Large scale Recommendations at Instagram

On behalf of Instagram Relevance

Meta AI

Agenda

01 Context

Trends in Recommendation Model Scaling

- In the past 3 years, IG head models have scaled 1000x from <1M FLOPs
- In the same period, training data has grown 5X and feature input size has grown 2X.
- Model Compute continues to be scaled at a rapid clip with Transformer-like architectures

- **Insight** - Keep scaling model complexity constrained by performance improvements and high ROI techniques.

The largest product improvements have stemmed from these step change innovations

A long way to go still

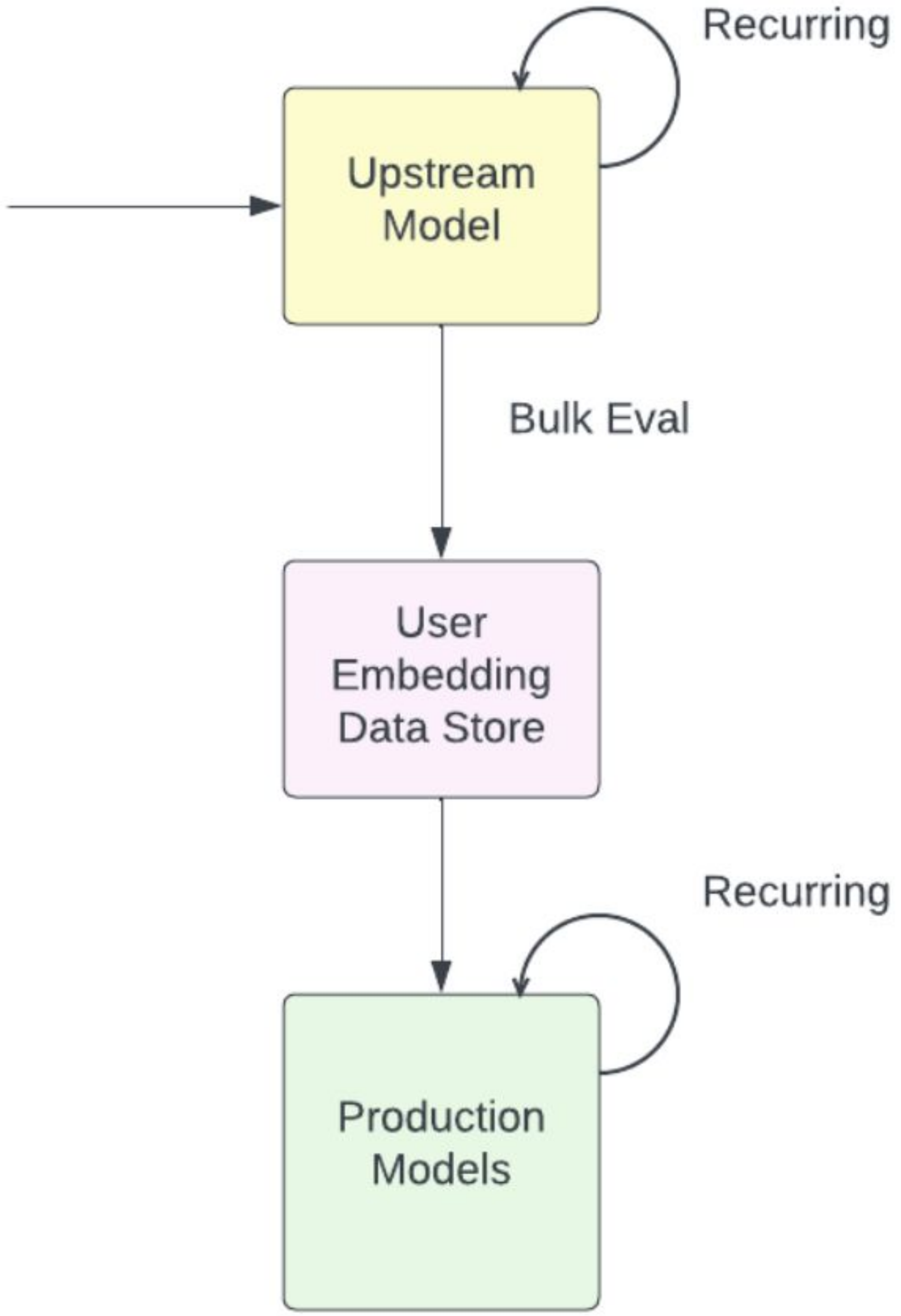
- We’re still capturing only a fraction of available user and ecosystem signals.
- There are still many parts of our recommender system that are understudied
  - Thinking through data feedback loops.
  - Blending to optimize for overall user experience.
- Balancing short term, long term interests with exploration.
- Embracing transformative architectures as they become available.

02 Modeling

Learning Strategy: Foundational Learning

This type of meta modeling architecture provides incredible benefits:

- **Multi modality:** Providing opportunities to our models to generalize to broader user understanding.
- **Cold start cases:** This setup helps us close the feedback loop by leveraging a holistic view of engagement.
- **Consistent baseline quality:** Perhaps the most important value add is the ability to provide consistency in recommendations.

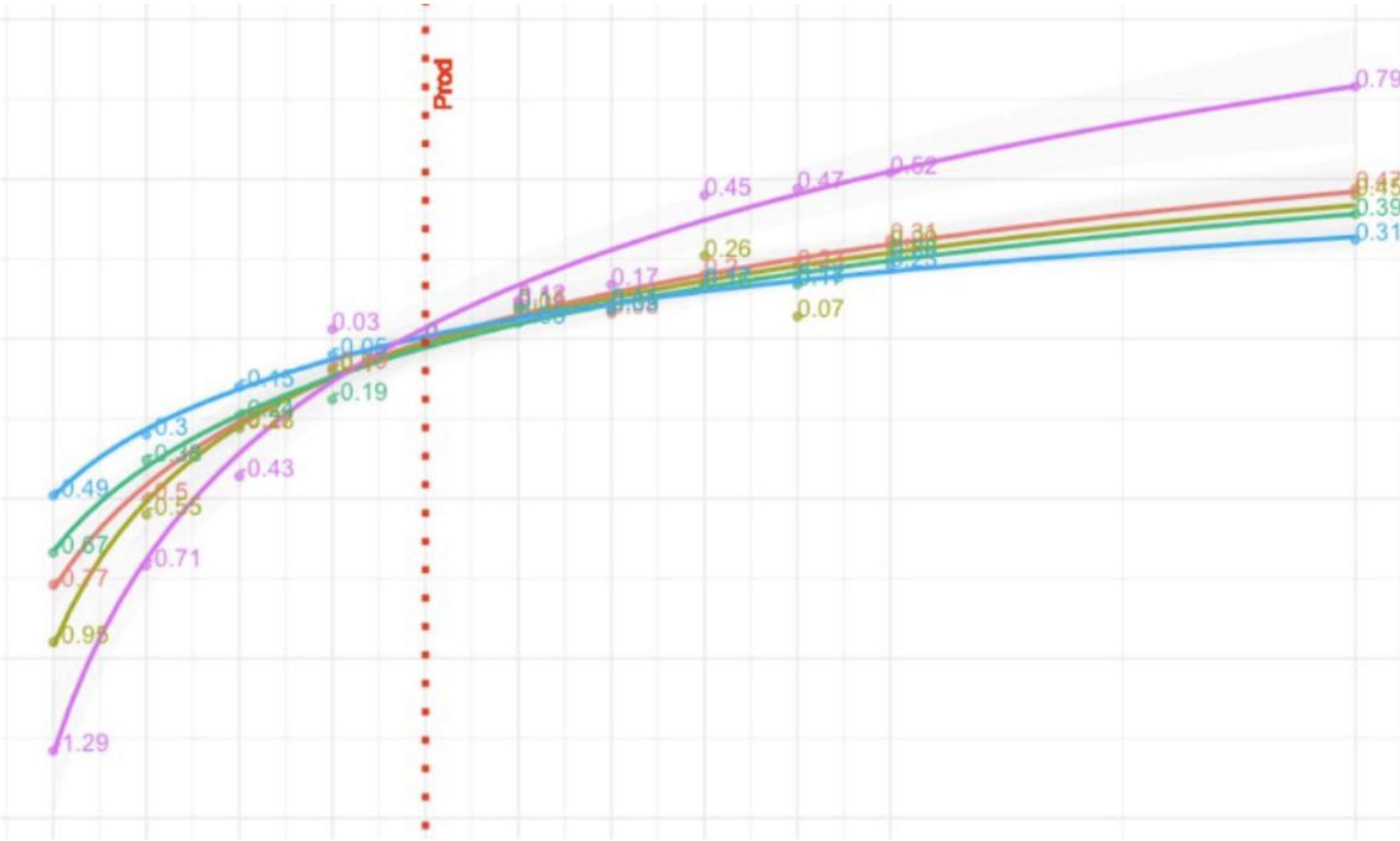


Scaling memoization

- Better retention of available entities (collisions, optimizer, dimension, ..)
- Longer retention and lookback window for users/medias.
- Higher sparsity, higher chance of hash collision because we are dealing with much bigger range of entities.
  - Solution: Decompose high cardinality IDs (semantic IDs)
- Encode heterogeneously (different surface, media type)
- Loss-less knowledge transfer between stages
- Data preprocessing changes with rich representation

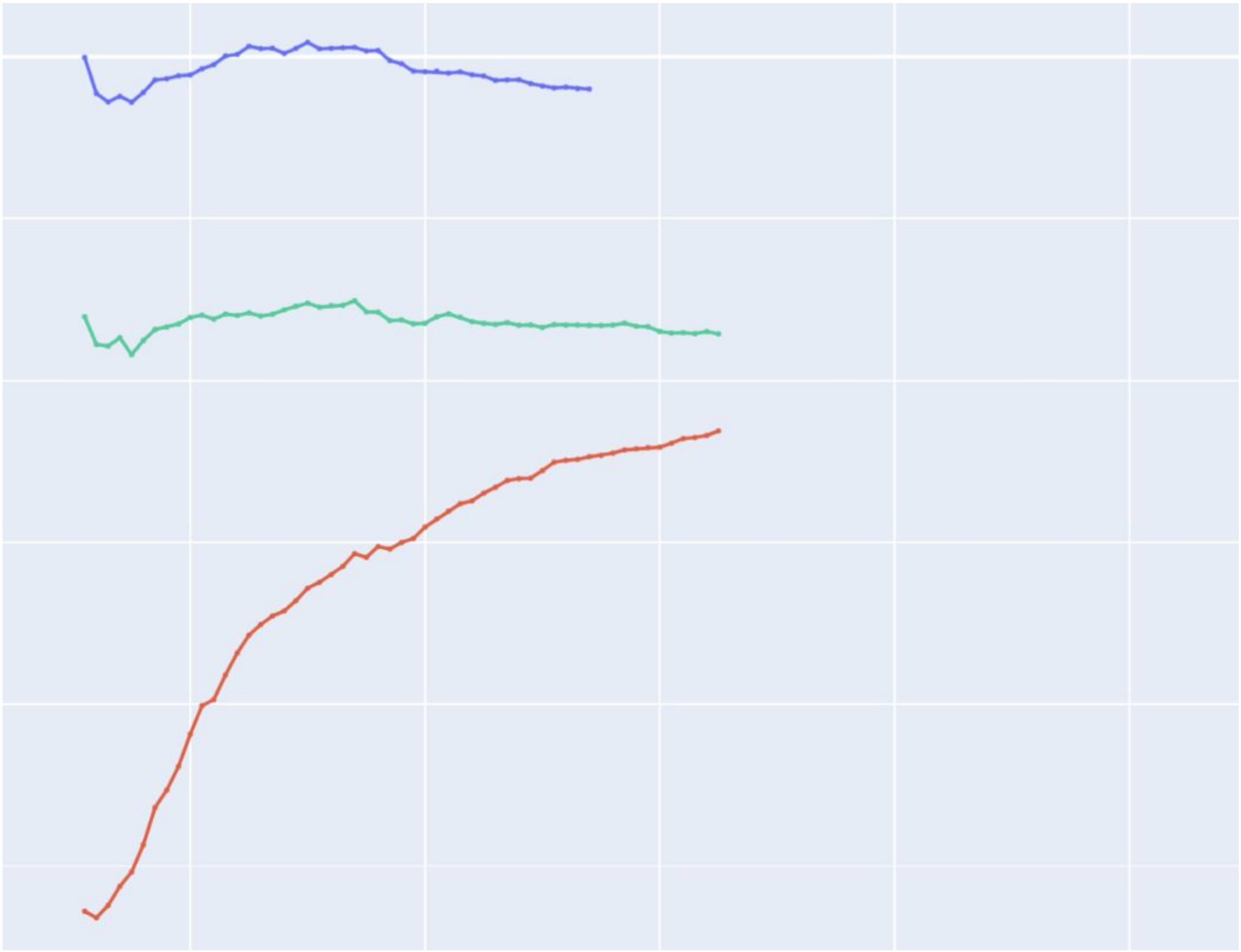
Data Upsampling

- Increasing data coverage for all key entities is a upstream of many other ideas and techniques that can be implemented in modeling.
- High cardinality sparse features have a high variance based on the sample size.
- High complexity when it comes to sampling strategy (across entities, event types, ..)
- Increases the stress on pooling logic.



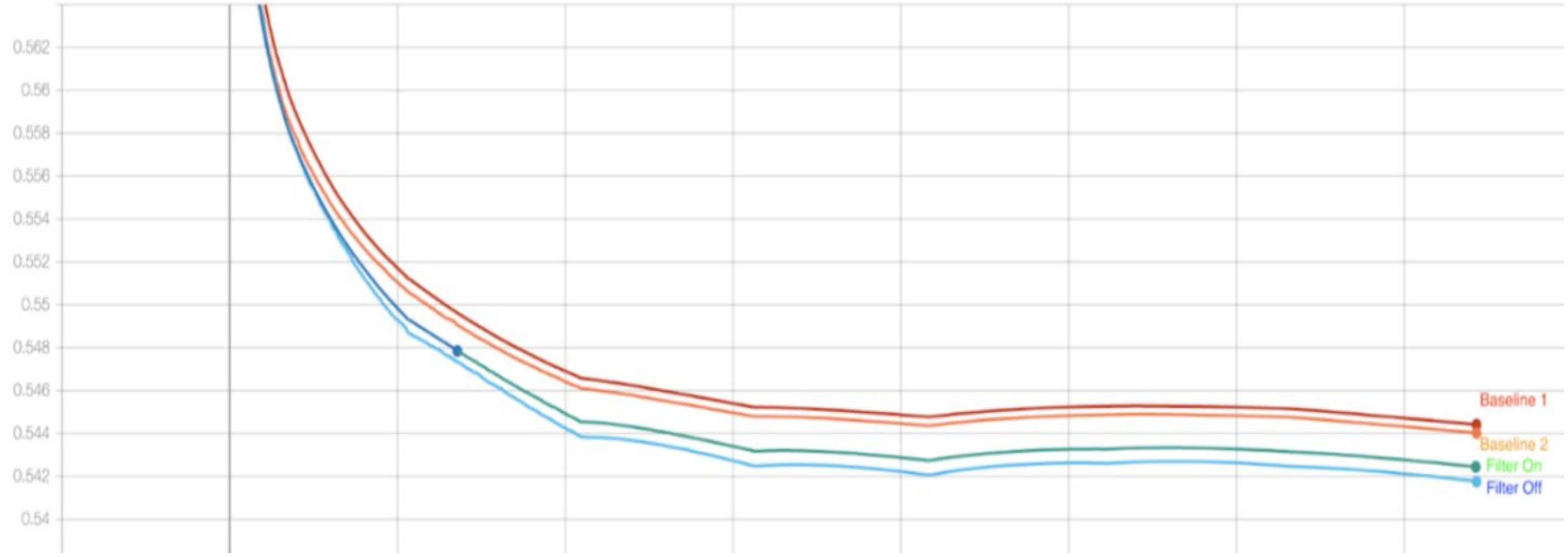
Larger Hash Size

- Increase hash_size for two sparse features from 500K to 50M (100x)
- There are more sophisticated ways to manage sparse collisions
  - They usually require additional memory to keep the correct accounting.
  - The cut-off between zero collision and controlled collision is another hyperparameter
  - Uncertainty where to allocate zero collisions



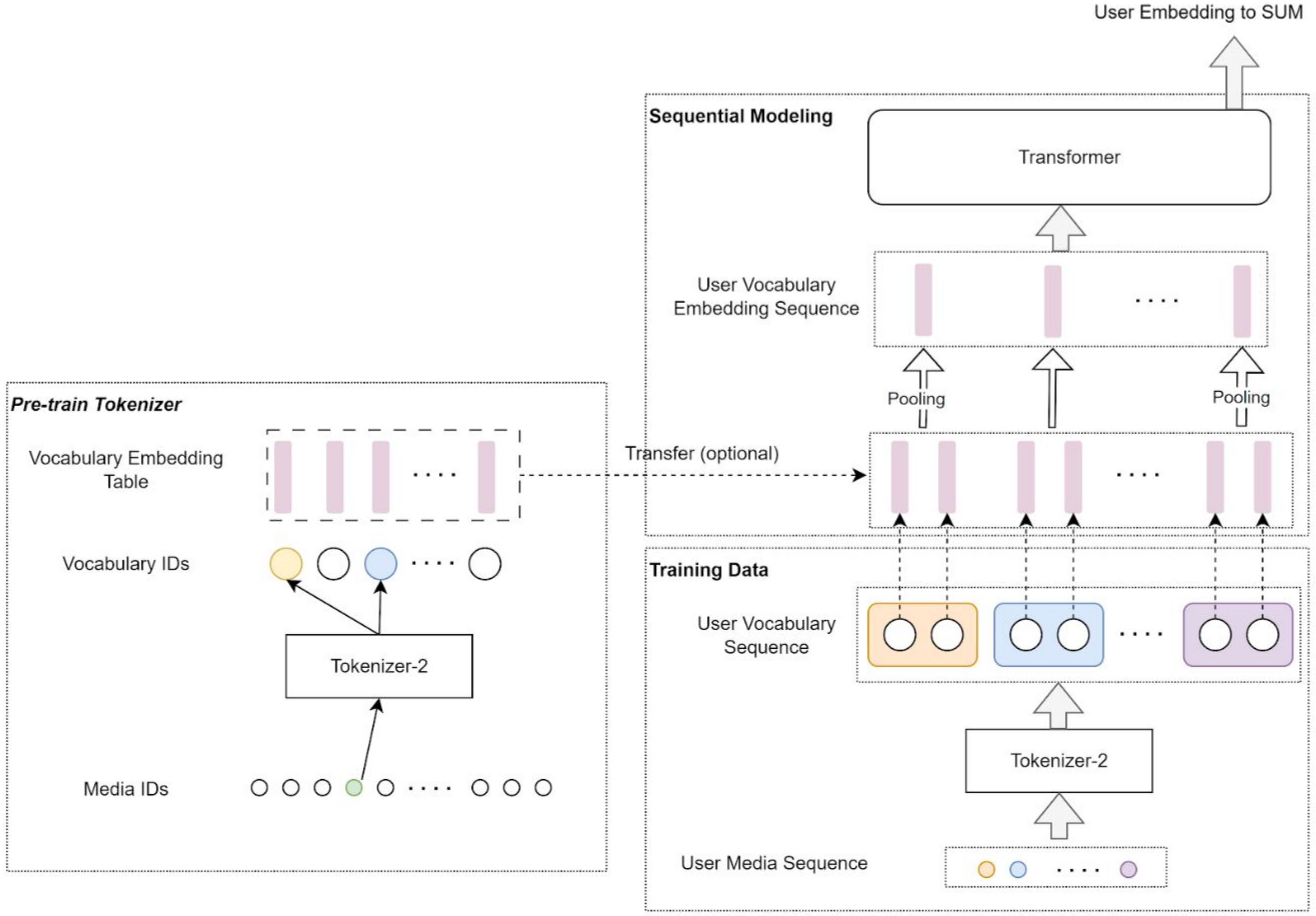
Rare-ID Filtering

- Uncertainty and variance is baked in rare-IDs.
- It needs to be evaluated against the performance on upsampled data.
- How else should we handle these rare IDs.



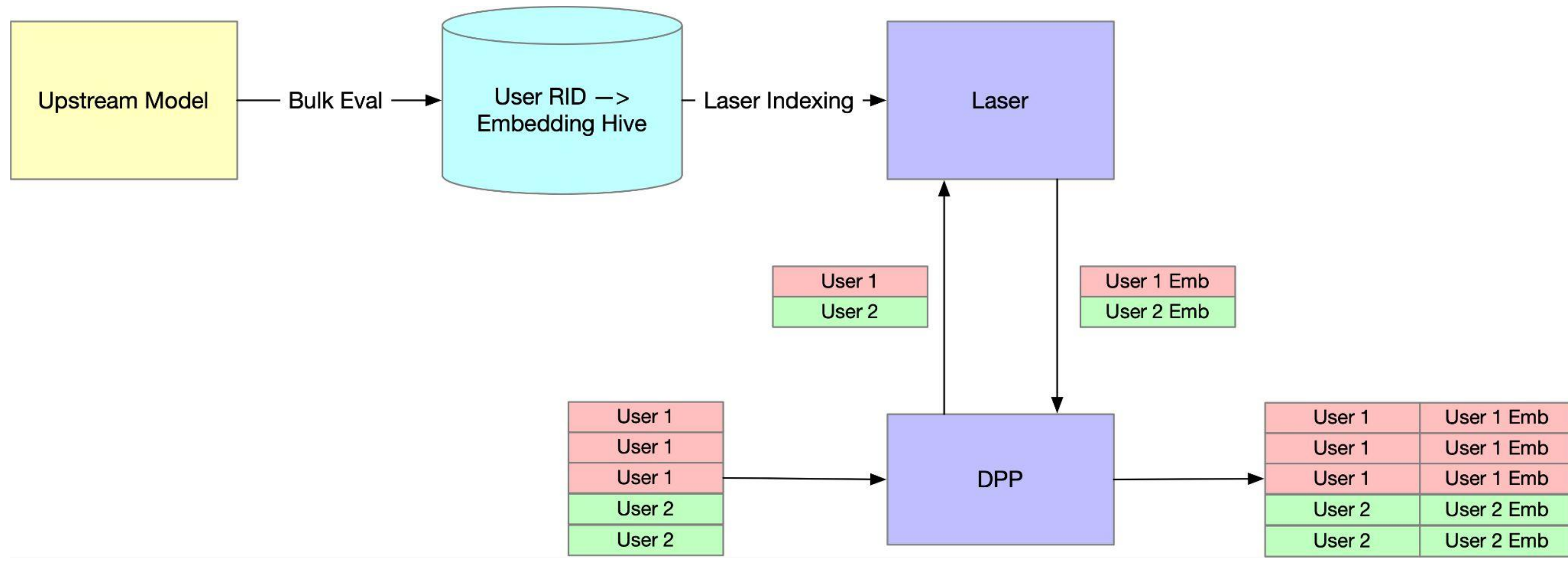
Modality-based Representation

- Challenges of ID-based representations
  - Entity ids are monotonically increasing
  - Embedding table size is bounded by GPU memories
- Proposal: modality-based representation
  - Multi modal semantic representation
  - Transfer learning the vocabulary embeddings to main model to warm up representations.



Downstream Model Experimentation

- Existing cycles takes around ~15 days to measure impact on downstream model. It consists of
  - Training a promising upstream model
  - Setting up recurring embedding export
  - Setting up exported embeddings for feature extraction
  - Get capacity approval for feature extraction
  - Begin extraction & wait for training data collection
  - Train downstream model with newly populated embedding

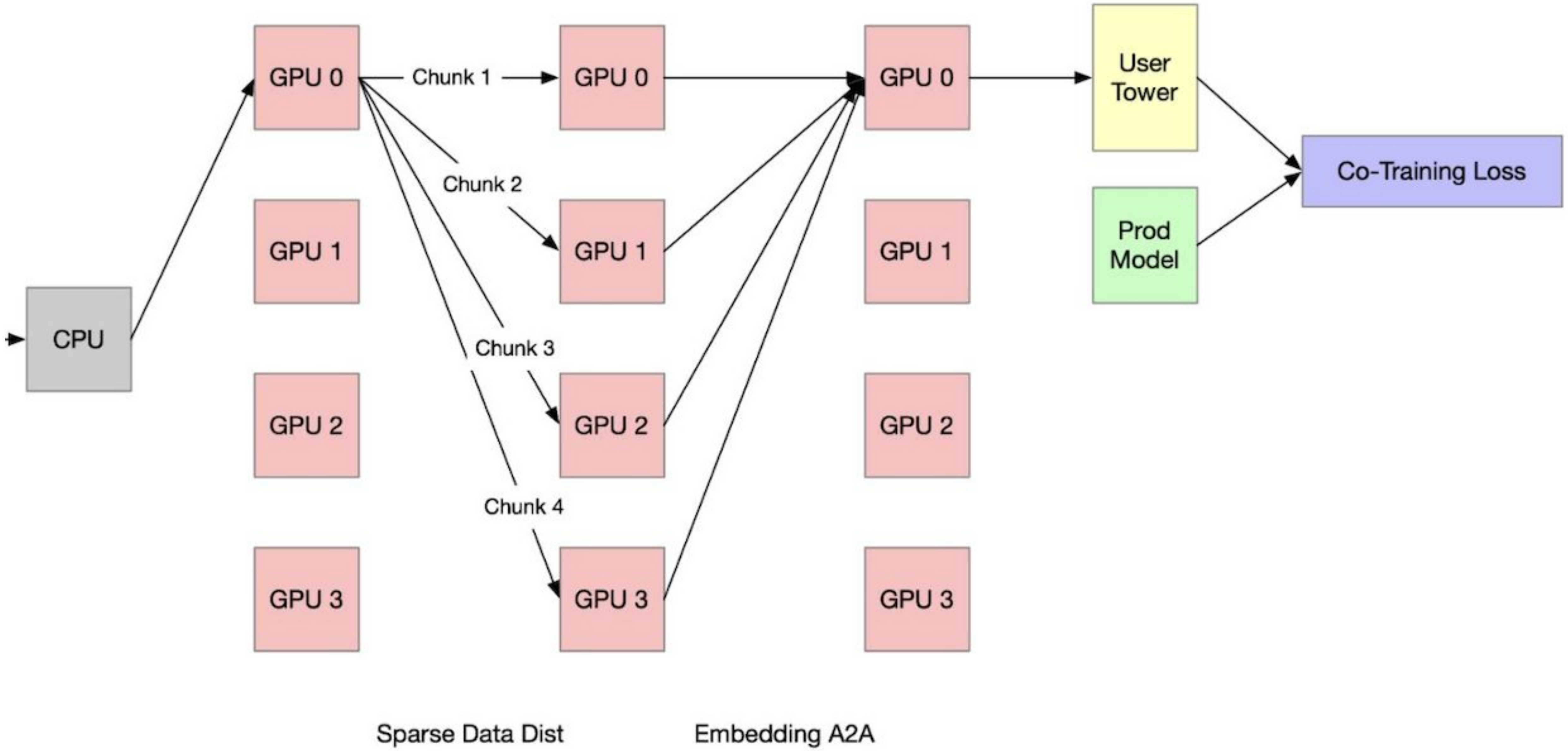


Improving Responsiveness

- Multi-faceted problem:
  - Chains of models and heuristics
  - Inputs freshness and lag variance
  - Weights freshness and trends
  - Content freshness and uncertainty
  - Delivery mechanisms
- For effective experiments you need to act on the entire stack at once.
- Different products are sensitive to different levers and techniques.
- There could be an offline and online path

03 Infra

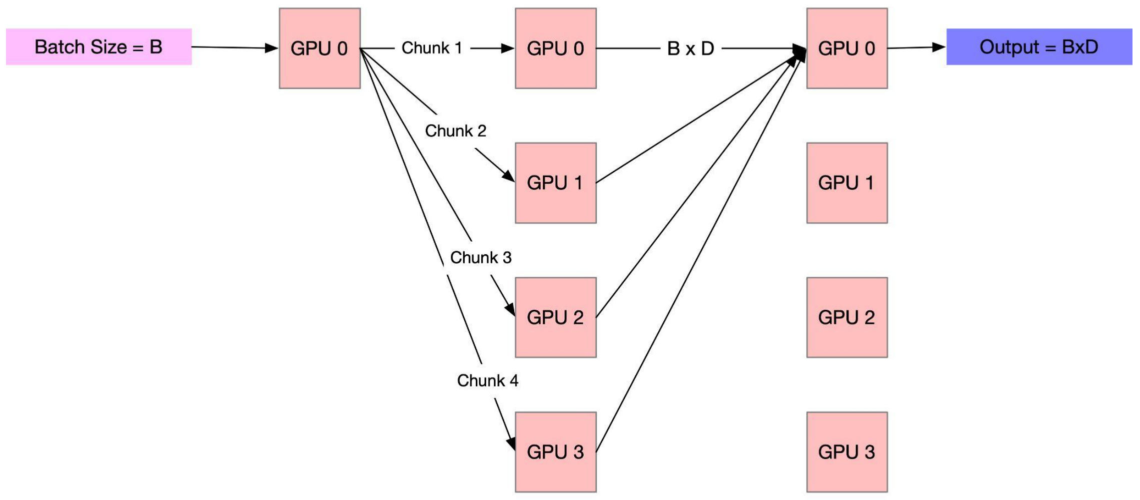
Training Overview



Pooling vs Sequence Embeddings

Pooling Embeddings

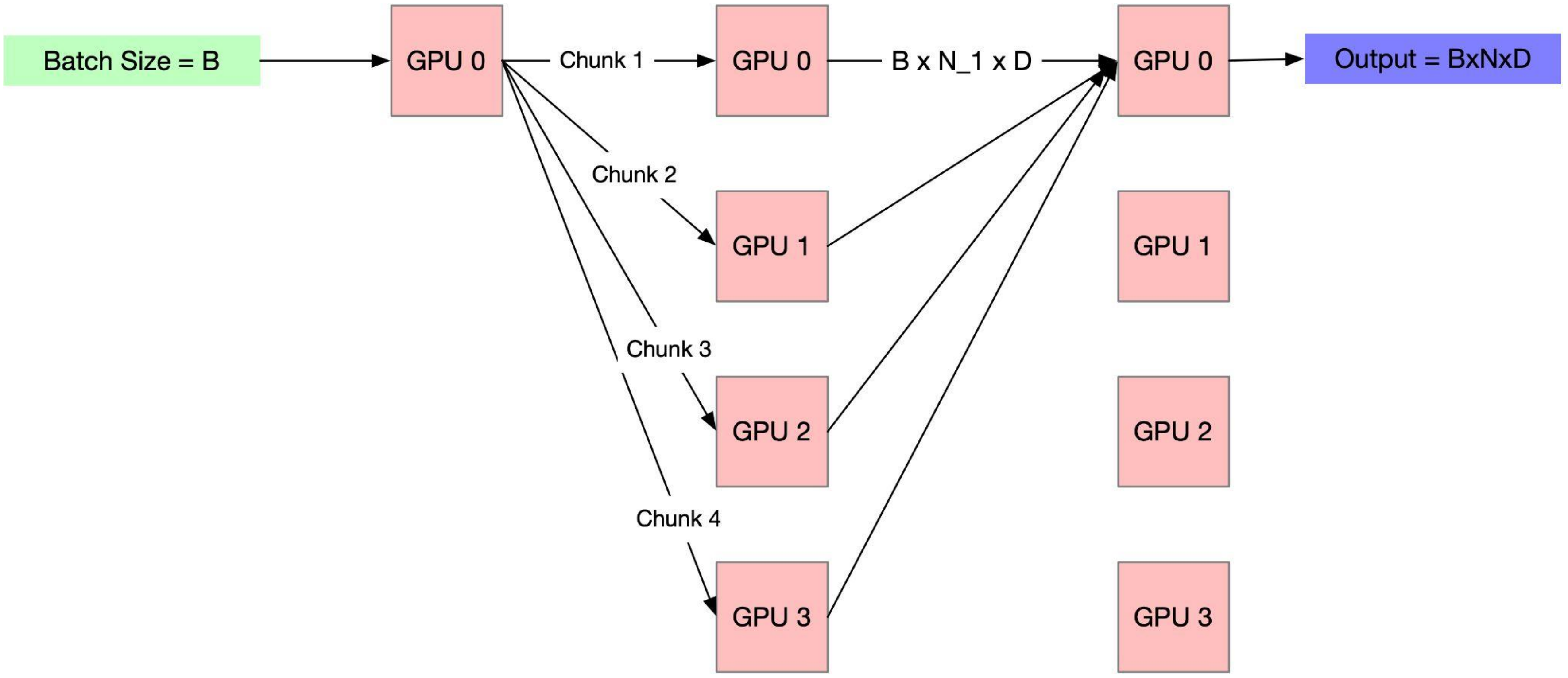
- **Pooling Embeddings** put less stress on GPU comms because each GPU is sum output embedding locally before sending to destination.
- Increasing length for pooling feature doesn't lead to explosion in compute because output of pooling features is always BxD irrespective of length.



Every GPU sums input chunk before communication

Sequence Embeddings

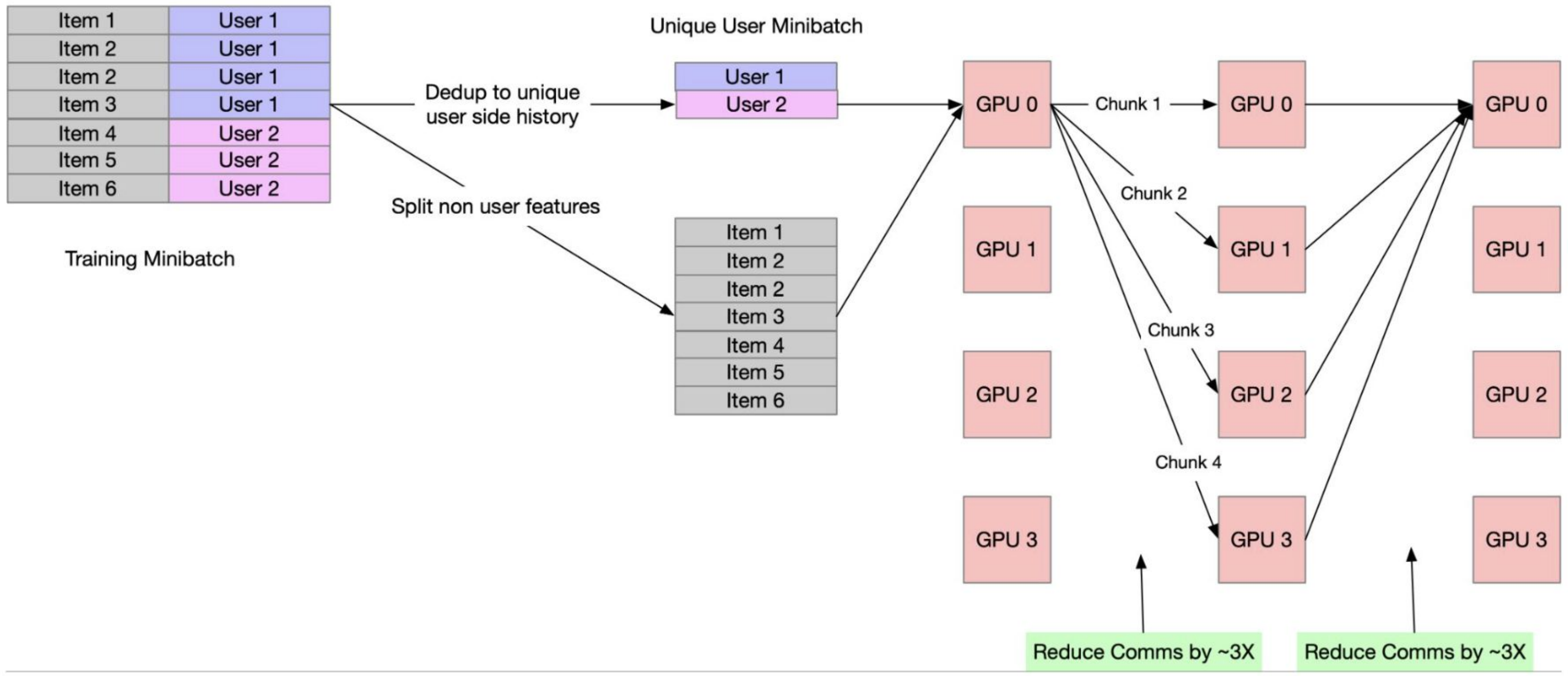
- **Sequence Embeddings** comm requirements scale w.r.t. input sequence length.



Every GPU communicates raw Embeddings

Solving Comms Bottlenecks

- Recommendation data serve & log data in batches - i.e. a user is served 5-8 recommendations at at time.
- User features for all these items remain the same due to nature of serving requests.
- We can exploit both properties to reduce comm bottleneck.
- This technique can lead to more than **~3X increase in Trainer QPS**.



Challenges

- Training reliability & resource usage continues to be a great challenge. Increased load on network, compute & bandwidth leads to death by thousand cuts.
  - Low reliability means slower iteration cycles - hence slower TTM (time to market)
  - Low reliability also means model freshness is affected in prod
- Embedding stability is paramount to obtain stable online wins
  - Constraining embedding space in model only guarantees temporary safety
  - Changing data patterns can introduce more model stability issues
- Increasing output embedding dimension gives higher gains but also leads to higher serving cost
  - Experimentation shows 2x, 3x embedding dim show significant gain but with major infrastructure cost

Meta AI