

CS 744: BAGPIPE

Shivaram Venkataraman

Spring 2025

ADMINISTRIVIA

- Course Project
 - Introduction grades
 - Checkins
- Midterm grading

Applications

Machine Learning

SQL

Streaming

Graph

Computational Engines

Scalable Storage Systems

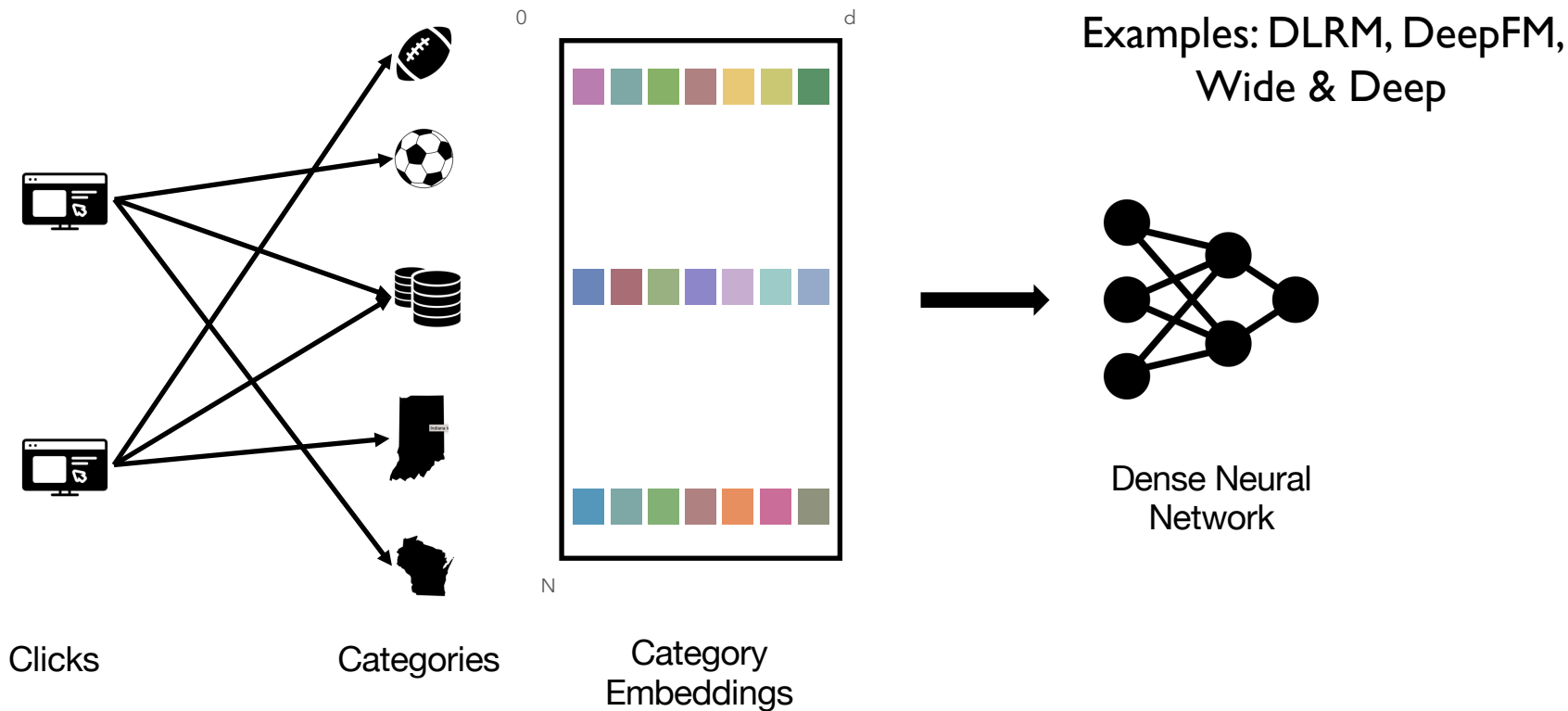
Resource Management

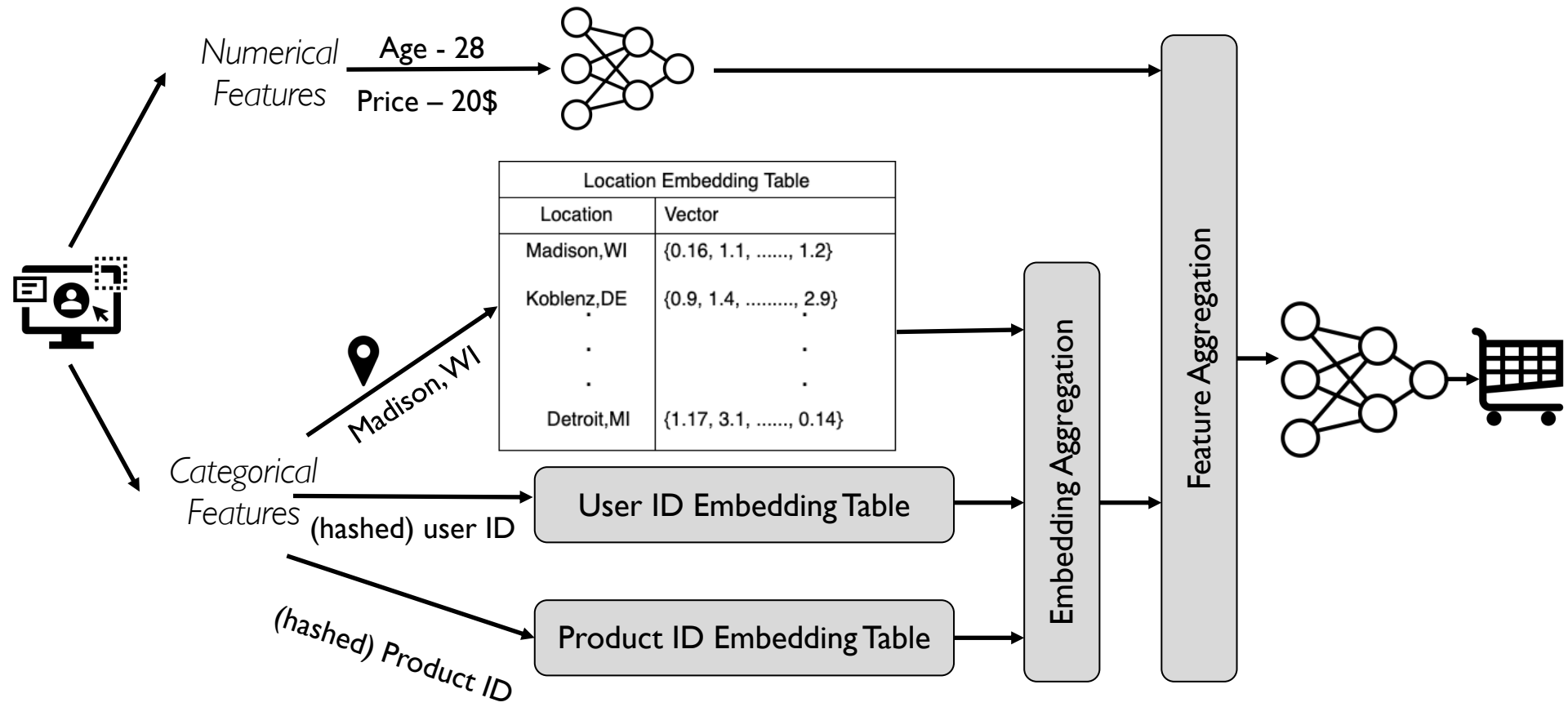


Datacenter Architecture



RECOMMENDATION MODELS





EMBEDDING TABLES

Convert categorical features to numerical features
Example: Geographic Location to a vector

Extremely memory intensive, could be up to TBs

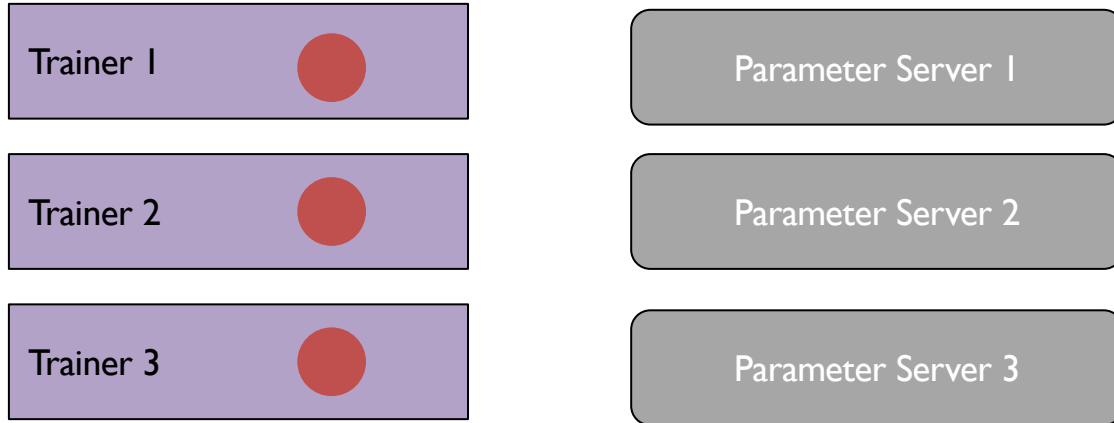
Have sparse access pattern

Geographical Location
Embedding Table

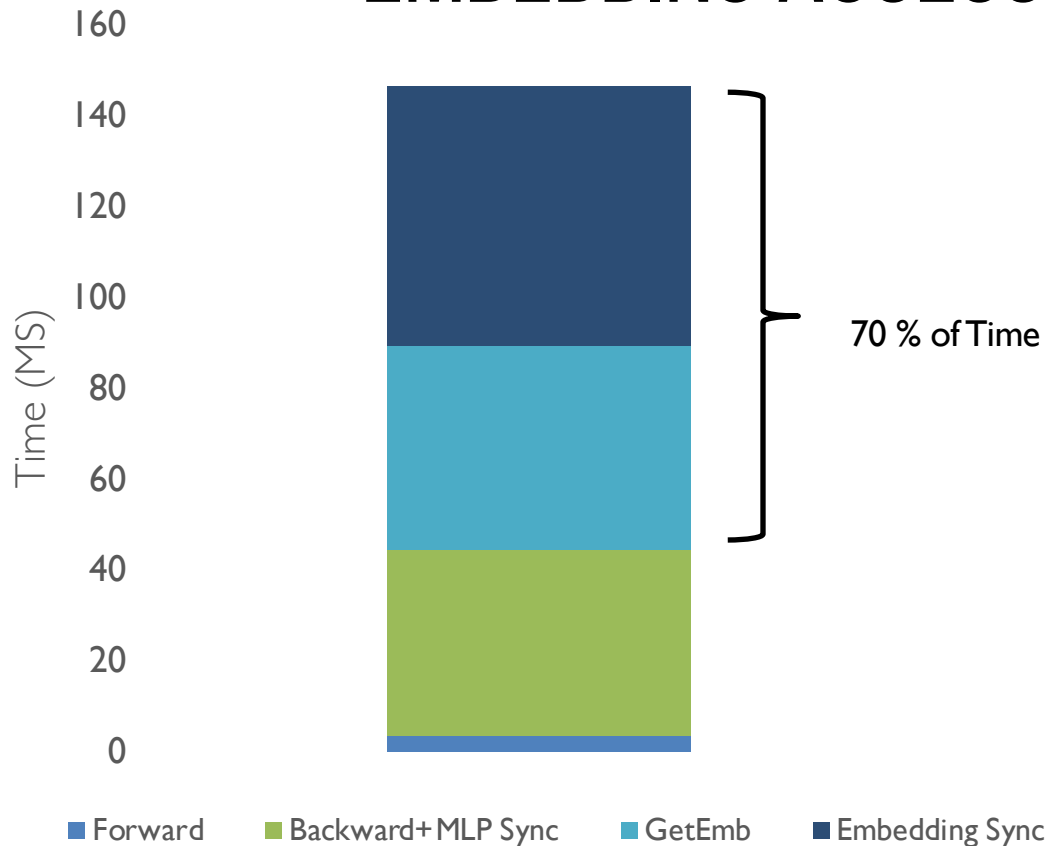
UserID Embedding Table

Product ID Embedding Table

DISTRIBUTED TRAINING ITERATION



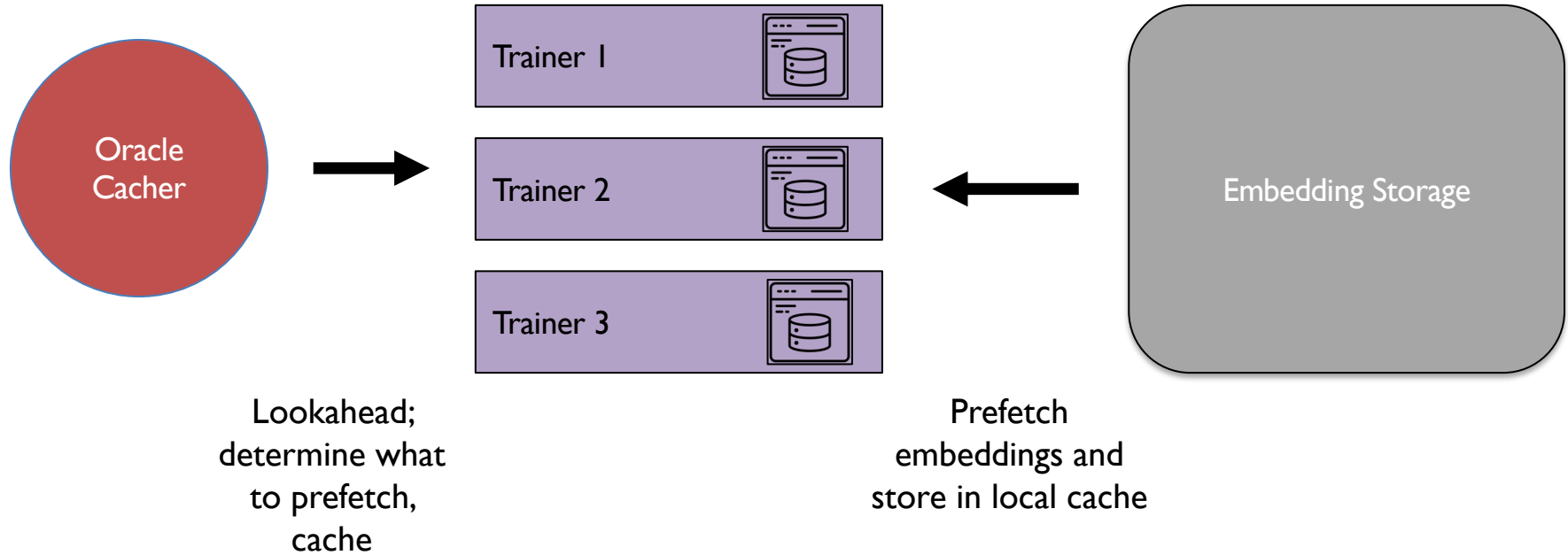
EMBEDDING ACCESS OVERHEADS



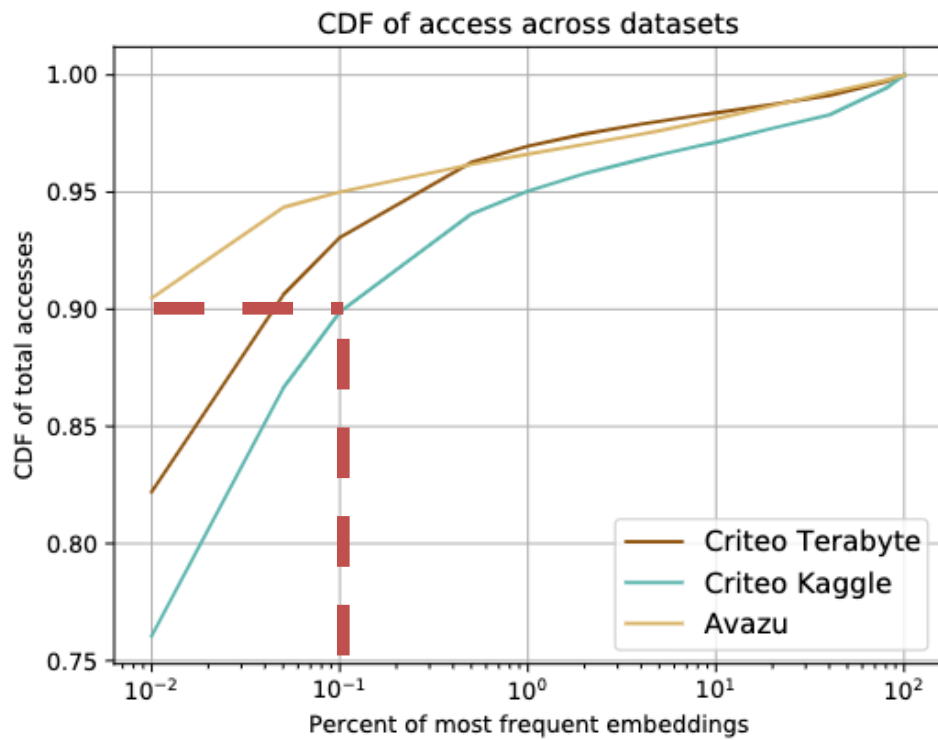
Setup:

- DLRM model
- 8 trainers (p3.2xlarge EC2 instances | V100 each node).
- Batch 2048 per machine.
- Criteo Terabyte Dataset

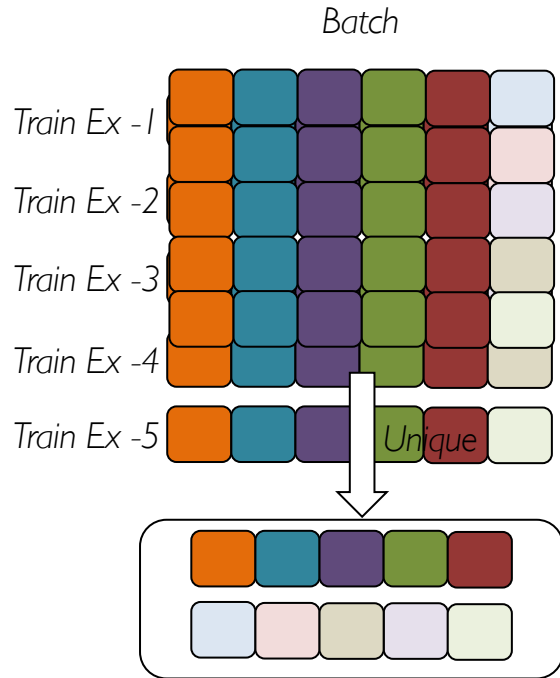
BAGPIPE DESIGN



EMBEDDING ACCESS PATTERNS



“ALL OR NOTHING”

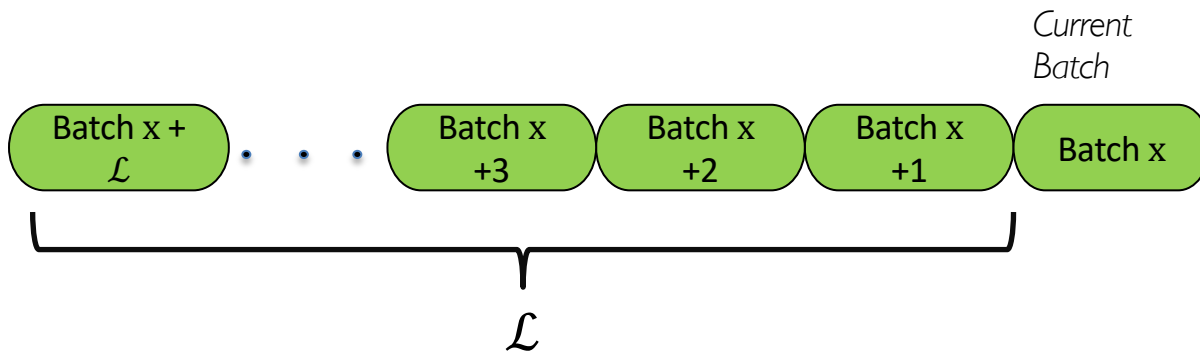


Models are trained with a batch of examples.

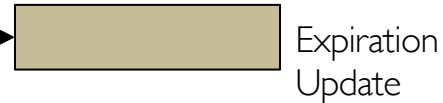
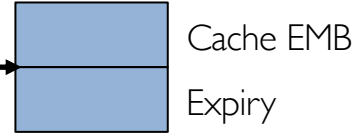
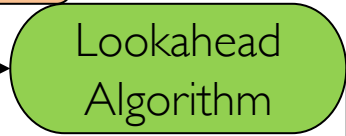
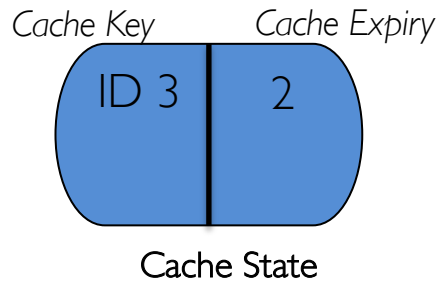
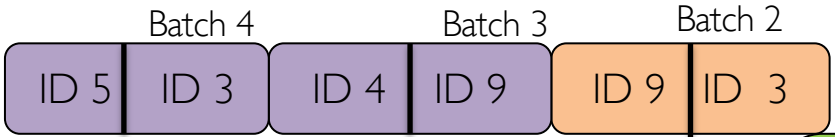
- For a batch only fetch unique embeddings
- Since hot embeddings are replicated, unique embeddings are comprised of long-tail accesses.

LOOKAHEAD ALGORITHM

Look at “ \mathcal{L} ” next batches ahead of current batch to extract access pattern of embeddings by future batches



Look-ahead Value of 2, Batch size of 2



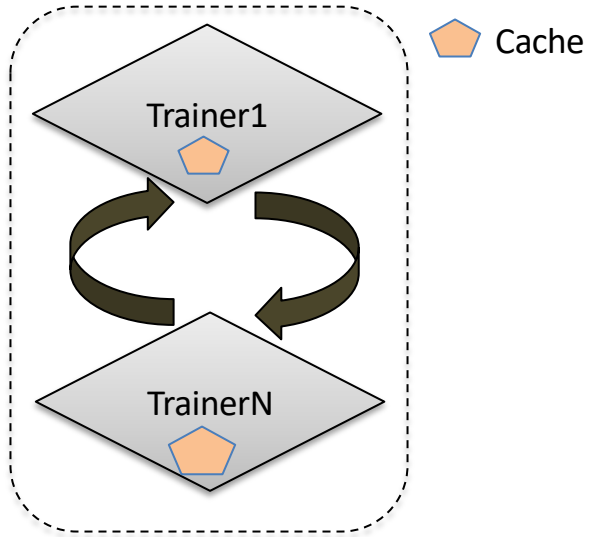
LOOKAHEAD GUARANTEES

An embedding used by batch x , will either be available in cache, or no preceding batch in range $[x - \mathcal{L}, x)$ has accessed it.

Consequently, we can prefetch embeddings used by batch x , once embeddings for batch $x - \mathcal{L}$ have been updated

CACHE SYNCHRONIZATION

At the end of each iteration, each trainer synchronizes caches



SUMMARY

Recommendation models: Embeddings access overheads

BagPipe: Efficient distributed training

- Lookahead to pre-fetch and cache embeddings

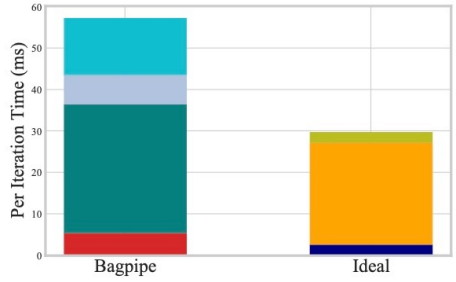
- Cache synchronization across trainers



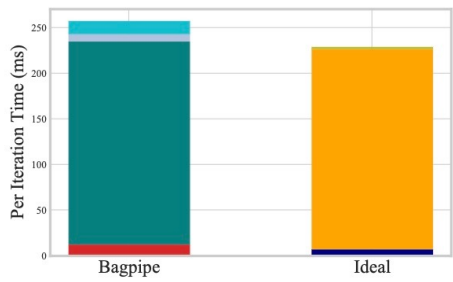
DISCUSSION

<https://forms.gle/9wKQVqNNWcGLJd7N9>

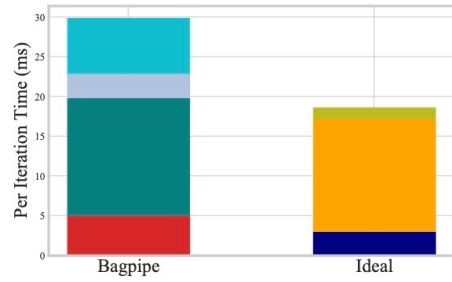
Consider a recommendation model trained on a graph where we use 2-hop neighbors. What are some challenges in using BagPipe-style ideas for such a workload?



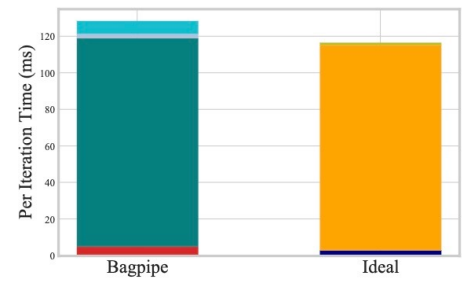
(a) DLRM: p3.2xlarge



(b) DeepFM: p3.2xlarge



(c) DLRM: g5.8xlarge



(d) DeepFM: g5.8xlarge

NEXT STEPS

Next class: Serverless computing