# Deep Learning Inference in Facebook Data Centers: Characterization, Performance Optimizations, and Hardware Implications

Jongsoo Park
Facebook AI System SW/HW Co-design Team

Sep-21 2018

# Team Introduction

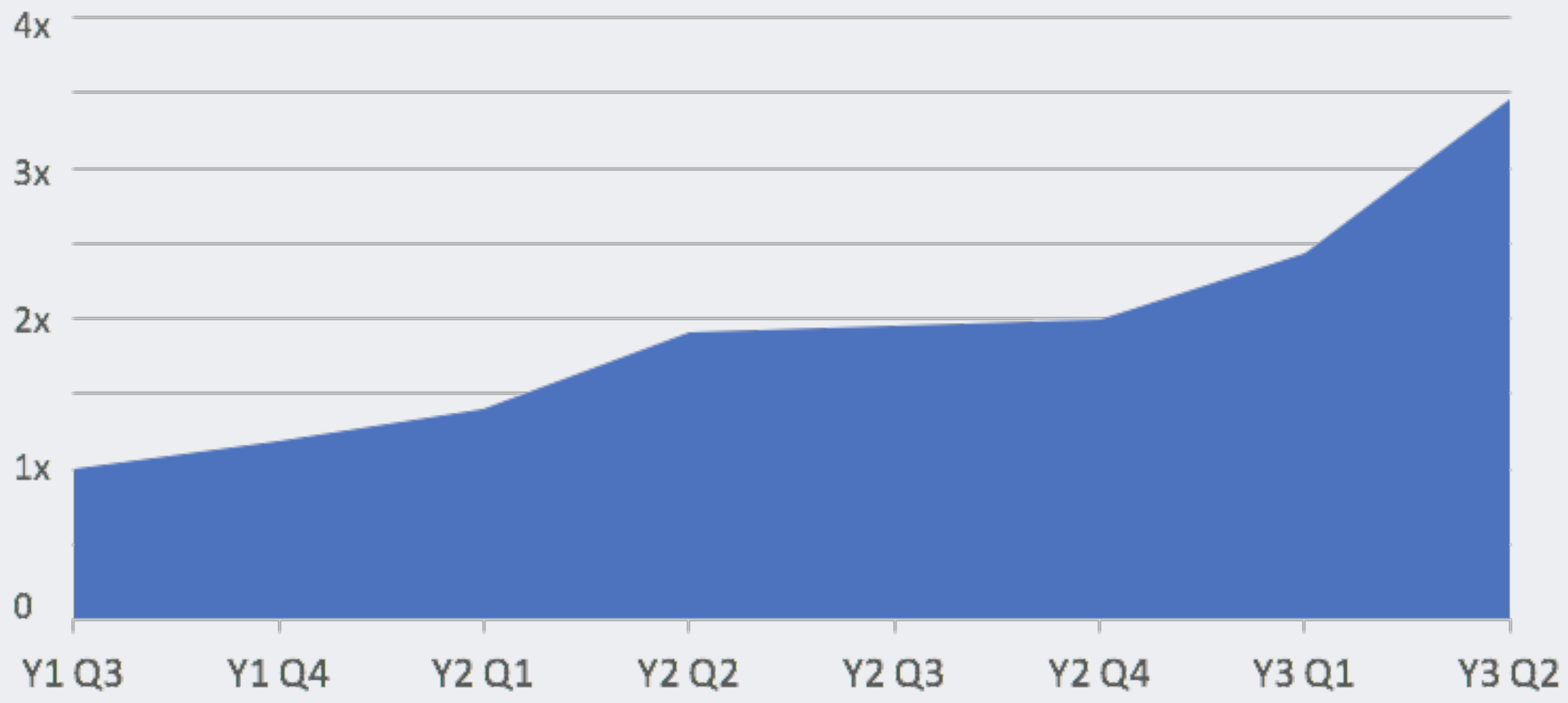
- AI System Co-design
- High performance numerical and architectural SW optimizations, HW performance modeling and recommendations through Machine Learning-driven Co-design
- Expertise
  - HPC and parallel algorithms
  - Computer architecture
  - Performance optimization and modeling
  - Numerical linear algebra, ML, and graph analytics

# Outline

- **Introduction to deep learning inference at Facebook**
- Computational characteristics
- Optimization experience on current HWs (Intel CPUs)
- SW/HW Co-design directions

# DL Inference in Facebook Data Centers

- Used for core services: personalization and integrity/security
- Diverse data types: images, videos, multi-lingual contents
- Scale to billions of users




Increase of server capacity for DL inference, Xiaodong Wang

# DL Application Domains

1. Ranking and recommendation: ads, feed, and search
2. Computer vision: image classification, object detection, and video understanding
3. Language: translation, content understanding

- Interactions among these: powering recommendation (1) with visual (2) and linguistic (3) content understanding

# Domain 1: Ranking and Recommendation

- Embedding tables demand
  - High memory capacity (>10s of GBs)
  - High memory bandwidth (low arithmetic intensity)
- HBMs are too small. NVMs are too slow

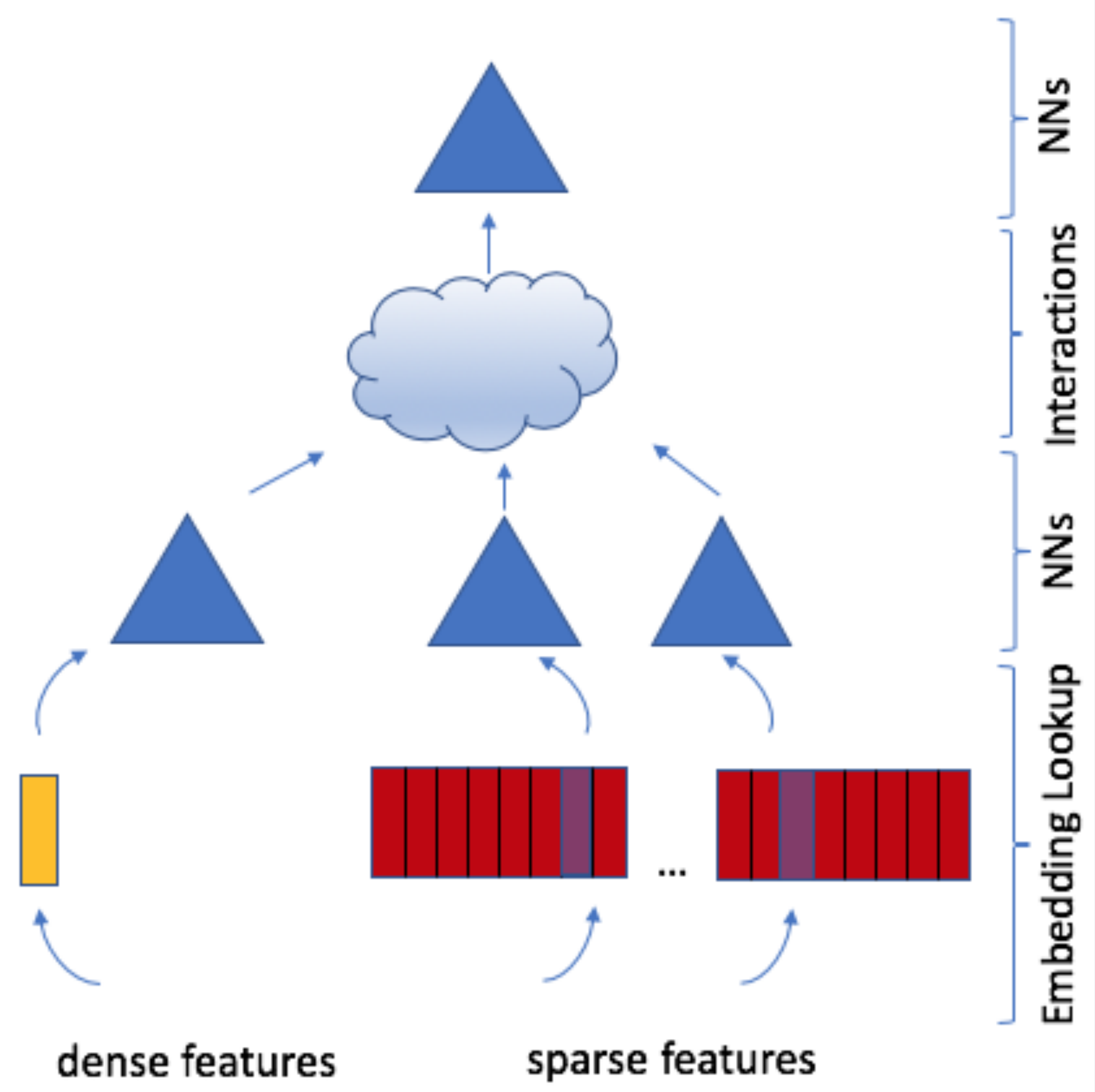
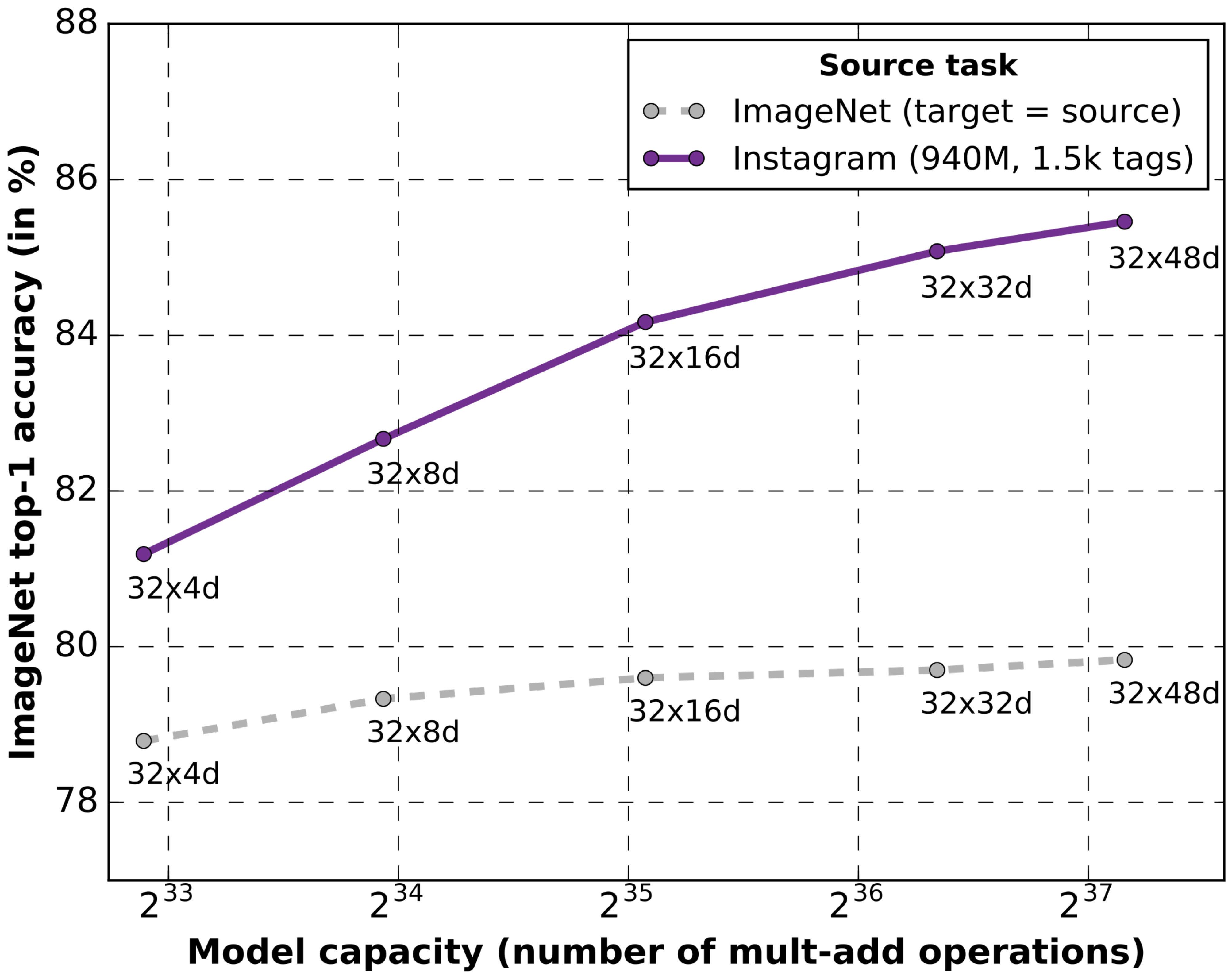



Figure credit: Maxim Naumov

# Domain 2: Computer Vision

- Classification
  - Bigger model + bigger data → higher accuracy



# Domain 2: Computer Vision

- Classification
  - Bigger model + bigger data → higher accuracy

- Object detection and video understanding
  - Bigger inputs than classification
  - FLOP-efficient models like ShuffleNet with depth-wise convolutions [2]

[1] Exploring the limits of weakly supervised pretraining. Mahajan et al.

[2] Rosetta: understanding text in images and videos with machine learning. Sivakumar et al.

# Domain 3: Language Models

- Small batch size for latency constraints
- Attention only models
- Multilingual models

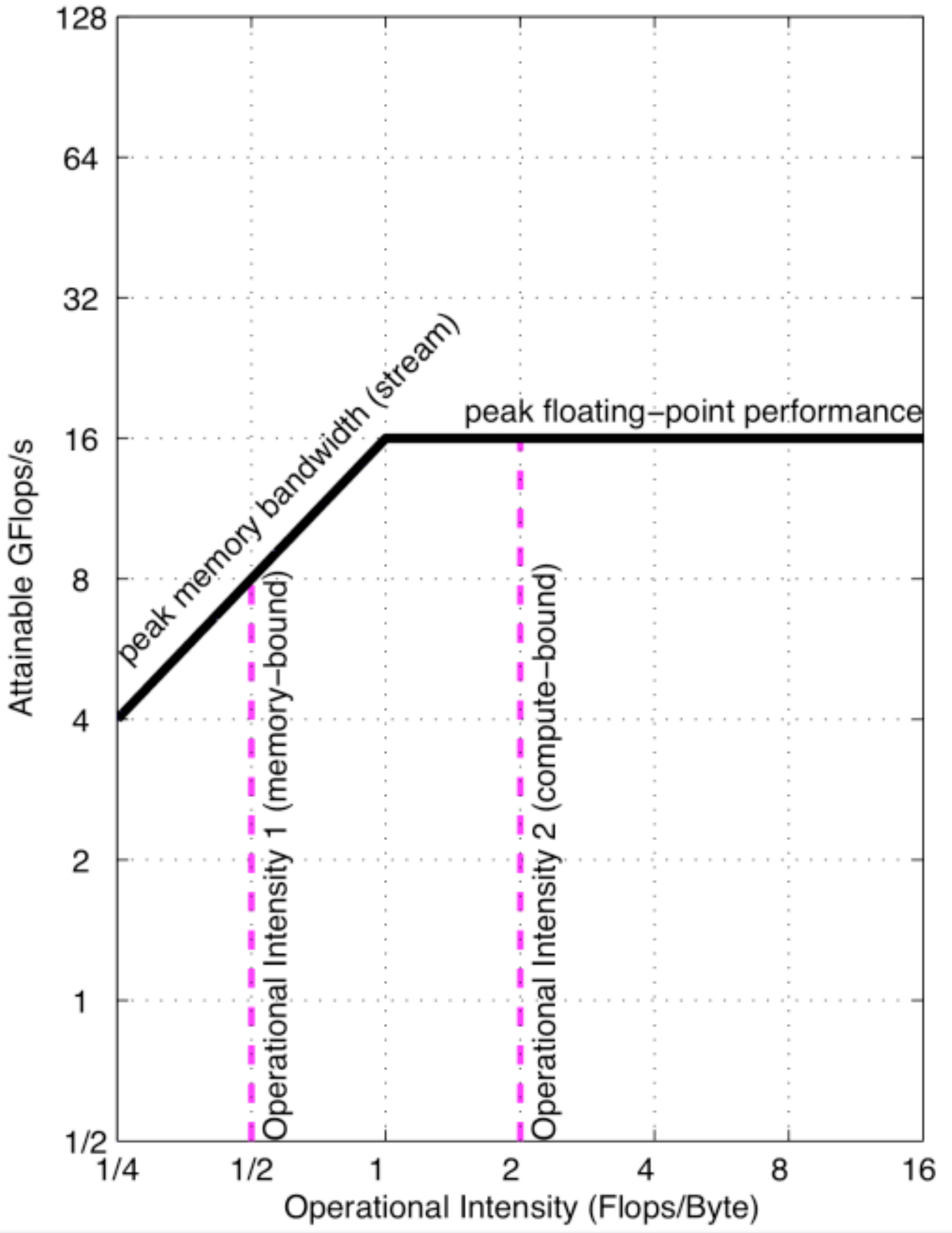
# Outline

- Introduction to deep learning inference at Facebook
- **Computational characteristics**
- Optimization experience on current HWs (Intel CPUs)
- SW/HW Co-design directions

# Roofline Model Recap

- Application flop/byte < System flop/byte → performance bound by memory BW

- Flop/byte w.r.t. parameters: drives off-chip BW need when parameters off chip and activations on chip
- Flop/byte w.r.t. parameters + activations: drives off-chip BW need when activations too big so need to be off chip, or on-chip BW need




Roofline: An Insightful Visual Performance Model for Floating-point Programs and Multicore Architectures. Williams et al.

# Resource Requirements

| Category | Model Types | Model Size (# params) | Max. Live Activations | Op. Intensity (w.r.t. weights) | Op. Intensity (w.r.t. act & weights) |
|---|---|---|---|---|---|
| Recommendation | FCs | 1-10M | > 10K | 20-200 | 20-200 |
| | Embeddings | >10 Billion | > 10K | 1-2 | 1-2 |
| Computer Vision | ResNeXt101-32x4-48 | 43-829M | 2-29M | avg. 380<br>Min. 100 | Avg. 188<br>Min. 28 |
| | Faster-RCNN (with ShuffleNet) | 6M | 13M | Avg. 3.5K<br>Min. 2.5K | Avg. 145<br>Min. 4 |
| | ResNeXt3D-101 | 21M | 58M | Avg. 22K<br>Min. 2K | Avg. 172<br>Min. 6 |
| Language | seq2seq | 100M-1B | >100K | 2-20 | 2-20 |

# Observation 1: big embedding with low op. intensity

| Category | Model Types | Model Size (# params) | Max. Live Activations | Op. Intensity (w.r.t. weights) | Op. Intensity (w.r.t. act & weights) |
|---|---|---|---|---|---|
| Recommendation | FCs | 1-10M | > 10K | 20-200 | 20-200 |
| | **Embeddings** | **>10 Billion** | > 10K | **1-2** | **1-2** |
| Computer Vision | ResNeXt101-32x4-48 | 43-829M | 2-29M | avg. 380<br>Min. 100 | Avg. 188<br>Min. 28 |
| | Faster-RCNN (with ShuffleNet) | 6M | 13M | Avg. 3.5K<br>Min. 2.5K | Avg. 145<br>Min. 4 |
| | ResNeXt3D-101 | 21M | 58M | Avg. 22K<br>Min. 2K | Avg. 172<br>Min. 6 |
| Language | seq2seq | 100M-1B | >100K | 2-20 | 2-20 |

- Interesting challenge for future memory system designs

# Observation 2: bigger models and activations

| Category | Model Types | Model Size (# params) | Max. Live Activations | Op. Intensity (w.r.t. weights) | Op. Intensity (w.r.t. act & weights) |
|---|---|---|---|---|---|
| Recommendation | FCs | 1-10M | > 10K | 20-200 | 20-200 |
| | Embeddings | >10 Billion | > 10K | 1-2 | 1-2 |
| Computer Vision | **ResNeXt101-32x4-48** | **43-829M** | **2-29M** | avg. 380<br>Min. 100 | Avg. 188<br>Min. 28 |
| | **Faster-RCNN (with ShuffleNet)** | 6M | **13M** | Avg. 3.5K<br>Min. 2.5K | Avg. 145<br>Min. 4 |
| | **ResNeXt3D-101** | 21M | **58M** | Avg. 22K<br>Min. 2K | Avg. 172<br>Min. 6 |
| Language | **seq2seq** | **100M-1B** | >100K | 2-20 | 2-20 |

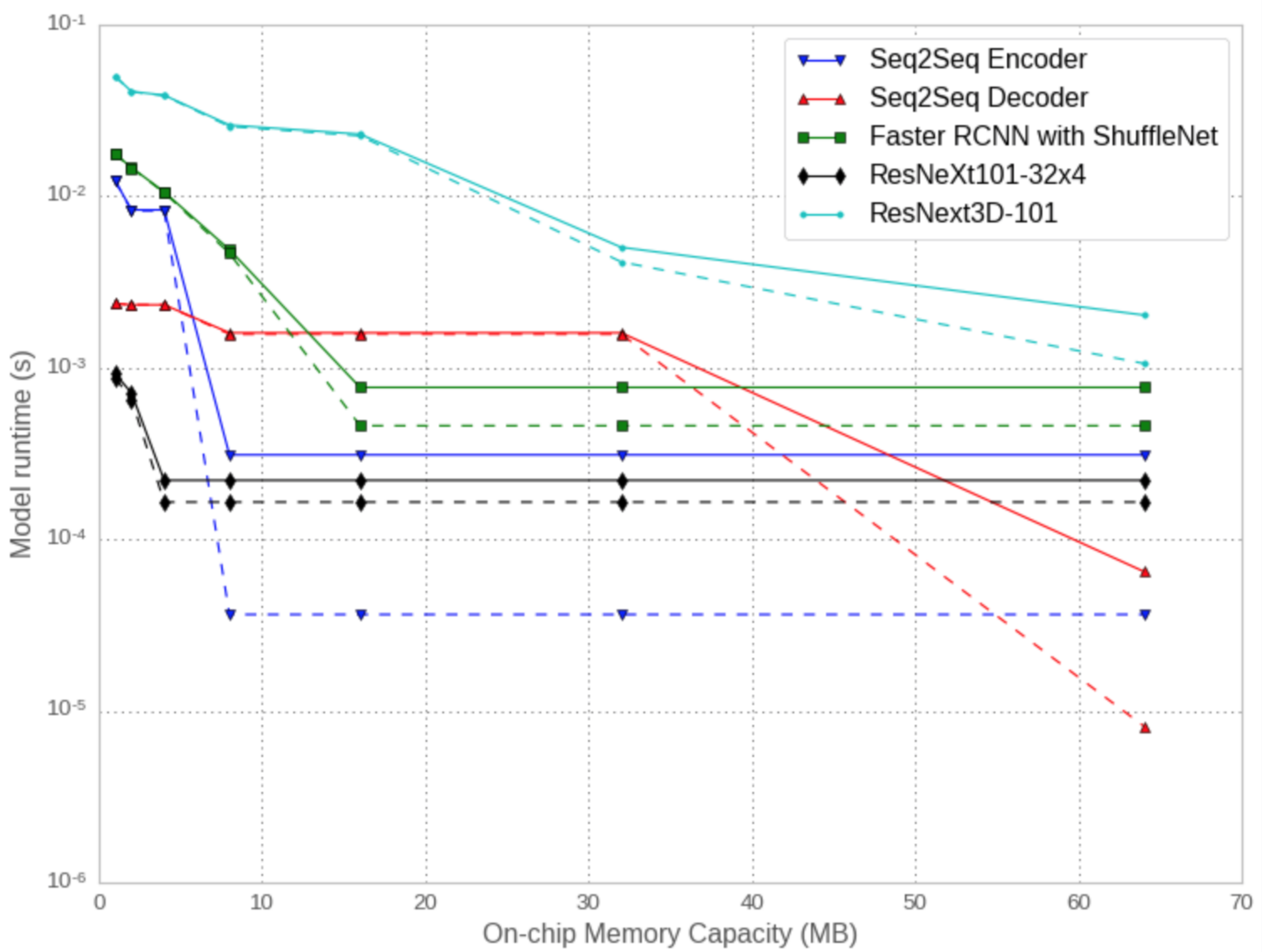
- Need large on-chip memory. Otherwise off-chip memory BW bound for small batch.

# Observation 3: tall-skinny matrix operations

| Category | Model Types | Model Size (# params) | Max. Live Activations | Op. Intensity (w.r.t. weights) | Op. Intensity (w.r.t. act & weights) |
|---|---|---|---|---|---|
| Recommendation | FCs | 1-10M | > 10K | 20-200 | 20-200 |
| | Embeddings | >10 Billion | > 10K | 1-2 | 1-2 |
| Computer Vision | ResNeXt101-32x4-48 | 43-829M | 2-29M | avg. 380<br>Min. 100 | Avg. 188<br>Min. 28 |
| | **Faster-RCNN (with ShuffleNet)** | 6M | 13M | Avg. 3.5K<br>Min. 2.5K | **Avg. 145<br>Min. 4** |
| | **ResNeXt3D-101** | 21M | 58M | Avg. 22K<br>Min. 2K | **Avg. 172<br>Min. 6** |
| Language | **seq2seq** | 100M-1B | >100K | 2-20 | **2-20** |

- e.g., depth-wise convolution
- Low utilization with big matrix-matrix unit
- Need high on-chip memory BW
- More on next slides

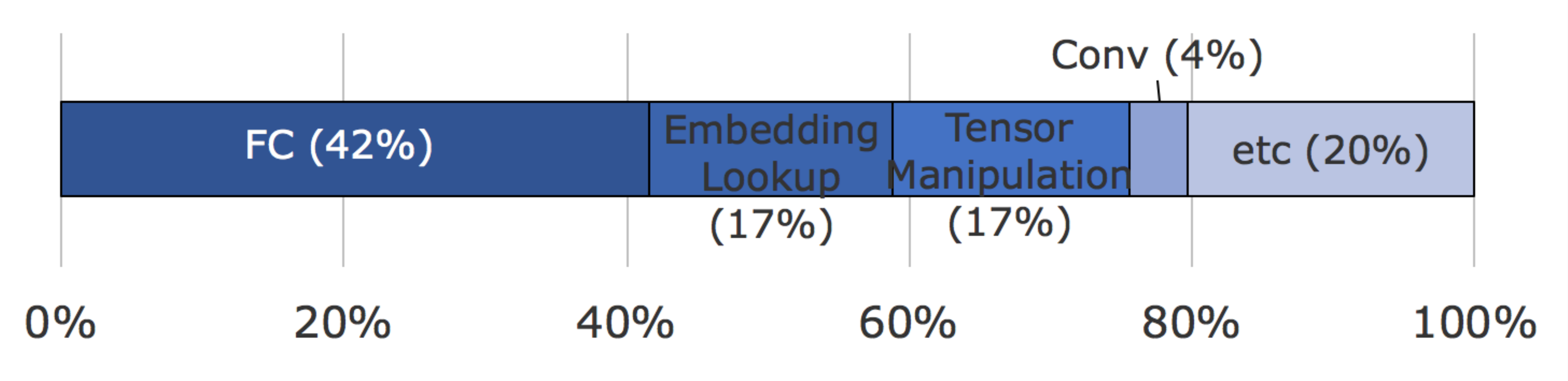
# Need for bigger and faster on-chip memory BW




- Runtime roofline analysis on a hypothetical accelerator with 100 int8 Top/s. Solid lines: 1 TB/s on-chip BW. Dashed lines: 10 TB/s on-chip BW.

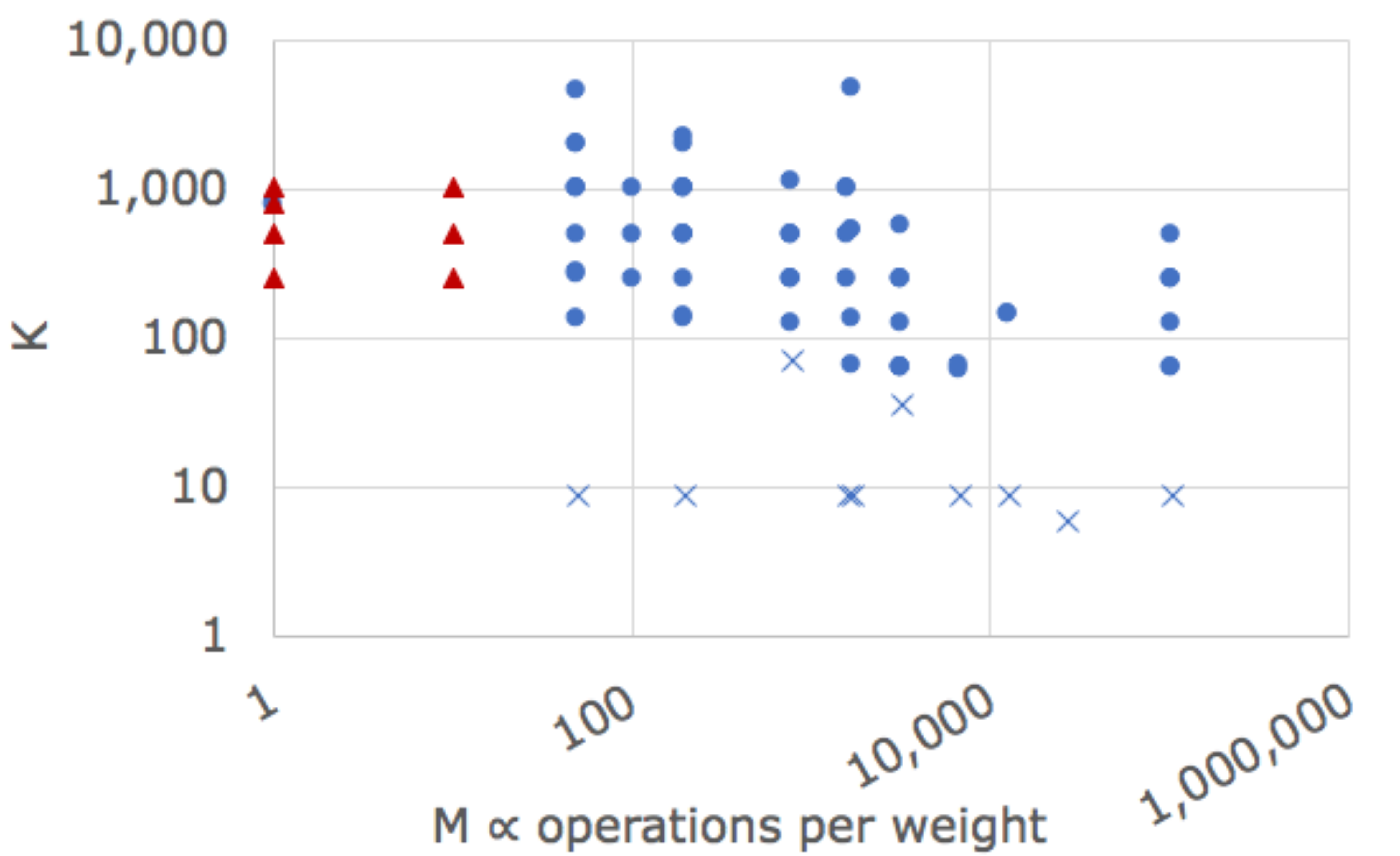
Figure credit: Martin Schatz

# Fleet-wide Caffe2 operator execution time breakdown

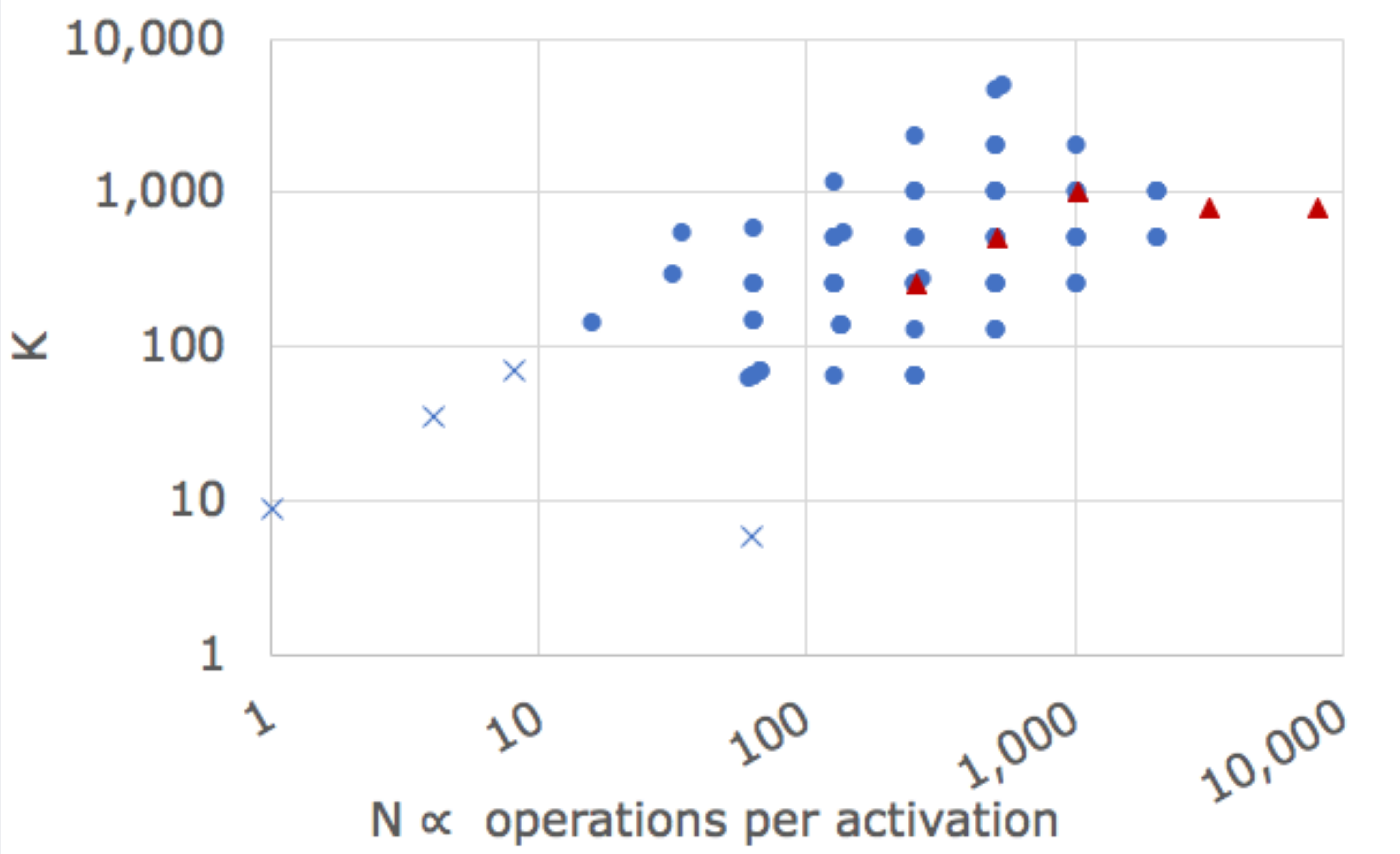



- FC is the most time consuming followed by embedding
- Conv is only 4%
- Tensor manipulation (concat, split, transpose, ...): good graph-level optimization targets

# Common matrix shapes




Activation matrices




Weight matrices

- Caffe convention: M-by-K activation matrix * K-by-N weight matrix
- ▲: FCs, X : group/depth-wise convolutions, ● : other convolutions
- Many shapes are not good targets of matrix-matrix units and with moderate op. intensity

# Outline

- Introduction to deep learning inference at Facebook
- Computational characteristics
- **Optimization experience on current HWs (Intel CPUs)**
- SW/HW Co-design directions

# Optimization Methodology

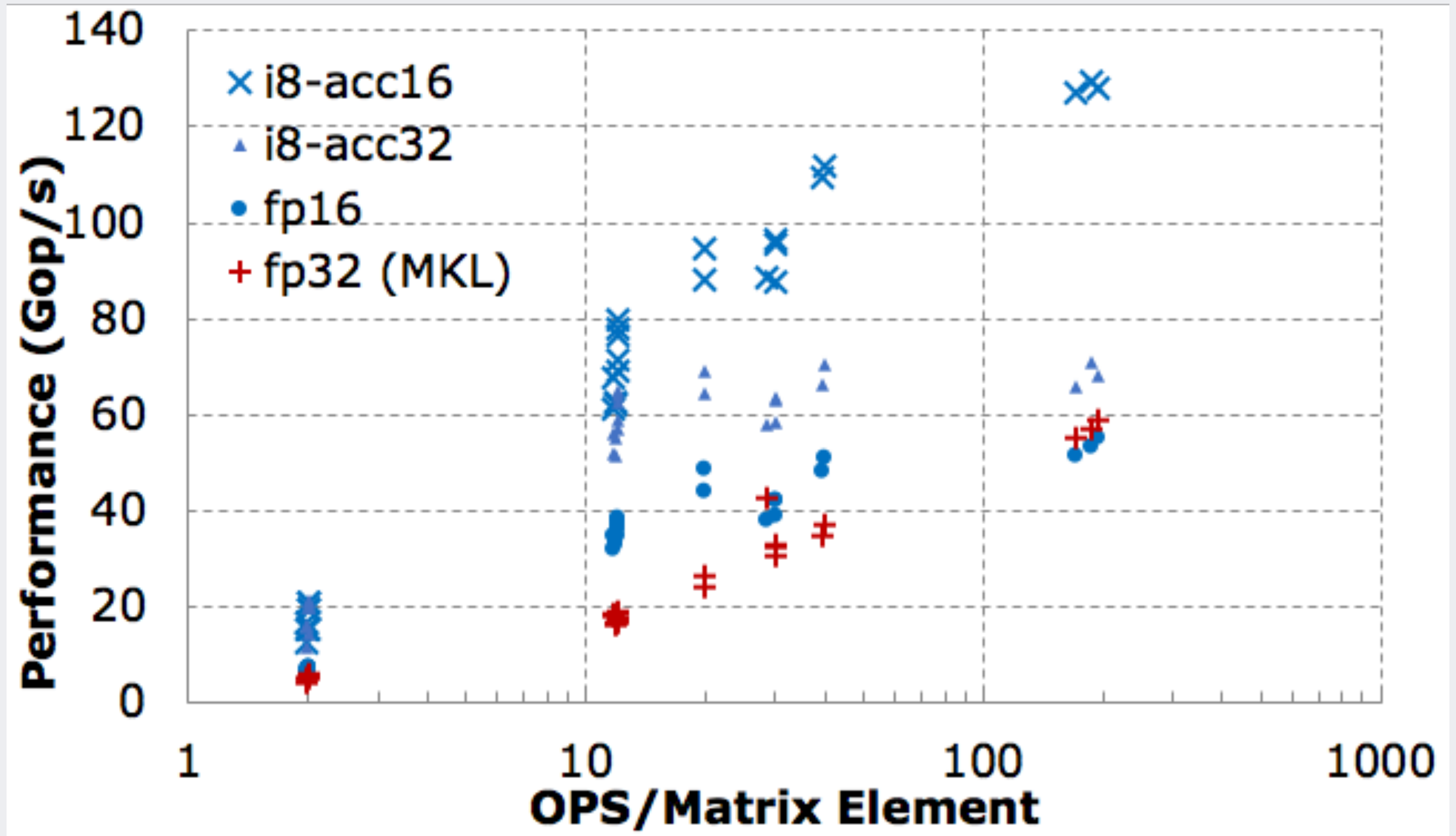
- Fleet-wide DL inference profiling
- **Reduced precision**
- Whole graph optimization

# Reduced-precision Inference

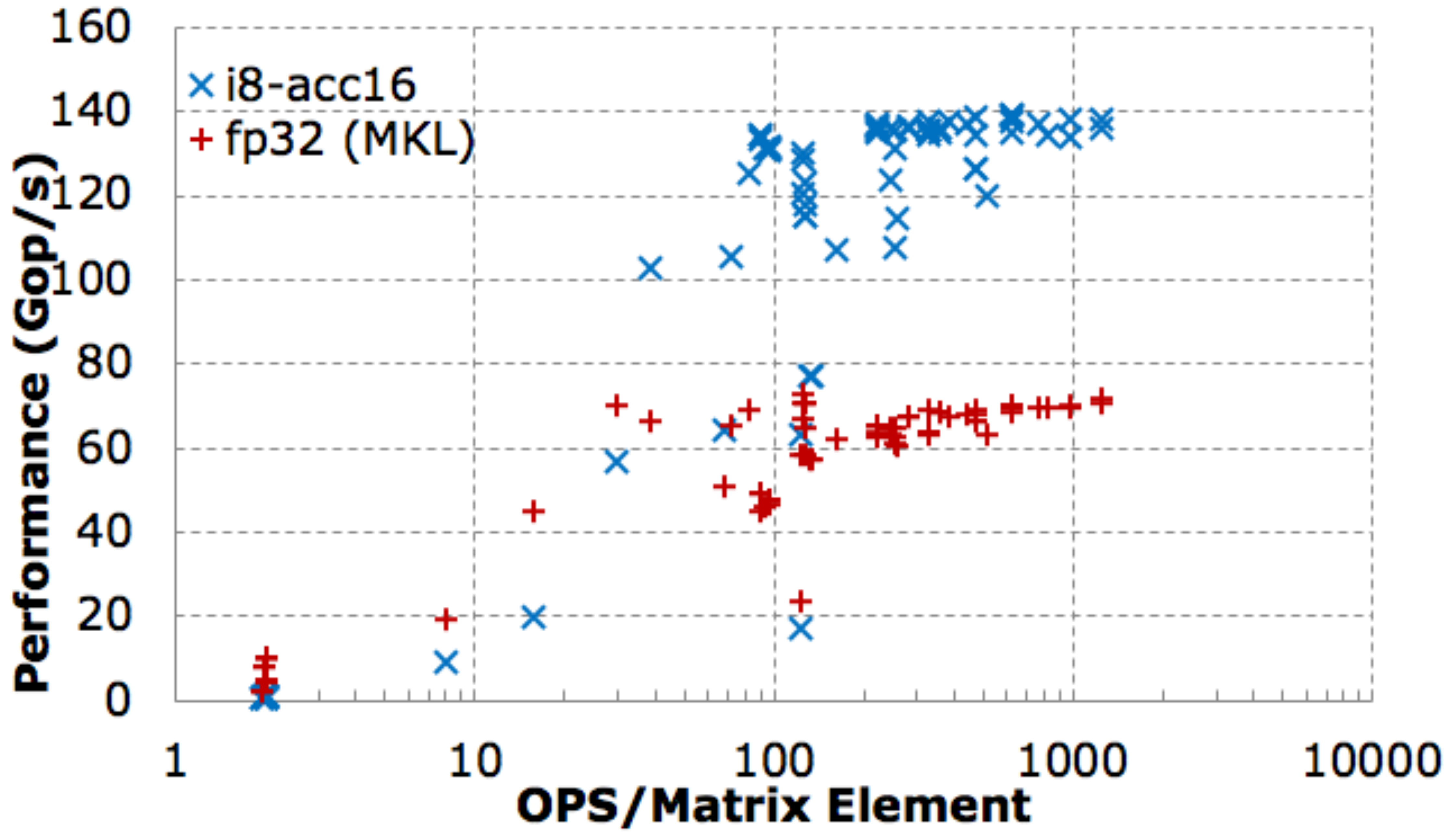
- Performance challenges in current Intel CPUs
  - 8-bit multiplication with 32-bit accumulation instruction throughput not much higher than fp32 (until VNNI is available)
- Accuracy challenges
  - Strict accuracy requirements in data center DL inference

# 16-bit accumulation for high op. intensity cases

Figure credit: Protonu Basu and Daya Khudia




FC




Conv

- Measured with 1 core of Intel Xeon E5-2680 v4 with turbo mode off
- i8-acc32 for low op. intensity case and i8-acc16 for high op. intensity case
- 1.7x in resnet50 and 2.4x in Rosetta (Faster-RCNN-ShuffleNet) over fp32

# Accuracy improving techniques

- resnet50 0.3% top-1 and 0.1% top-5 drop. Similarly small accuracy drops in Faster-RCNN-ShuffleNet, ResNeXt, ResNeXt3D, ...

- **Outlier-aware quantization**
- L2 error minimizing quantization: find a scale and zero_point that minimizes L2 error (similar to Nvidia TensorRT's KL divergence minimization)
- Fine-grain quantization: per output feature quantization (FC), per output channel quantization (Conv), per-entry quantization (Embedding)
- Quantization-aware training: fake quantization (similar to TF)
- Selective quantization: skip layers with high quantization errors (e.g., first Conv layer)
- Net-aware quantization: propagation range constraints (e.g., operators followed by ReLU or sigmoid)

# Outlier-aware Quantization

$$Y = X * W^T = X * (W_1 + W_2)^T$$

$$W_1(i, j) = W(i, j) \text{ if } |W(i, j)| < \text{outlier\_threshold, else } 0$$

$$W_2(i, j) = W(i, j) \text{ if } |W(i, j)| >= \text{outlier\_threshold, else } 0$$

- $W_1$ : dense matrix with small values. Can compute with 16-bit accumulation
- $W_2$ : sparse matrix with big values. Compute with 32-bit accumulation

# Outline

- Introduction to deep learning inference at Facebook
- Computational characteristics
- Optimization experience on current HWs (Intel CPUs)
- **SW/HW Co-design directions**

# DL models are diverse and changing fast

- AlexNet is not interesting to us
- Not all matrix operations have “nice” square matrix shapes
- Video, object detection, multilingual language models demand big on-chip memory. However, solely relying on SRAM without off-chip memory interface is risky
- Embedding lookups demand high capacity and bandwidth memory

# DL inference in data centers vs. inference at edge devices

| | **Data Center** | **Edge Devices** |
|---|---|---|
| **Reduced Precision** | Wants to maintain accuracy. Fp16 fallback can be useful | Trade-off accuracy for energy-efficiency and latency constraints |
| **Model Pruning** | Should focus on speeding up inference (exception: embeddings) | Should focus on model size |

# Q&A