SIDE : Semantic ID Embedding for effective learning from sequences

Dinesh Ramasamy, Shakti Kumar, Chris Cadonic, Jiaxin Yang,
Sohini Roychowdhury, Esam Abdel Rhman, Srihari Reddy

Meta Platforms, Inc.

AdKDD‘25, Toronto, Canada
August 04, 2025

Meta AI

Agenda

Motivation and Overview

SIDE Encoders

Ads Ranking

Results

01 Motivation and Overview

Compression is Important

- User histories typically range in order **O(10^3)** to **O(10^4)**
- Features are large embeddings of *d* dimensions
- *d* is large ~ 400-1000
- Significant storage and inference cost
- Need to reduce this to
  - Increase dimension X: use longer sequences
  - Increase dimension Y: use more features
  - Increase dimension Z : pack more information
- improve ads recommendations + **reduce feature capacity cost**

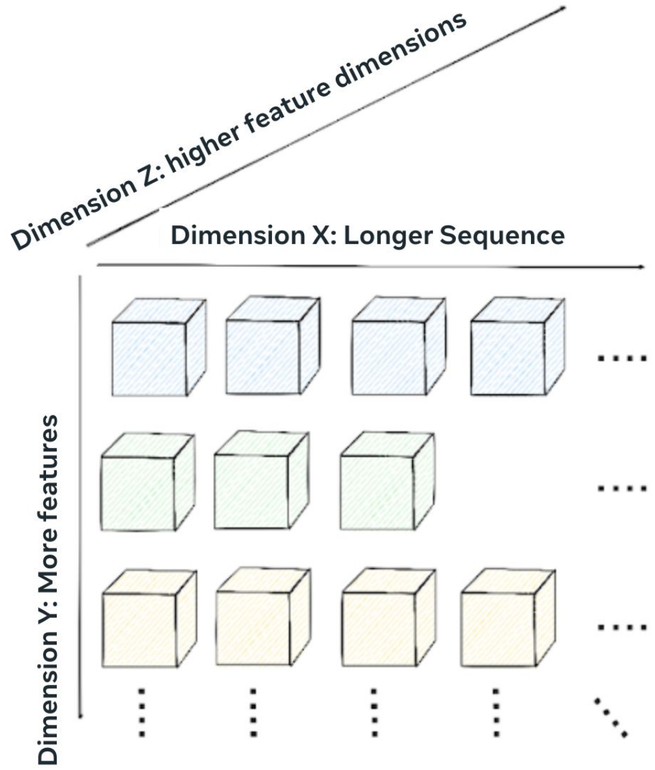


Figure 1: Data scaling issues for user history feature

Why SIDE?

SIDE: Semantic ID Embeddings

Contribution 1

- SIDE compresses these large *d* embeddings to a feature of length *L* << *d*
  - eg. *L* = 25, *d* = 400
  - generated via our novel quantization technique **Discrete PCA (DPCA)**

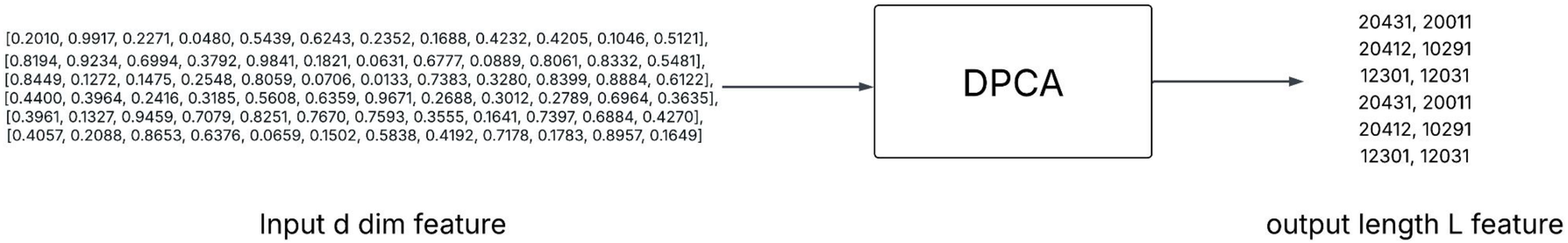


Figure 2: DPCA

Why SIDE?

SIDE: Semantic ID Embeddings

Contribution 2

- SIDE compresses multi inputs to single output
  - via our novel fusion technique **VQ Fusion**

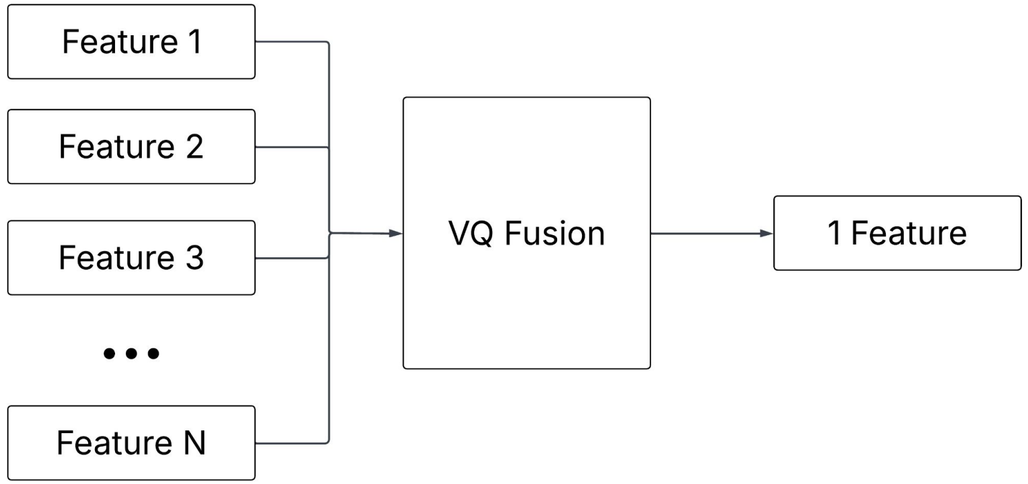


Figure 3: VQ Fusion

Why SIDE?

SIDE: Semantic ID Embeddings

Contribution 3

- **SIDE doesn't need embedding tables**
- Can express much larger embedding space in Ads Ranking
- Save storage
  - 90% model size is embedding tables

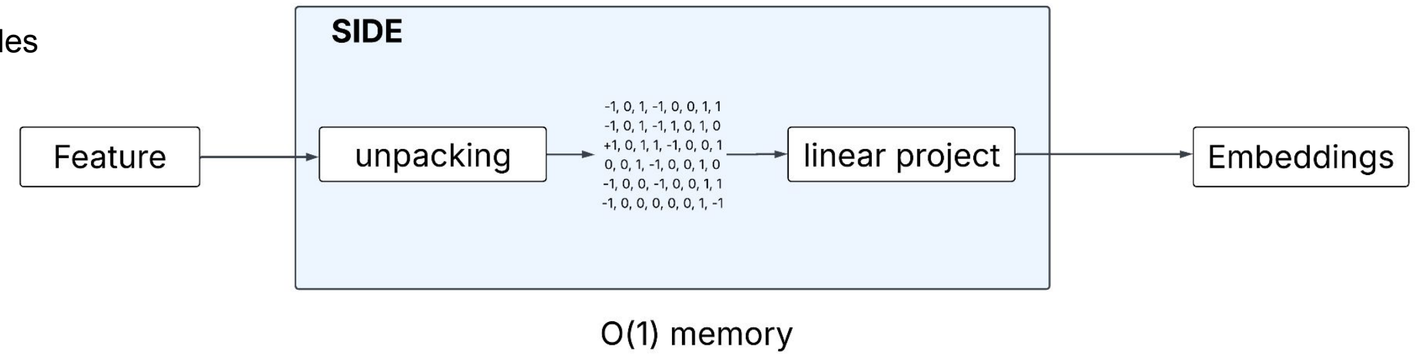


Figure 4: Embedding Free Lookup

02 SIDE Encoder

Encoders

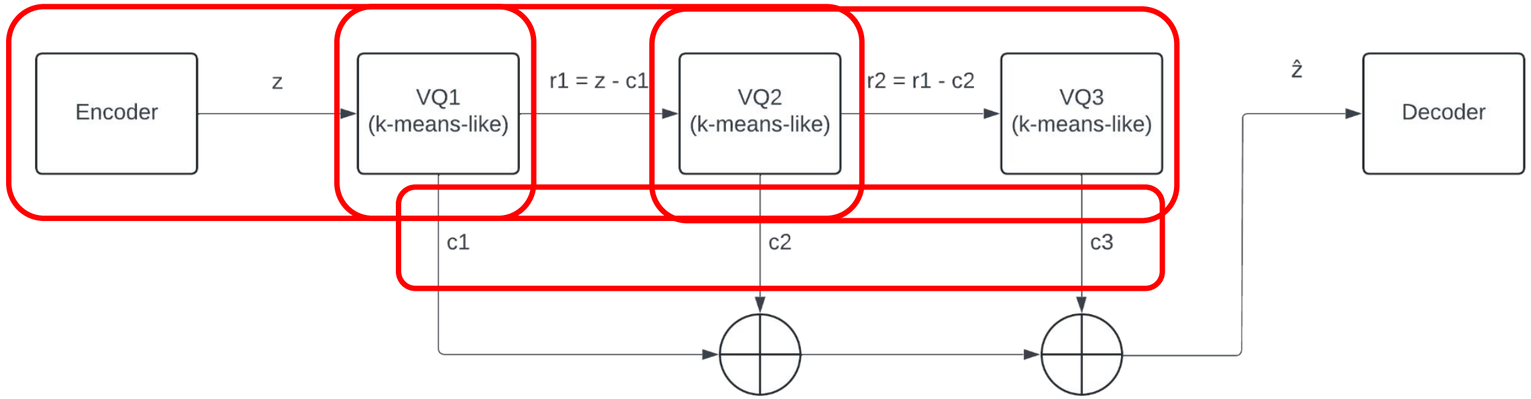


Figure 5: Residual Quantization for training SID Encoders

- Vector quantization developments have historically been feasible for compressing embeddings into a mapped hierarchical Semantic ID (SID) space that can build a high quality representation of an input signal
  - Residual Quantization (RQ)[0] here specifically enables hierarchical learning
  - Trained as an autoencoder that leverages:
    - Reconstruction loss
    - Regularization such as codebook loss and commitment loss

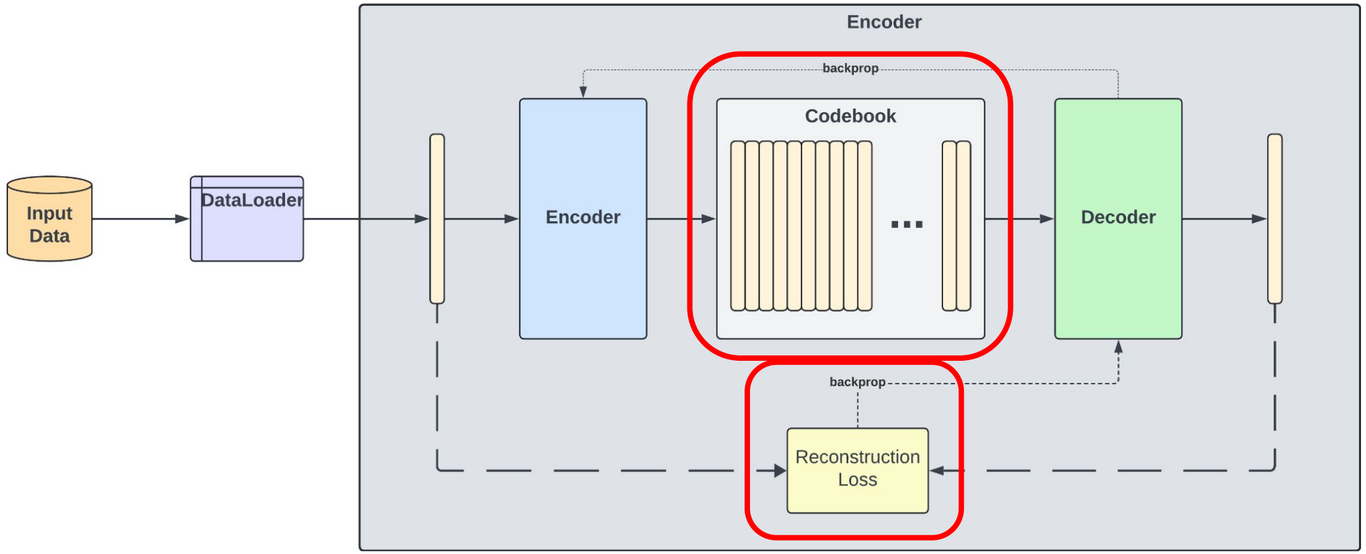


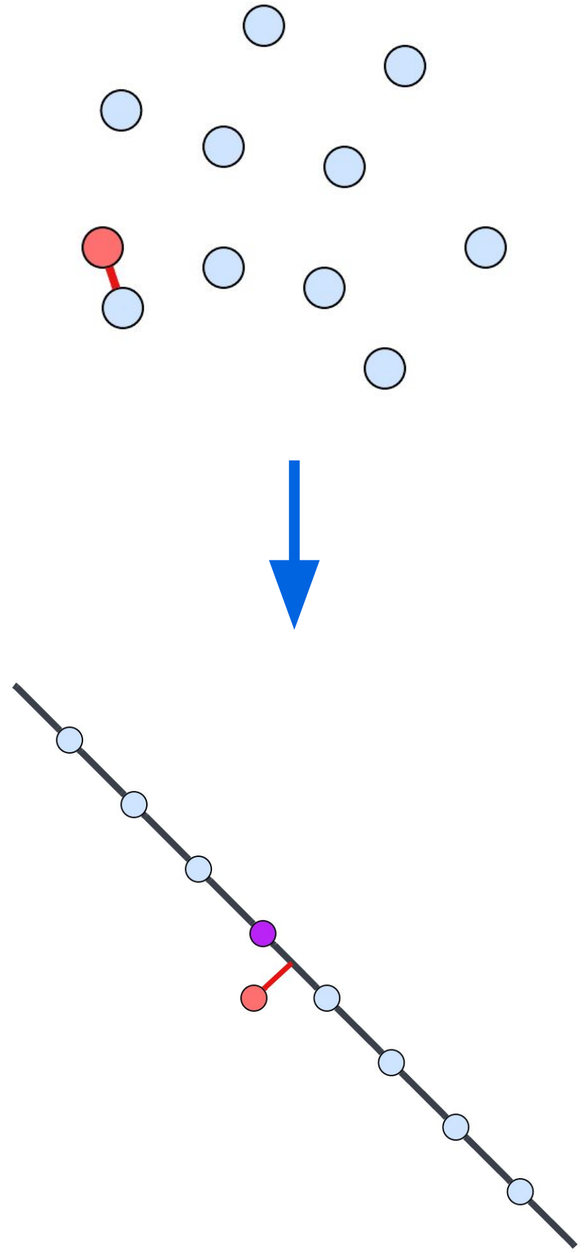
Figure 6: Learning and exporting SIDs

[0] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022. Autoregressive Image Generation using Residual Quantization.

Novel Encoder Development

- Relearning ID-to-codeword embedding mapping requires large amounts of data and parameterization/memory in v0 model
  - Thus we developed an embedding table-free **SID** to **embedding** algorithm (**SIDE**)

- Developed an extension to *k*-means VQ-VAE to instead have **structured co-linear codewords**



Novel Encoder Development

- Relearning ID-to-codeword embedding mapping requires large amounts of data and parameterization/memory in v0 model
  - Thus we developed an embedding table-free **SID** to **embedding** algorithm (**SIDE**)
- Developed an extension to *k*-means VQ-VAE to instead have **structured co-linear codewords**
- Additionally add targeted pre-encoder model heads to handle **multi-input (VQ-Fusion)**
  - Modality support (e.g., video embeddings, image embeddings)
  - Signal type support (e.g., embeddings, categorical features, KNN IDs)

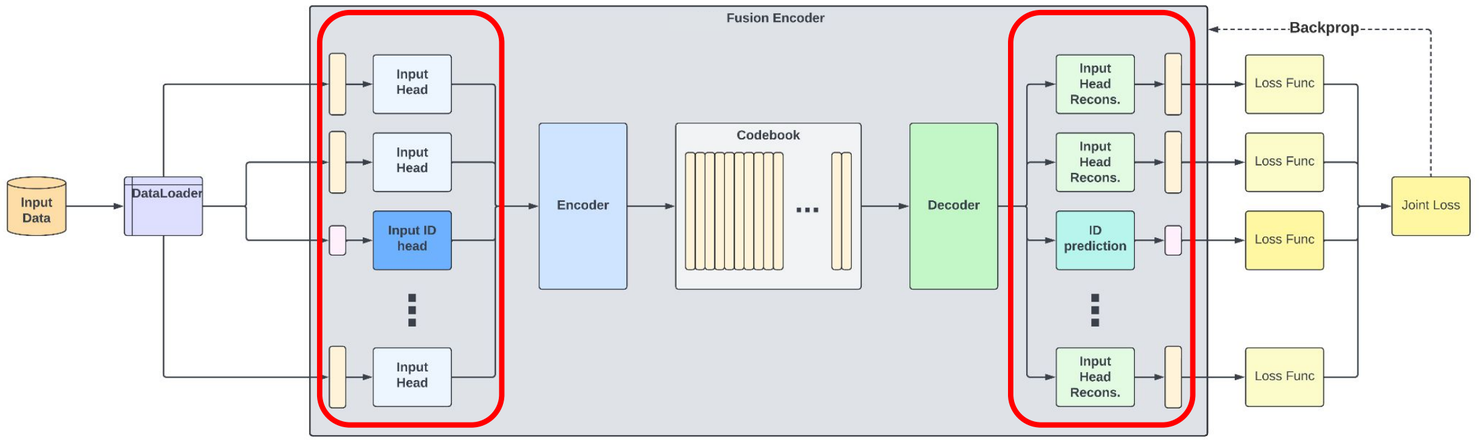
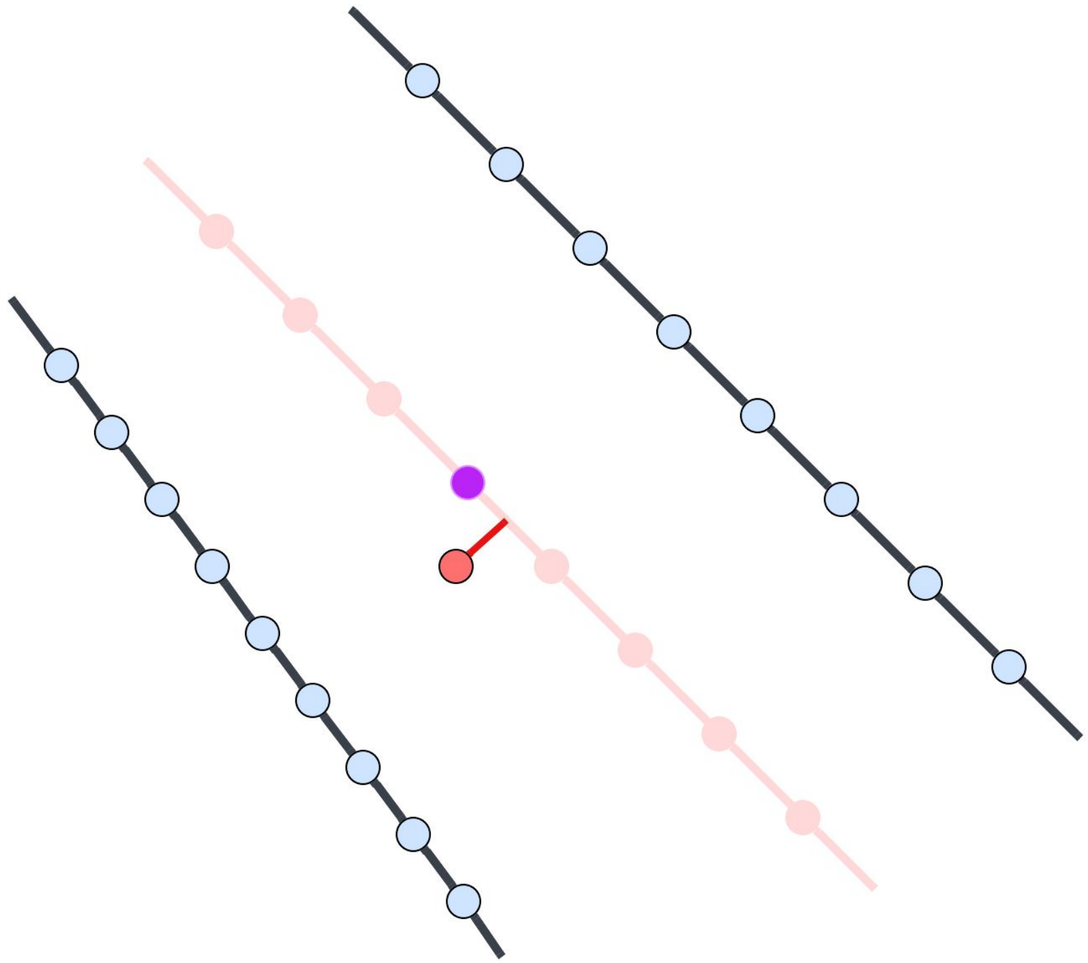


Figure 7: Learning SIDs for VQ-Fusion: combining multiple upstream inputs

Structured Quantization: FSQ vs. DPCA



- Finite Scalar Quantization (FSQ) extends lookup-free Quantization (LFQ)
  - No learned embedding mapping table to SIDs
  - FSQ extends codebook layers with more buckets per dimension
- High-level inference algorithm
  - Choose “line” closest to given point to corresponding *k*-th codeword group and closest line u_k
  - Estimate distance to closest line using inner product
  - Quantize projection weight s_k with L bins to identify codeword index l_k

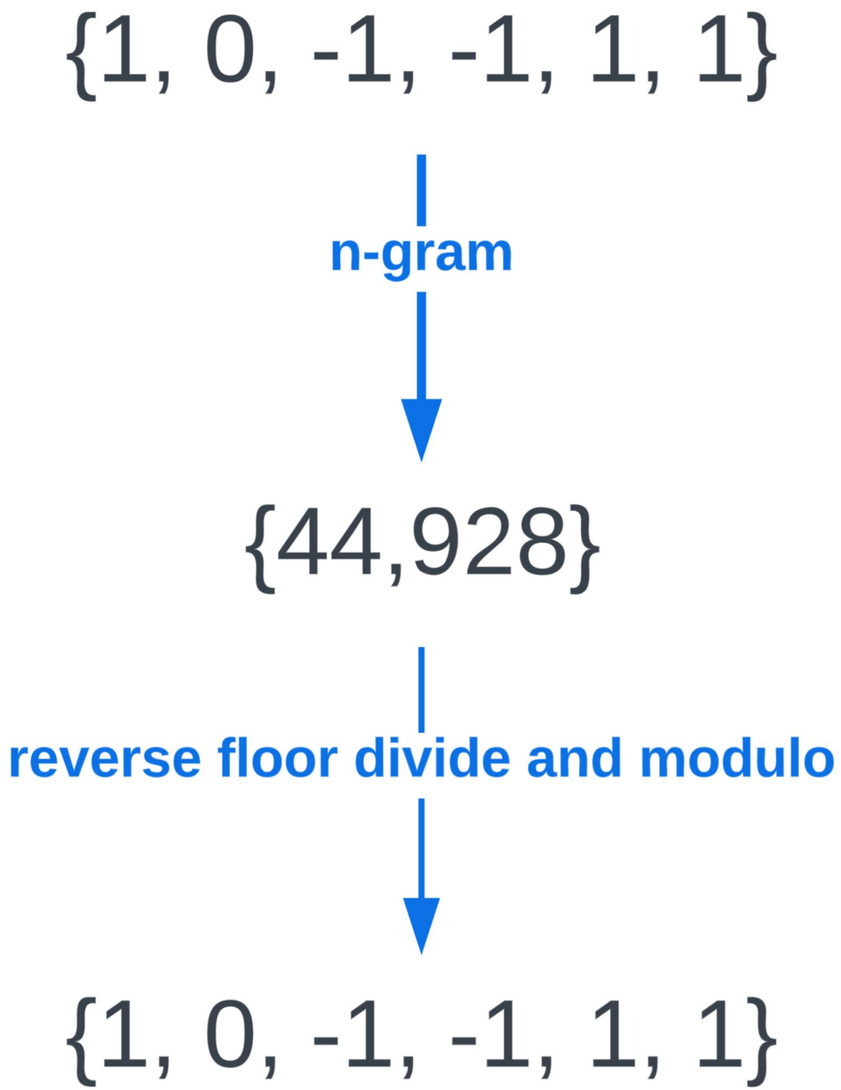
- **DPCA**
  - Allowing scalar values {-1, 0, 1} as codebook signed distance (as in 1.58-bit LLMs[1]) to quantize to bins we develop **Discrete-PCA (DPCA)**
    - No assertion on orthogonal “component vectors”

[1] Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits. arXiv:2402.17764

Structured Quantization: FSQ vs. DPCA

- Finite Scalar Quantization (FSQ) extends lookup-free Quantization (LFQ)
  - No learned embedding mapping table to SIDs
  - FSQ extends codebook layers with more buckets per dimension
- High-level inference algorithm
  - Choose “line” closest to given point to corresponding *k*-th codeword group and closest line u_k
  - Estimate distance to closest line using inner product
  - Quantize projection weight s_k with L bins to identify codeword index l_k

- **FSQ/DPCA → SIDE**
  - Convert to collection of *n*-gram SID(s) by collecting codebook values across layers
    - Can be reversibly floor/modulo converted to a representative embedding



[1] Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits. arXiv:2402.17764

SIDE

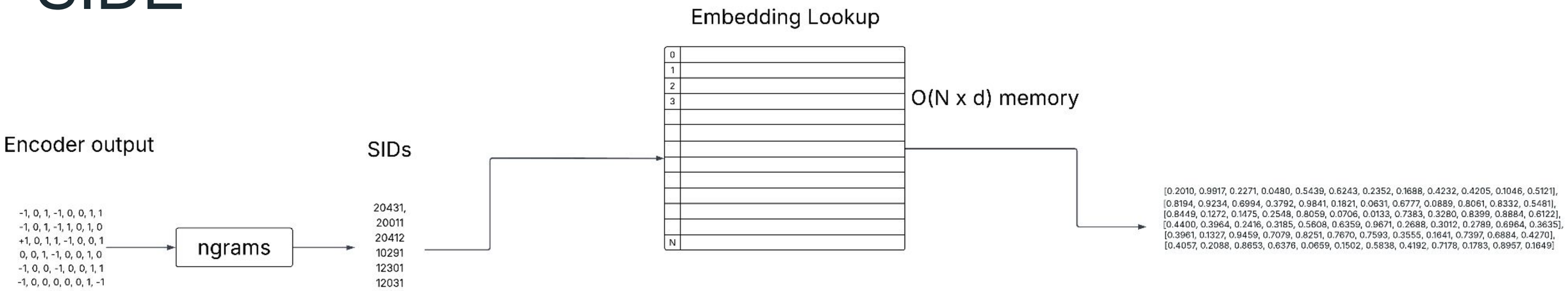


Figure 8: SIDE saves big embedding tables

SIDE

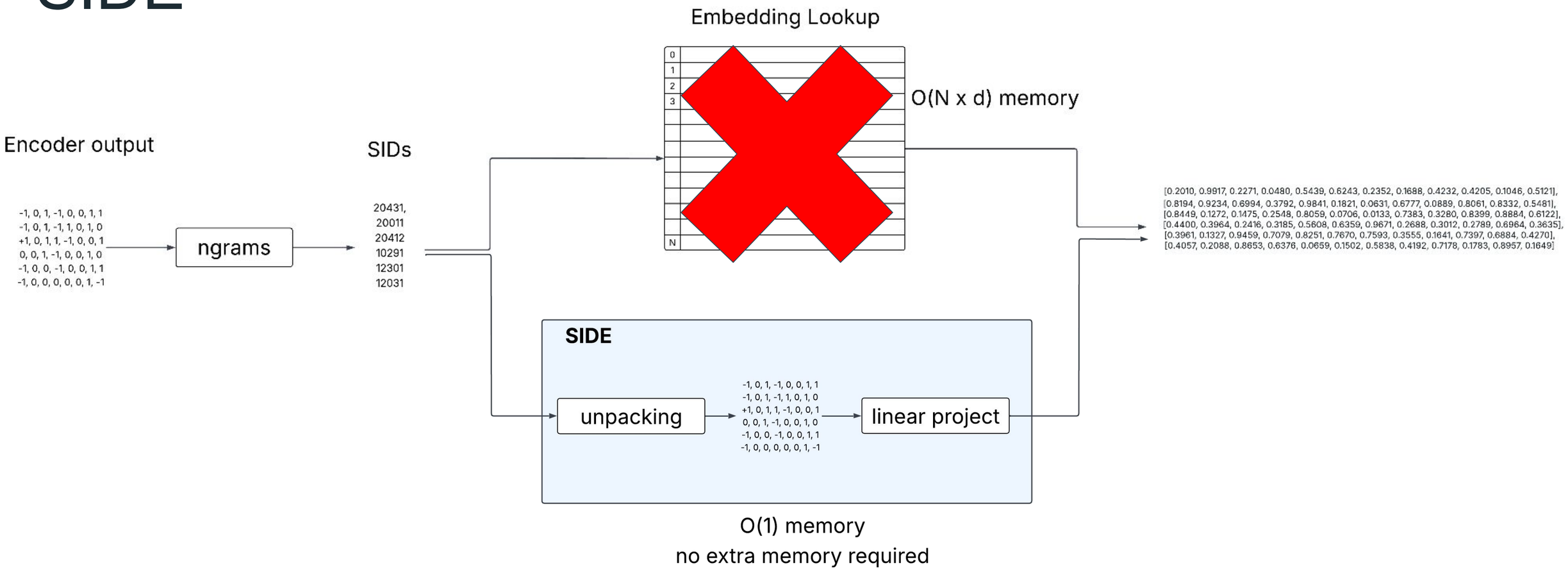


Figure 8: SIDE saves big embedding tables

03 Ads Ranking

Ads Ranking

- Can pack SIDs by various methods, ngrams OR bitpacked
- User / ad features are (batch x compressed Length)
- Unpack in Ads Ranking
- Interact with other features

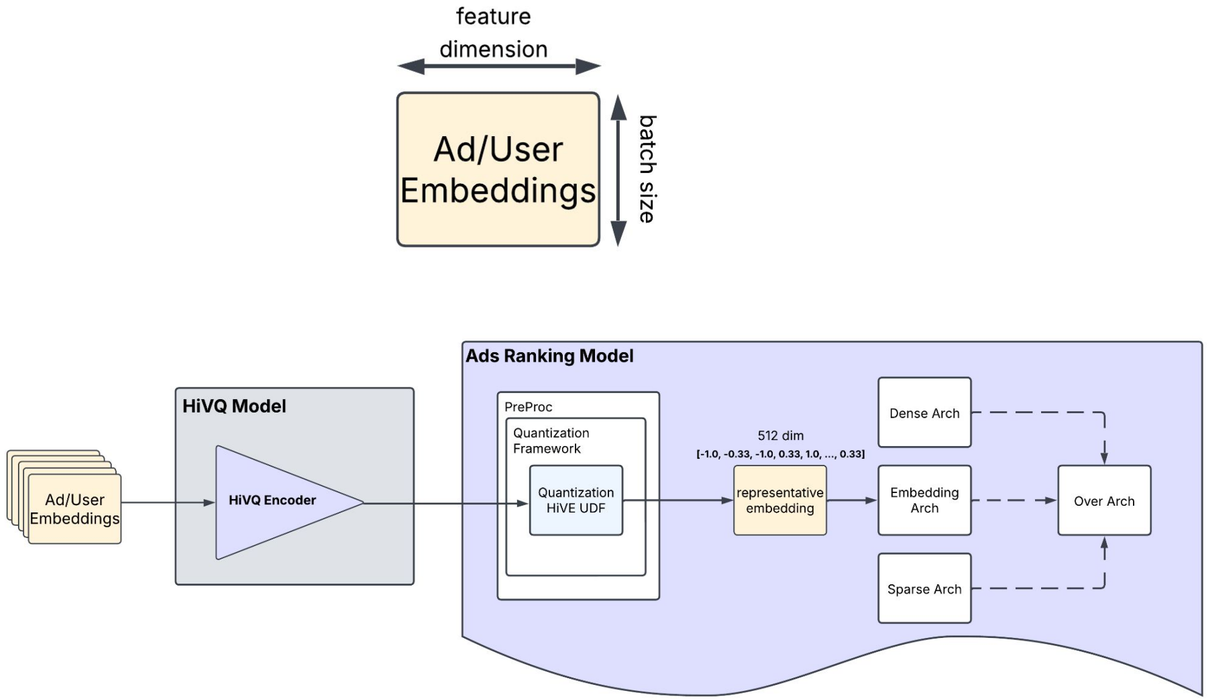
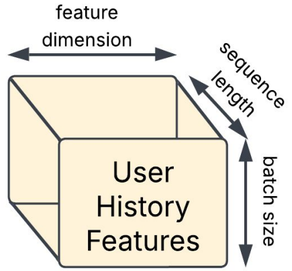


Figure 9: Ads Ranking unpacking for user/ad embeddings



Ads Ranking

- User history features are (batch x sequence x compressed Length)
- Unpack in Ads Ranking
- Summarize the sequence axis
  - produce (batch x compressed Length)
- Interact with other features

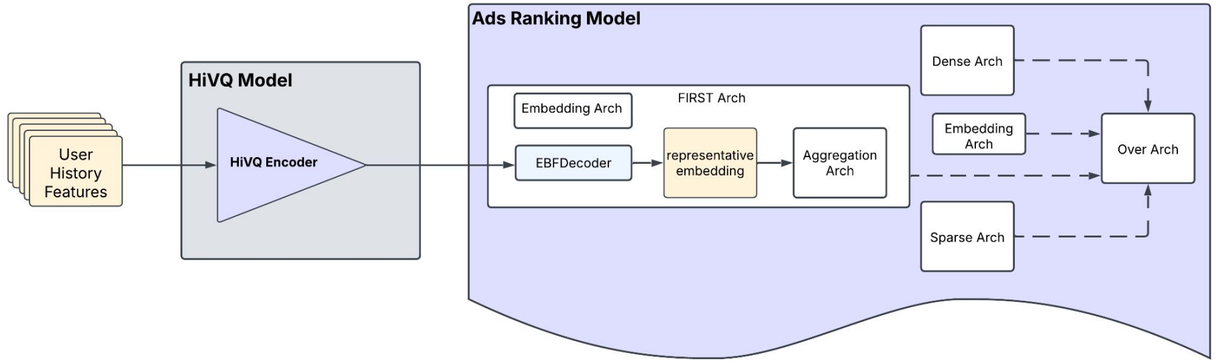


Figure 10: Ads Ranking unpacking for engagement sequence features

04 Results

Results

- For simplicity: RoI = $(NE_{\text{click}} + NE_{\text{conv}})/2C$
- C = cost of features ~ feature length

- Table 2
  - k-means features with length 50 have cost $46.54kW$
  - SID features have length 3 and cost $10.84kW$
  - SID improves RoI by 5.57 times over traditional k-means

- Table 3
  - Cost for SIDE is 1/3 x SID – just one ID from SID to unpack
  - RoI boost 5.33x for the DPCA encoded feature as EBF

- Table 4
  - For VQ-fusion RoI improves by 7.40x over SID

Method	k-means	SID
Incremental Cost	×1	×0.23
Click NE gain	**0.0108%**	0.0085%
Conversion NE gain	0.0037%	**0.0101%**
RoI	×1	×5.57

Table 2: 3: Incremental data and feature cost and normalized entropy gain relative to the baseline for a DPCA encoded ad feature. Baseline is a large-scale production ads-ranking model without the specific ad-features.

Method	SID	SIDE
Incremental cost	×1	×0.33
Click NE gain	0.0082%	**0.0185%**
Conversion NE gain	0.0086%	**0.0111%**
RoI	×1	×5.33

Table 3: Normalized entropy gain relative to the baseline for a DPCA encoded feature used as user history. Baseline is the large-scale production ads-ranking model.

Method	SID	SIDE
Incremental cost	×1	×0.33
Click NE gain	0.0100%	**0.0284%**
Conversion NE gain	0.0067%	**0.0124%**
RoI	×1	×7.40

Table 4: Normalized entropy gain for VQ fusion encoded feature comprising 6 individual signals, used as user history. Baseline = 6 different 1:1 encoded DPCA features used as SID.

Results

Prefix n-gram length	SID	SIDE
3	0.029%	0.044%
4	0.020%	0.046%

Table 5: Effect of prefix n-gram length on NE, keeping all other Encoder hyper-parameters fixed.

- Varying SID construction hyperparameters and its impact in NE gains shown in Table 5, experiment setup as in Table 4.
- We observe that for SID, as more n-gram IDs are introduced (by making the prefix n-gram longer from 3 to 4) in the ads-ranking model, the NE gains reduce due to higher noise due to increase in hash collisions based on the embedding table size limitations. However, our proposed SIDE technique avoids hash collisions while unpacking the 4th n-gram ID leading to NE increase.
- We also assess the effect of increasing the number of scalar quantizer codewords on the ads-ranking NE gains. Using 4 scalar quantizer codewords NE gain for SID and SIDE are 0.018% and 0.035% respectively. Extending this analysis to 64 codewords results in NE gains for SID and SIDE as 0.015% and 0.044%

Thank you