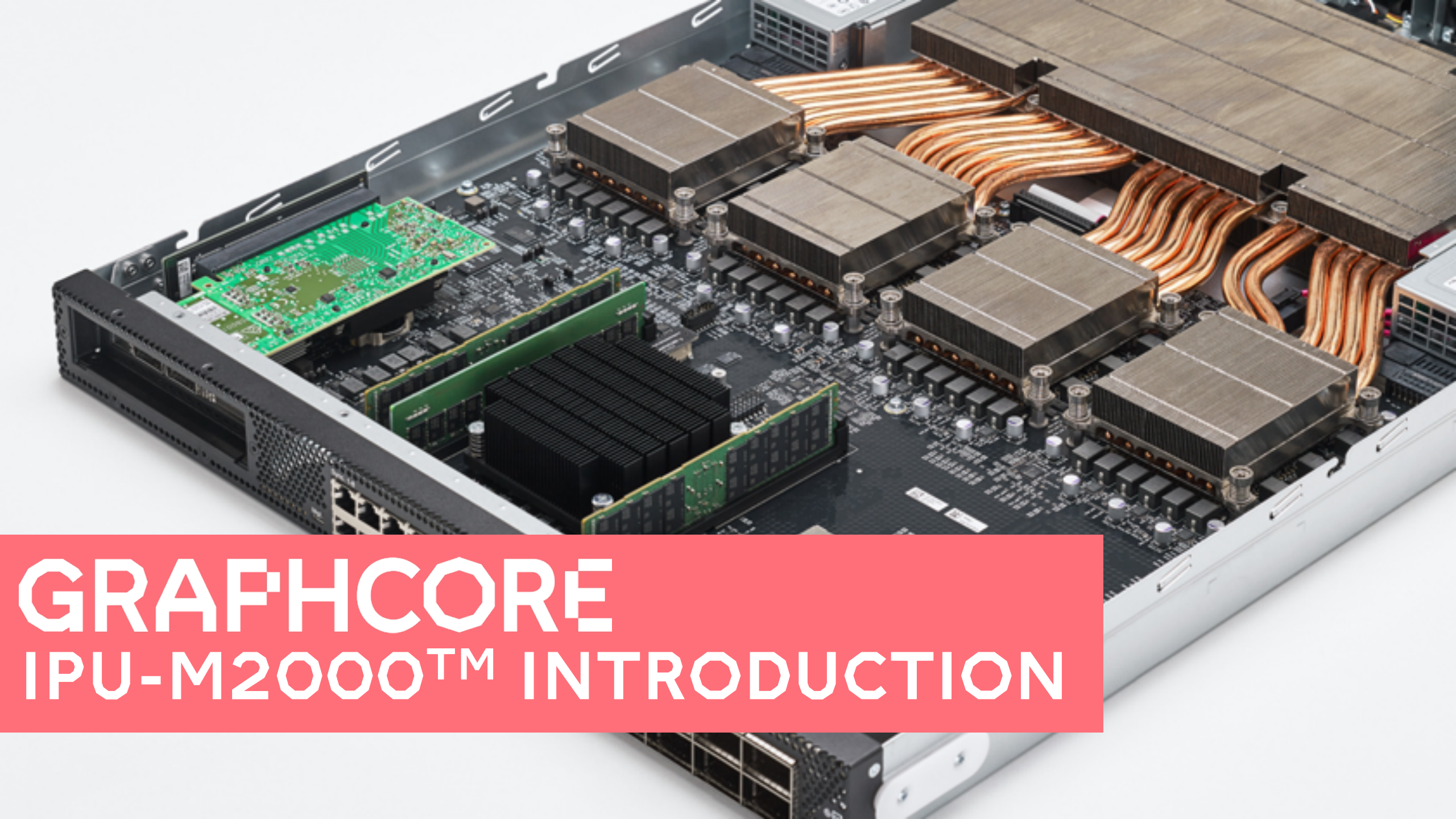
NOTES:
EfficientNet-B0 | Real Data (ImageNet)
IPU: Graphcore 2x C2 (SDK 1.2 GA) mixed-precision using TensorFlow @ 600W TDP (EN-B0 modified version uses Group Dim 16)
GPU: 2x GPU (FP32) using TensorFlow @ 600W TDP (note: GPU throughput drops with modified version of EN-B0)
GPU results using public Google repo (https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet/)

“IPU’s novel architecture is already enabling us to explore new techniques that have been inefficient or simply not possible before.”

CITADEL

“We were able to train one of our proprietary probabilistic models In 4.5 minutes instead of 2 hours. That’s 26x faster. ”

carmot CAPITAL



GRAPHCORE

IPU-M2000™ INTRODUCTION

COLOSSUS MK2

the world's most complex processor

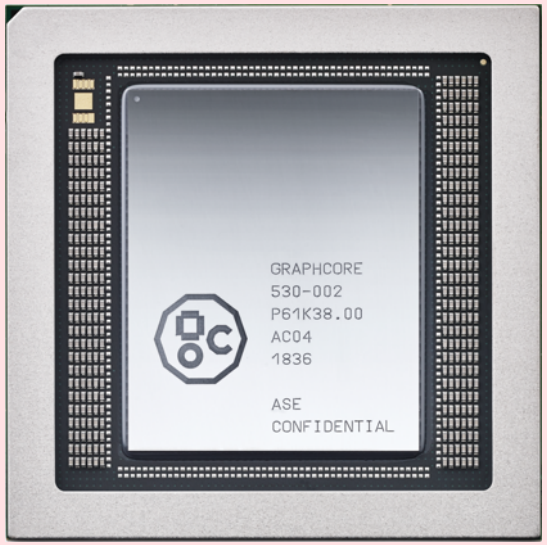
59.4Bn transistors, TSMC 7nm @ 823mm²

250TFlops AI-Float | 900MB In-Processor-Memory™

1472 independent processor cores

8832 separate parallel threads

>8x step-up in system performance vs Mk1



GC200 IPU

GC200 IPU PROCESSOR

IPU-Tiles™

1472 independent IPU-Tiles™ each with an IPU-Core™ and In-Processor-Memory™

IPU-Core™

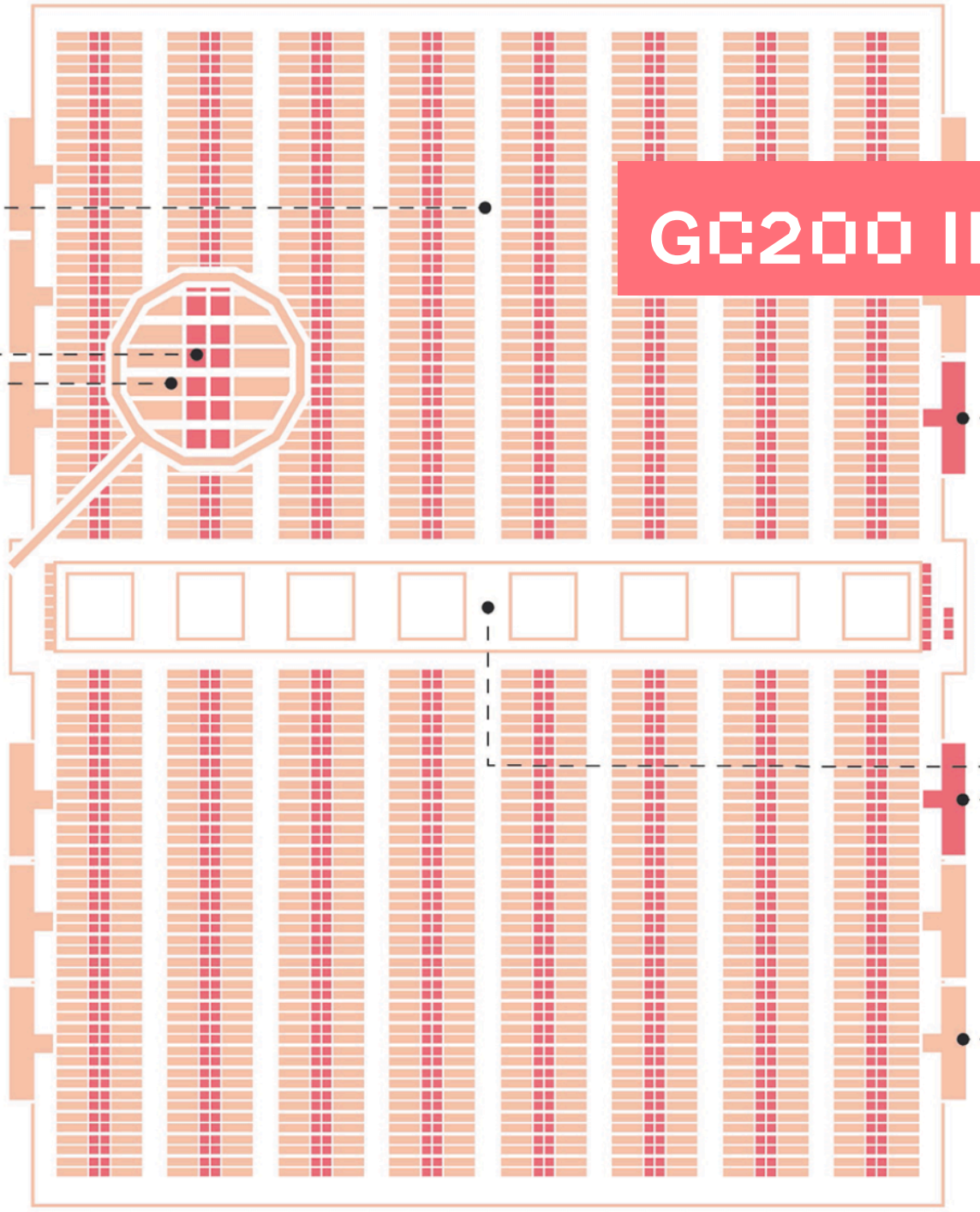
1472 independent IPU-Core™

8832 independent program threads executing in parallel

In-Processor-Memory™

900MB In-Processor-Memory™ per IPU

47.5TB/s memory bandwidth per IPU



IPU-Exchange™

8 TB/s all to all IPU-Exchange™
Non-blocking, any communication pattern

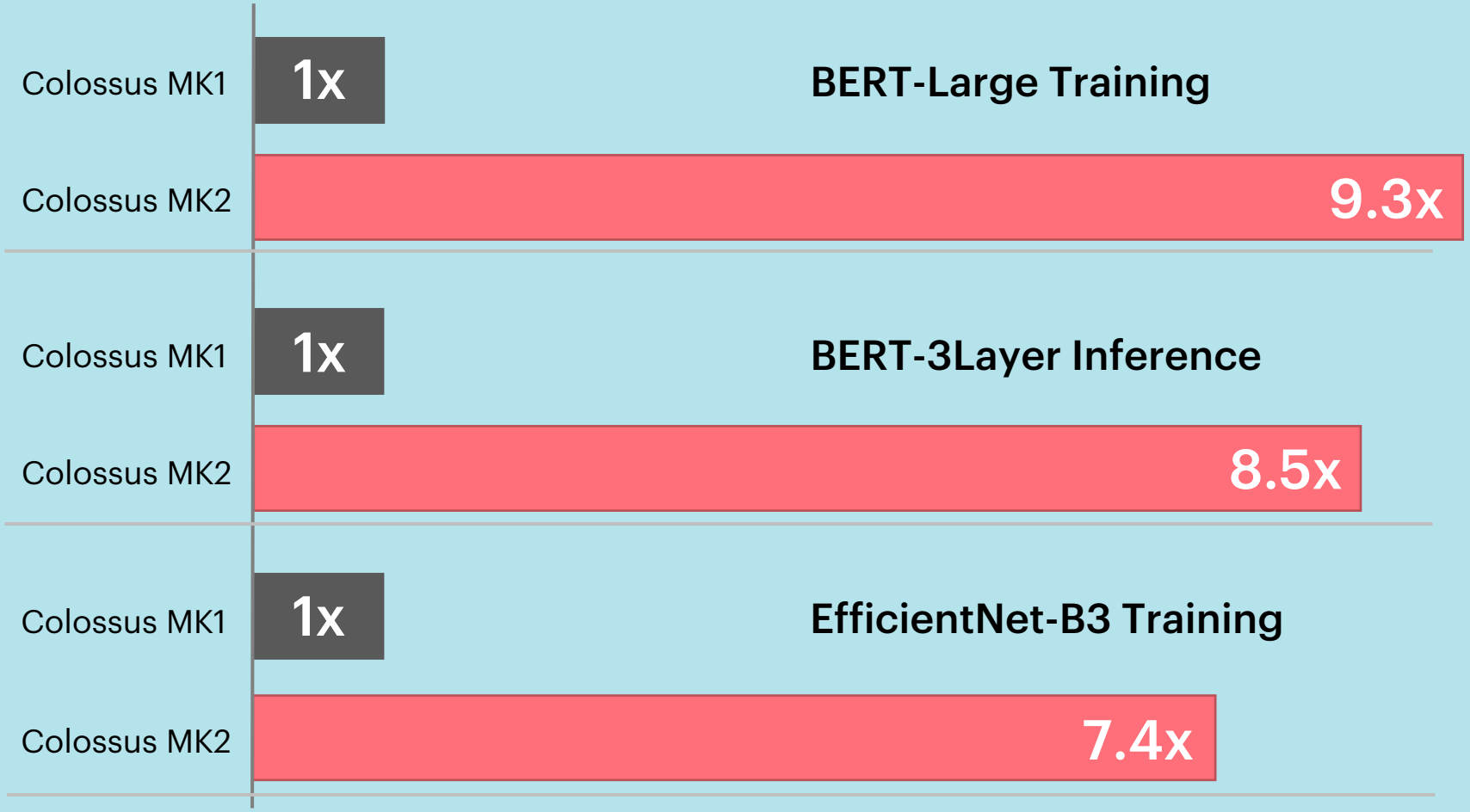
PCIe

PCI Gen4 x16
64 GB/s bidirectional bandwidth to host

IPU-Links™

10 x IPU-Links,
320GB/s chip to chip bandwidth

COLOSSUS Mk2 PERFORMANCE

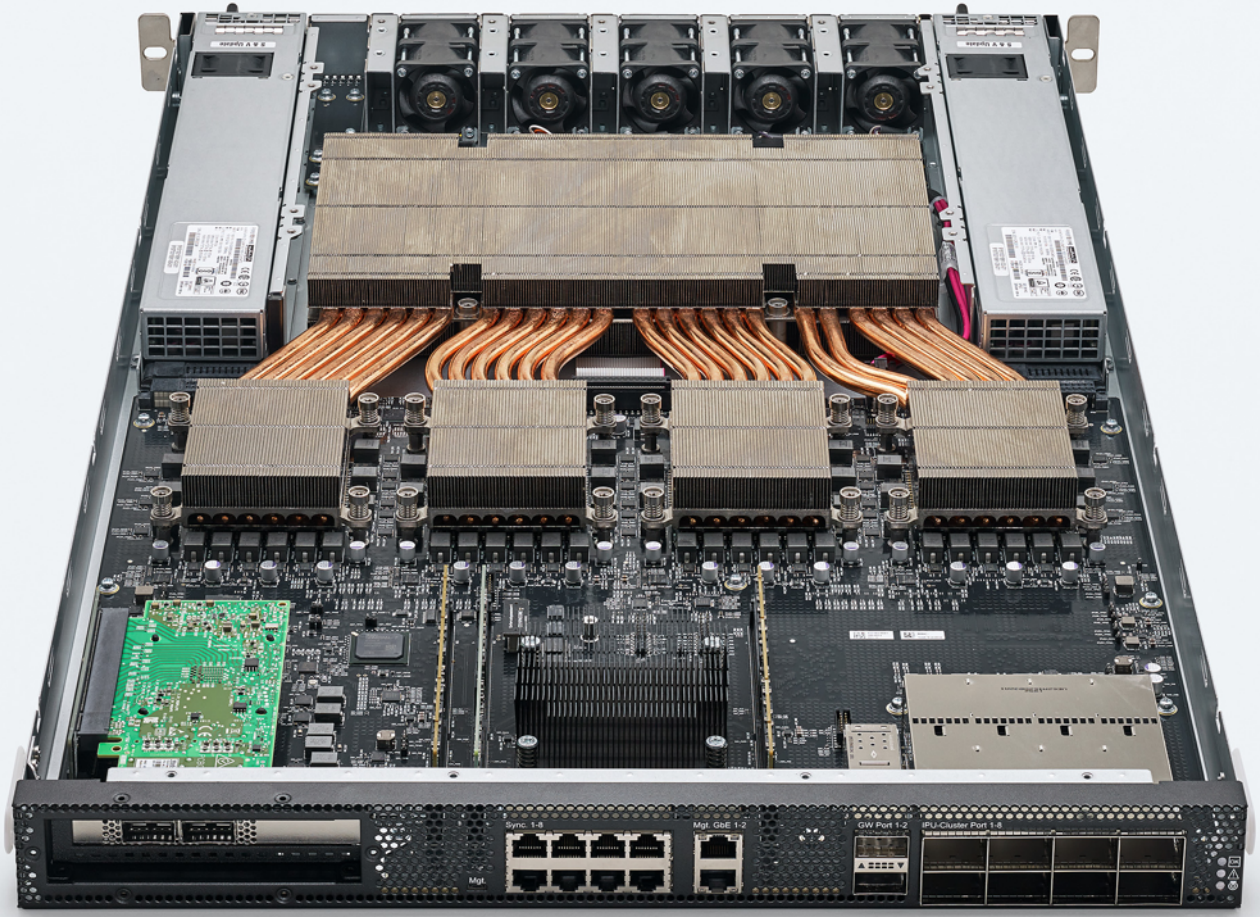


Note: 8x C2 PCIe IPU-Processors with IPU-LINK vs. 8x M2000 M2000 with IPU-FABRIC

NOT A STAC BENCHMARK

IPU-MACHINE
M2000

1 PetaFlop IPU compute
2.8Tbps IPU-Fabric™



M2000 PRODUCT CONFIGURATIONS

x1 M2000

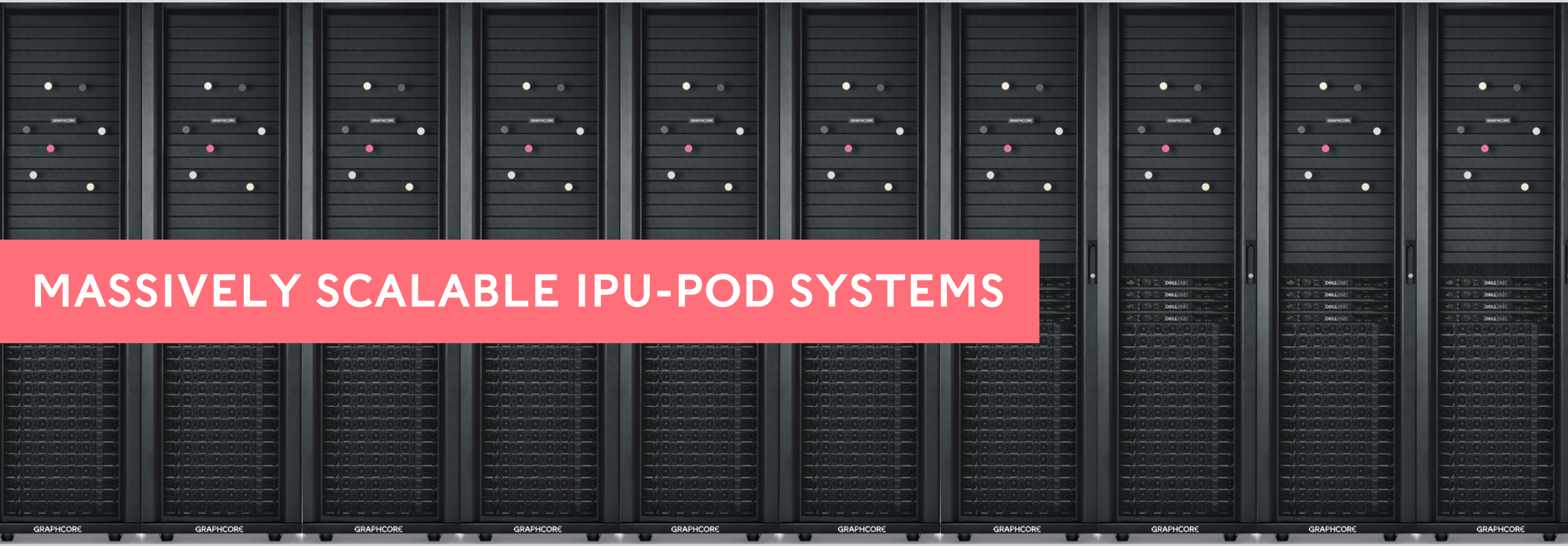
x4 M2000

x8 M2000

STANDALONE

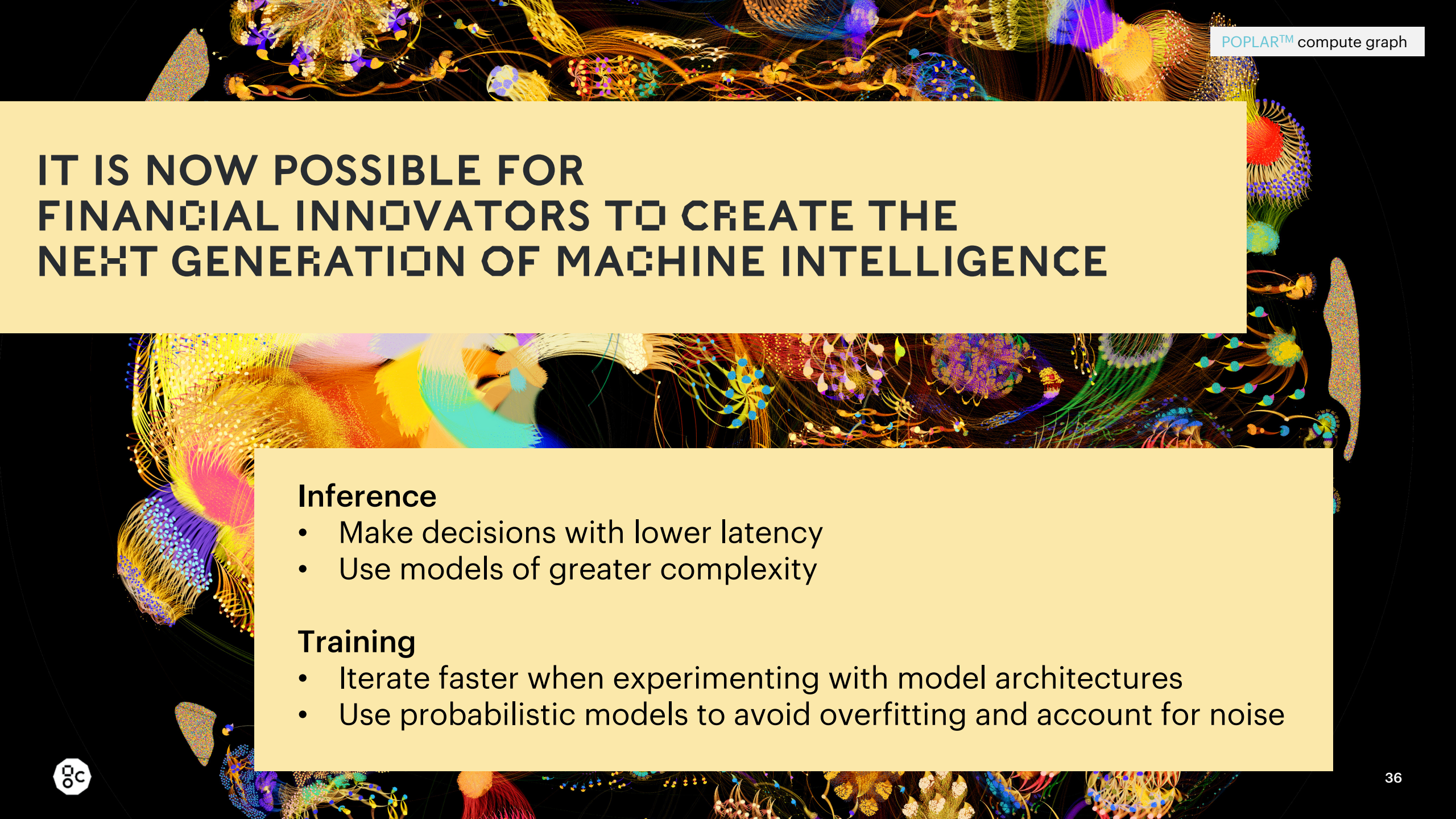
IPU-POD$_{64}$

COMPUTE + DATA + COMMUNICATIONS



MASSIVELY SCALABLE IPU-POD SYSTEMS

IPU-POD



IT IS NOW POSSIBLE FOR FINANCIAL INNOVATORS TO CREATE THE NEXT GENERATION OF MACHINE INTELLIGENCE

Inference

- Make decisions with lower latency
- Use models of greater complexity

Training

- Iterate faster when experimenting with model architectures
- Use probabilistic models to avoid overfitting and account for noise

THANK YOU

Alexander Tsyplikhin
alext@graphcore.ai