

# **TensorDIMM: A Practical Near-Memory Processing Architecture for Embeddings and Tensor Operations in Deep Learning**

KAIST

**Youngeun Kwon**, Yunjae Lee, and Minsoo Rhu

# Research Scope

# DL architecture research so far

## Primarily focused on “dense” DNN layers (e.g., CNNs, RNNs, ...)

2014 47th Annual IEEE/ACM International Symposium on Microarchitecture

**DaDianNao: A Machine-Learning Supercomputer**

Yunji Chen<sup>1</sup>, T...

2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture

**EIE: Efficient Inference Engine on Compressed Deep Neural Net**

Song Han\* Xingyu L...  
Mark A. ...  
{songhan, xyli, huizi, jing...

2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture

**GANAX: A Unified MIMD-SIMD Acceleration for Generative Adversarial Networks**

Amir Yazdanbakhsh<sup>1</sup> Kambiz Samadi<sup>2</sup> Nam Sung Kim<sup>3</sup> Hadi Esmaeilzadeh<sup>4</sup>  
Alternative Computing Technologies (ACT) Lab  
Georgia Institute of Technology <sup>1</sup>Qualcomm Technologies, Inc. <sup>2</sup>University of Illinois at Urbana-Champaign <sup>3</sup>UC San Diego  
ayazdanbakhsh@gatech.edu ksamadi@qti.qualcomm.com nskim@uiuc.edu hadi@erg.ucsd.edu

**Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks**

Yu-Hsin Chen<sup>1</sup>, Joel Emer<sup>2\*</sup>

<sup>1</sup>EECS, MIT  
Cambridge, MA 02139

\*{yhchen, jsemer, s...

**Abstract**—Deep convolutional neural networks (CNNs) are widely used in modern AI systems for their superior accuracy but at the cost of high computational complexity. The complexity comes from the need to simultaneously process hundreds of filters and channels in the high-dimensional convolutions, which involve a significant amount of data movement. Although highly-parallel compute paradigms, such as SIMD/SIMT, effectively address the computation requirement to achieve high throughput, energy consumption still remains high as data movement can be more expensive than computation. Accordingly, finding a dataflow that supports parallel processing with minimal data movement cost is crucial to achieving energy-efficient CNN processing without compromising accuracy. In this paper, we present a novel dataflow, called *restationary* (RS), that minimizes data movement energy con...

**SCNN: An Accelerator for Compressed-sparse Convolutional Neural Networks**

Angshuman Parashar<sup>1</sup>  
Rangharajan Venkatesan<sup>2</sup> Bruce ...  
NVIDIA<sup>1</sup> Massachusetts Ins...

**Abstract**—State-of-the-art deep neural networks have hundreds of millions of connections and are both computationally and memory intensive, making them difficult to deploy on embedded systems with limited hardware resources and power budgets. While custom hardware helps the cost of fetching weights from DRAM is two orders of magnitude more expensive than ALU operations, and dominates the total power. Previously proposed “Deep Compression” makes to fit large DNNs (AlexNet and VGGNet) fully SRAM. This compression is achieved by pruning the connections and having multiple connections share weights. We propose an energy-efficient inference engine that performs inference on this compressed network accelerates the resulting sparse matrix-vector multiplication with weight sharing. Going from DRAM to SRAM (120x energy saving; Exploiting sparsity saves 10x sharing gives 8x; Skipping zero activations from 1x to another 3x. Evaluated on nine DNN benchmarks 189x and 13x faster when compared to CPU implementations of the same DNN without compression. Our SCNN accelerator has a processing power of 102 GOPS working on a 100mm<sup>2</sup> die. Our SCNN accelerator can leverage highly-parallel compute paradigms, such as SIMD/SIMT, throughput may not scale accordingly due to the accompanying bandwidth requirement, and the energy consumption remains high as data movement can be more expensive than computation [11–13]. In order to achieve energy-efficient CNN processing without compromising throughput, we need to develop dataflows that support parallel processing with minimal data movement. The differences in data movement energy cost based on where the data is...

**Abstract**—Generative Adversarial Networks (GANs) are one of the most recent deep learning models that generate synthetic data from limited genuine datasets. GANs are on the frontier as further extension of deep learning into many domains (e.g., medicine, robotics, content synthesis) requires massive sets of labeled data that is generally either unavailable or prohibitively costly to collect. Although GANs are gaining prominence in various fields, there are no accelerators for these new models. In fact, GANs leverage a new operator, called transposed convolution, that exposes unique challenges for hardware acceleration. This operator first inserts zeros within the multidimensional input, then convolves a kernel over this expanded array to add information to the embedded zeros. Even though there is a convolution stage in this operator, the inserted zeros lead to underutilization of the compute resources when a conventional convolution accelerator is employed. We propose the GANAX architecture to alleviate the sources of inefficiency associated with the acceleration of GANs using conventional convolution accelerators, making the first GAN accelerator design possible. We propose a reorganization of the output computations to allocate compute rows with similar patterns of zeros to adjacent processing engines, which also avoids inconsequential multiply-adds on the zeros. This compulsory adjacency reclaims data reuse across these neighboring processing engines, which had otherwise diminished due to the inserted zeros. The reordering breaks the full SIMD execution model, which is prominent in convolution accelerators. Therefore, we propose a unified MIMD-SIMD design for GANAX that leverages repeated patterns in the computation to create distinct microprograms that execute concurrently in SIMD mode. The interleaving of MIMD and SIMD modes is performed at the granularity of single microprogrammed operation. To amortize the cost of MIMD execution, we propose a decoupling of data access from data processing in GANAX. Two tasks: (1) training — in which the parameters of a neural network are learned by observing massive numbers of training examples, and (2) inference — in which a trained neural network is deployed in the field and classifies the observed data. Today, training is often done on GPUs [27] or farms of GPUs, while inference depends on the application and can employ CPUs, GPUs, FPGAs or specially-built ASICs. During the training process, a deep learning expert will typically architect the network, establishing the number of layers, the operation performed by each layer, and the connectivity between layers. Many layers have parameters, typically filter weights, which determine their exact computation. The objective of the training process is to learn these weights, usually via a stochastic gradient...

**ABSTRACT**

Convolutional Neural Networks (CNNs) have become a fundamental technology for machine learning. High extreme energy efficiency are critical for deployment especially in mobile platforms such as autonomous vehicles and electronic personal assistants. This paper introduces SCNN (SCNN) accelerator architecture, which improves performance and energy efficiency by exploiting the zero-valued from network pruning during training and zero-valued that arise from the common ReLU operator. Specifically, SCNN employs a novel dataflow that enables maintaining the sparse weights and activations in a compressed encoding, which eliminates unnecessary data transfers and reduces storage requirements. Furthermore, the SCNN dataflow facilitates efficient delivery of those weights and activations to a multiplier array, where they are extensively reused; product accumulation is performed in a novel accumulator array. On contemporary neural networks, SCNN can improve both performance and energy by a factor of 2.7x and 2.3x, respectively, over a comparably provisioned dense CNN accelerator.

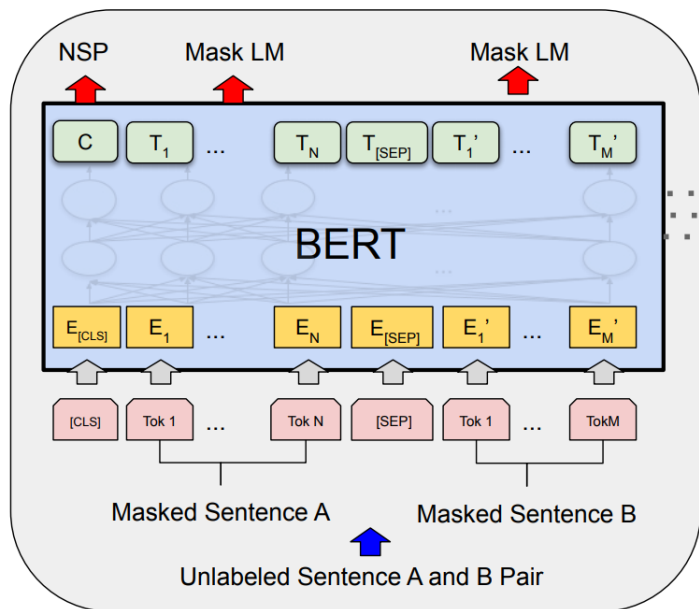
**CCS CONCEPTS**

• Computer systems organization → Architectures; Parallel ar...

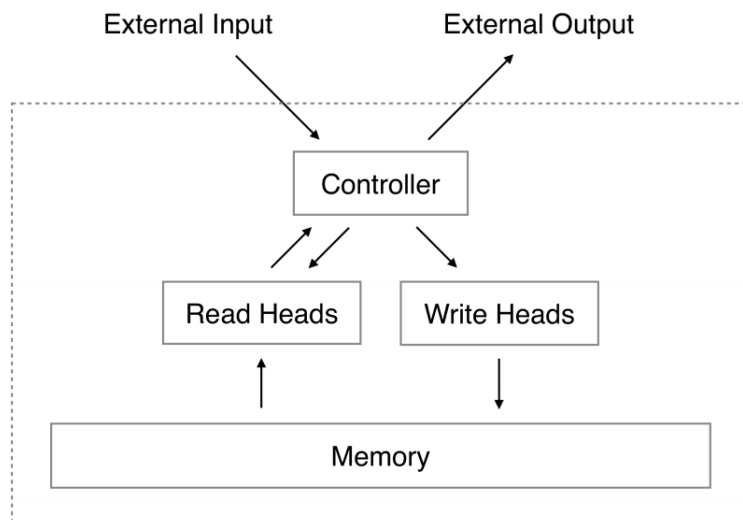
- \* Chen et al., “DaDianNao: A Machine-Learning Supercomputer”, ISCA-2014
- \* Chen et al., “Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks”, ISCA-2016
- \* Han et al., “EIE: Efficient Inference Engine on Compressed Deep Neural Network”, ISCA-2016
- \* Parashar et al., “SCNN: An Accelerator for Compressed-sparse Convolutional Neural Networks”, ISCA-2017
- \* Yazdanbakhsh et al., “GANAX: A Unified MIMD-SIMD Acceleration for Generative Adversarial Networks”, ISCA-2018

# Emerging DL applications?

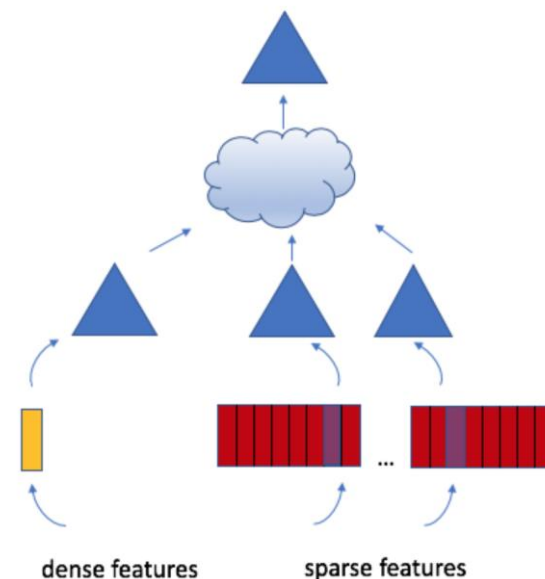
“Non” conventional DNN layers are causing a bottleneck



**BERT**



**Neural Turing Machine**



**Recommendation**

\* Devlin et al., “Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding”, arxiv.org, 2018

\* Graves et al., “Neural Turing Machines”, arxiv.org, 2014

\* Naumov et al., “Deep Learning Recommendation Model for Personalization and Recommendation Systems”, arxiv.org, 2019

# Personalized recommendation models

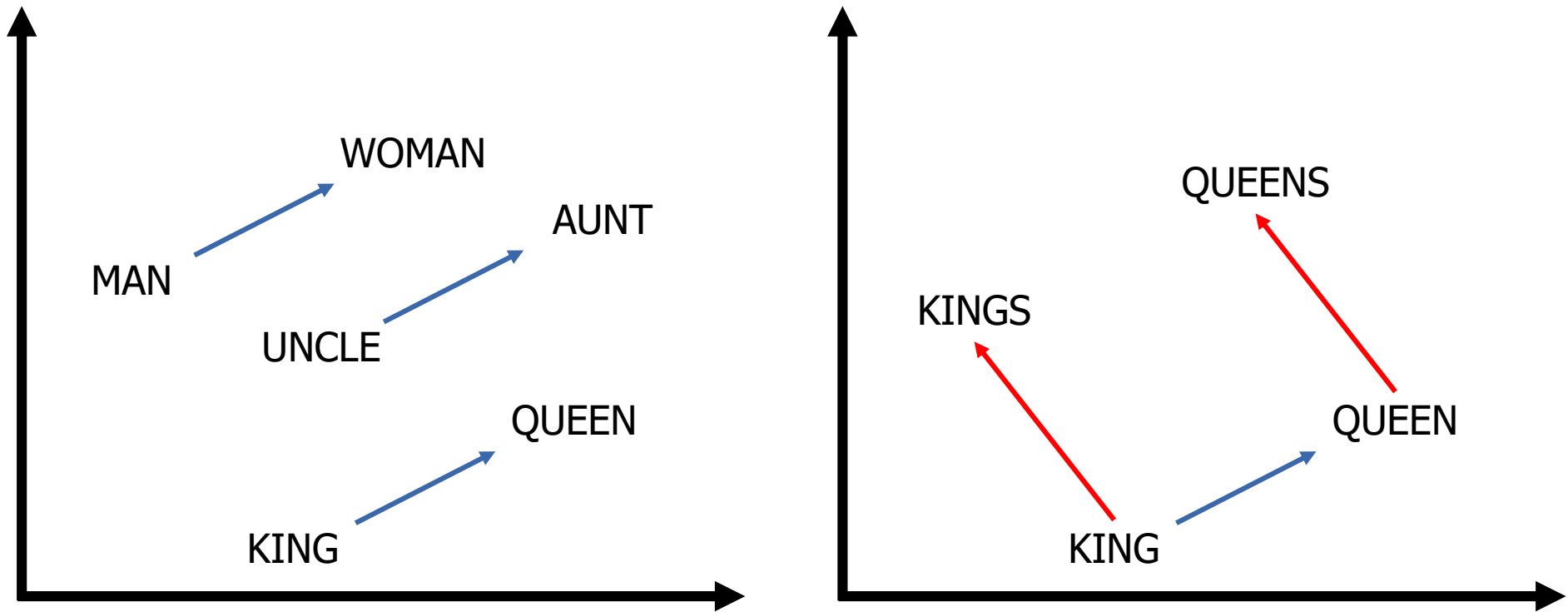
“Sparse” embedding layers (rather than dense DNNs) are the bottleneck

The image is a collage of four screenshots demonstrating personalized recommendations:

- Restaurants:** A screenshot of a restaurant search interface. A red box highlights a list of restaurant results, including 'Conchito Tacos', 'Pizza Rustica', 'Ardenic Restaurant', 'Max & Erma's', and 'Sandman Express Downtown'.
- Top Free iPhone Apps:** A screenshot of the 'Top Free iPhone Apps' list. A red box highlights the top row of app icons, including Candy Crush, Facebook Messenger, Facebook, Instagram, iTunes U, YouTube, New Words, Snapchat, and Pandora.
- Amazon.com:** A screenshot of the Amazon.com homepage. A red box highlights the 'Recommended for You' section, which features three book recommendations: 'Google Apps Deciphered: Compute in the Cloud to Streamline Your Desktop', 'Google Apps Administrator Guide: A Private-Label Web Workspace', and 'Googlepedia: The Ultimate Google Resource (3rd Edition)'.
- Netflix:** A screenshot of the Netflix homepage. A red box highlights the 'Trending Now' section, which features a row of movie and TV show thumbnails including 'Prison Break', 'Narcos', 'The Mindy Project', 'Modern Family', 'Peaky Blinders', and 'Arrow'.

# What is an embedding?

Projection of sparse features into dense vector dimension (e.g., word2vec)



# What is an embedding?

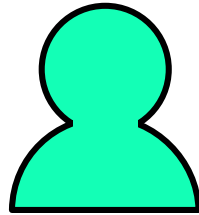
Stored as a large look-up table containing millions-to-billions of entries

User ID	Embedding (vector)
0: Sam	[0.49, 0.52, 0.23, 0.69, 0.32, ...]
1: Harry	[0.24, 0.27, 0.13, 0.09, 0.79, ...]
2: Matt	[0.31, 0.71, 0.46, 0.91, 0.07, ...]
3: John	[0.83, 0.43, 0.81, 0.57, 0.09, ...]
4: Elicia	[0.31, 0.83, 0.23, 0.69, 0.86, ...]
...	...
N: Danny	[0.77, 0.18, 0.71, 0.59, 0.46, ...]

N: can be millions

# Recommendation model 101

Goal: predict a preference of user-item pair



Sam

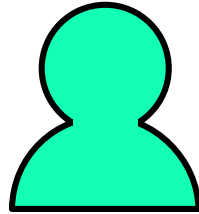


# Recommendation model 101

Goal: predict a preference of user-item pair



Movie 0
Movie 1
Movie 2
Movie 3
Movie 4
Movie 5
Movie 6
Movie 7
...
Movie N

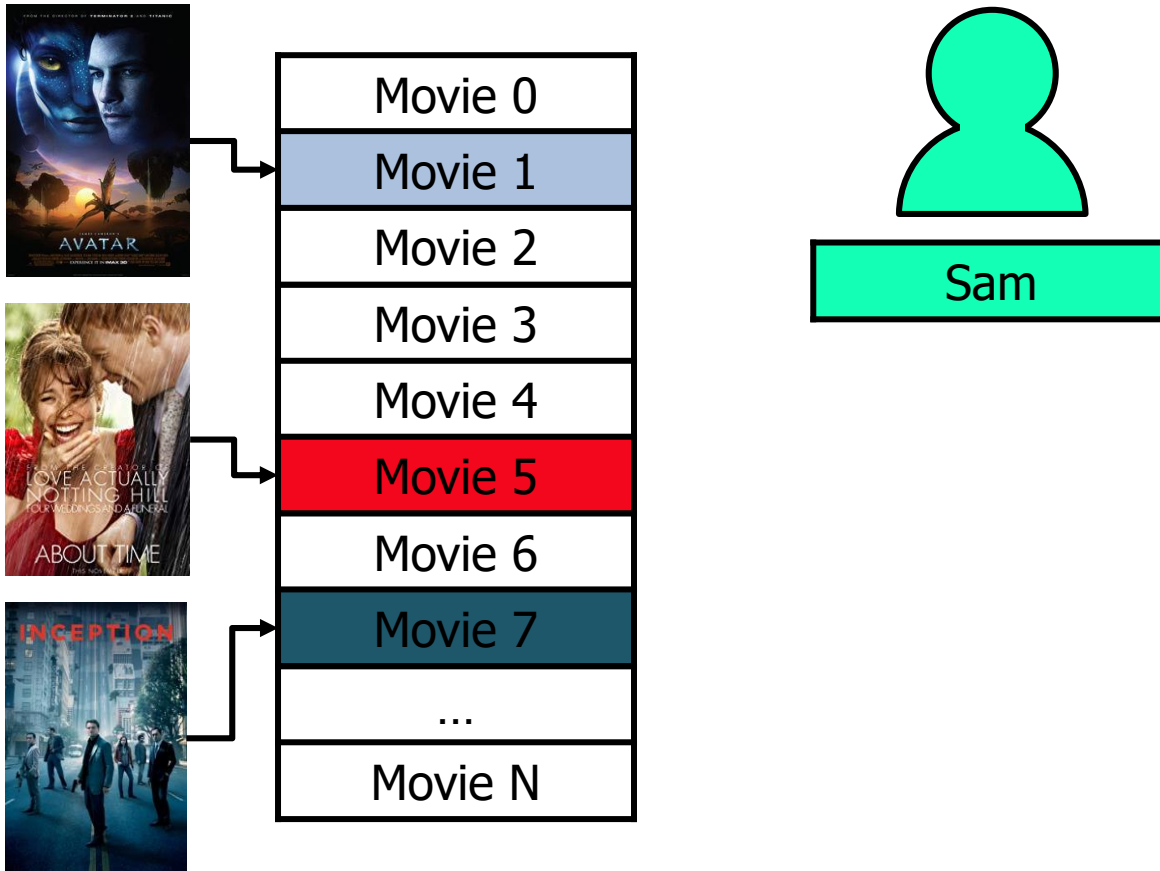


Sam

[Embedding table]

# Recommendation model 101

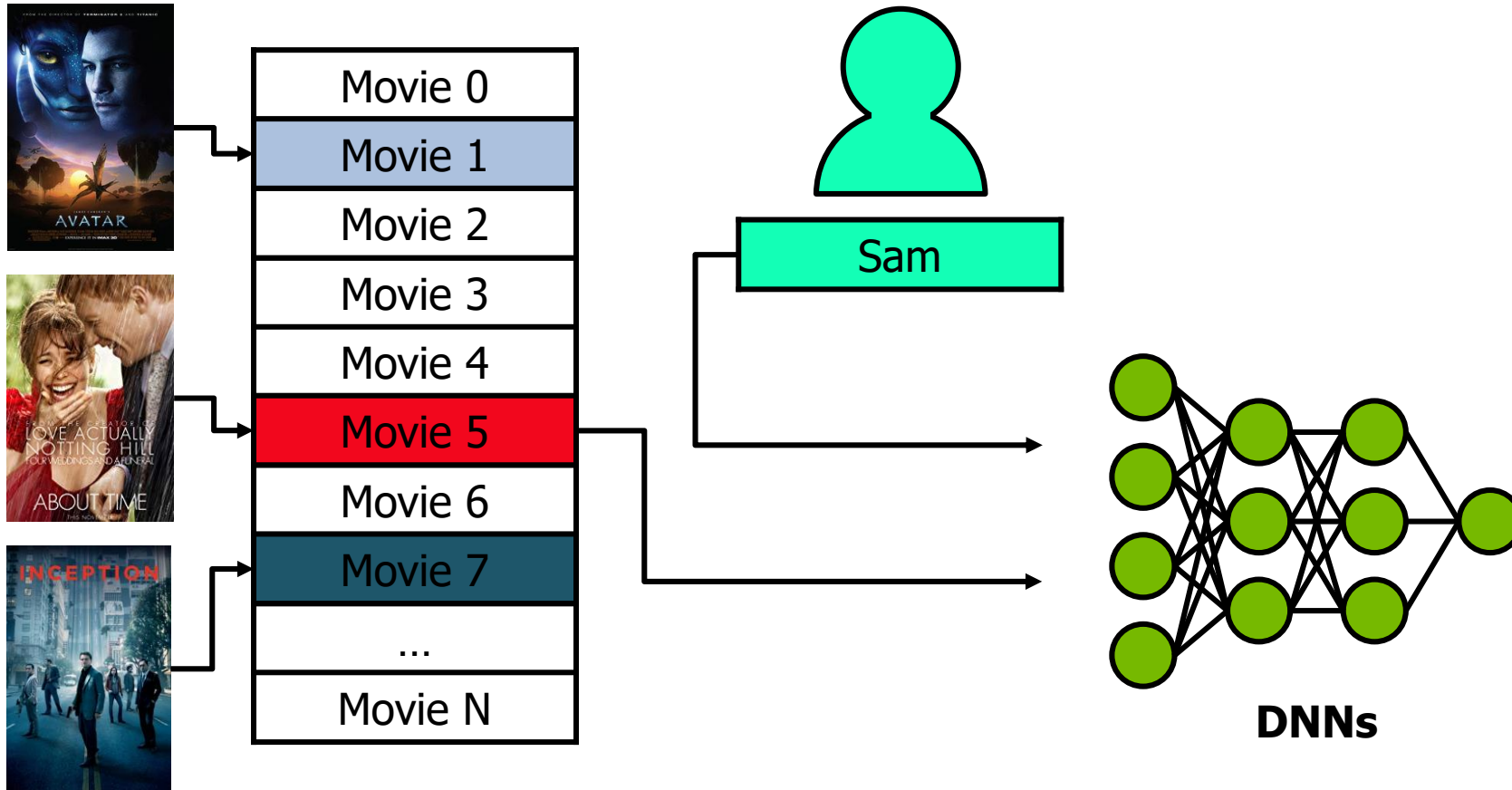
Goal: predict a preference of user-item pair



[Embedding table]

# Recommendation model 101

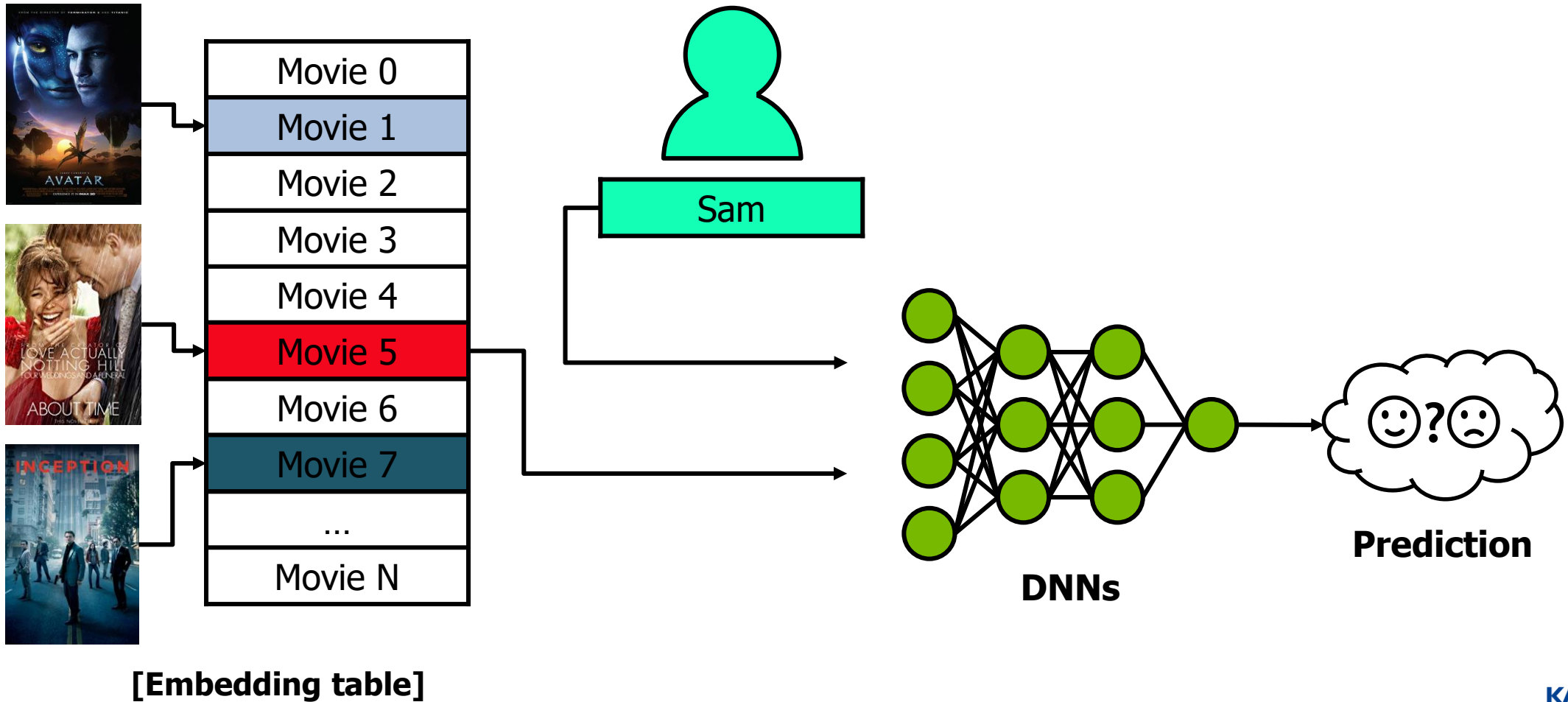
Goal: predict a preference of user-item pair



[Embedding table]

# Recommendation model 101

Goal: predict a preference of user-item pair



# **Key Primitives in Embedding Layers**

# #1: Embedding lookup (gather)

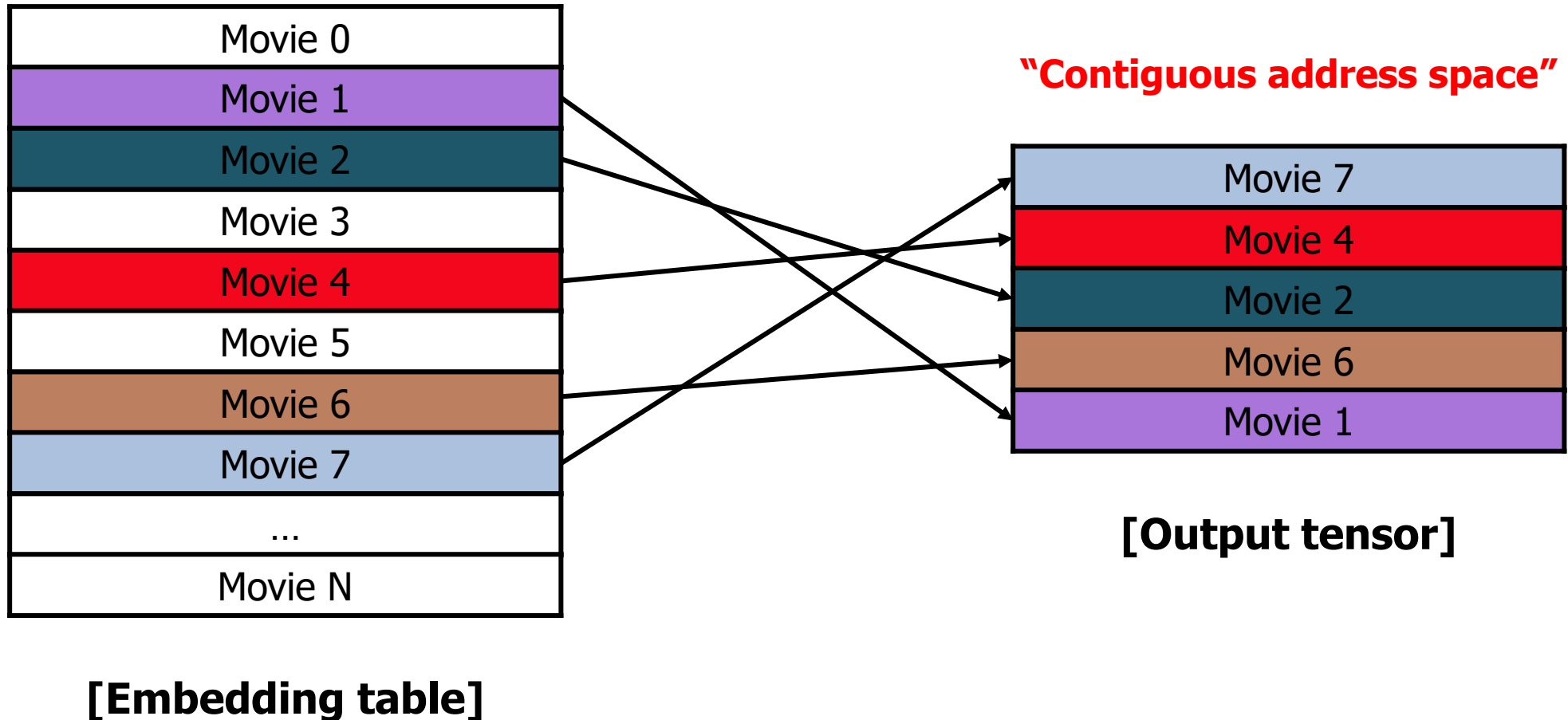
Copying target embeddings into contiguous address space

Movie 0
Movie 1
Movie 2
Movie 3
Movie 4
Movie 5
Movie 6
Movie 7
...
Movie N

**[Embedding table]**

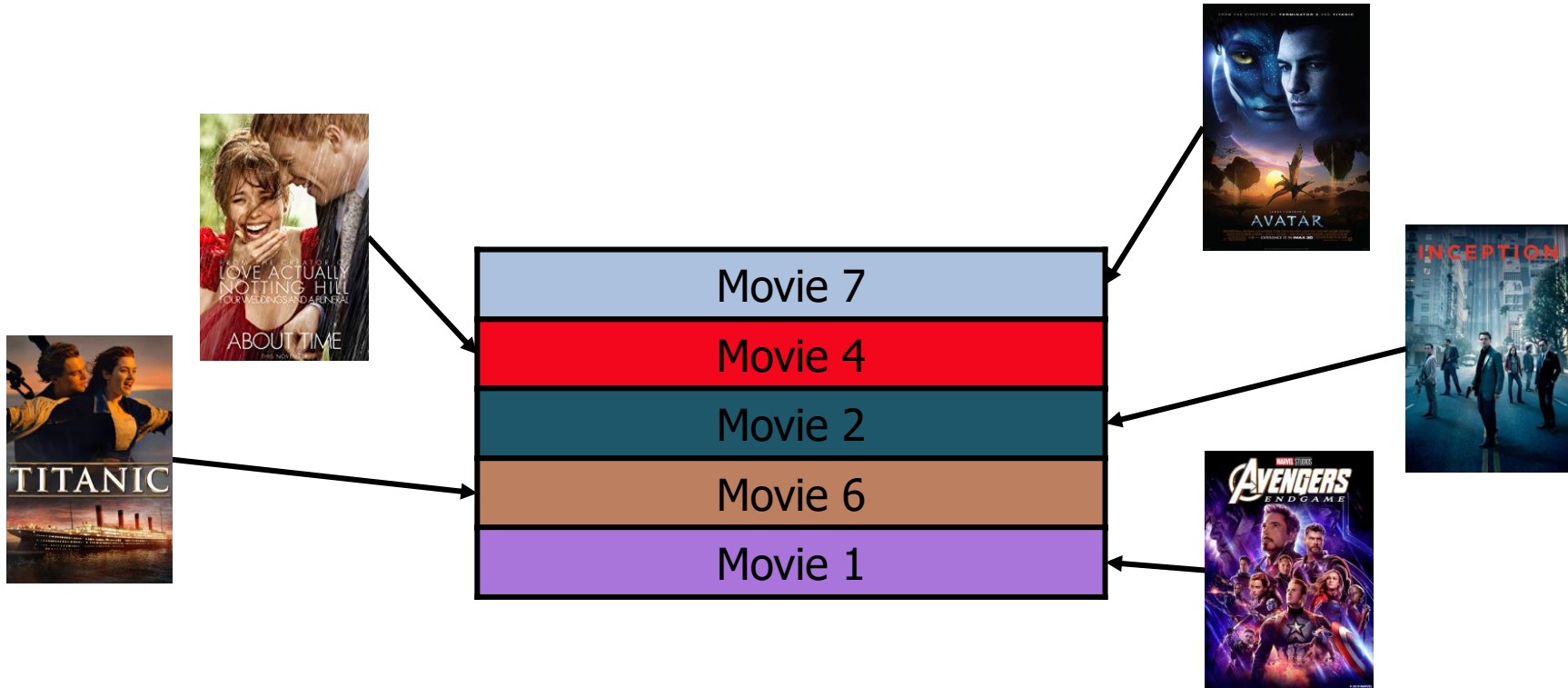
# #1: Embedding lookup (gather)

Copying target embeddings into contiguous address space



# #2: Tensor operation (reduction)

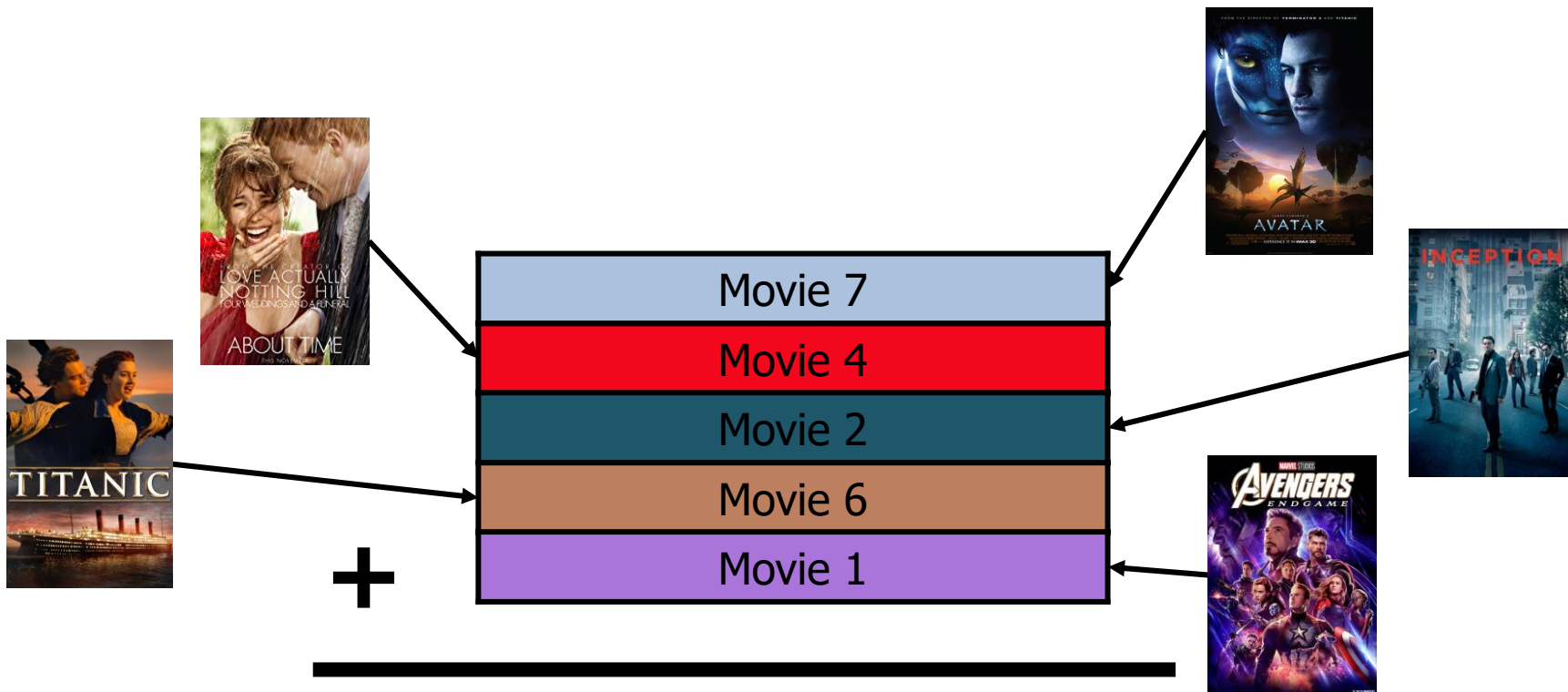
e.g., Averaging multiple embeddings, element-wise addition/multiplication





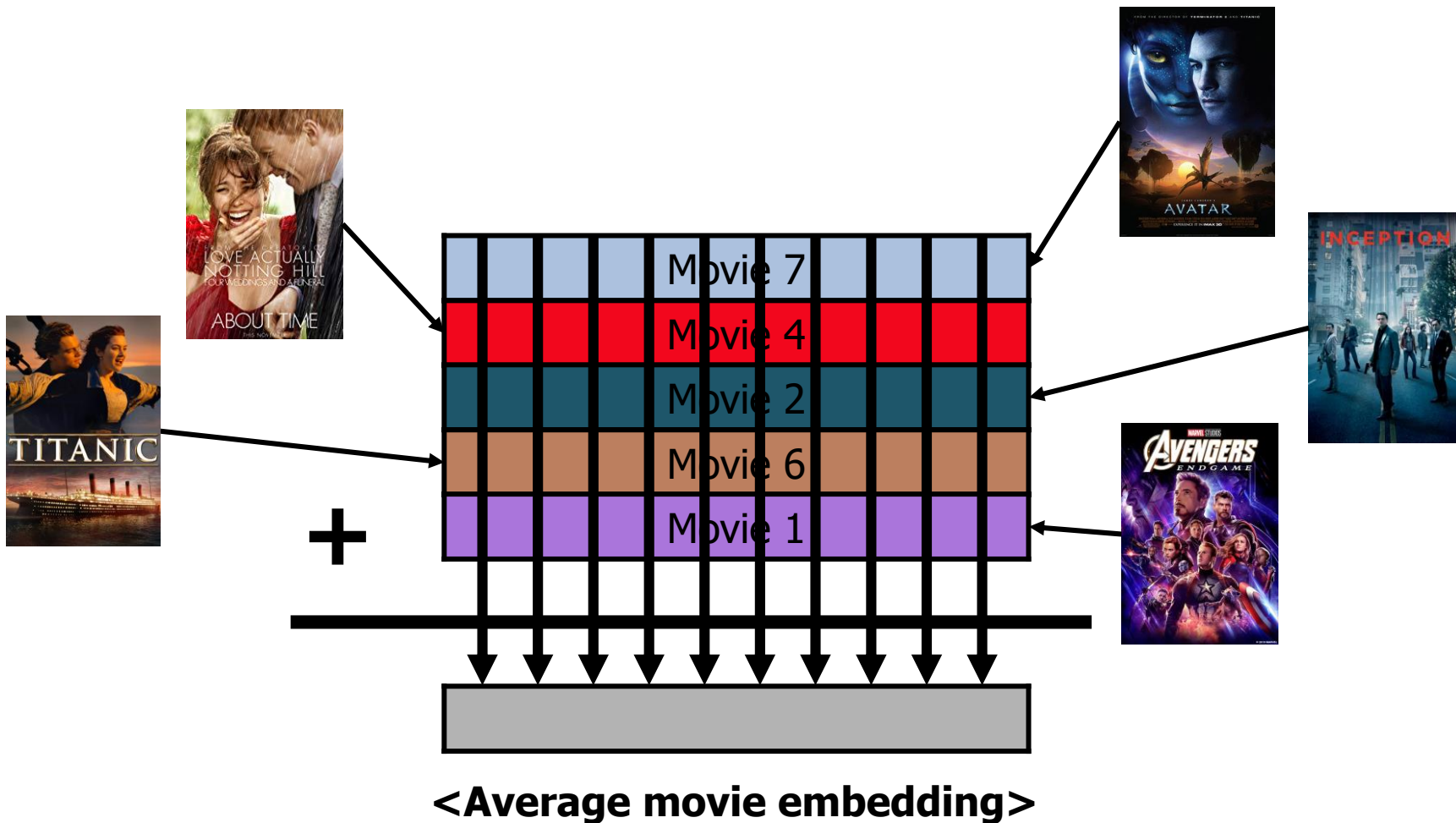
# #2: Tensor operation (reduction)

e.g., Averaging multiple embeddings, element-wise addition/multiplication



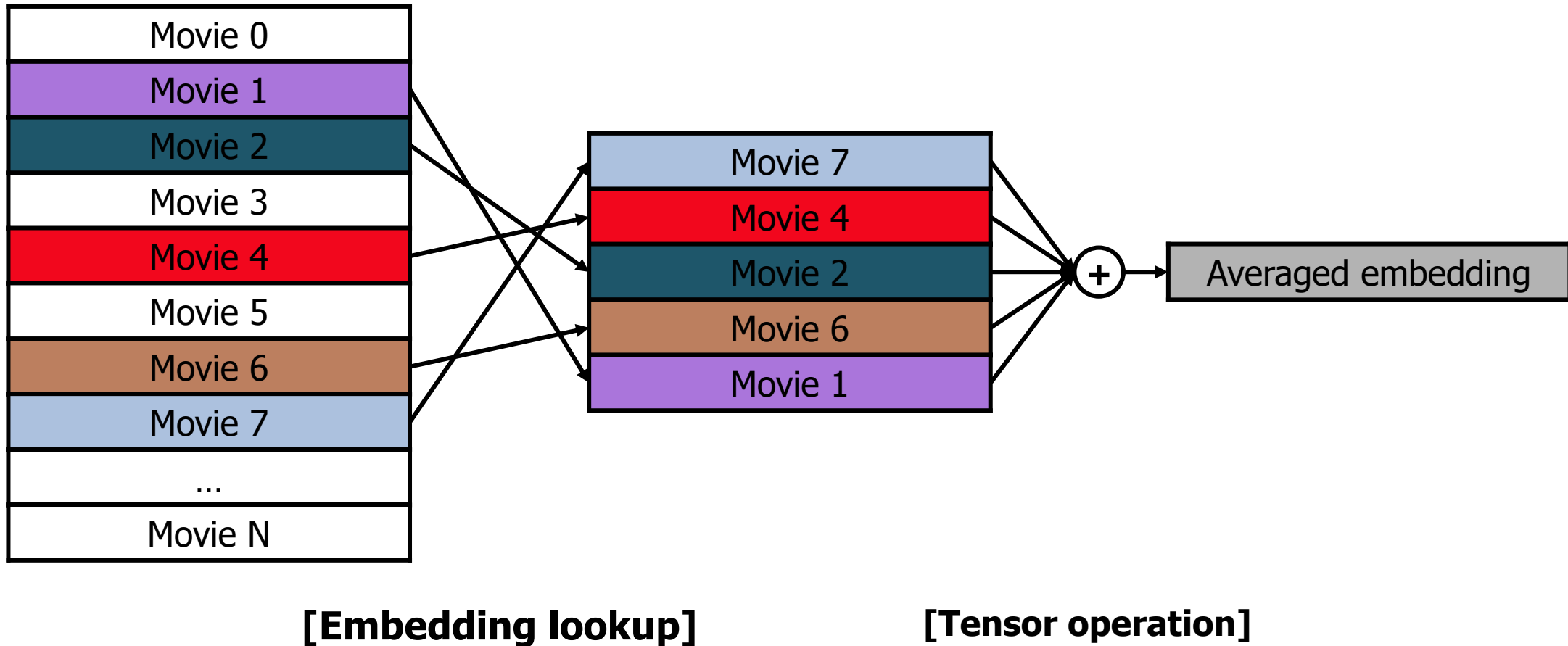
# #2: Tensor operation (reduction)

e.g., Averaging multiple embeddings, element-wise addition/multiplication



# Key challenges of embedding layers

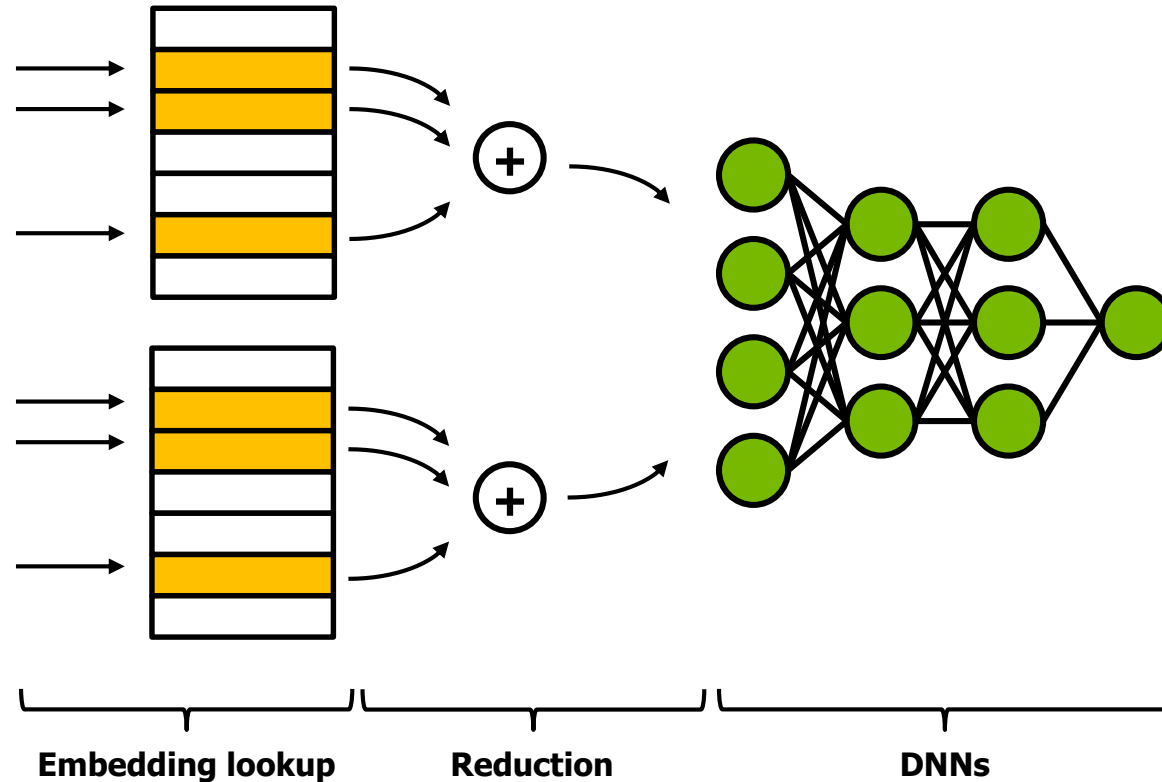
Embedding gathers/reductions are extremely memory-bandwidth sensitive



# **Current Solutions for Recommendation Systems**

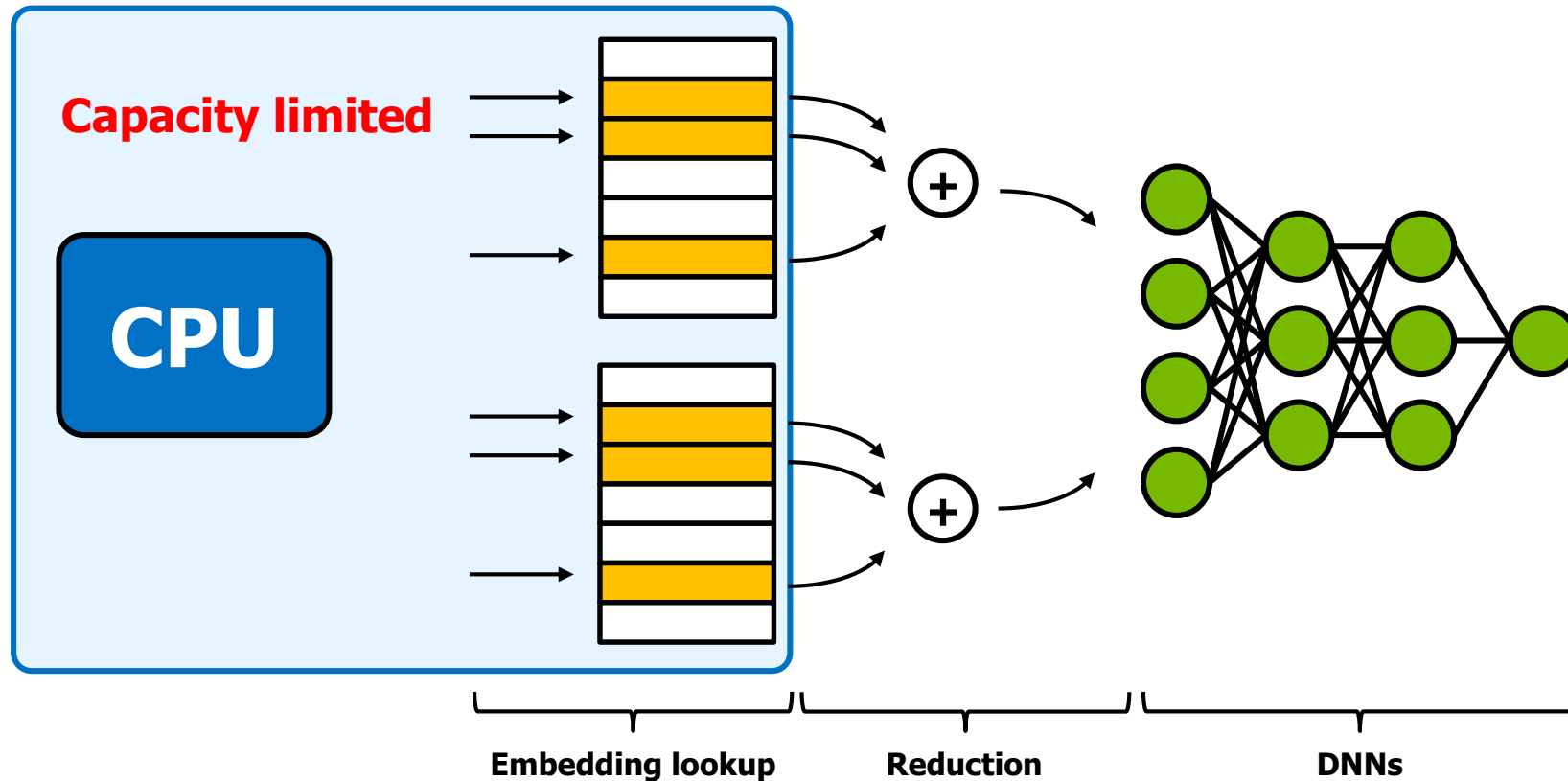
# The memory wall for "Inference"

Size of embedding tables can reach hundreds of GBs



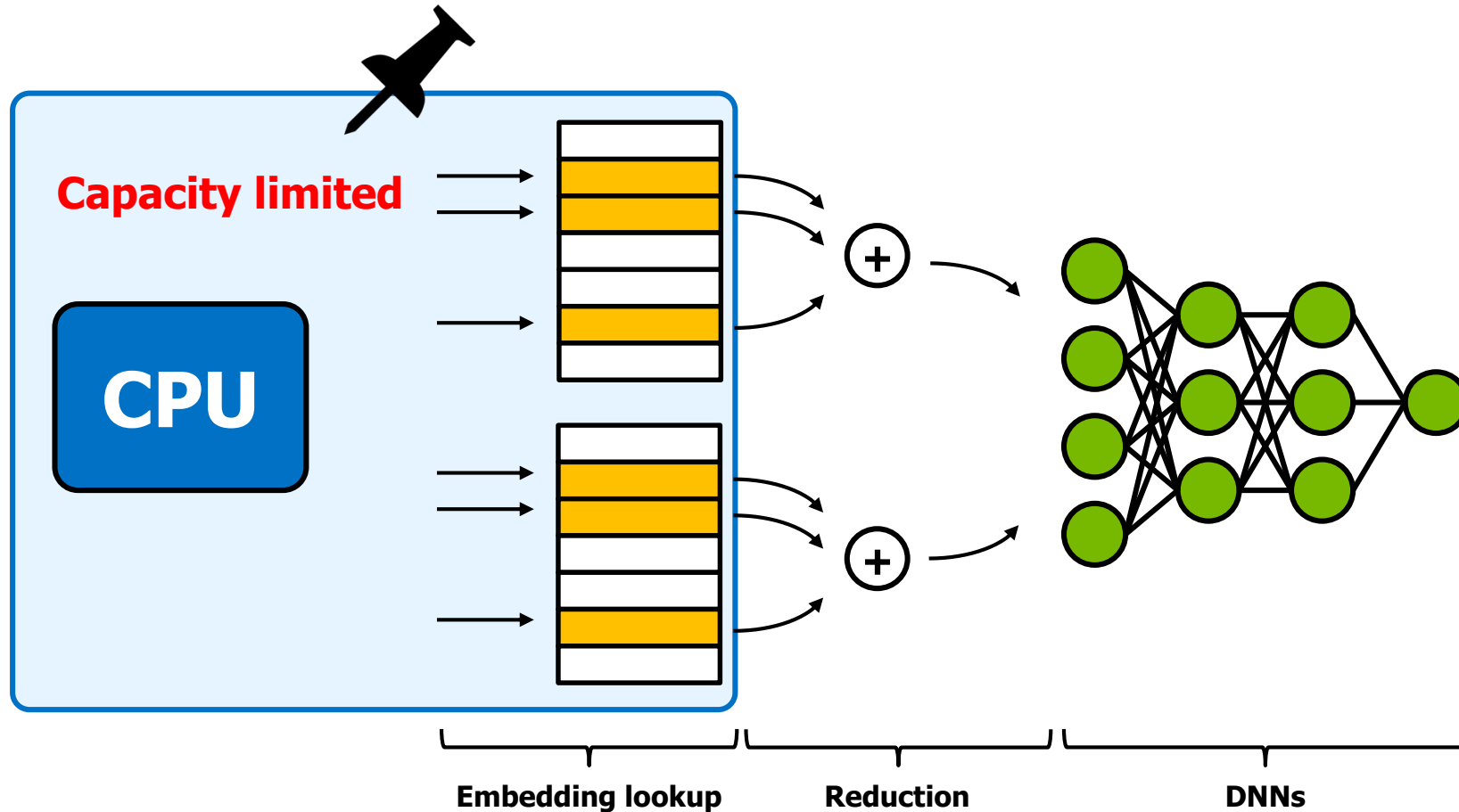
# The memory wall for "Inference"

Size of embedding tables can reach hundreds of GBs



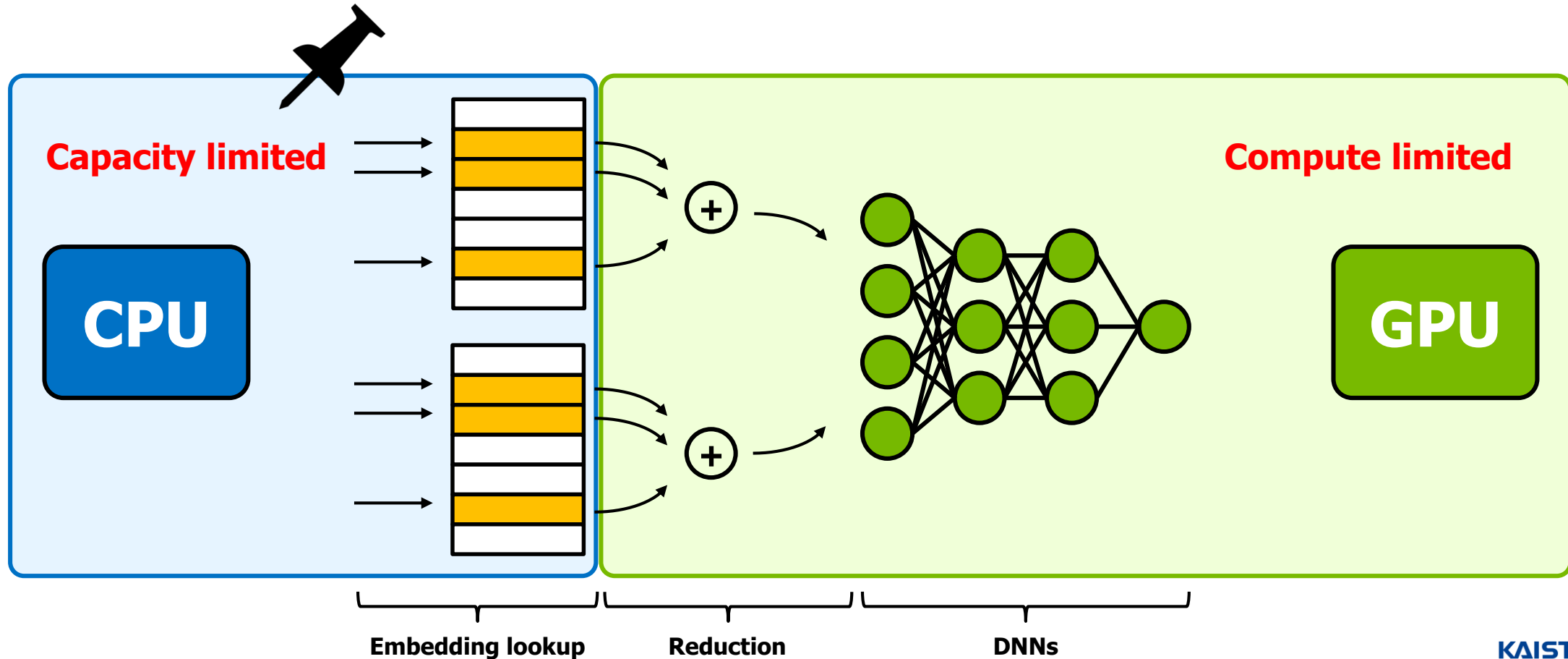
# The memory wall for "Inference"

Size of embedding tables can reach hundreds of GBs



# Design#1: Hybrid CPU-GPU approach

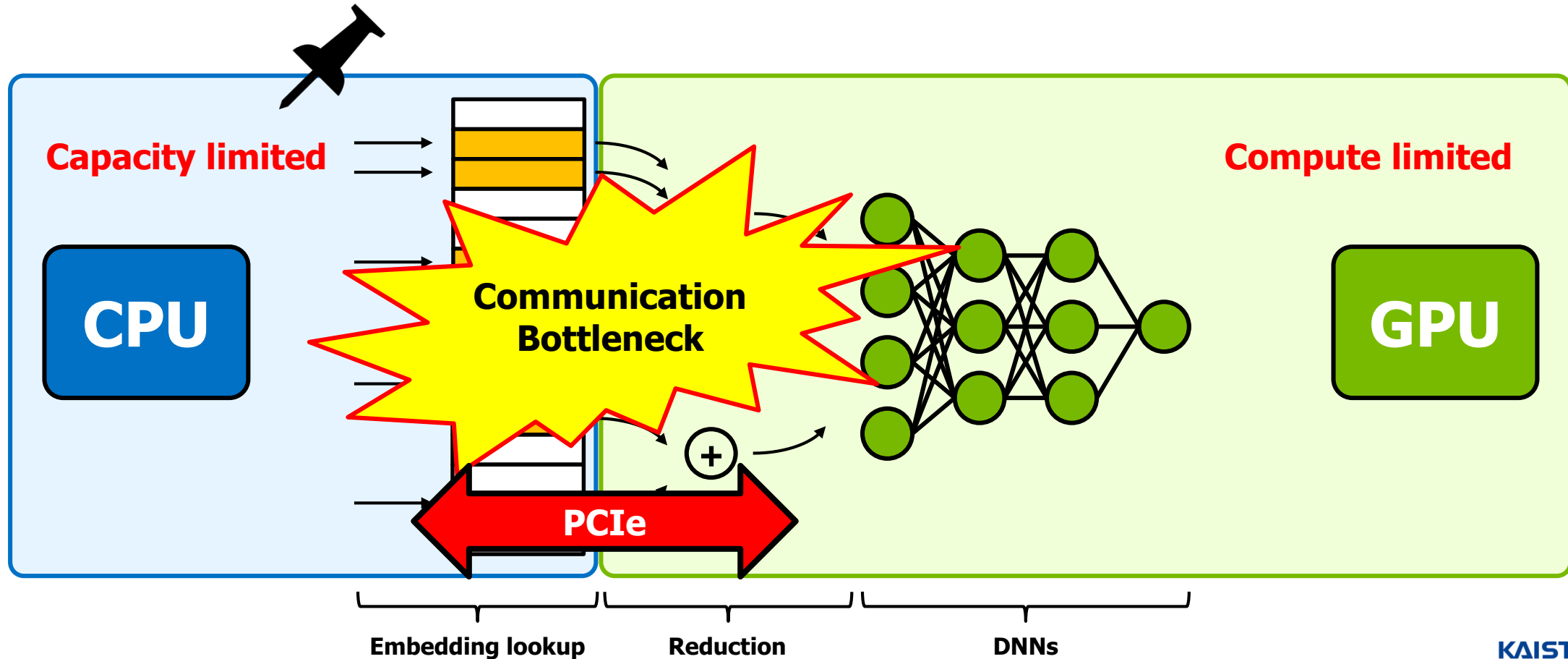
CPU stores entire embedding tables, but DNNs executed using GPUs





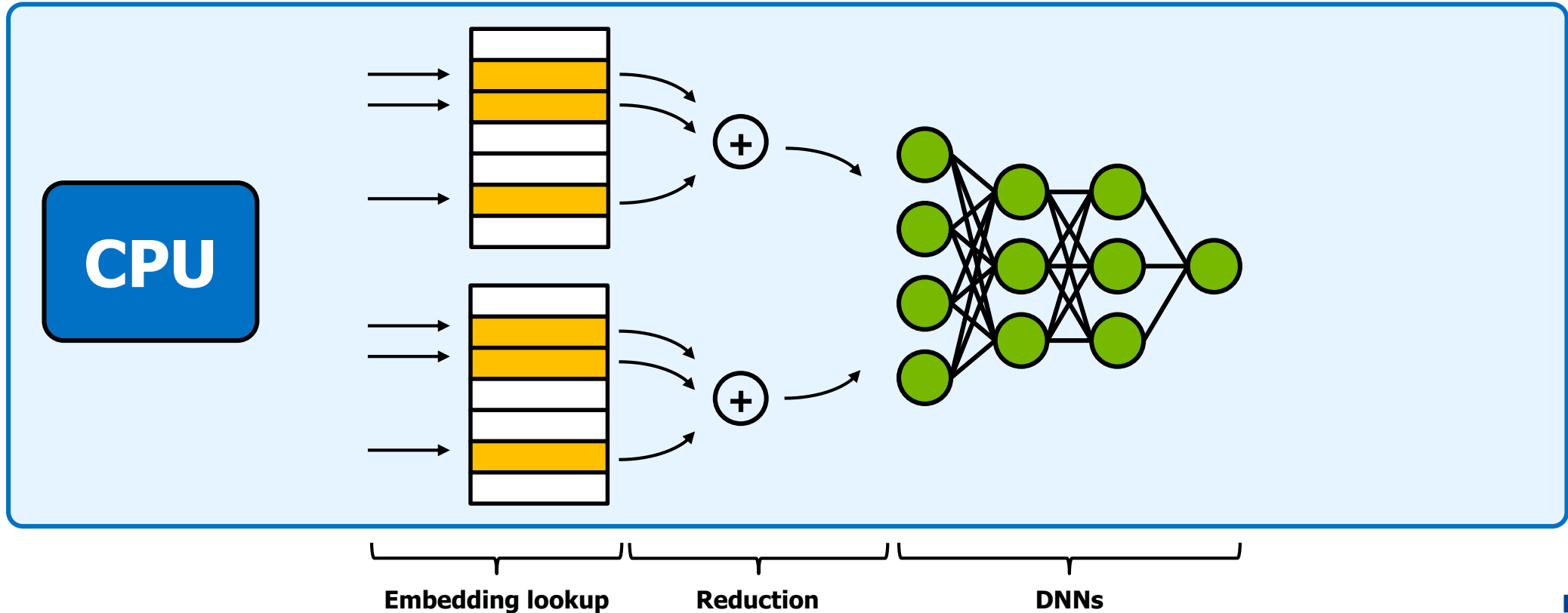
# Design#1: Hybrid CPU-GPU approach

Challenges: need to copy multiple embeddings via narrow PCIe channel



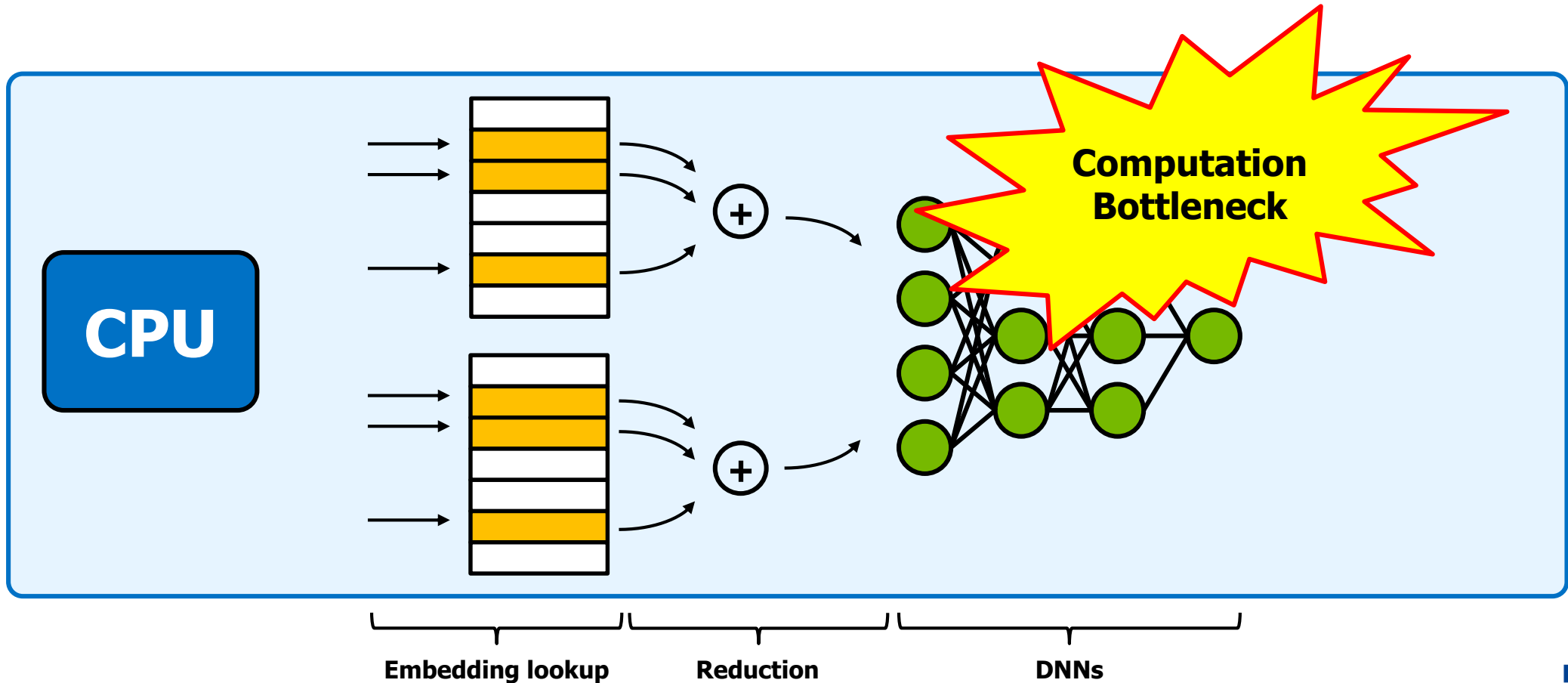
# Design#2: CPU-only approach

The CPU handles the entire steps of inference



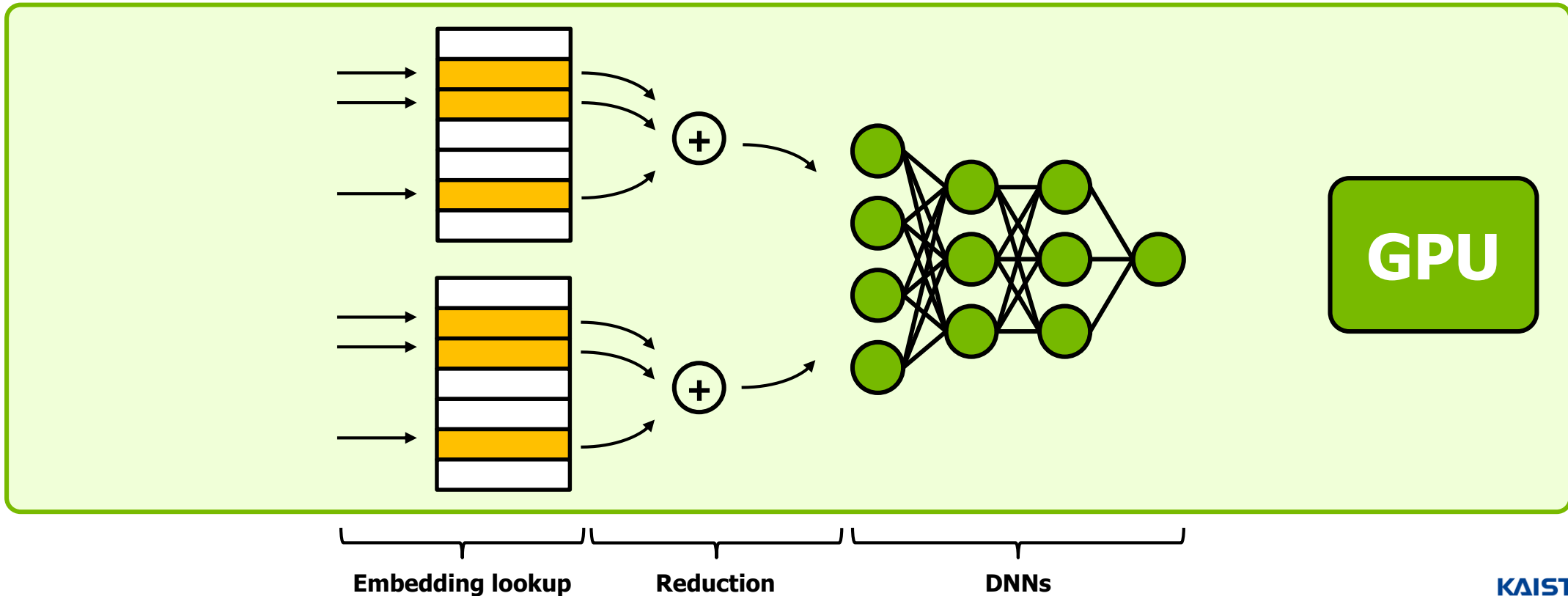
# Design#2: CPU-only approach

Challenges: low throughput of CPUs slows down DNN computation



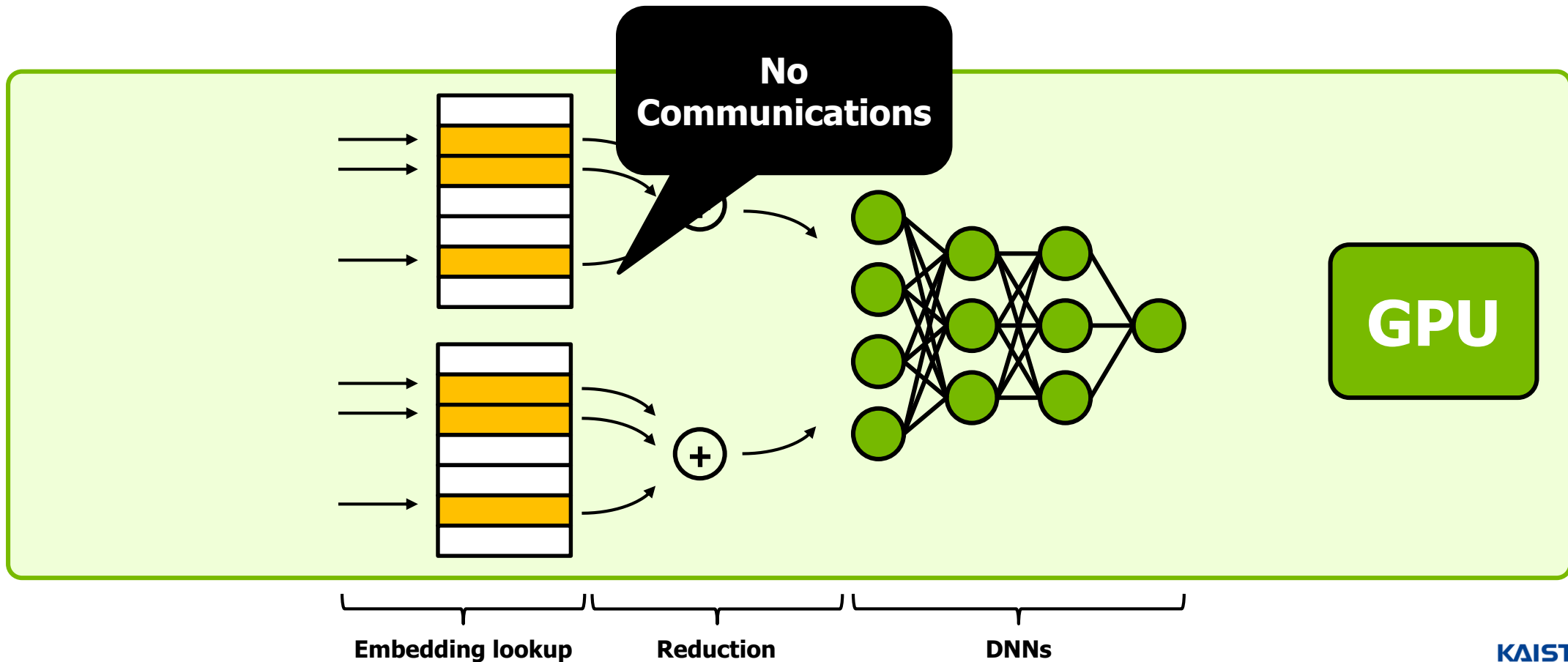
# Design#3: GPU-only approach?

Unbuildable, oracular solution assuming infinite GPU memory capacity



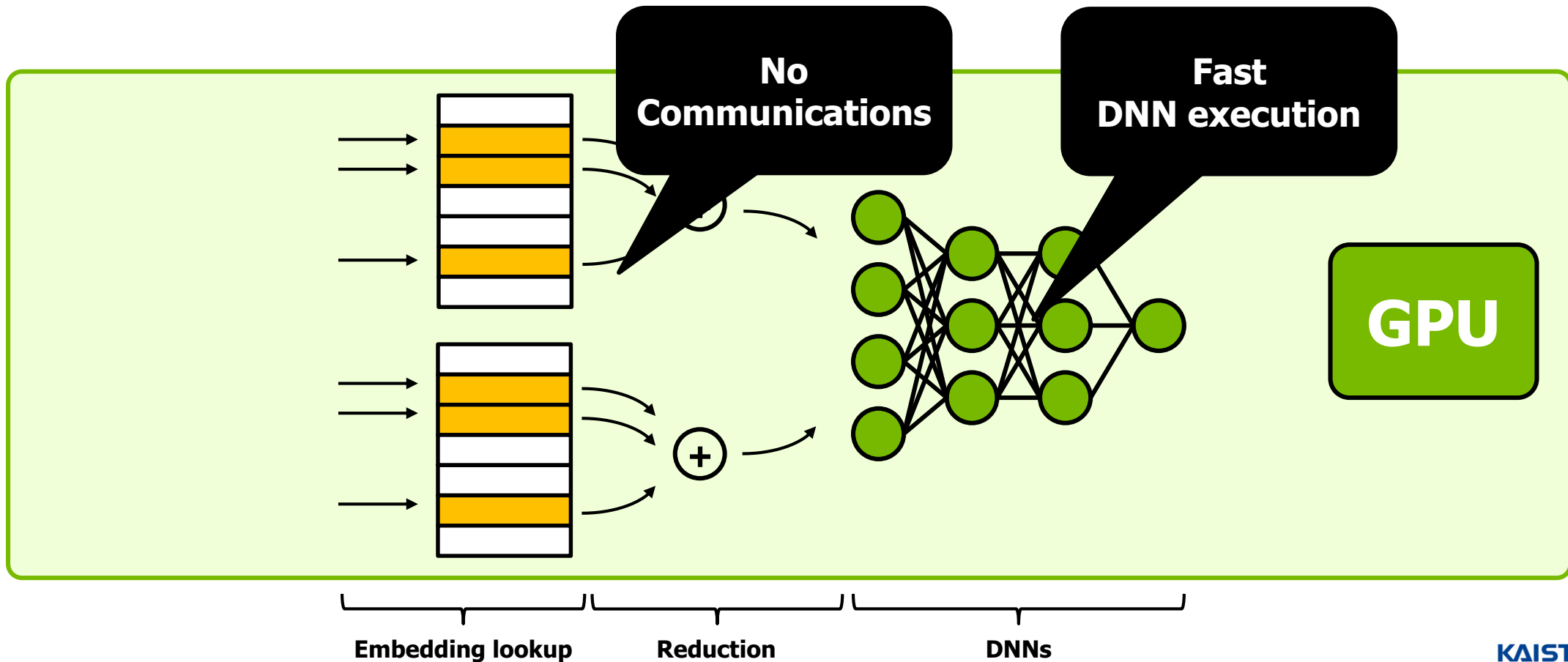
# Design#3: GPU-only approach?

Unbuildable, oracular solution assuming infinite GPU memory capacity



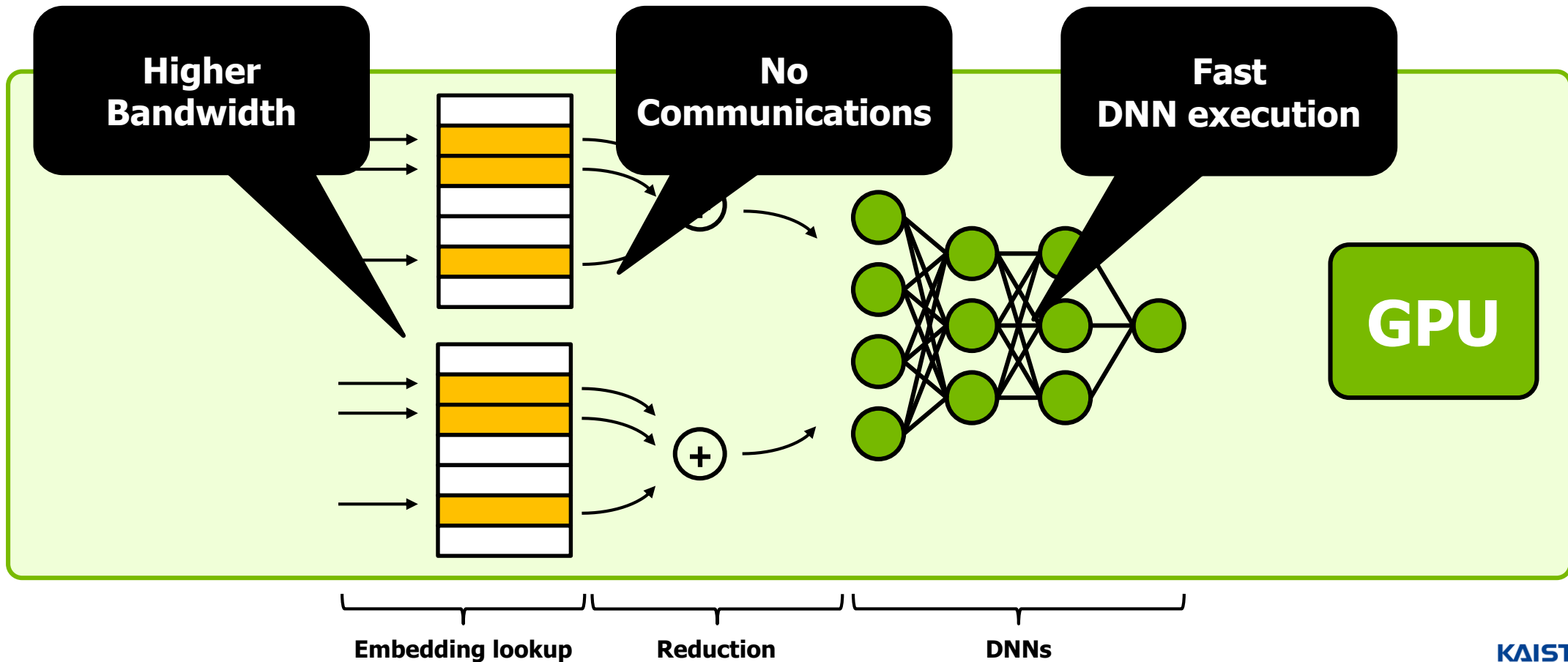
# Design#3: GPU-only approach?

Unbuildable, oracular solution assuming infinite GPU memory capacity



# Design#3: GPU-only approach?

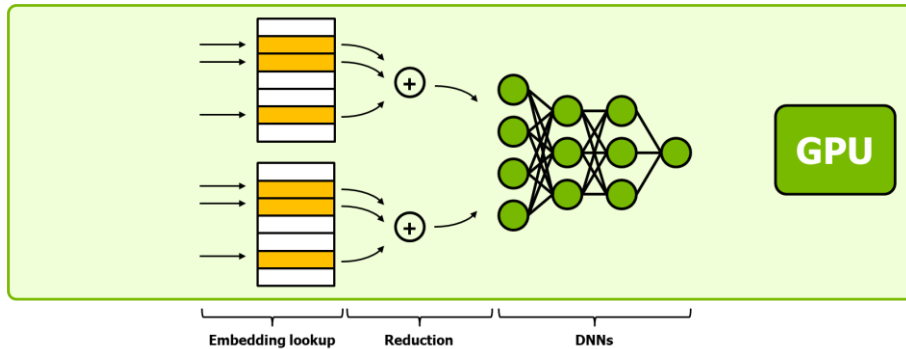
Unbuildable, oracular solution assuming infinite GPU memory capacity



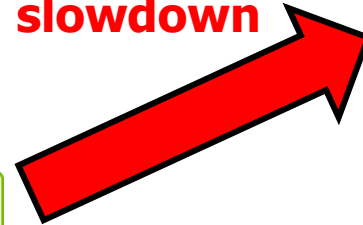
# Key challenges of existing solutions?

CPU-only and hybrid CPU-GPU (vs. GPU-only)

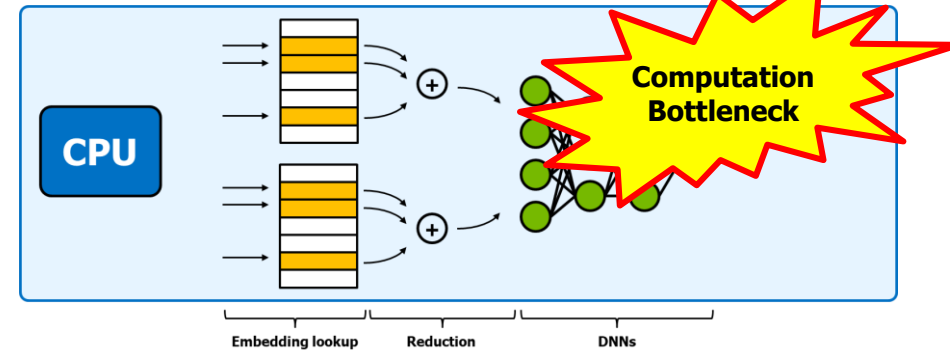
**(Oracular) GPU-only**



**10x  
slowdown**



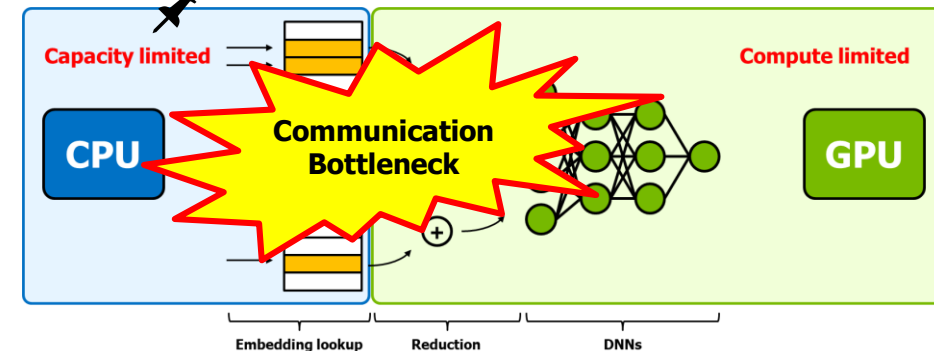
**CPU-only**



**7.3x  
slowdown**



**Hybrid CPU-GPU**

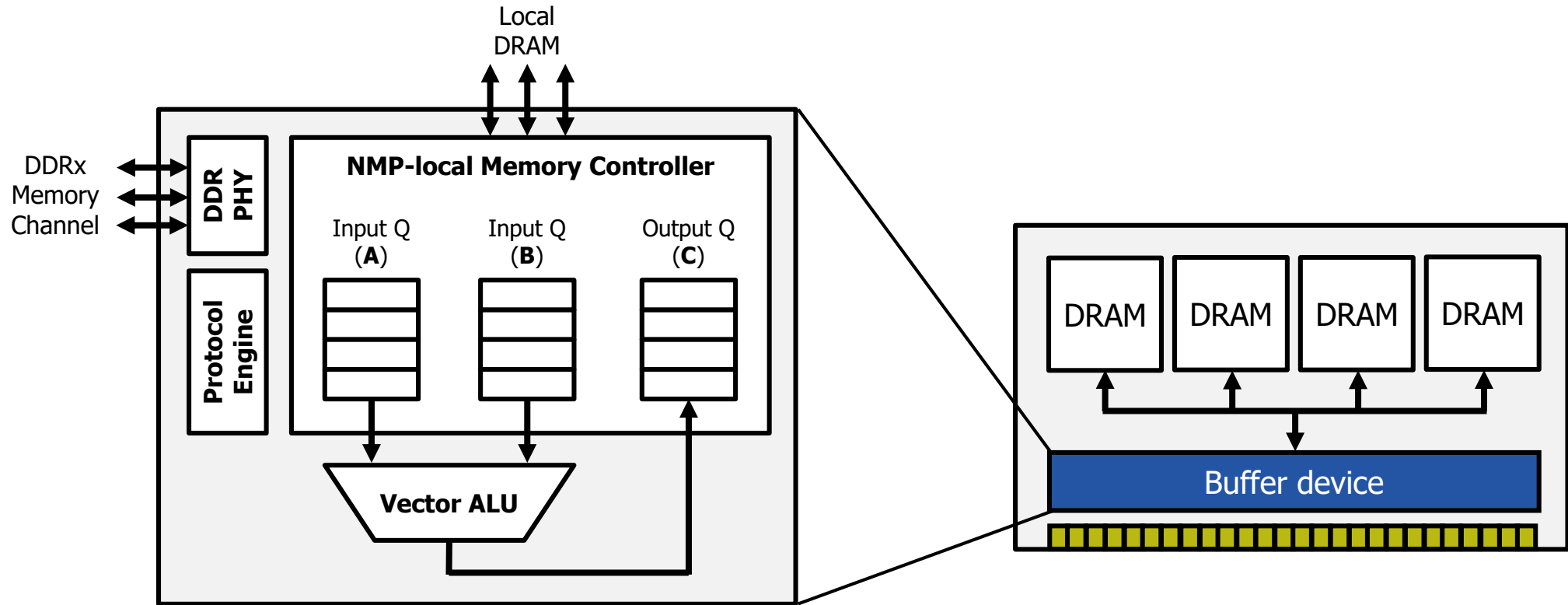




# **Our Approach: “Near”-Memory Acceleration** **(so Near-Memory Processing, NMP)**

# TensorDIMM: a NMP for embeddings

Augment buffer device to add NMP cores for embedding gathers/reductions

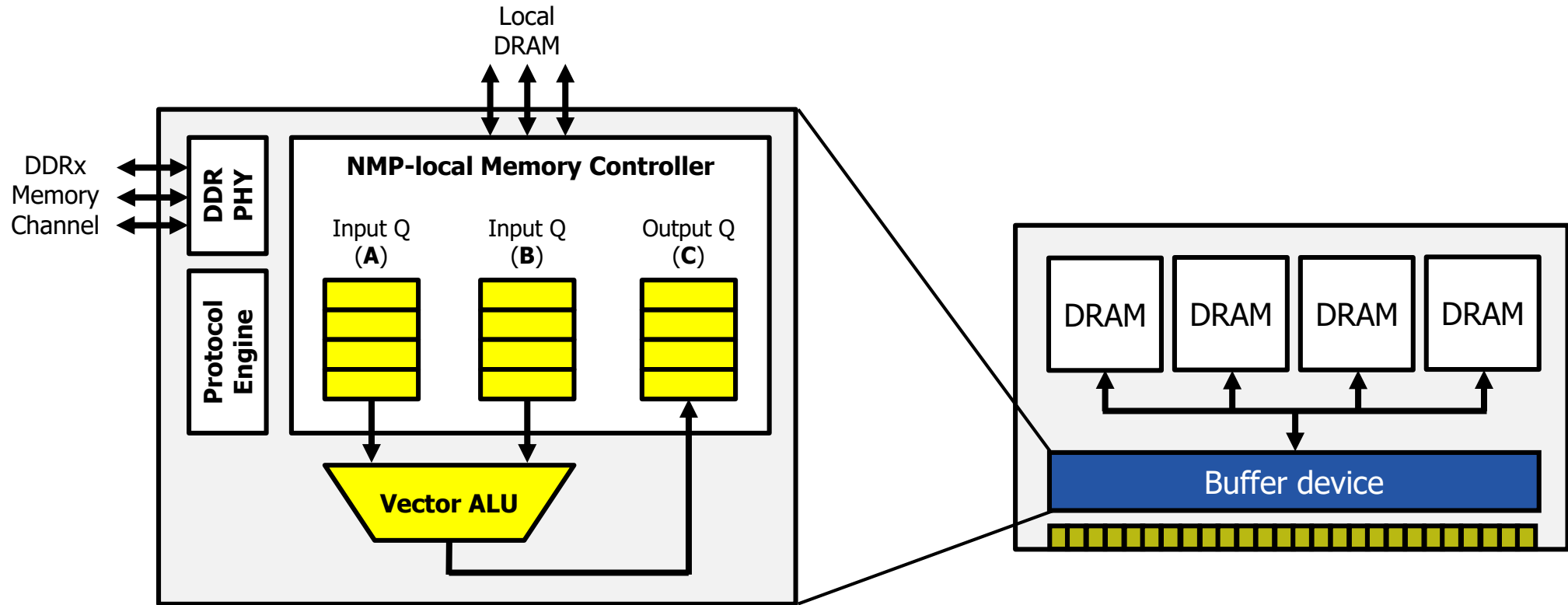


(a) NMP core

(b) TensorDIMM

# TensorDIMM: a NMP for embeddings

Augment buffer device to add NMP cores for embedding gathers/reductions

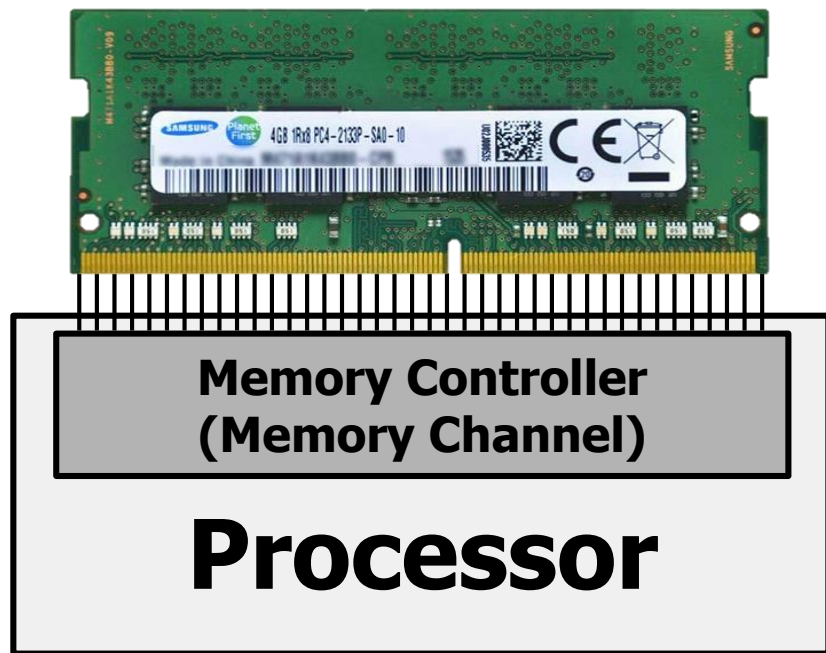


(a) NMP core

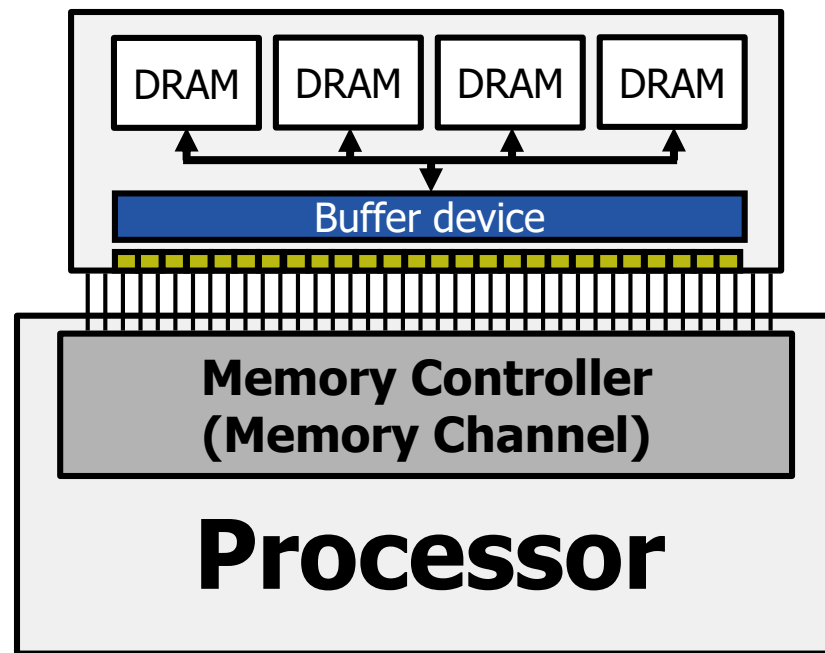
(b) TensorDIMM

# Key advantage of TensorDIMM

“Effective” memory bandwidth scales proportional to the # of DIMMs/ranks



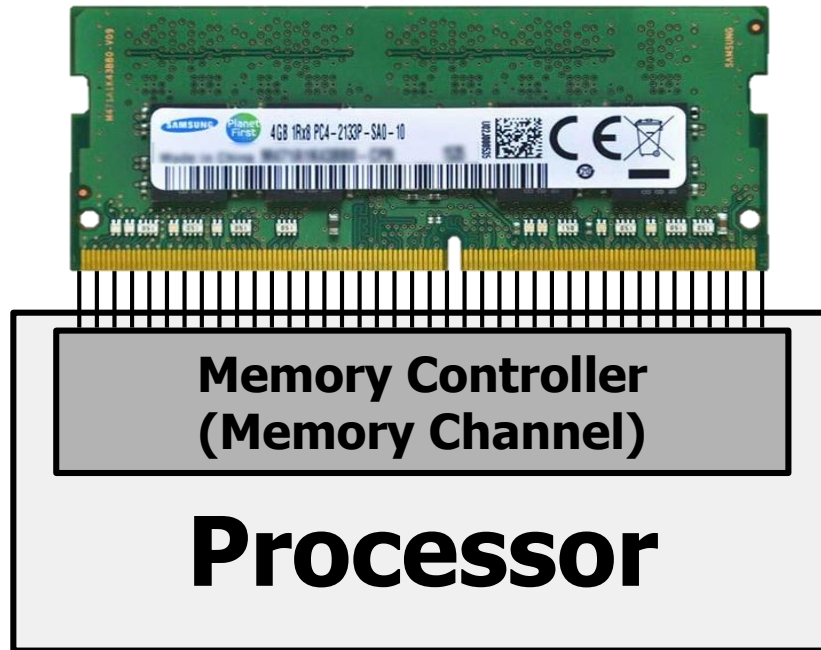
Current system



TensorDIMM approach

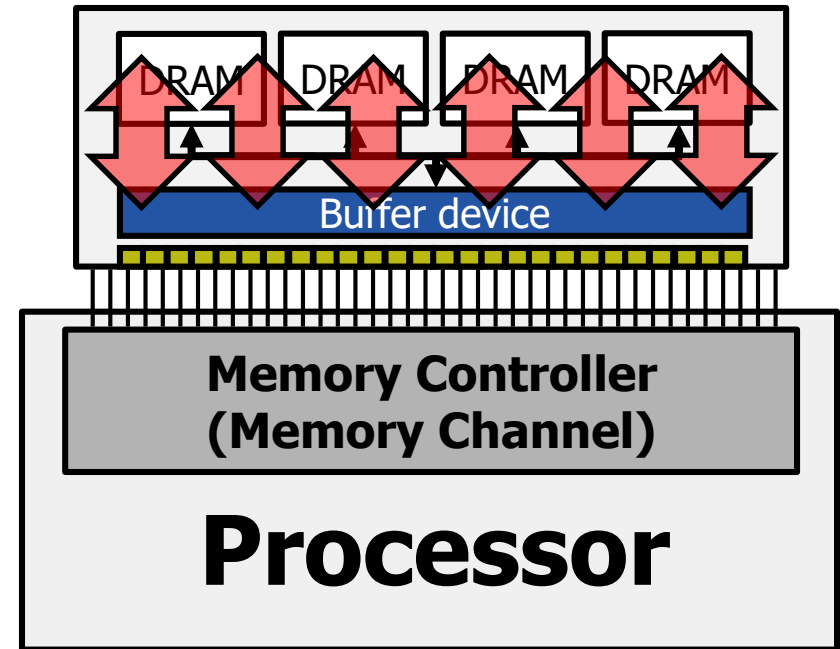
# Key advantage of TensorDIMM

“Effective” memory bandwidth scales proportional to the # of DIMMs/ranks



Current system

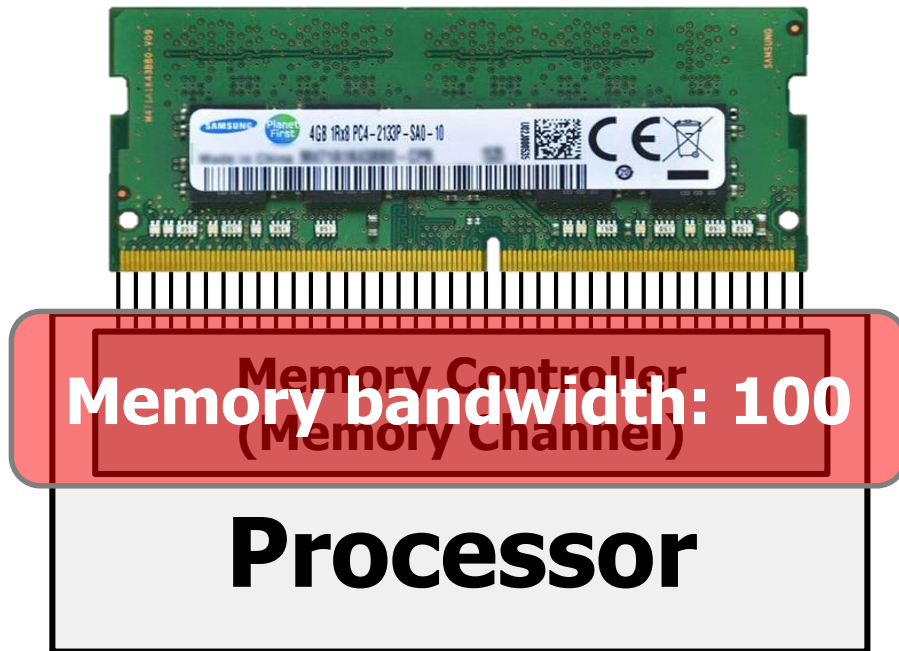
Embedding **gathers/reductions** are done **“locally”** within a DIMM



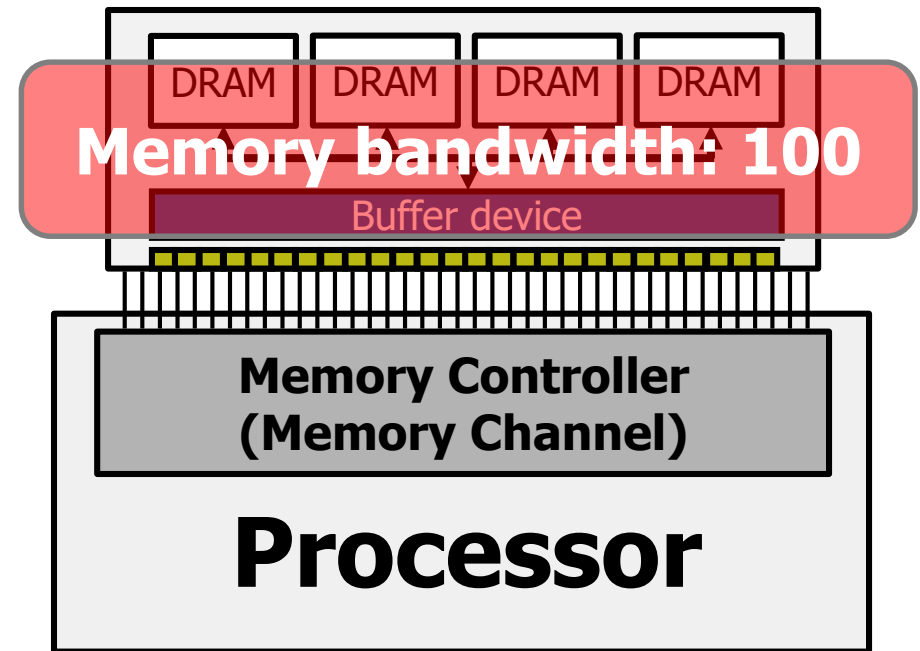
TensorDIMM approach

# Key advantage of TensorDIMM

“Effective” memory bandwidth scales proportional to the # of DIMMs/ranks



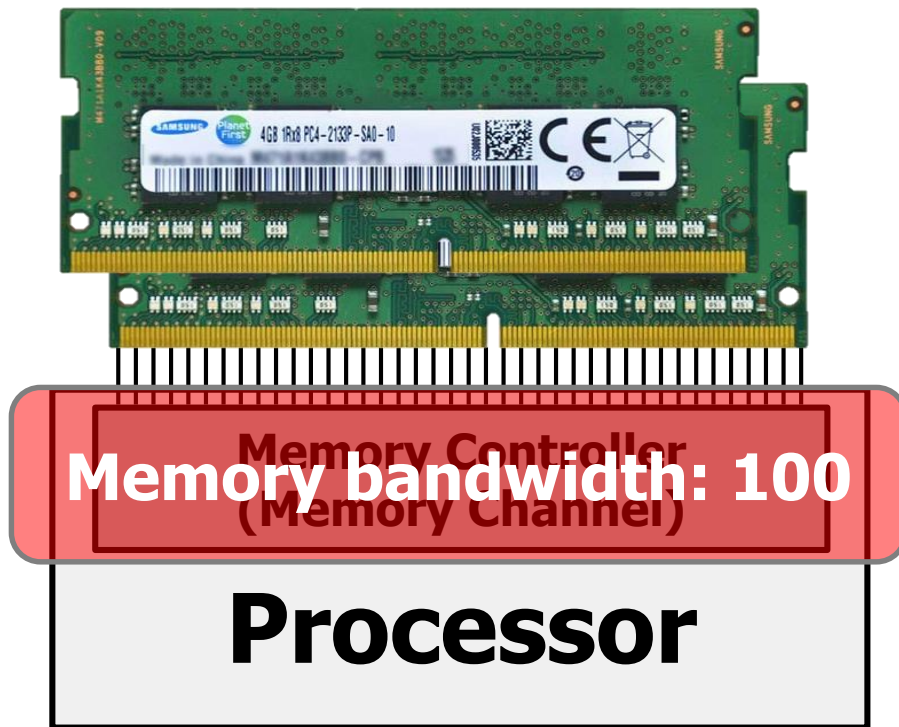
Current system



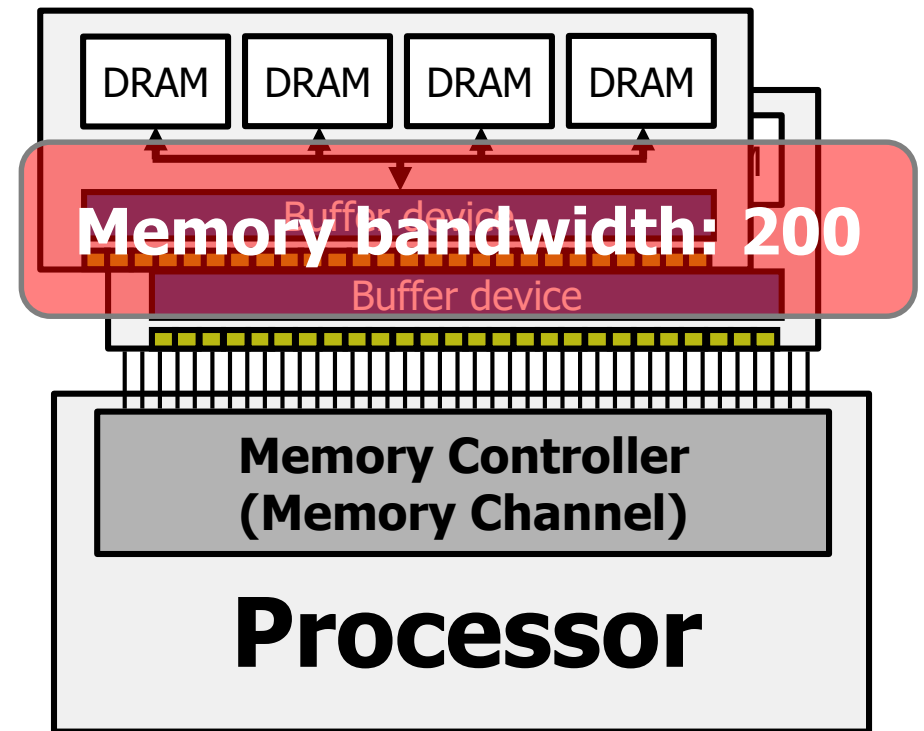
TensorDIMM approach

# Key advantage of TensorDIMM

“Effective” memory bandwidth scales proportional to the # of DIMMs/ranks



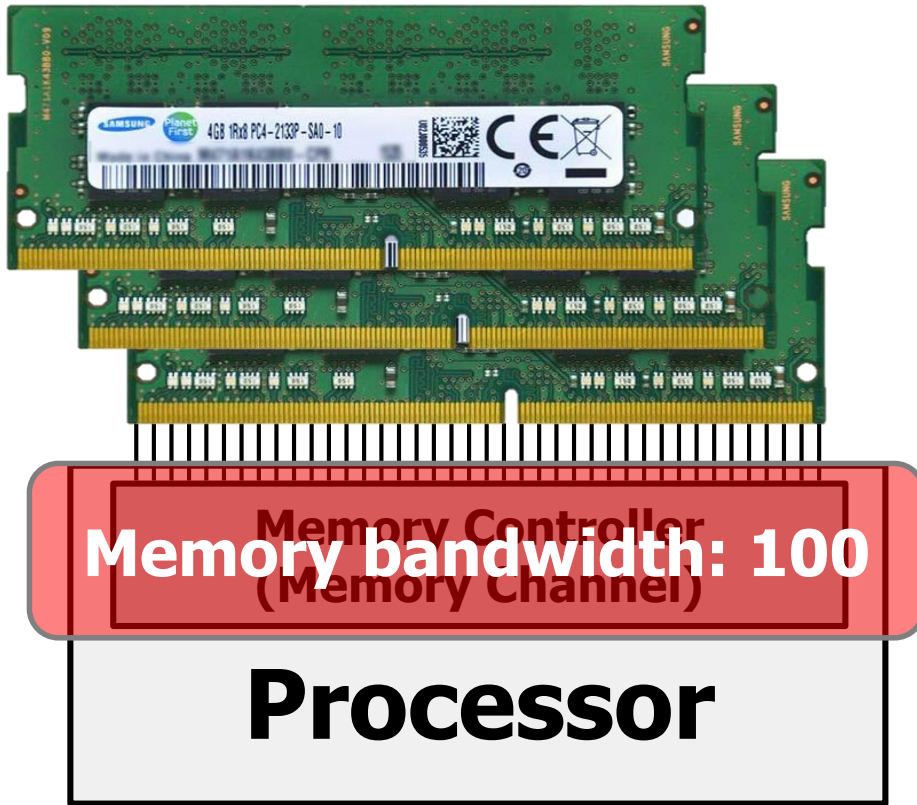
Current system



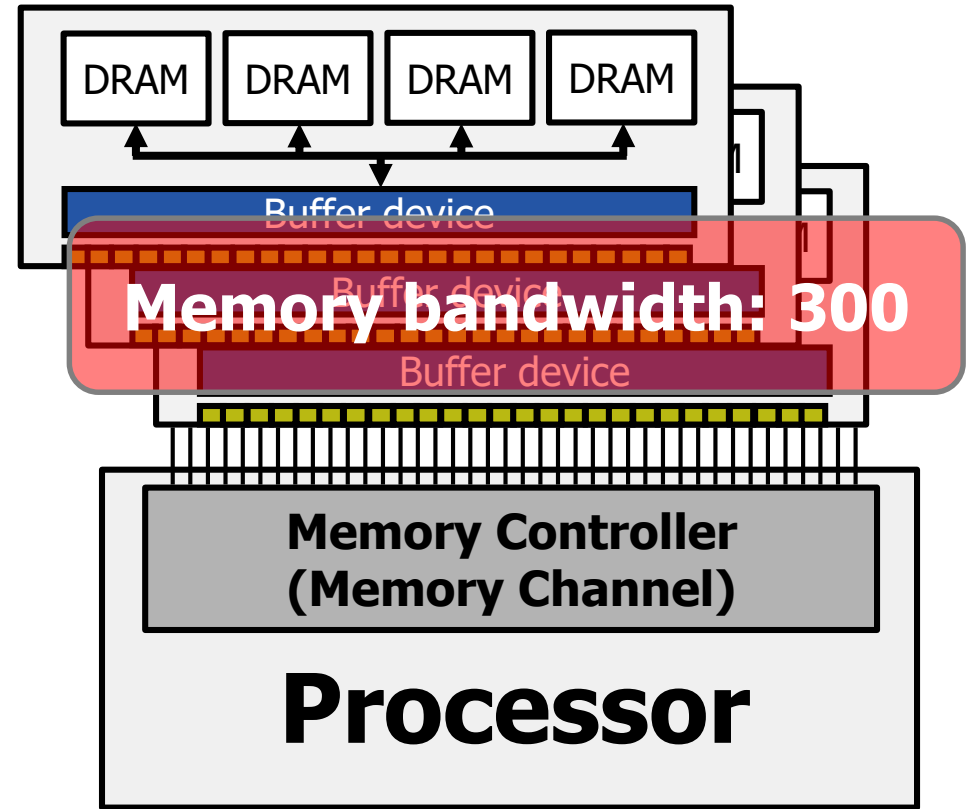
TensorDIMM approach

# Key advantage of TensorDIMM

“Effective” memory bandwidth scales proportional to the # of DIMMs/ranks



Current system

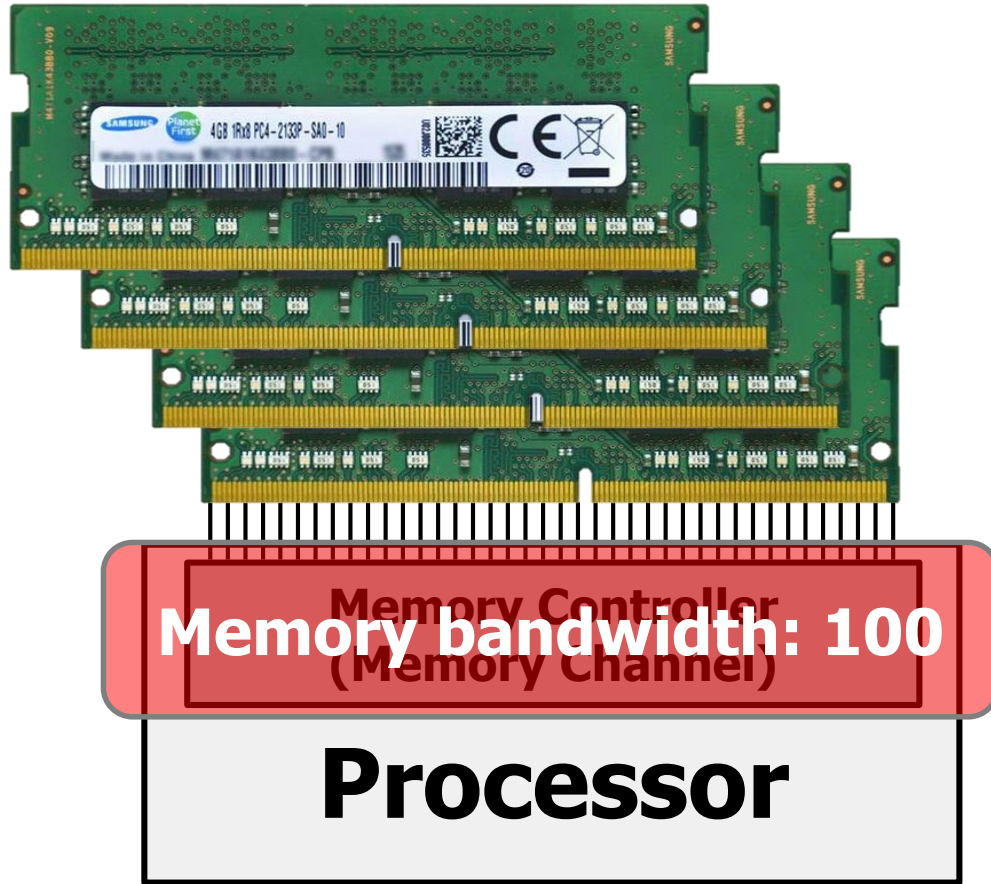


TensorDIMM approach

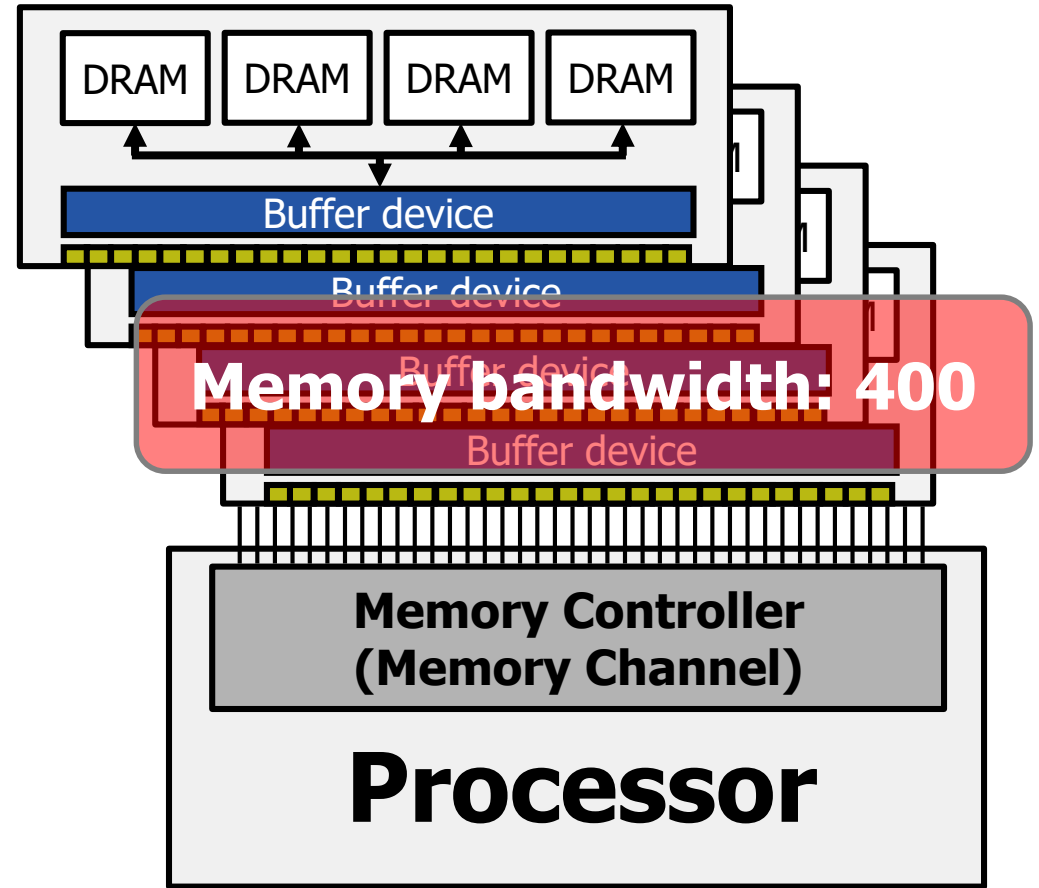


# Key advantage of TensorDIMM

“Effective” memory bandwidth scales proportional to the # of DIMMs/ranks



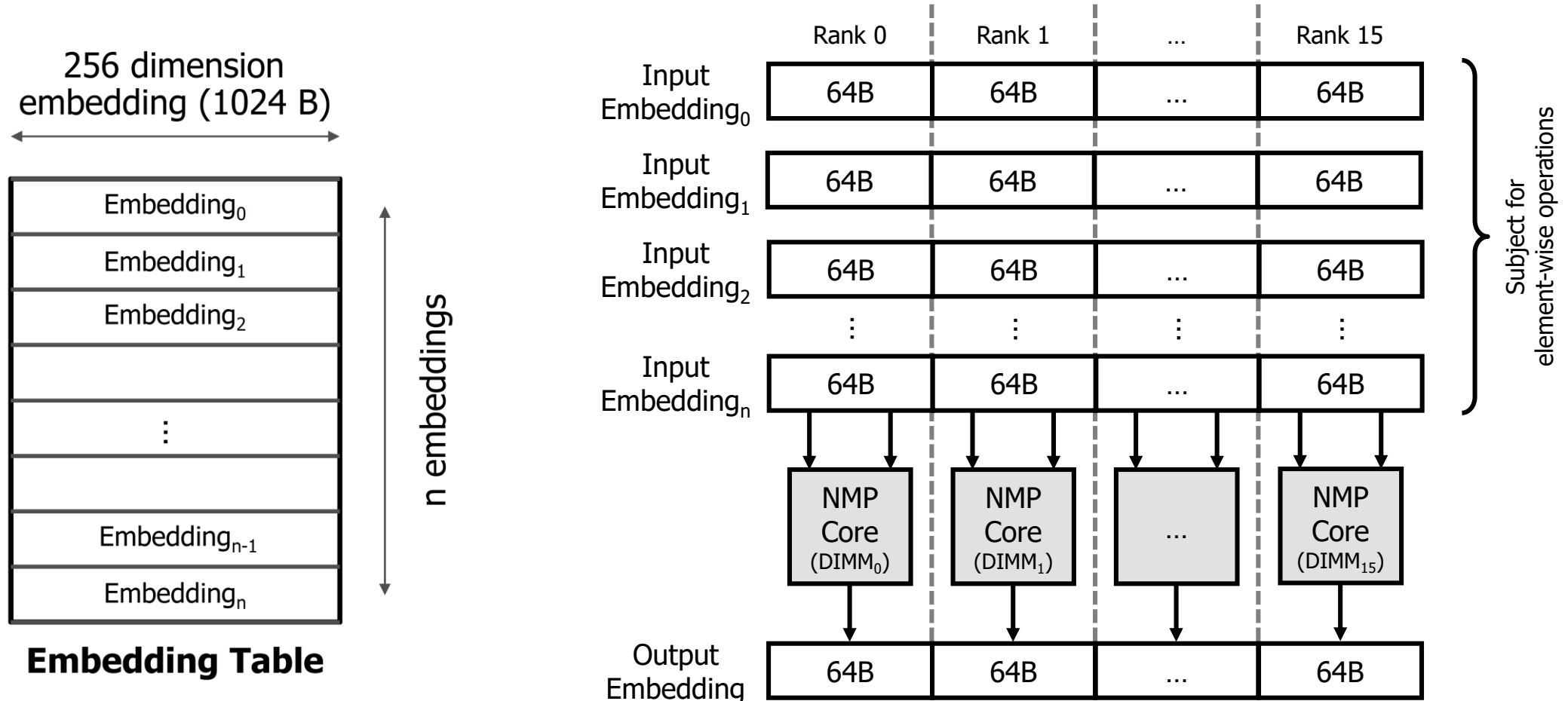
Current system



TensorDIMM approach

# Mapping embedding tables in DRAMs

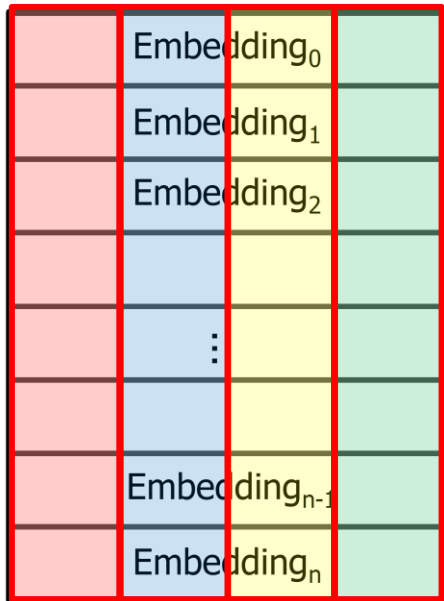
Leverage rank-level parallelism for maximal bandwidth utilization



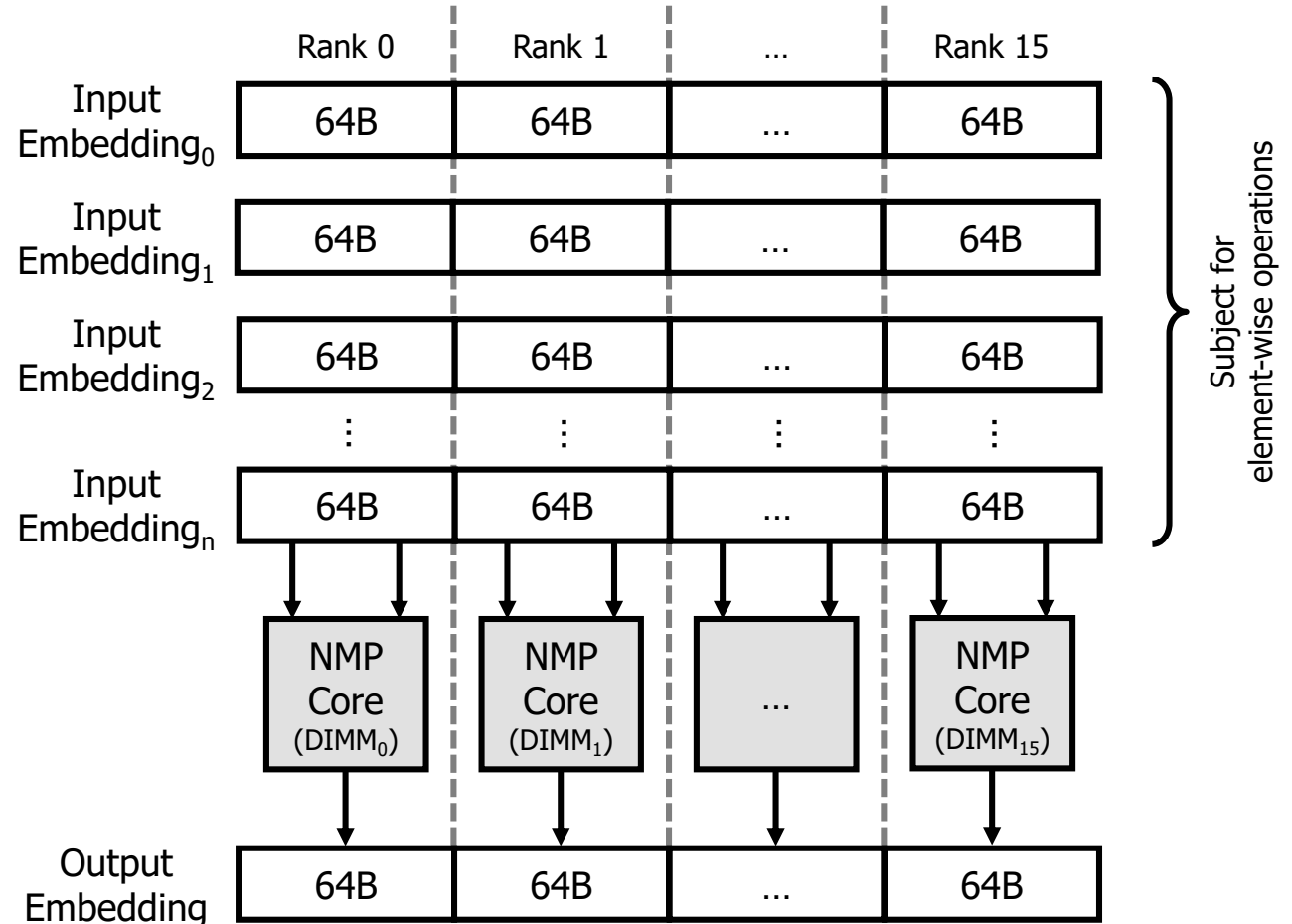
# Mapping embedding tables in DRAMs

Leverage rank-level parallelism for maximal bandwidth utilization

256 dimension  
embedding (1024 B)



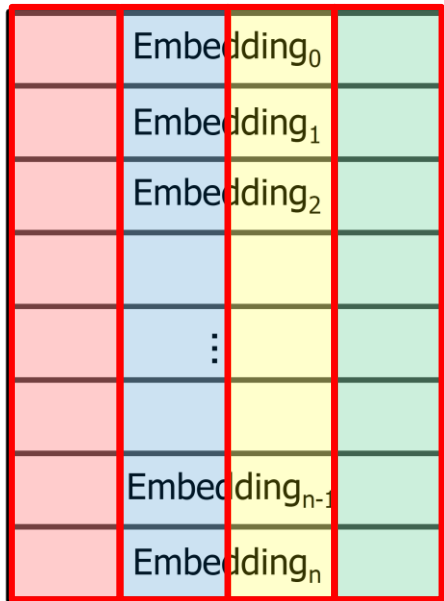
**Embedding Table**



# Mapping embedding tables in DRAMs

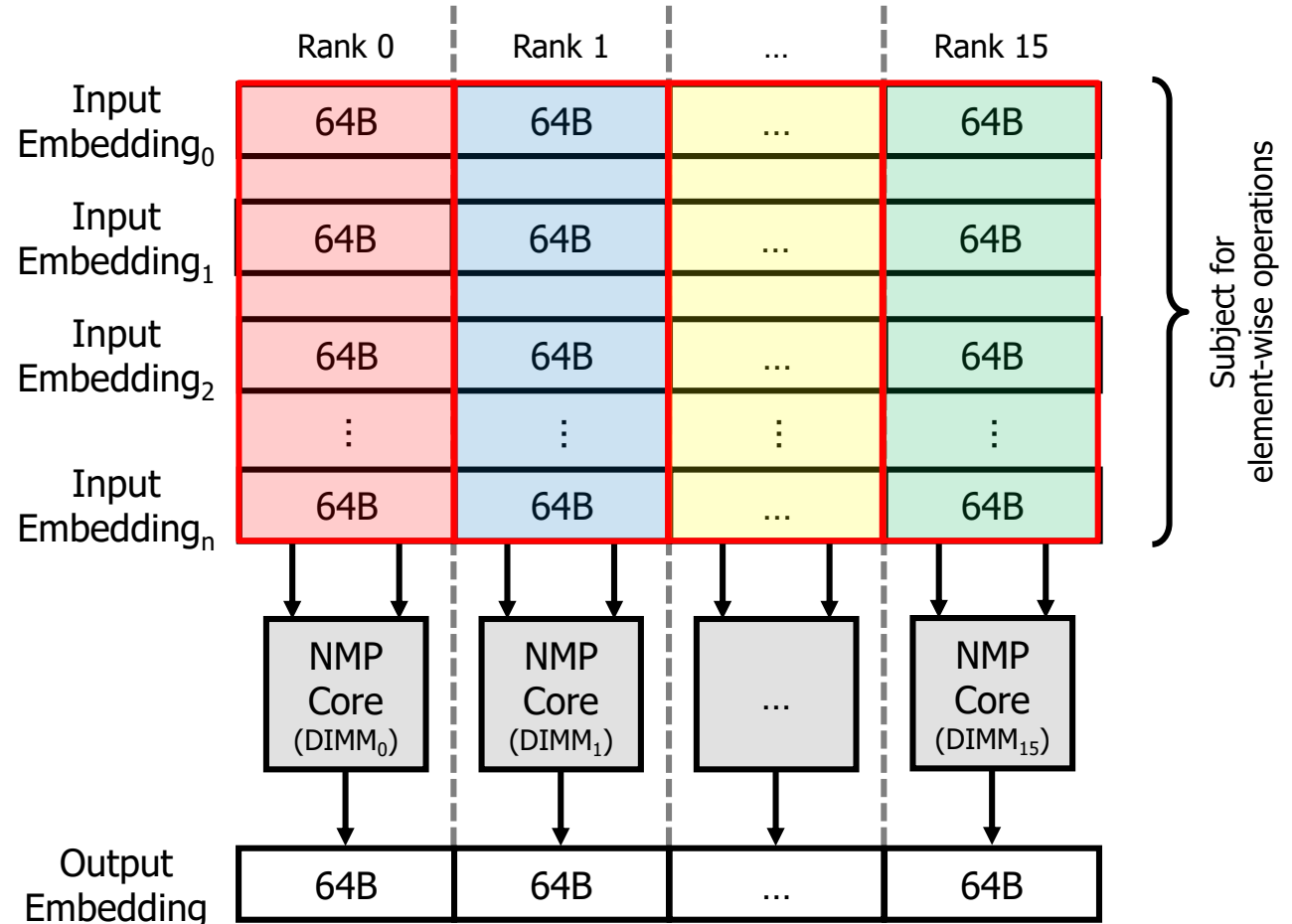
Leverage rank-level parallelism for maximal bandwidth utilization

256 dimension  
embedding (1024 B)



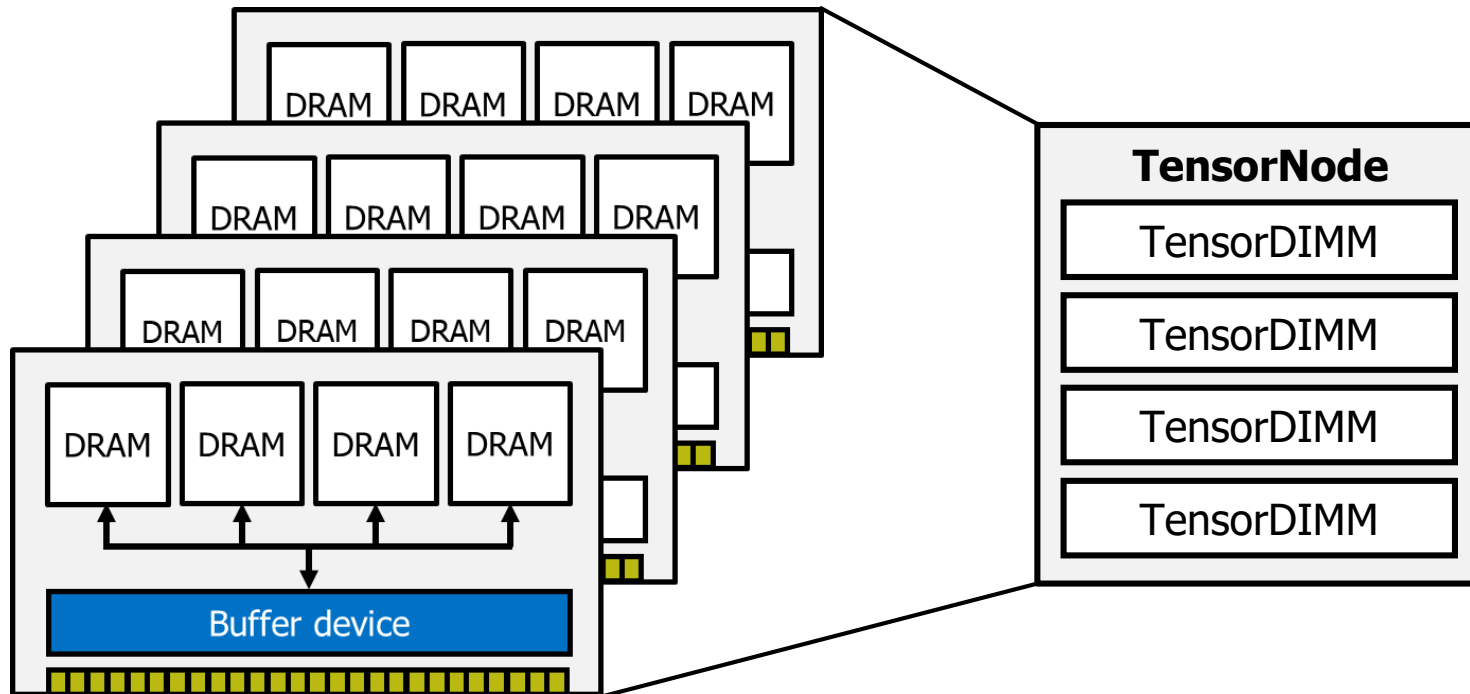
**Embedding Table**

n embeddings



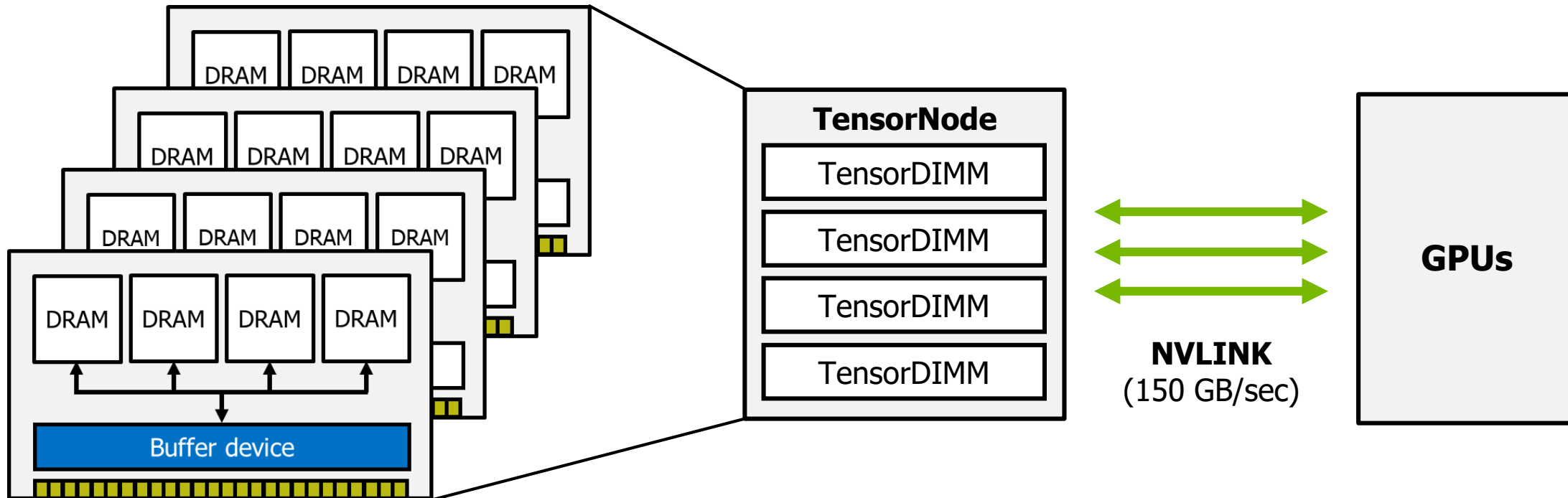
# Tensor“Node” using TensorDIMMs

A pooled memory architecture aggregated with multiple TensorDIMMs



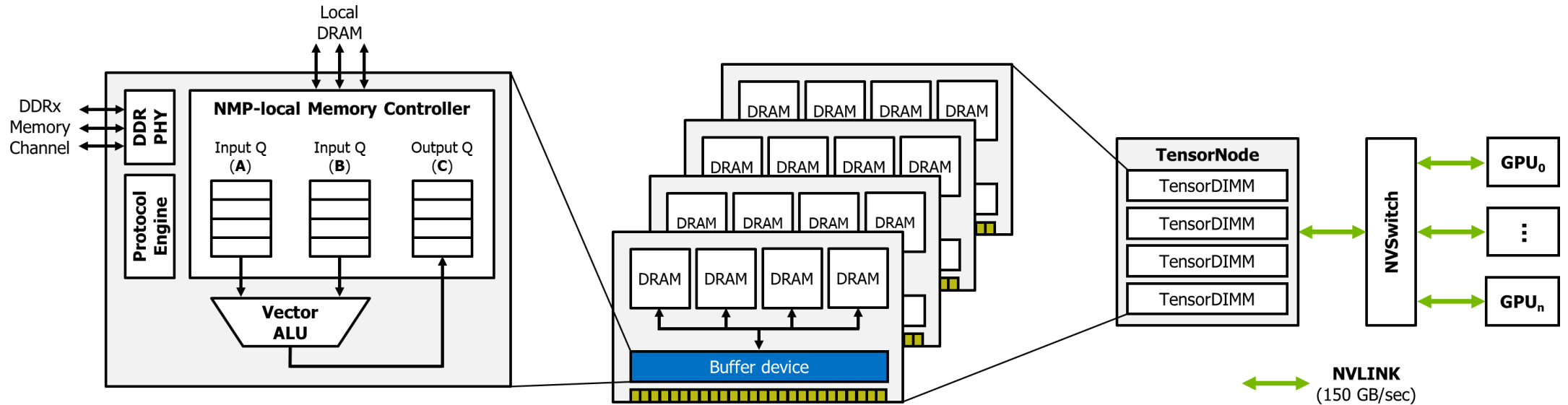
# TensorNode as “remote” memory pools

Utilize high-speed links (e.g. NVLINK) for inter-device communication



# Putting everything together

A platform for scalable expansion of both memory bandwidth and capacity



(a) NMP core

Addresses the  
**memory bandwidth**  
challenge

(b) TensorDIMM

Addresses the  
**memory capacity**  
challenge

(c) System architecture with TensorNode

Addresses the  
**compute & communication**  
challenges

# Evaluation



# Evaluation methodology

Combination of cycle-level simulation and emulation on real ML systems

- ❑ Cycle-level DRAM simulator (Ramulator\*)
  
- ❑ Proof-of-concept software prototype on real ML systems (NVIDIA DGX-1V)

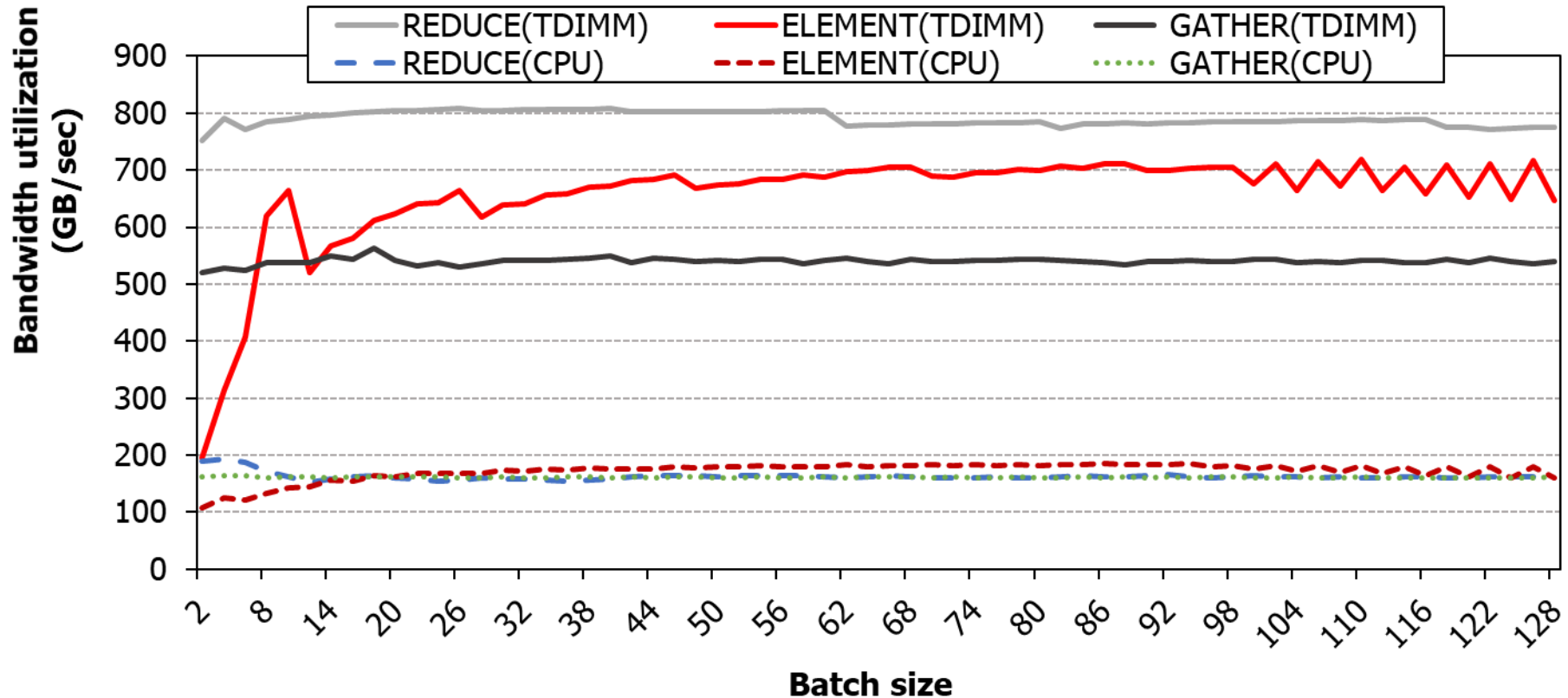
# Evaluation methodology

Combination of cycle-level simulation and emulation on real ML systems

- ❑ Cycle-level DRAM simulator (Ramulator\*)
  - Memory bandwidth for embedding gathers/reductions under our address mapping
- ❑ Proof-of-concept software prototype on real ML systems (NVIDIA DGX-1V)

# Memory bandwidth utilization

Effective bandwidth scales proportional to number of ranks (avg 4x↑)



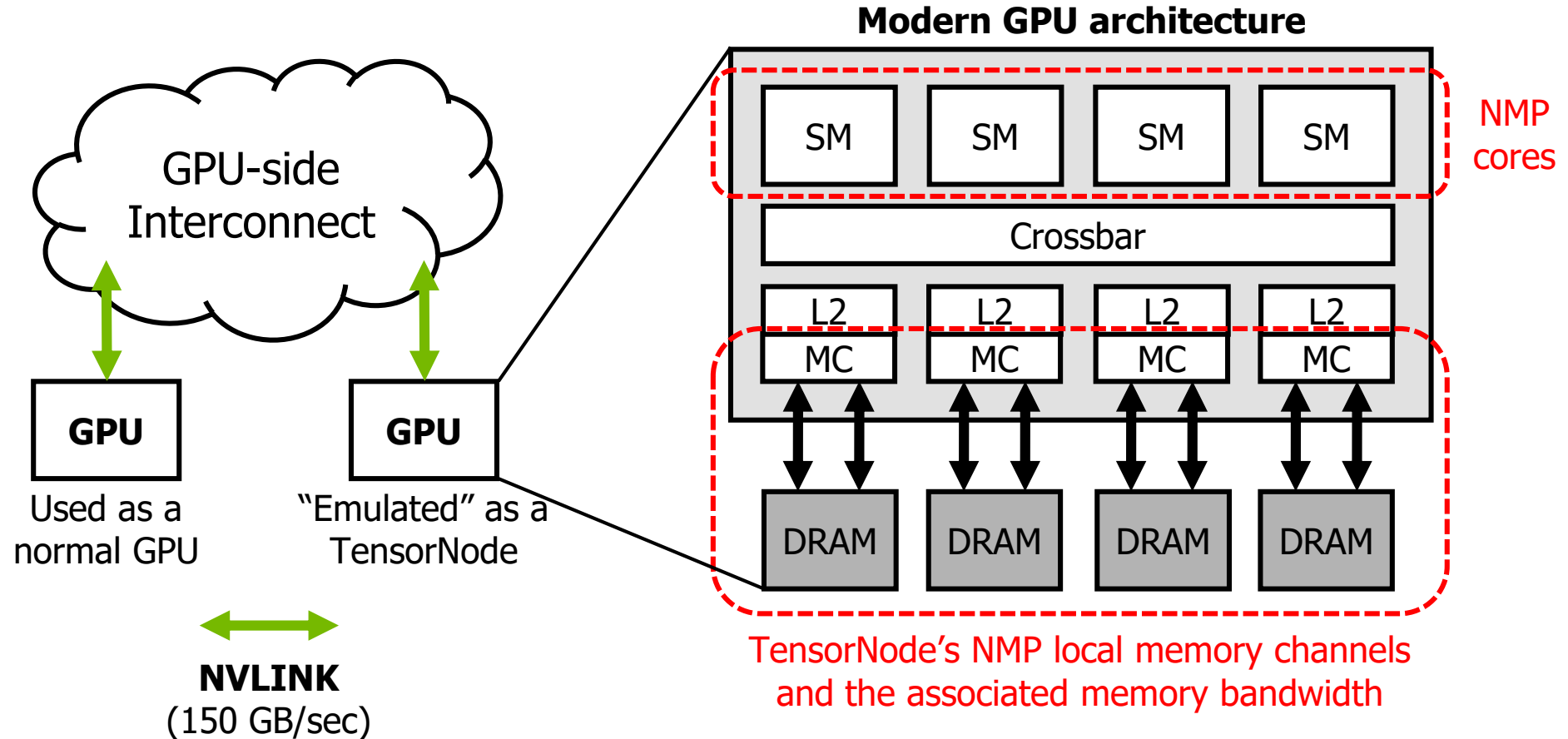
# Evaluation methodology

Combination of cycle-level simulation and emulation on real ML systems

- ❑ Cycle-level DRAM simulator (Ramulator\*)
  - Memory bandwidth for embedding gathers/reductions under our address mapping
- ❑ Proof-of-concept software prototype on real ML systems (NVIDIA DGX-1V)
  - Intel's Math Kernel Library (MKL)
  - NVIDIA cuDNN / cuBLAS
  - In-house CUDA implementation of other layers
  - NVIDIA DGX-1V
    - Eight NVIDIA V100 GPUs
    - Two Intel Xeon E5-2698 v4

# TensorNode system modeling

A proof-of-concept software prototype to emulate TensorDIMM



# Evaluation methodology

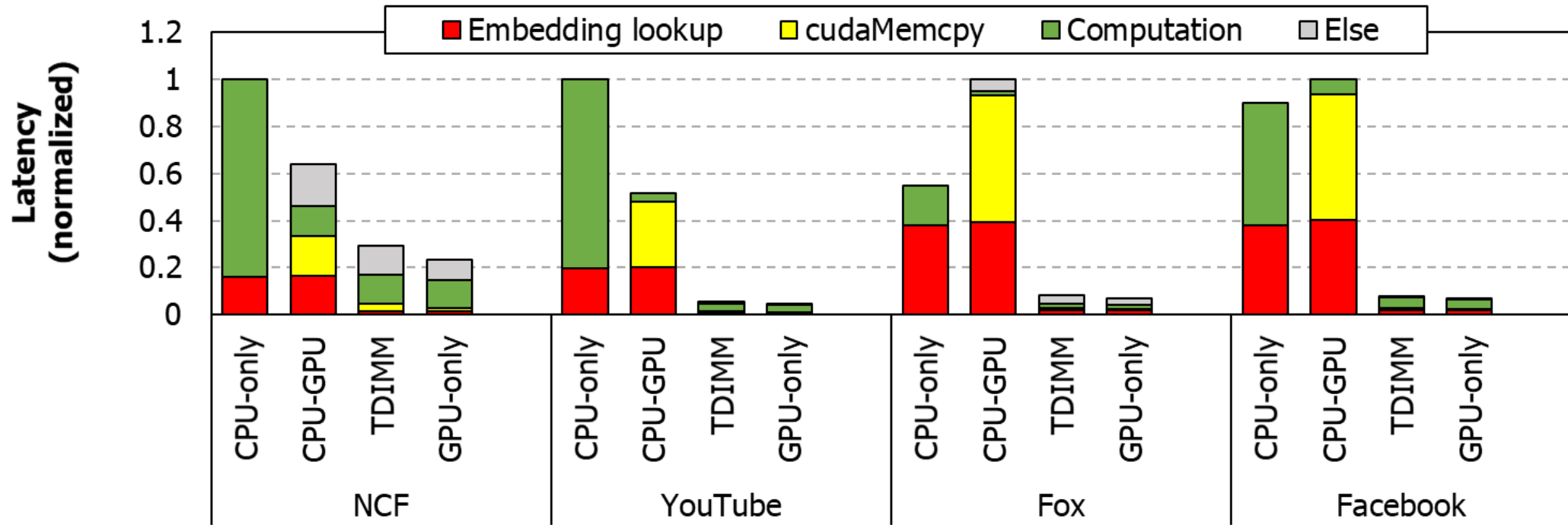
A proof-of-concept software prototype to emulate TensorDIMM

Four system design points

- CPU-only
- Hybrid CPU-GPU
- TensorDIMM (ours)
- GPU-only (oracle)

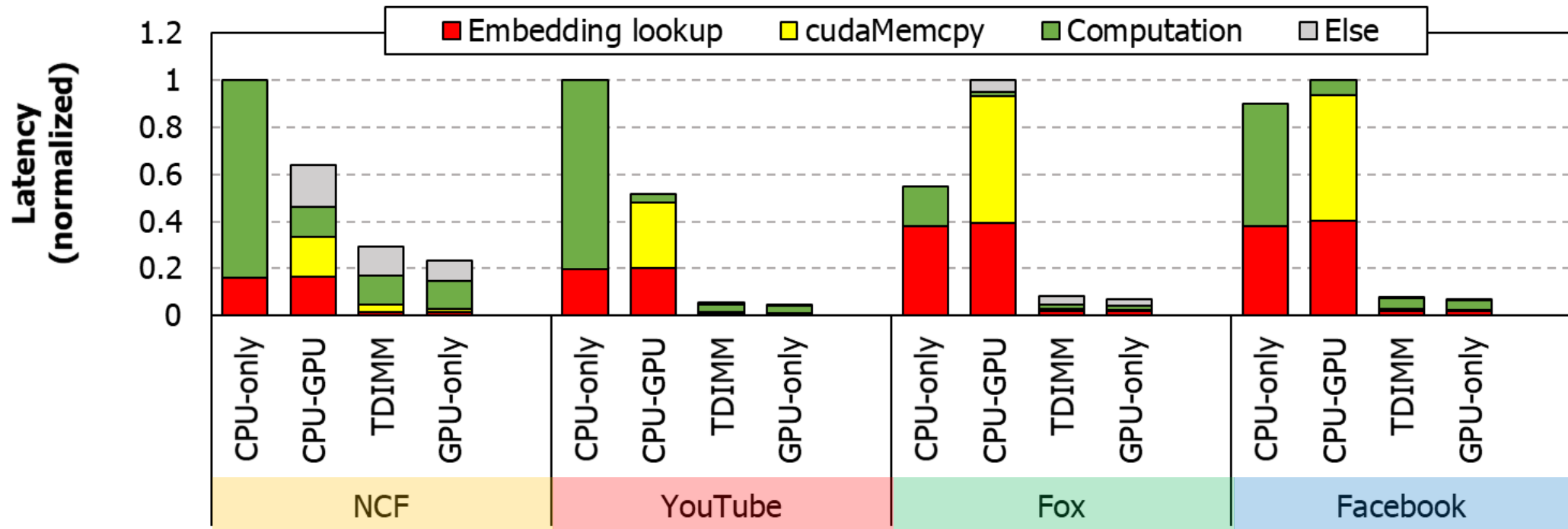
# Latency breakdown

TensorDIMM helps reduce both embedding/MLP latency



# Latency breakdown

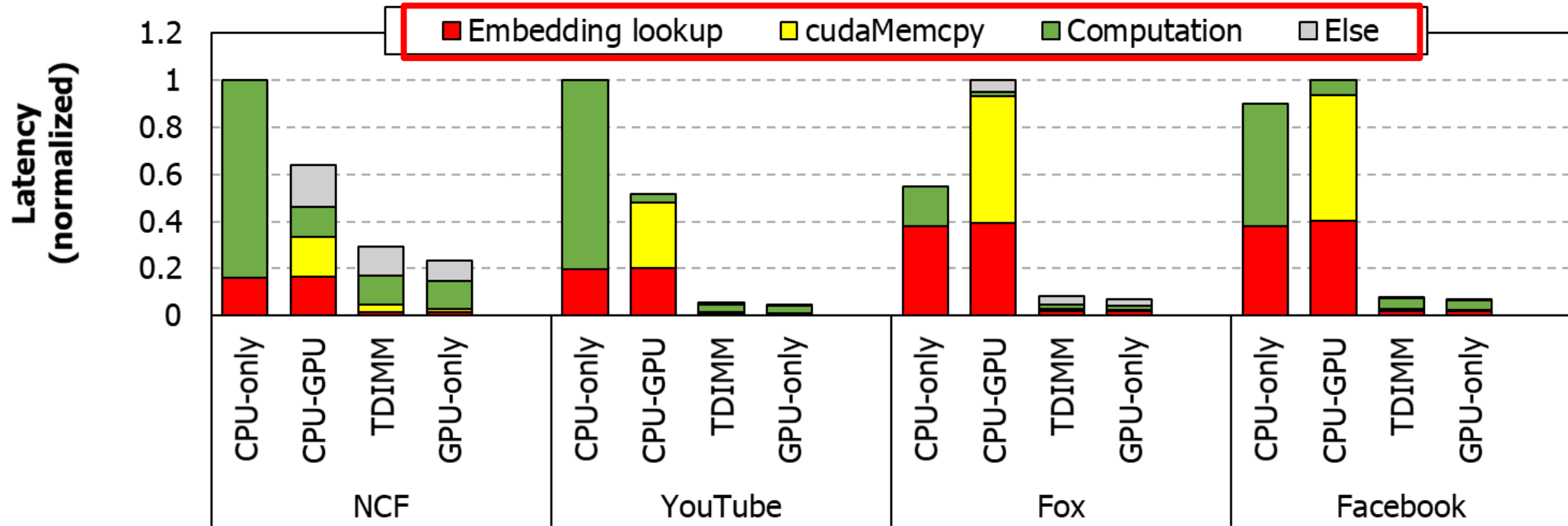
TensorDIMM helps reduce both embedding/MLP latency





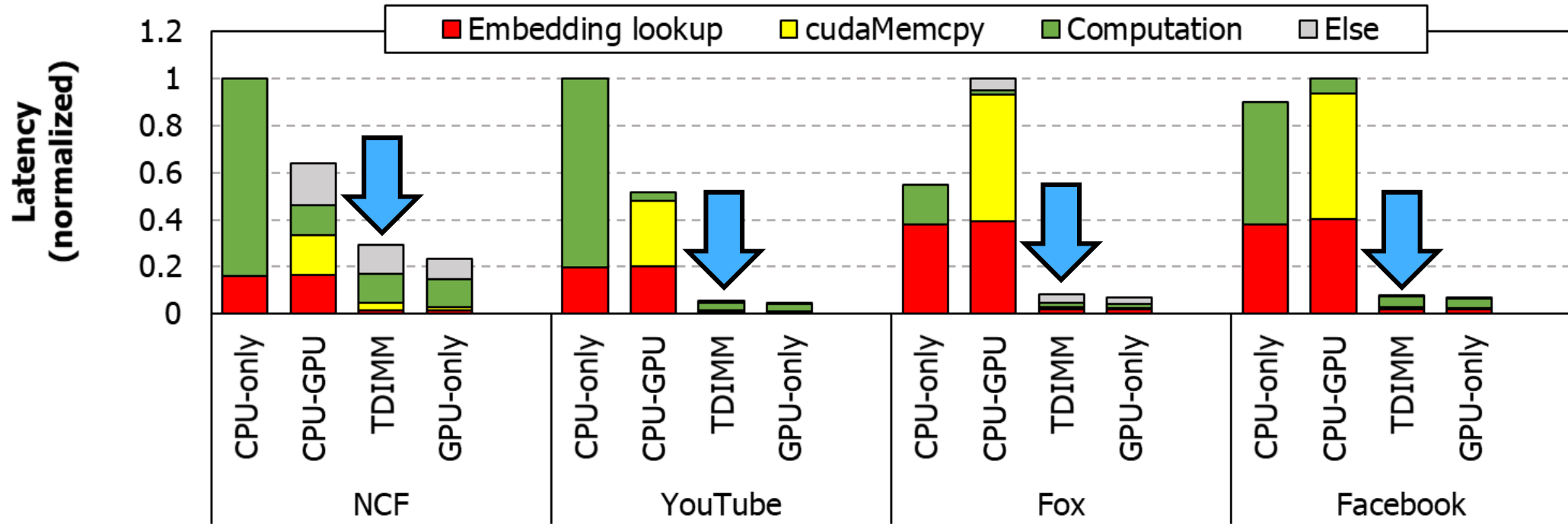
# Latency breakdown

TensorDIMM helps reduce both embedding/MLP latency



# Latency breakdown

TensorDIMM achieves overall 6-9x speedup against the baselines



## TensorDIMM:

# A Near-Memory Processing Architecture for Sparse Embedding Layers

The “**first**” architectural solution tackling sparse embedding layers

A “**practical**” near-memory processing solution for an important AI workload

Average “**6~9x**” performance improvement on state-of-the-art DNN-based recommendation models

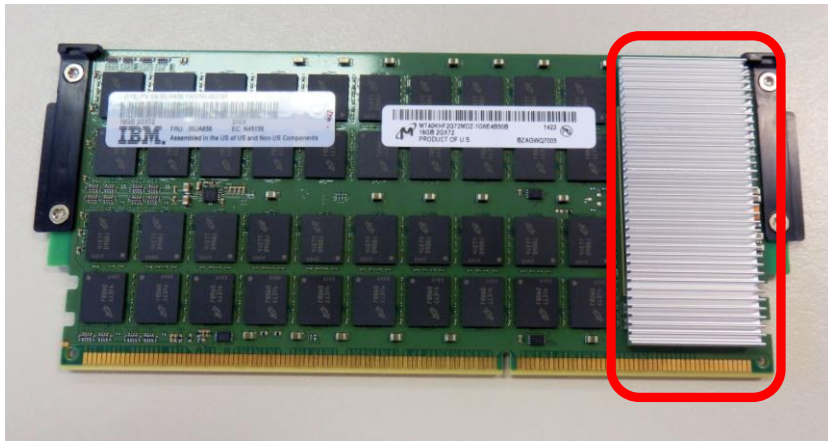
**Questions?**

# Backup Slides

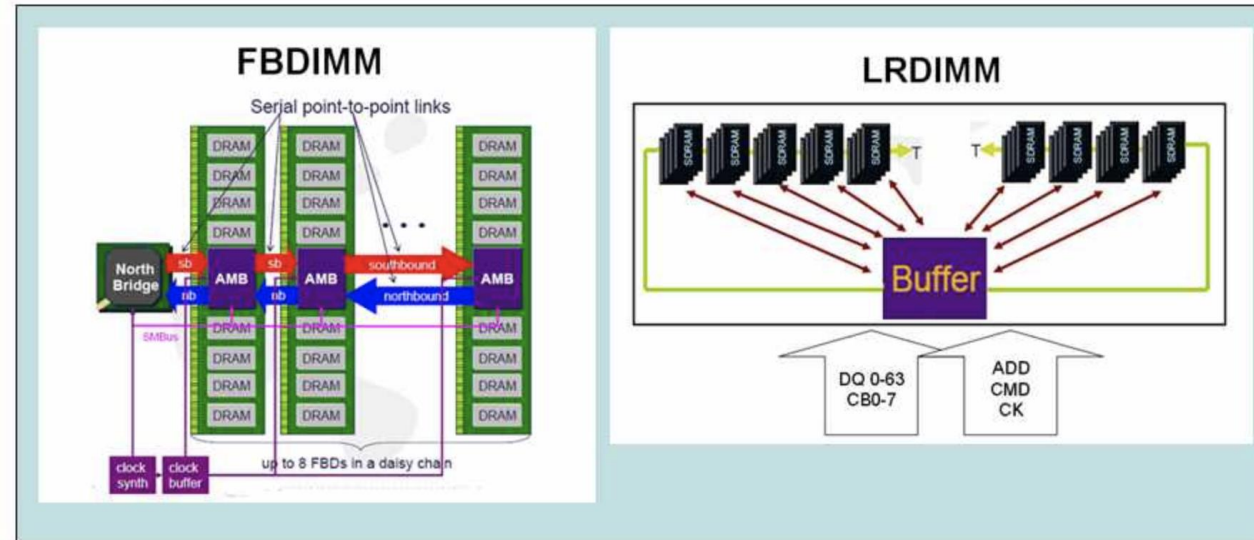
# TensorDIMM design overheads

It's not free, but adding custom logics within DIMM has been done before

## FBDIMM versus LR-DIMM



IBM centaur DIMM





OCP  
GLOBAL  
SUMMIT

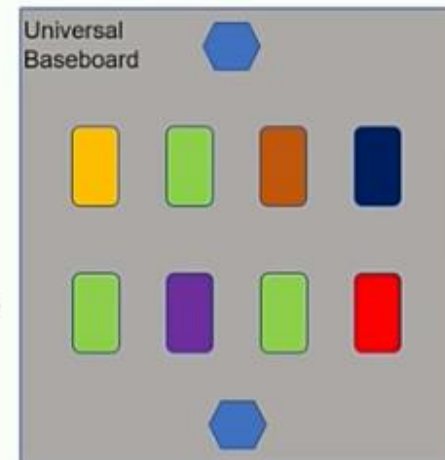
## Heterogenous OAMs

These Modules need not be of the same type

Each one may be suited for a specific application/task

xPUs, FPGA, CPU, GPU, ASICs, SoCs, **Memory**, ...

Chained, pipelined processing stages



Open. Together.

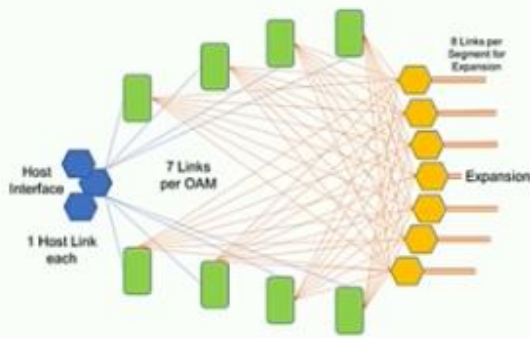
San Jose Convention Center | San Jose, California

March 14-15, 2019

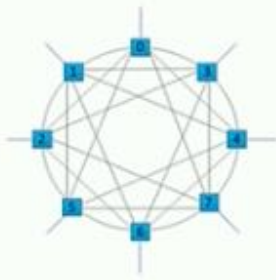


# OCP GLOBAL SUMMIT

## With different interconnect topologies



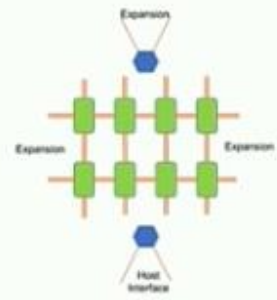
A Grid of interconnected OAMs,  
Max Bisection BW  
One Hop Away  
Ready for Expansion



With **six** inter-OAM Links  
and one Host Link



With **seven** inter-OAM Links  
and one Host Link



**Six** inter-module Links may  
create a 3D Mesh or Torus



### Open. Together.

## San Jose Convention Center | San Jose, California

March 14-15, 2019



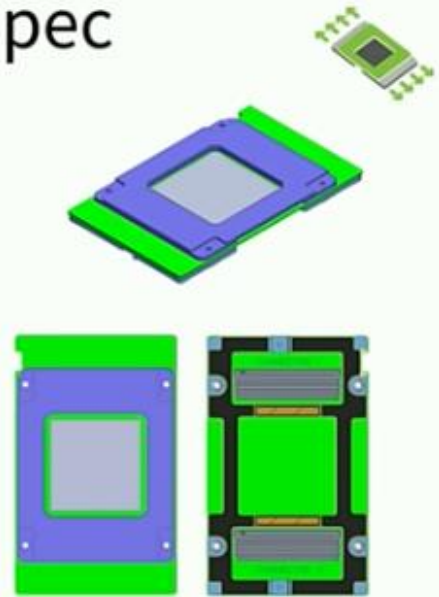




**OCP**  
GLOBAL  
SUMMIT

## OCP Accelerator Module Spec

- Support both 12V and 48V as input
- Up to 350w(12V) and up to 700w(48V) TDP
- 102mm x 165mm
- Support single or multiple ASIC(s) per Module
- Up to **eight** x16 Links (Host + inter-module Links)
  - Support one or two x16 High speed link(s) to Host
  - Up to seven x16 high speed interconnect links
- Expect to support up to 450W (air-cooled) and 700W (liquid-cooled)
- Up to 8\* Modules per system
- System management and debug interfaces



OCP  
SUMMIT

Open. Together.

San Jose Convention Center | San Jose, California

March 14-15, 2019