

Introducing Disaggregated Multi-Tower: a New Modeling Technique for Efficient Large-Scale Recommendations

AI SYSTEM CO-DESIGN TEAM

Liang Luo
Research Scientist



Agenda

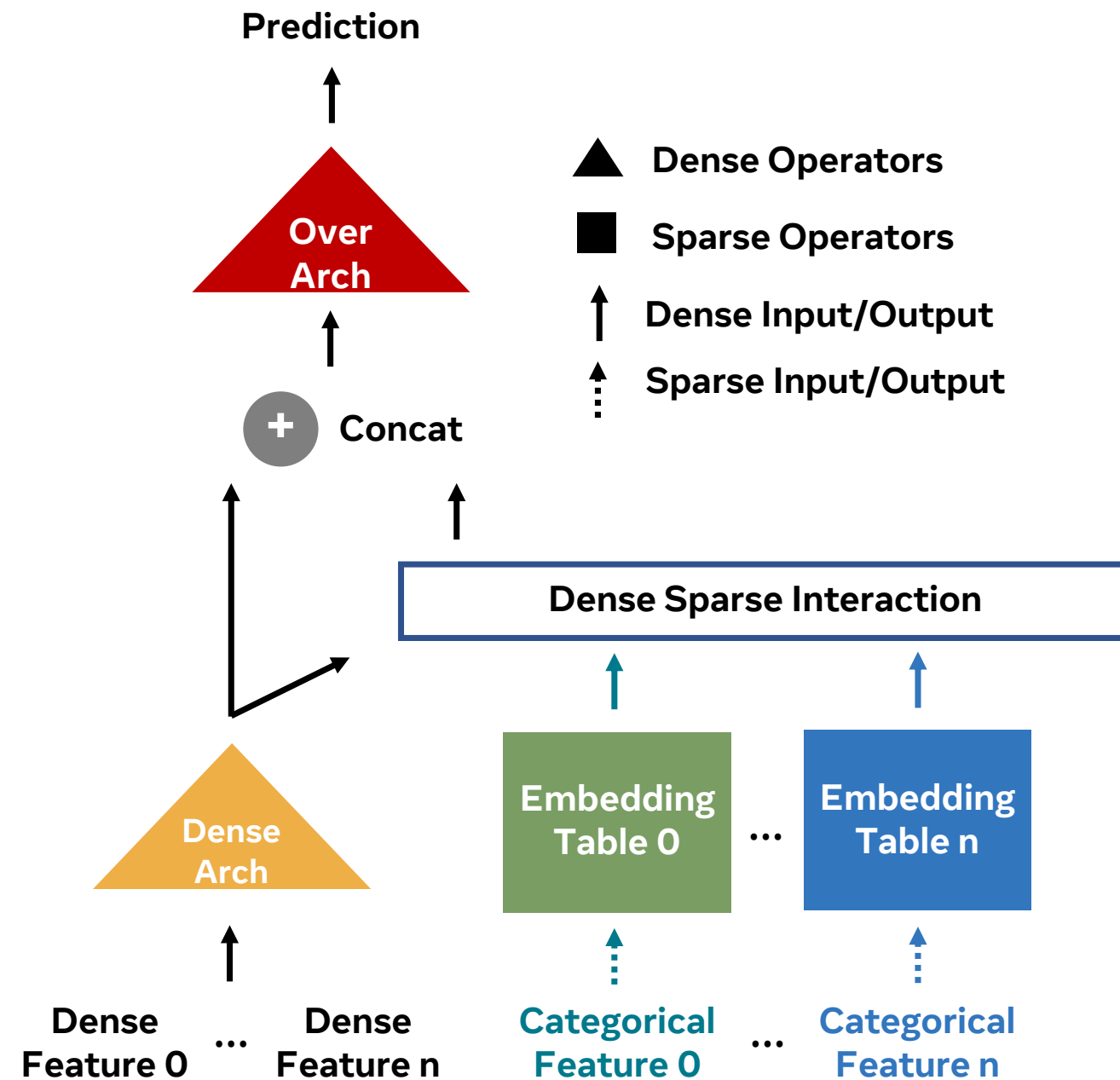
Deep Learning-based
Recommendation Models (DLRMs)

DLRM Training Bottlenecks

Disaggregated Multi-Tower (DMT)

Effectiveness of DMT in Large-Scale
Recommendation

Deep Learning Recommendation Model



- Modern recommendation models consist of large embedding tables and a dense network.
- Left is Meta's DLRM.
- DLRM has both dense and categorical input features.
- Categorical features are converted to dense representations (embeddings) by the embedding tables.
- Embeddings then interact with dense features in the dense network, which produces a prediction.

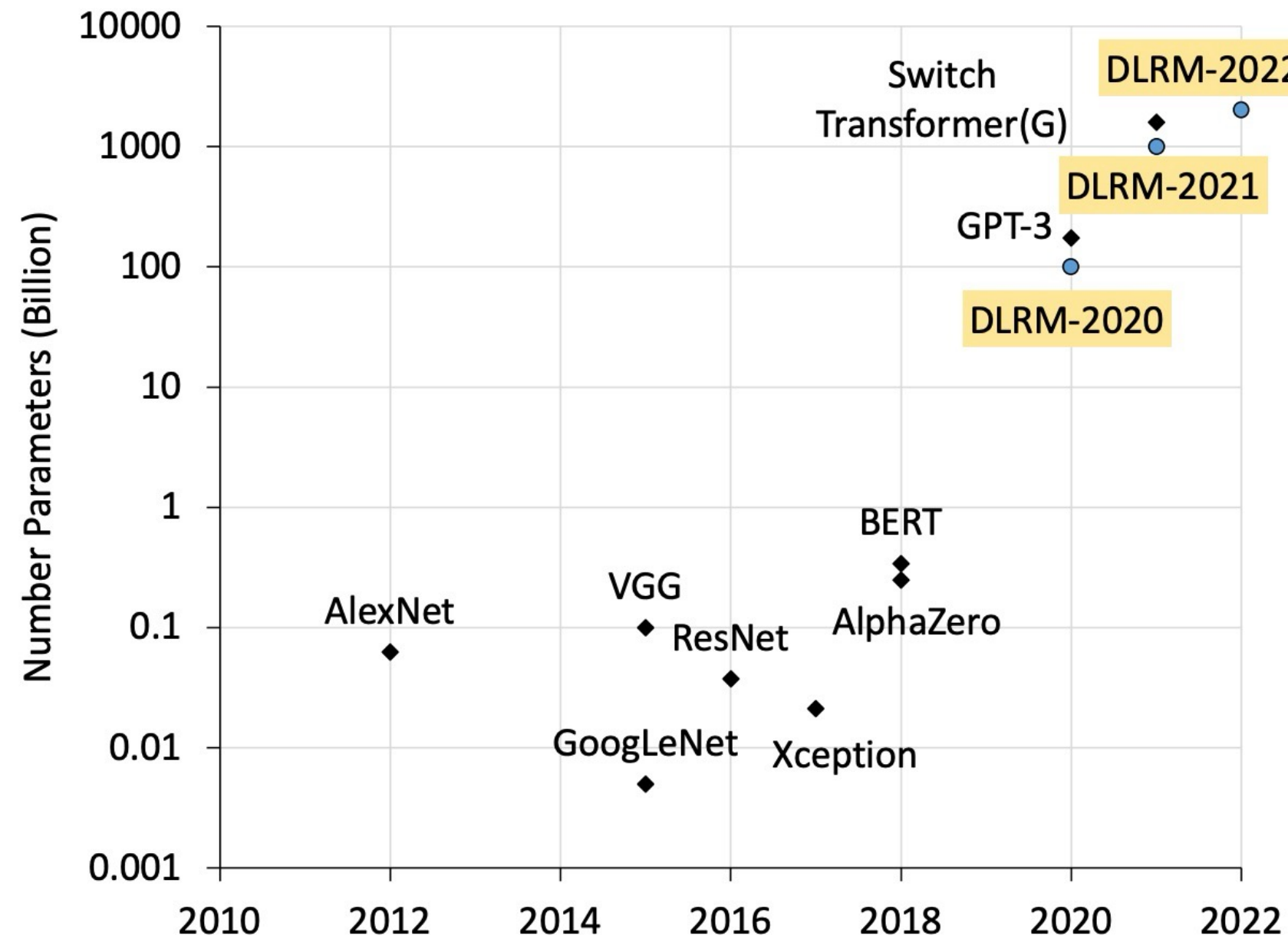
Naumov, M., Mudigere, D., Shi, H.-J. M., Huang, J., Sun-daraman, N., Park, J., Wang, X., Gupta, U., Wu, C.-J., Azzolini, A. G., et al. Deep learning recommendation model for personalization and recommendation systems.



OUR CHALLENGE

Modern recommendation models are so large...
... that I can't even fit them onto this screen

... and they keep getting bigger!

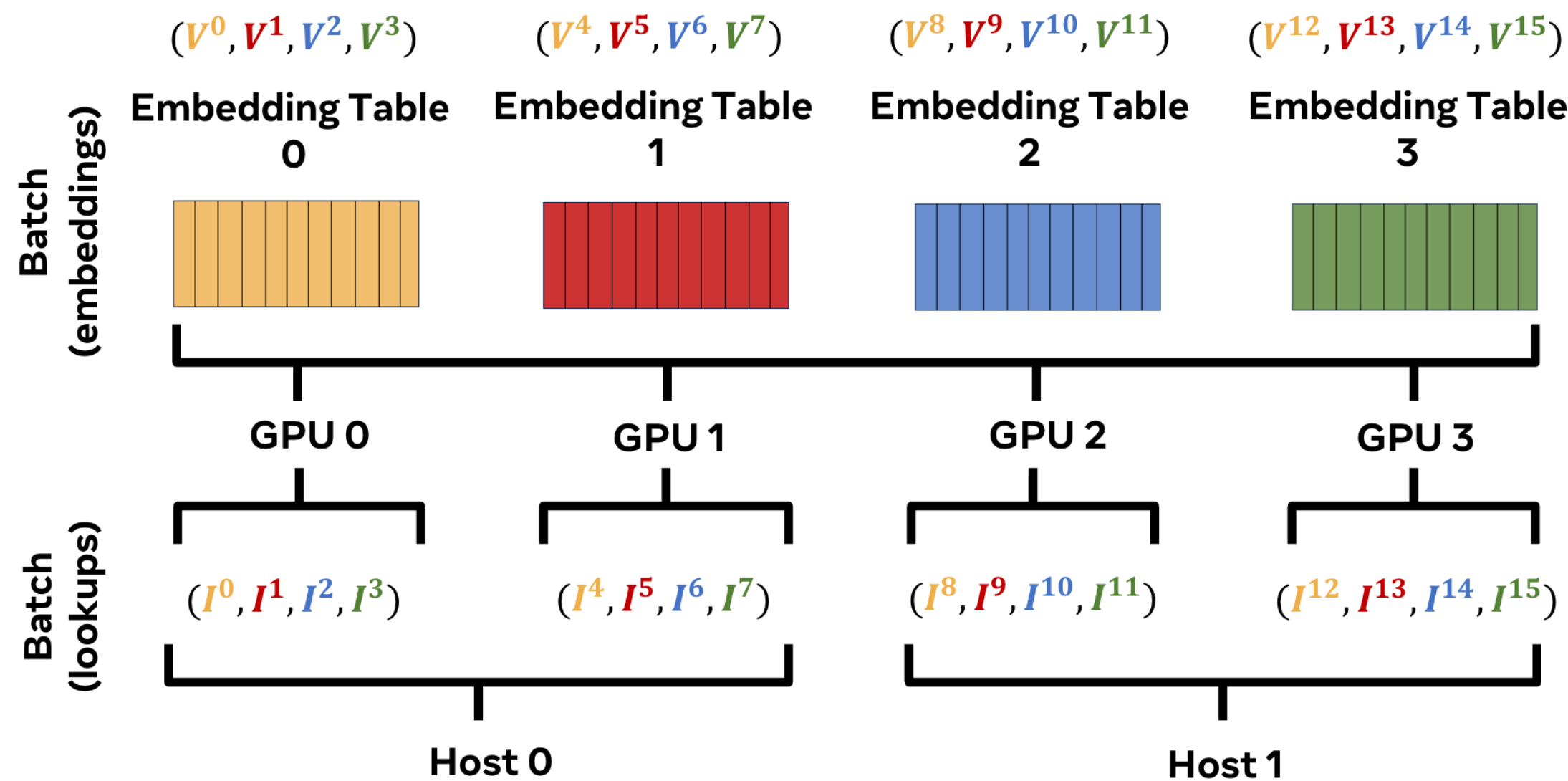


- Vast majority of parameters are embedding tables.
- High performance, GPU training is a must.
- Distributed training is a must.

Mudigere, D., Hao, Y., Huang, J., Tulloch, A., Sridharan, S., Liu, X., Ozdal, M.,
Nie, J., Park, J., Luo, L., et al.

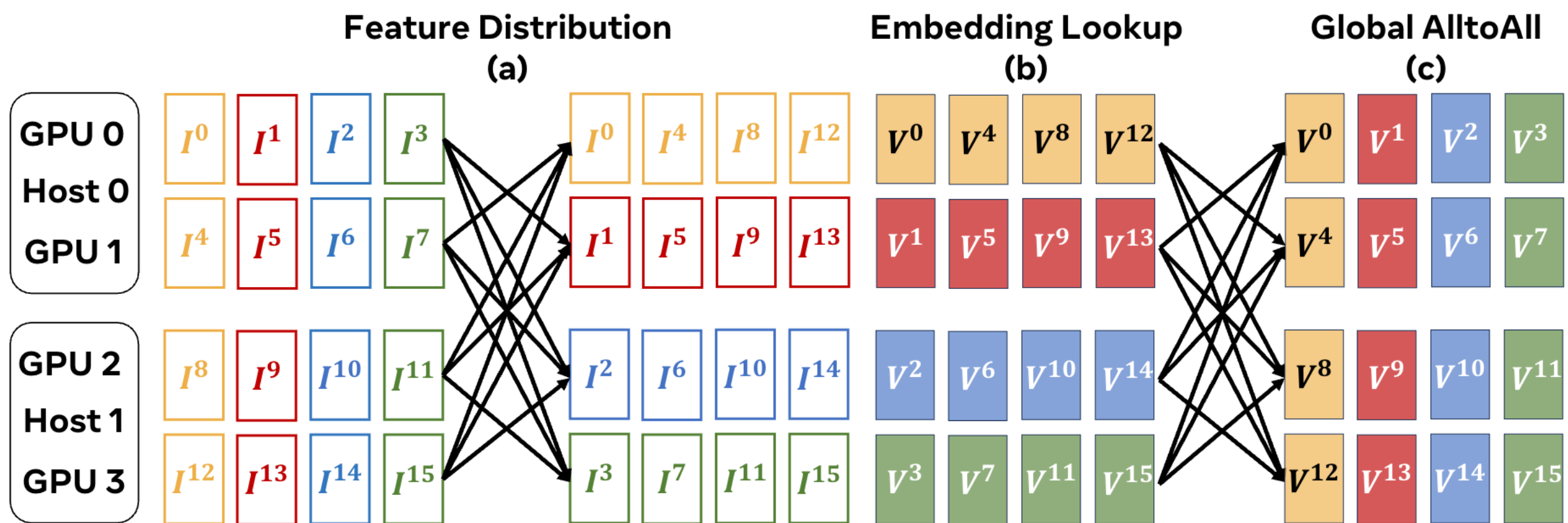
High-performance, distributed training of large-scale deep learning
recommendation models

The Distributed Embedding Lookup Process



- Converts discrete objects into dense representations.
- Example shows 4 GPUs on 2 hosts, each contains 1 embedding table, color-coded.

The Distributed Embedding Lookup Process: An Operational View

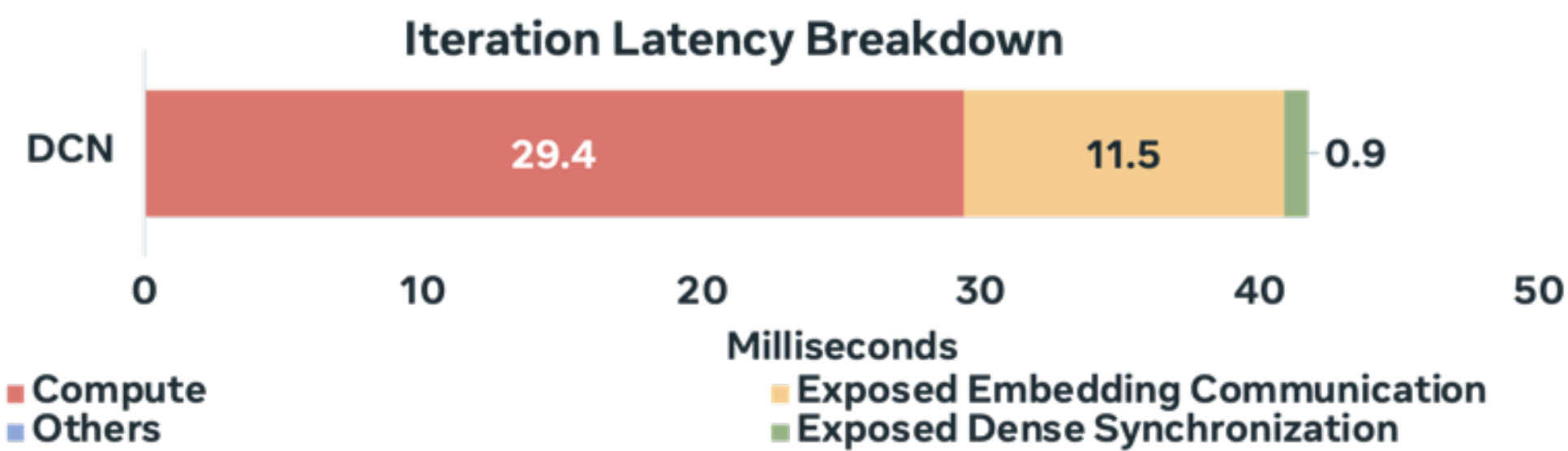


- Step a: a global feature AlltoAll routes local inputs to the GPU that holds the embedding table for the features.
- Step b: each GPU performs a local embedding lookup, producing embeddings.
- Step c: another global embedding AlltoAll transports embedding results back to each GPU.

OUR CHALLENGE

Training Efficiency of DLRLMs on SOTA Hardware and Frameworks can be Improved.

Up to 30% of Iteration Time Wasted on Explicitly Waiting for Embedding Communication

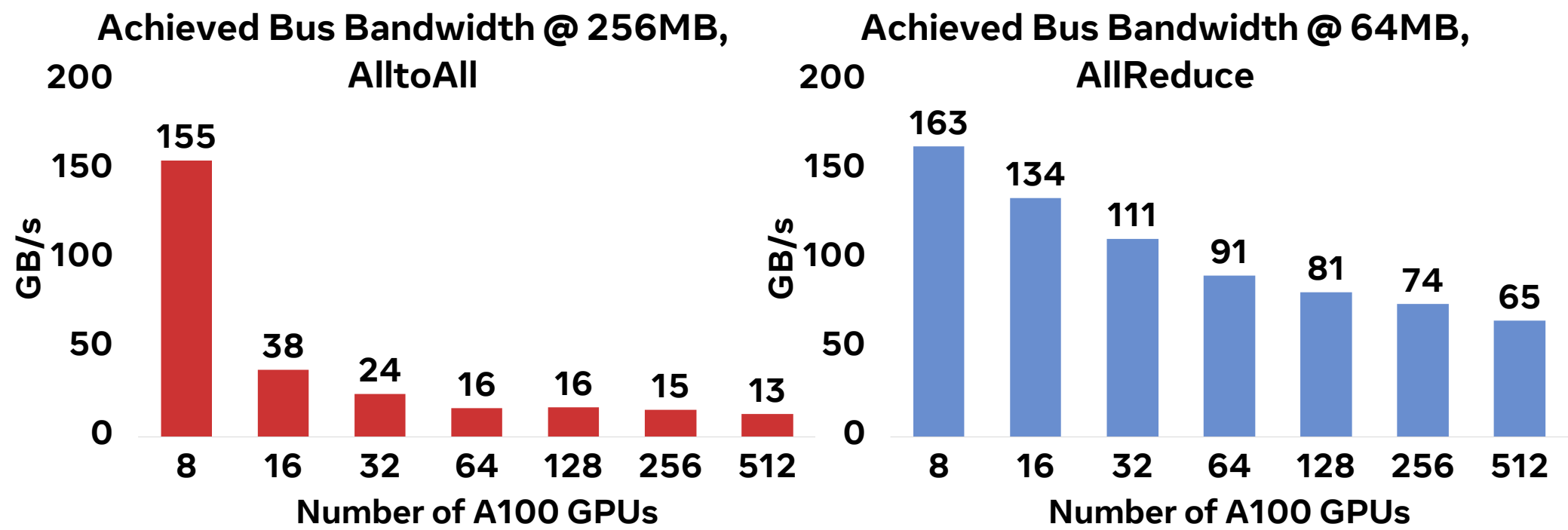


- Measured on a 64-H100 Cluster training a modern recommendation model using TorchRec.
- Applied SOTA optimizations including communication quantization and overlapping of compute and communication.

UNDERSTANDING THE TRAINING BOTTLENECK

NCCL Scalability and Model Architecture Contributed to this Bottleneck


NCCL Scalability and Model Architecture Contributed to this Bottleneck



- NCCL AlltoAll and AllReduce Scalability on an A100 cluster: degrading Collectives performance.
- Global feature interaction, together with the training paradigm, dictate multiple rounds of global AlltoAlls for embedding lookup.
- The training paradigm fails to exploit locality within a single host.
- Likely to cause even larger problem on future hardware, as the compute capacity continues to overshadow network and memory bandwidth.


A TOPOLOGY-AWARE TECHNIQUE FOR EFFICIENT LARGE-SCALE RECOMMENDATION

Disaggregated Multi-Tower




Semantic- Preserving Tower Transform

SPTT decomposes the monolithic distributed global embedding lookup into parallel AlltoAlls to exploit data center locality with local data shuffles



Hierarchical Feature Interaction via Tower Modules

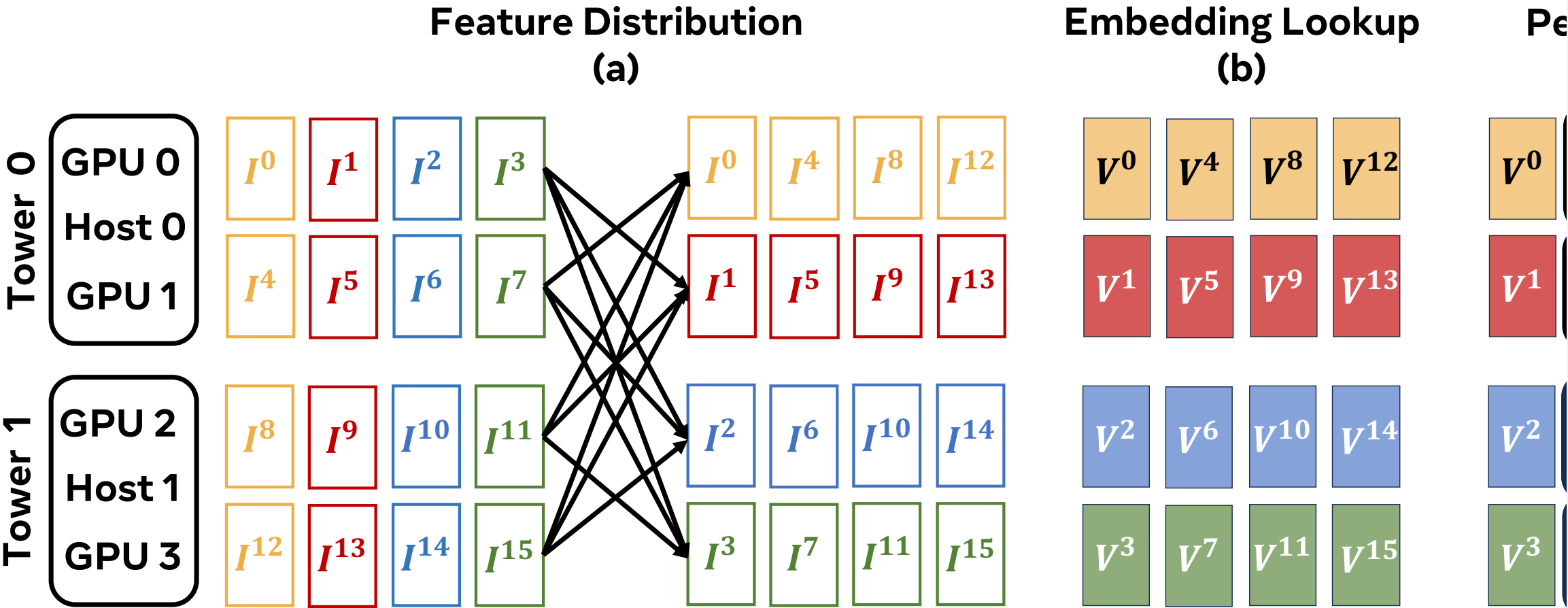
TMs are synergistic modules with SPTT that are attached to towers to reduce model complexity and communication volume through hierarchical feature interaction



Learned, Balanced Feature Partition

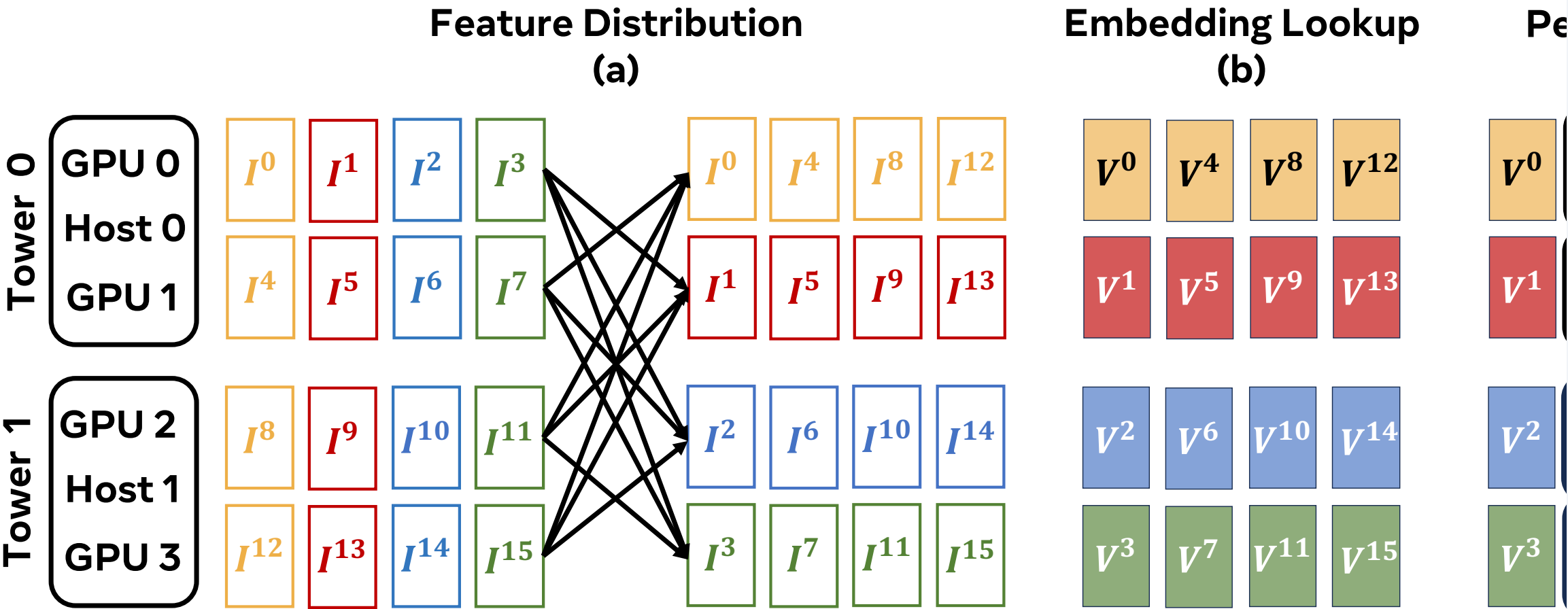
BFP systematically creates towers with meaningful feature interactions and load balanced assignments to preserve model quality and training throughput via learned embeddings

Semantic-Preserving Tower Transform



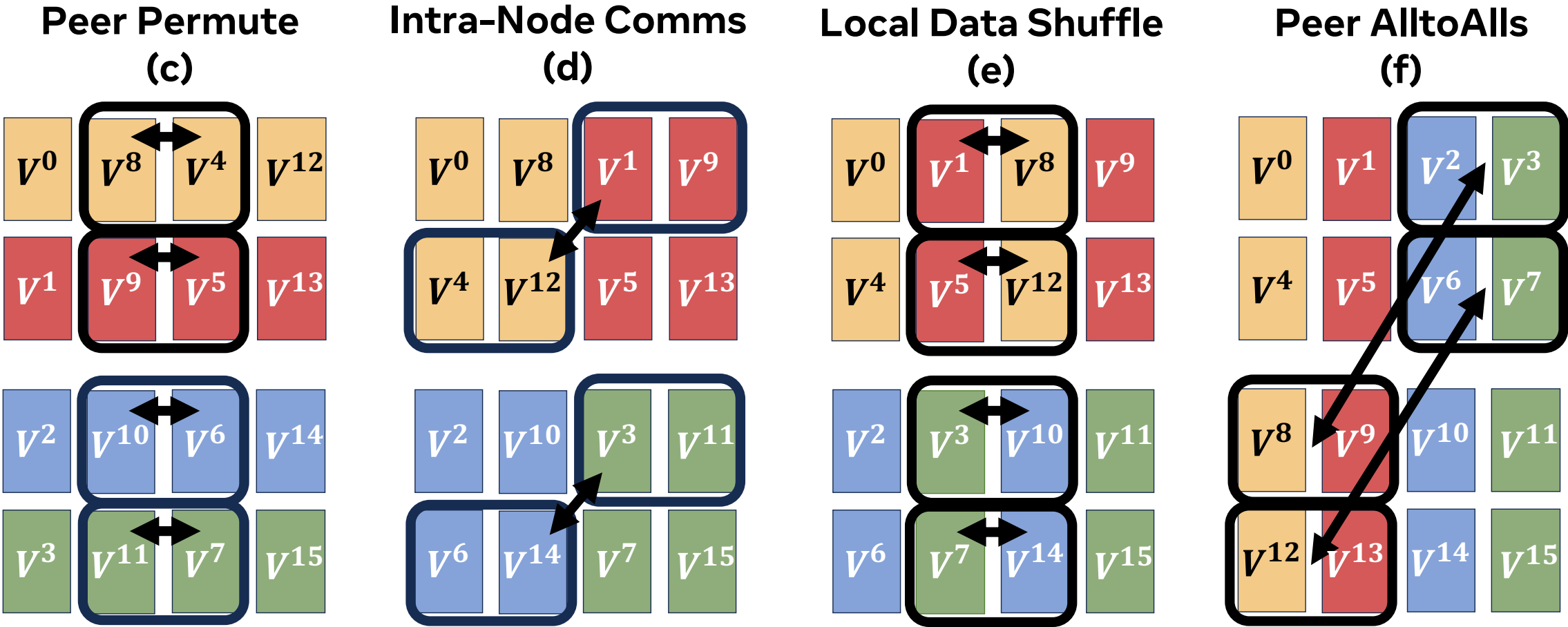
SPTT decomposes the monolithic distributed global embedding lookup into parallel AlltoAlls to exploit data center locality with local data shuffles.

Semantic-Preserving Tower Transform



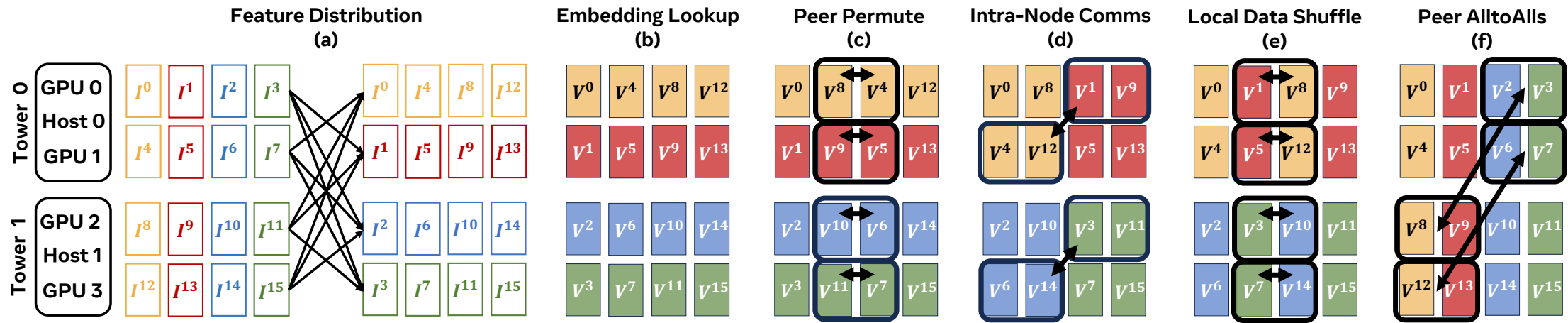
- SPTT decomposes the monolithic distributed global embedding lookup into parallel AlltoAlls to exploit data center locality with local data shuffles.
- Proceed to embedding lookup in the classical case (step a-b).
 - Decomposes the embedding lookup results distribution global AlltoAll into parallel Peered AlltoAlls through a series of local shuffles.
 - Example showing two towers on two hosts, each with two GPUs.

Semantic-Preserving Tower Transform



- Step c: performs a local peer shuffle, allowing data to be peer-ordered (local GPU rank, host rank).
- Step d: performs an intra-host shuffle, so that a GPU carries all embedding results for all its peers for all features assigned to its tower. Implemented as an AlltoAll operations leveraging NVLink.
- Step e: performs another local shuffle, so that the features are in the right order of (peers, features).
- Step f: performs parallel AlltoAll collectives cross-host. Each AlltoAll consists of only peer GPUs.

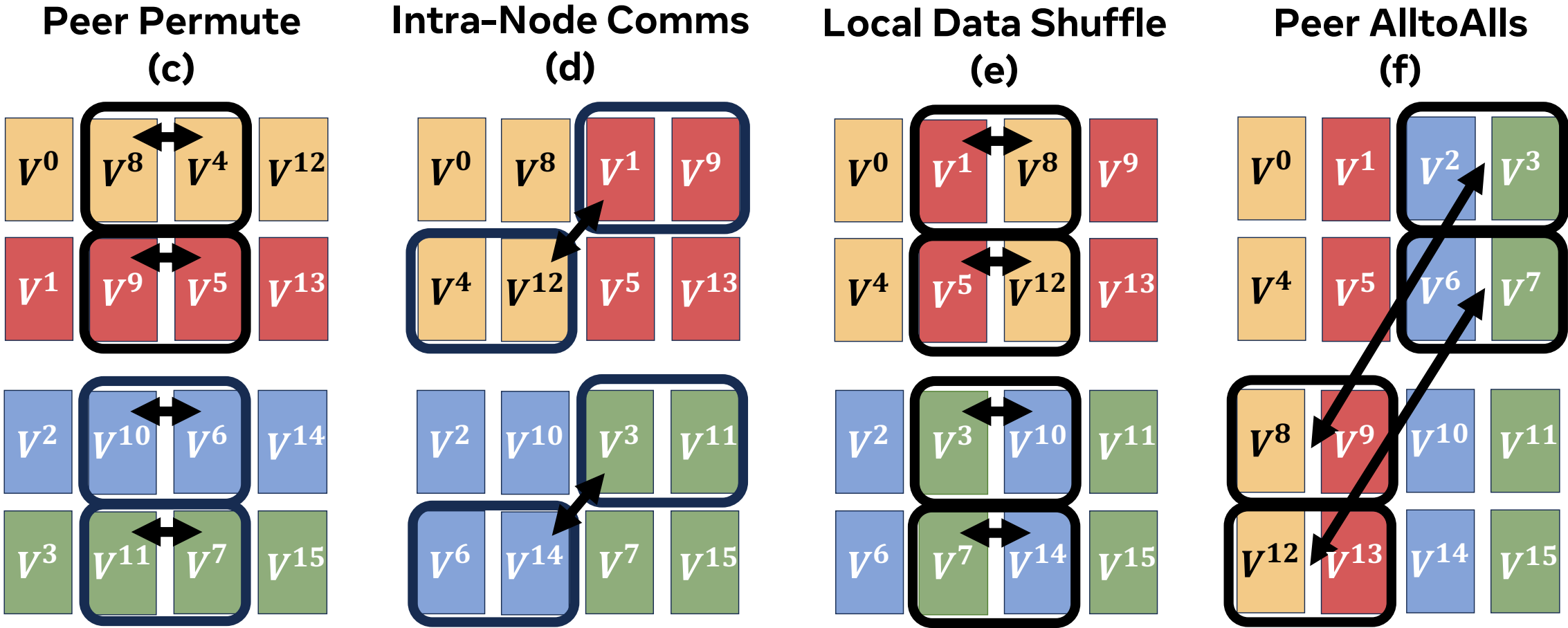
Semantic-Preserving Tower Transform



SPTT does not reduce total bytes on wire, and adds additional overheads in local shuffles, but it:

- Improves cross-host AlltoAll efficiency by reducing world size
- Fully utilizes NVLink for efficient intra-host data shuffles
- Sets up the proper data layout for more data volume compression through hierarchical feature interaction.

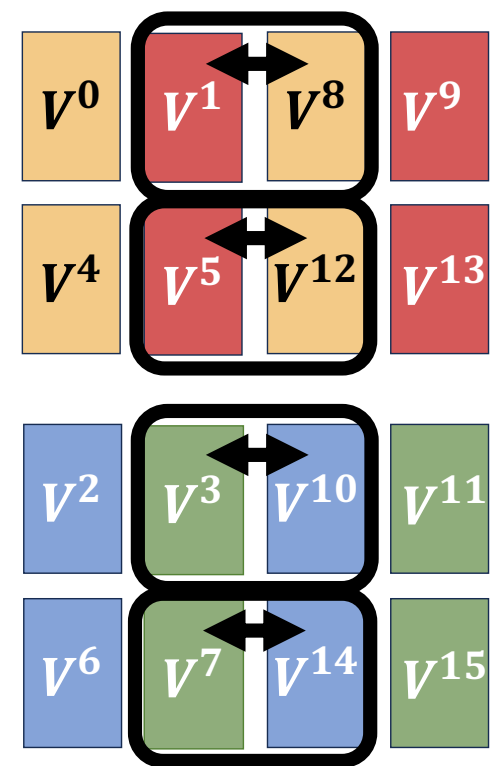
Hierarchical Feature Interaction via Tower Modules



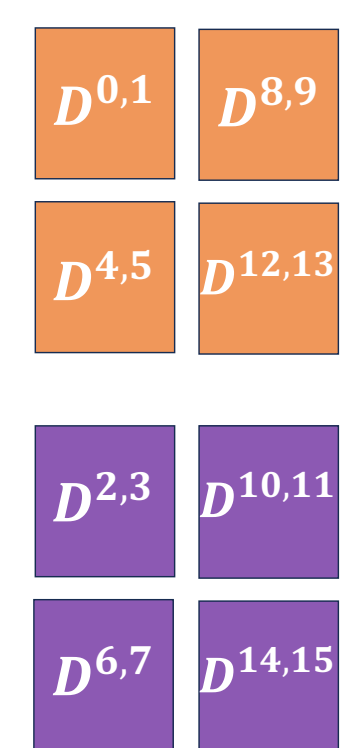
TMs are synergistic modules with SPTT that are attached to towers to reduce model complexity and communication volume through hierarchical feature interaction.

Hierarchical Feature Interaction via Tower Modules

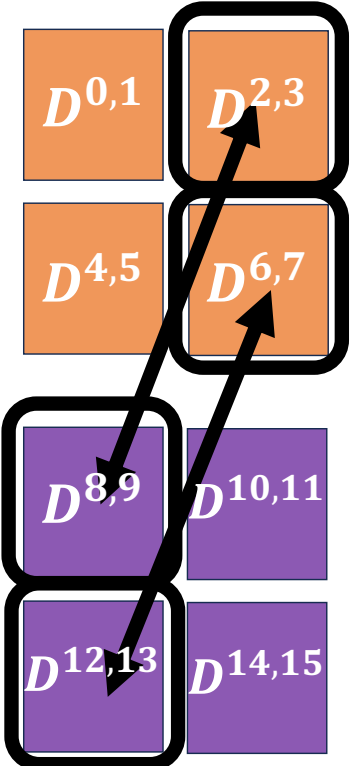
Local Data Shuffle
(e)



Tower Module
(f)



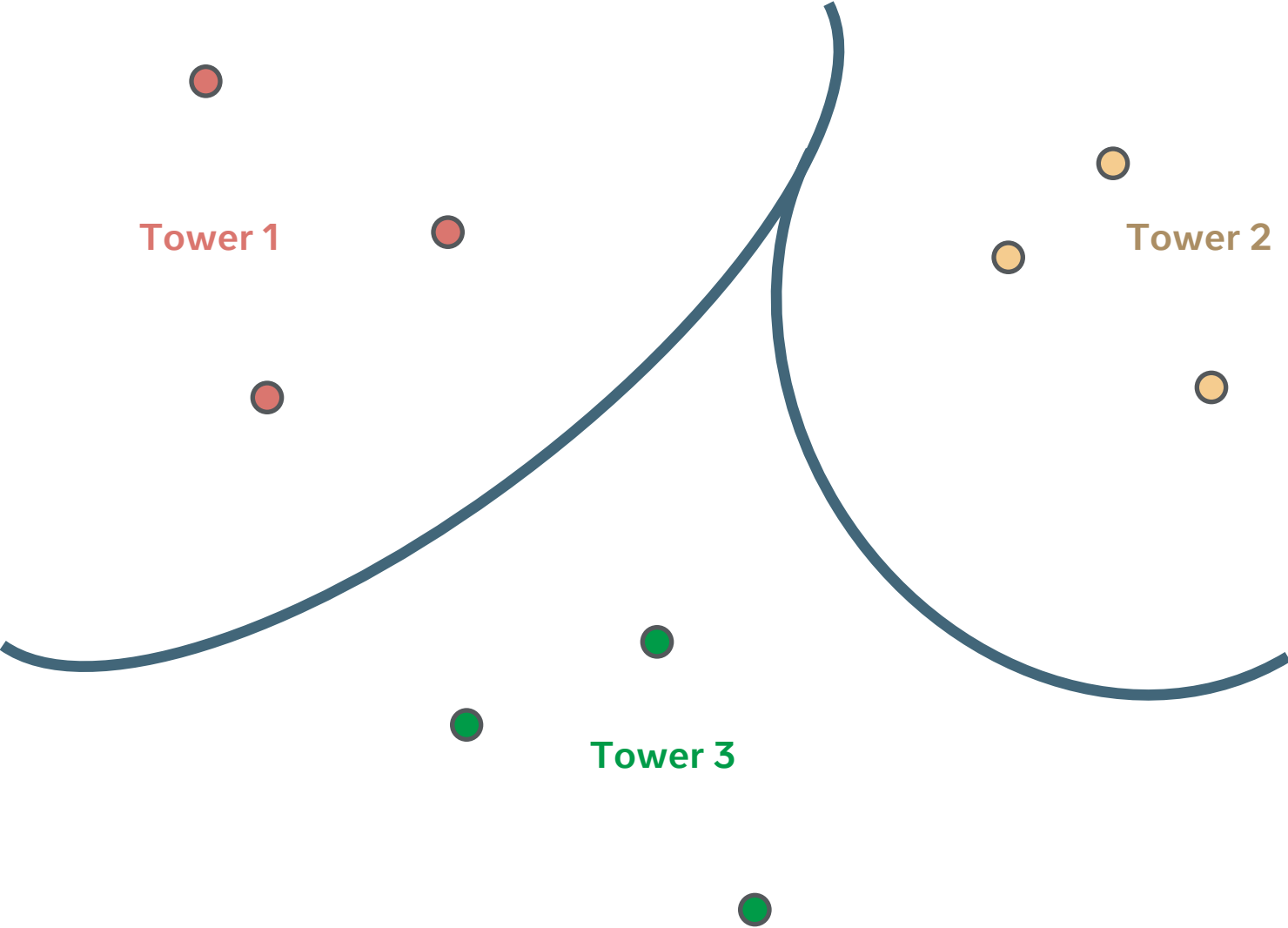
Peer AlltoAlls
(g)



TMs are synergistic modules with SPTT that are attached to towers to reduce model complexity and communication volume through hierarchical feature interaction.

- Step e in SPTT ensures each GPU has all embeddings for all the features for its peers, making it a perfect place to set up dense modules to further compress embedding.
- Step f: applies Tower Modules to each GPU, interacting only features assigned to each tower, producing a small embedding digest for a group of embeddings.
- Step g: the Peered AlltoAll now transports these digests instead of raw embeddings, reducing network cost.

Learned, Balanced Feature Partition



BFP systematically creates towers with meaningful feature interactions and load balanced assignments to preserve model quality and training throughput via learned partitions.

Learned, Balanced Feature Partition

Given a kernel K and an interaction matrix I where:

$$I_{i,j} = |K(F_i, F_j)|$$

Define a transformation f that converts I to a distance matrix D where:

$$D = f(I)$$

Find for each feature a set of coordinates (x, y) to embed it on a plane, such that they minimize:

$$\sum_{i=1}^{|F|} \sum_{j=1}^{i-1} \|d_{i,j} - D_{i,j}\|_2$$

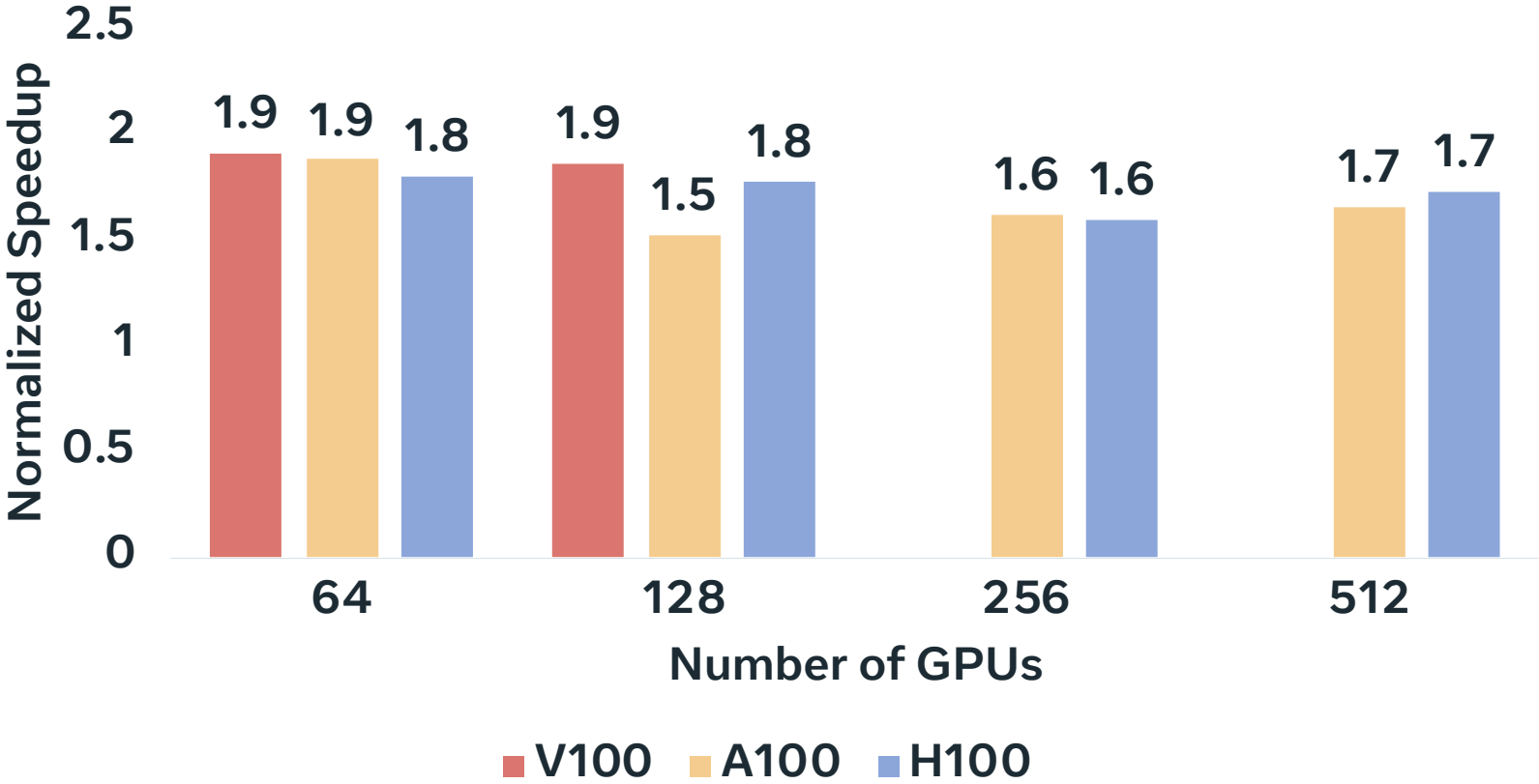
Where

$$d_{i,j} = \|x_i - x_j\|_2 + \|y_i - y_j\|_2$$

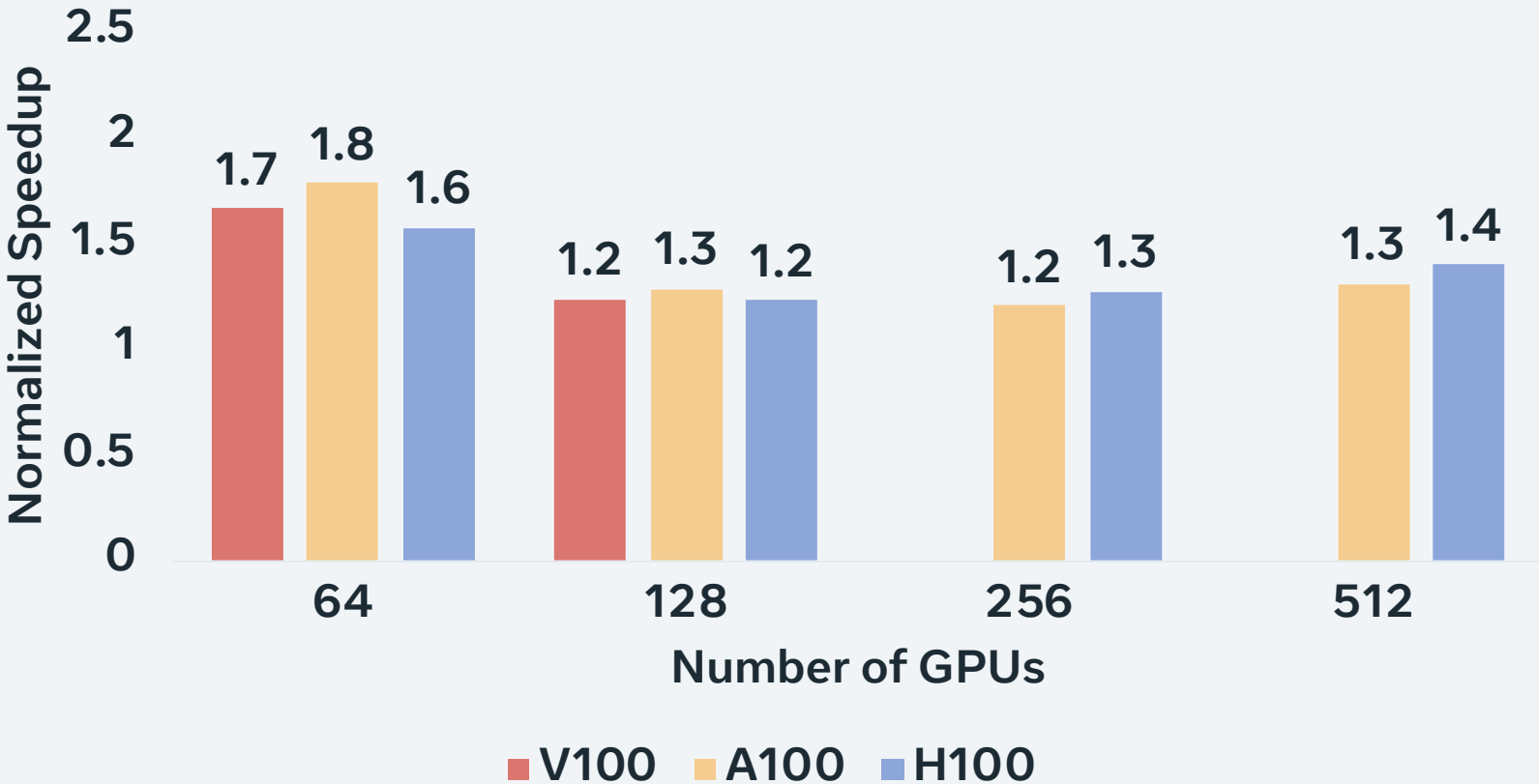
Given a set of input features \mathbf{F} , partition them into G towers such that the interactions within a group is meaningful, and the workload in each tower is balanced.

- Perform the embedding step on the left.
- Run a constrained K-Means algorithm on the embedding plane, such that features with meaningful interaction are kept together.
- The constrained K-Means process also ensures the group sizes are balanced, so each partition has balanced same workload.
- Typically, K is the cosine similarity, and I is simply the dot product matrix, and f can simply be $1 - I$ (1 minus identity) or I (identity).

Up to 1.9X Throughput on DLRM

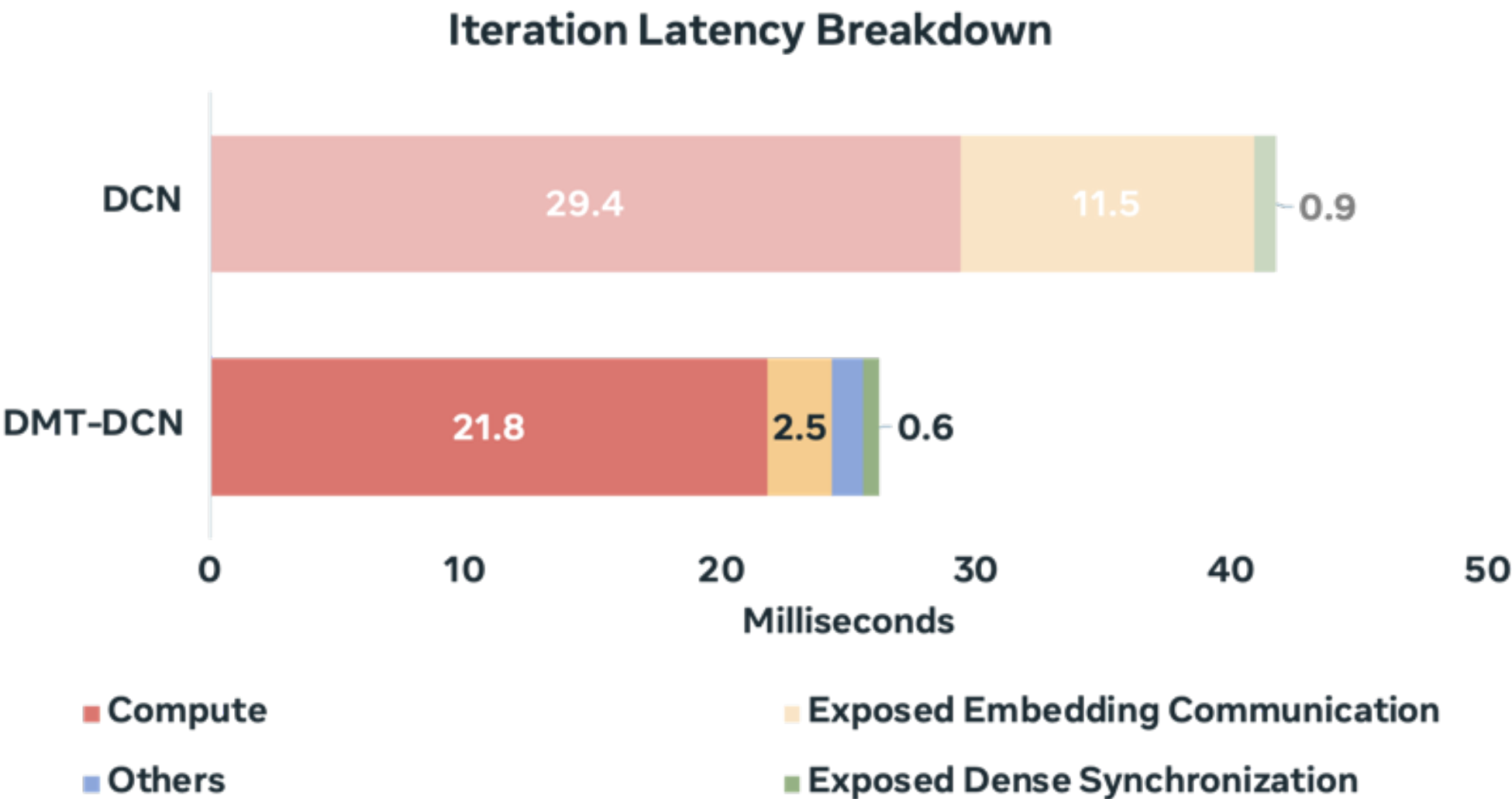


Up to 1.8X Throughput on DCN



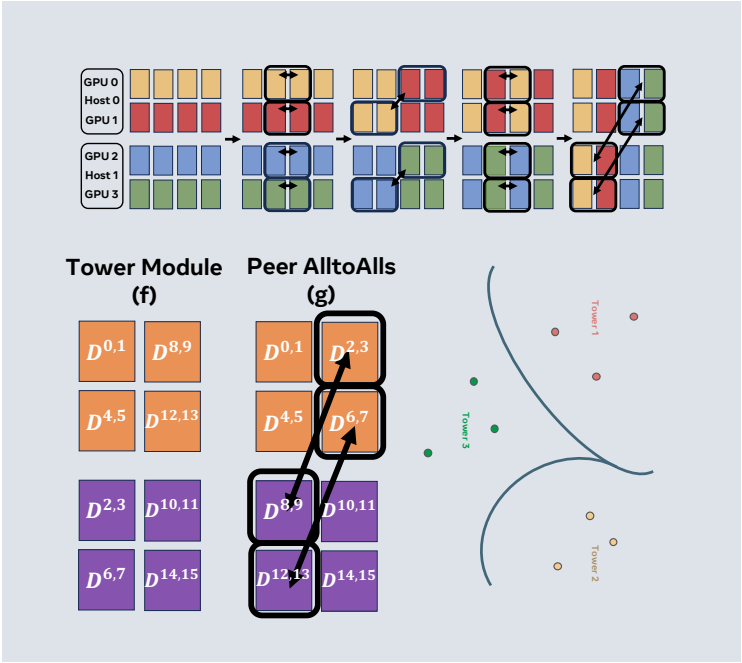
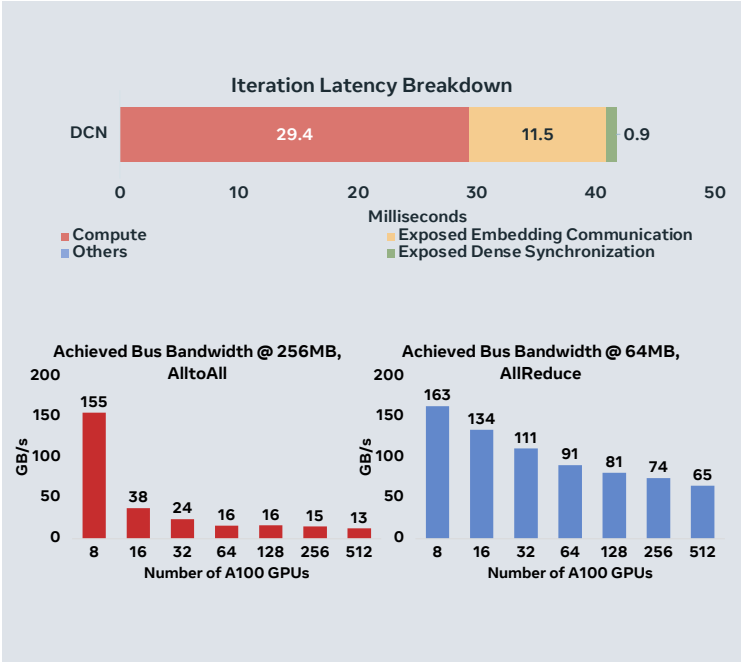
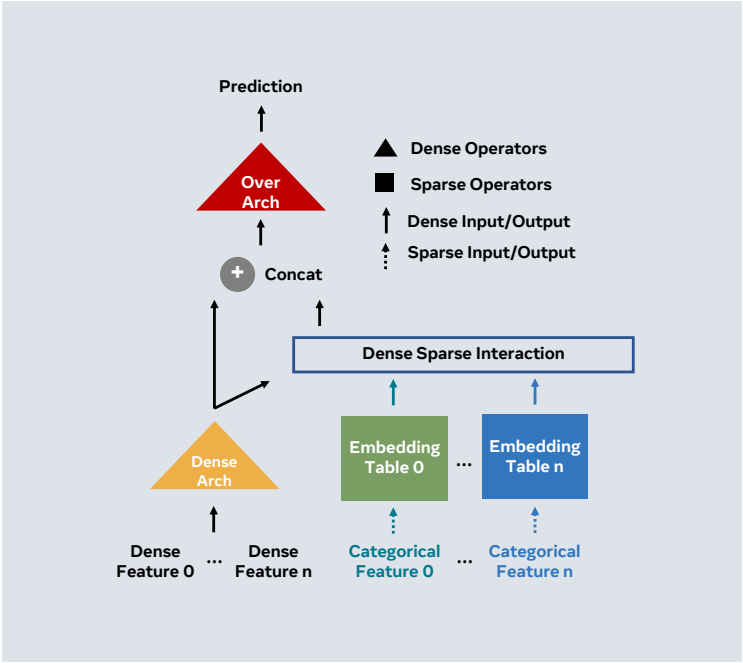
Experiment results obtained versus official TorchRec implementation. Speedup achieved with on-par AUC metrics.

Ablation Study: Where does the Gain Come From?



- Significantly reduced compute and overhead.
- Significantly reduced exposed embedding communication cost.

CONCLUSION



DLRMs

Deep learning-based models with large embedding tables.

Training Bottleneck

NCCL scalability and model and architecture make DLRMs hard to train.

DMT

Topology-aware modeling techniques that tackles the training bottlenecks via SPTT, TM and BFP.

Paper Available

Checkout the DMT paper for a more detailed description of DMT and experimental setups.

 Meta AI