

From a Static Multi-Level Small Semantic Codebook to a Dynamic Single-Level Large Semantic Codebook for Generative Recommendation

Tianlu Xie¹, Xin Ku¹, Mingjie Sun¹, Yunhao Sha¹, Lixiang Wang¹
Peng Wang¹, Yiyu Wang¹, Wenjin Wu¹, Zhaojie Liu¹, Peng Jiang¹, Wenwu Ou¹

¹Kuaishou Technology, Beijing, China

{xietianlu, kuxin, sunmingjie, shayunhao,
wanglixiang03, wangpeng16, liubin21, wuwenjin,
zhaotianxing, jiangpeng, luocheng10}@kuaishou.com

Abstract

Generative recommendation represents each item with a sequence of discrete Semantic IDs (SIDs) and predicts that sequence to retrieve the next item. Typical systems construct a multi-level semantic representation through residual quantization. In our industrial setting, the common three-level realization uses two semantic codes followed by a collaborative disambiguation code. The additional semantic level lengthens autoregressive decoding while creating a large hierarchical space that is sparsely occupied under each first-level code. Online systems also face **codebook drift**. As new items continually arrive and exposure distributions change, the partitions of a static codebook become increasingly misaligned with current traffic. We propose a **single-level large semantic codebook** that replaces the multi-level residual semantic representation with one semantic token and retains a separate collaborative disambiguation token to reduce item collisions. We further introduce an **exposure-aware dynamic codebook update mechanism** that combines temporal weight decay, exponential moving-average center updates, and an exposure-weighted penalty on SID changes. Finally, we develop an **offline evaluation framework** covering representation quality, semantic-code utilization, cluster load, full-SID collision, and temporal stability. Across the two public datasets, the two-level SID improves mean Recall@10 by **5.0%–8.8%** and mean NDCG@10 by **4.1%–5.1%** for OneRec-V1. The corresponding gains for OneRec-V2 are **7.1%–8.7%** and **3.8%–8.5%**. Under the fixed-date KuaiRec evaluation, dynamic updating further improves Recall@10 by **1.4%** and **2.7%**, and NDCG@10 by **7.0%** and **2.7%**, for OneRec-V1 and OneRec-V2, respectively. Across the decoder, LazyAR, and MTP serving architectures, the shorter SID reduces estimated autoregressive-decoding FLOPs by **47.93%–48.70%** and increases single-card QPS by **28.57%–47.0%**. In a five-day online A/B test serving 2.5% of production traffic, the two-level SID improves the primary consumption metric by **0.792%**.

Keywords: generative recommendation, generative retrieval, semantic IDs, item tokenization

1 Introduction

Many large-scale recommender systems encode user queries and item candidates as continuous vectors and retrieve candidates through approximate nearest-neighbor search [13]. Generative recommendation offers a different formulation: it represents each item as a sequence of discrete

Semantic IDs (SIDs) and generates the SID sequence of the next item. TIGER established this paradigm by quantizing content embeddings into hierarchical semantic codewords and predicting them with an encoder-decoder Transformer [13]. Subsequent studies have made item tokenization a central research problem, since the identifiers must preserve semantic structure, incorporate collaborative signals, and maintain balanced code usage [2, 4, 20, 26, 28].

Most SID-based generative recommenders construct hierarchical, multi-token item representations with **multi-level residual semantic quantization** [7, 13, 20, 28]. The first semantic code approximates the item embedding, and each subsequent semantic code quantizes the remaining residual to refine the representation. This design keeps the vocabulary at each decoding step manageable. However, an autoregressive recommender predicts the codes one step at a time and typically maintains multiple candidate paths with beam search. Every additional level therefore introduces another sequential prediction step and enlarges the search process [7].

In the industrial system considered in this work, a typical deployed **three-level SID** exposes a limitation of this standard design. SID1 and SID2 perform semantic residual quantization, whereas the collaborative SID3 only disambiguates items that remain indistinguishable under the semantic codes and thereby reduces full-SID collisions. An analysis of approximately 1.50 billion industrial samples reveals a marked gap between the global coverage of semantic SID2 and its conditional use under SID1. Although 93.31% of the SID2 vocabulary appears globally, each active SID1 is associated with only 2.48% of that vocabulary on average. **SID2 therefore creates a large but conditionally sparse hierarchical space while still requiring an additional decoding step.** Figure 1 presents both the full distribution and its summary statistics.

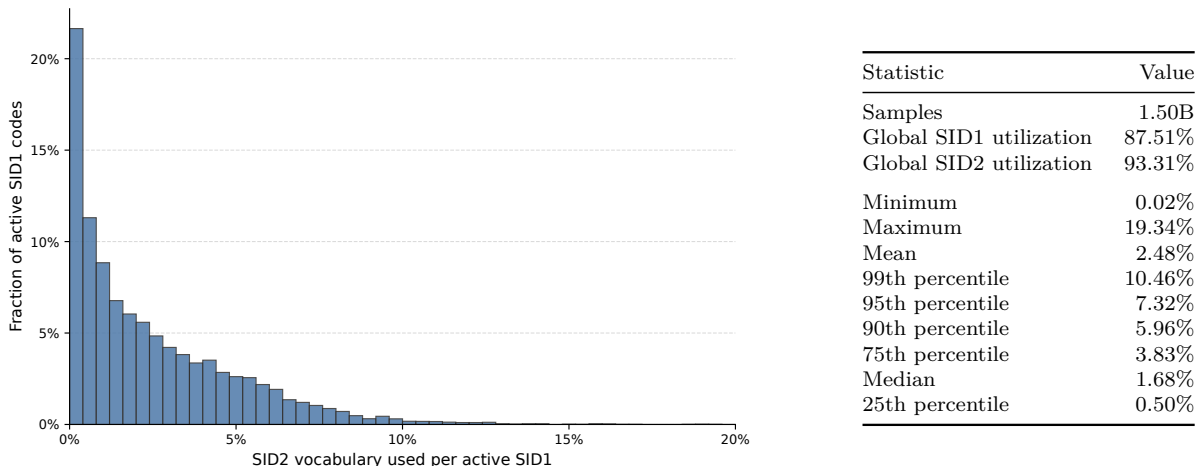


Figure 1: Global and conditional utilization of the two semantic levels in the deployed three-level SID. The histogram shows, for each active SID1, the fraction of the SID2 vocabulary observed under that code. The table reports global utilization and summarizes the conditional utilization distribution.

Parallel SID generation has been explored as one way to reduce sequential dependencies, but independently predicted tokens are not guaranteed to compose a valid catalog identifier. RPG therefore uses graph-guided decoding to exclude invalid SIDs [7], while LLaDA-Rec introduces adapted beam search for its diffusion decoder [14]. Thus, parallel prediction still requires validity recovery or constrained search. We pursue a different direction: **reduce the number of semantic codes** that must be generated and avoid invalid cross-level combinations at the source.

Motivated by the large-vocabulary prediction capability of modern language models, we replace

the multi-level residual semantic representation with a **single-level large semantic codebook**. A separate collaborative disambiguation code is retained only to reduce collisions. It is not part of semantic reconstruction. The resulting SID contains one semantic code and one disambiguation code, shortening the typical industrial sequence from three codes to two and reducing the number of autoregressive decoding steps.

Structural simplification alone does not address **codebook drift in a non-stationary online item corpus**. New items arrive continuously, item embeddings evolve, and exposure distributions shift with user interests and traffic allocation. A semantic codebook fitted to a fixed historical window can therefore become stale. Its centers and partitions no longer match the items that dominate current traffic. Recent studies have questioned static SID assignment, developed dynamic SIDs, or introduced adaptive item indexing for streaming recommendation [4, 15, 23]. We therefore introduce an **exposure-aware dynamic update mechanism** that maintains temporally decayed exposure weights, updates active centers with an exponential moving average, and penalizes exposure-weighted SID changes to improve the stability of high-exposure items.

In industrial systems, directly validating a new codebook through the complete recommendation pipeline is expensive. Each candidate codebook requires the item corpus to be retokenized and the downstream generative recommender to be retrained and evaluated. This long feedback loop makes online trial-and-error unsuitable for rapid codebook development. We therefore build a **codebook-level offline evaluation framework** that measures representation quality, semantic-code utilization, cluster load, full-SID collision, and temporal stability without requiring a full recommender training run for every candidate codebook. When a dataset provides meaningful content-type annotations, we additionally analyze the composition of those types within clusters. These diagnostics enable efficient codebook screening and comparison. Final recommendation quality is still evaluated by training generative recommenders on public datasets and industrial data.

Across the two public datasets, the two-level SID improves mean Recall@10 by **5.0%–8.8%** and mean NDCG@10 by **4.1%–5.1%** for OneRec-V1. The corresponding gains for OneRec-V2 are **7.1%–8.7%** and **3.8%–8.5%**. Under the fixed-date KuaiRec evaluation, dynamic updating further improves Recall@10 by **1.4%** and **2.7%**, and NDCG@10 by **7.0%** and **2.7%**, for OneRec-V1 and OneRec-V2, respectively. In industrial codebook evaluation, dynamic PV-S2 improves reconstruction similarity by **1.50%** relative to static PV-S2. Across the decoder, LazyAR, and MTP serving architectures, estimated autoregressive-decoding FLOPs decrease by **47.93%–48.70%**, while single-card QPS increases by **28.57%–47.0%**. Finally, in a five-day online A/B test serving 2.5% of production traffic, the two-level SID improves the primary consumption metric by **0.792%**.

The main contributions of this work are as follows:

- **Single-level large semantic codebook.** We consolidate multiple semantic quantization levels into one large semantic codebook and append a separate collaborative disambiguation code, shortening the complete SID and reducing autoregressive decoding steps.
- **Exposure-aware dynamic codebook.** We construct the dynamic codebook using temporally decayed exposure weights, exponential moving-average center updates, and an exposure-weighted SID-switching penalty, thereby balancing adaptation to changing traffic with assignment stability.
- **Offline codebook evaluation framework.** We develop multidimensional diagnostics that enable efficient codebook screening before the costly process of retokenizing items and retraining the downstream generative recommender.

2 Related Work

2.1 Semantic ID Tokenization and Codebook Design

Semantic ID construction. Generative recommendation formulates next-item prediction as sequence generation over item identifiers, making identifier design central to both the model’s output space and the structure shared across items. Atomic identifiers preserve item distinctiveness but expose no relations among items, whereas textual identifiers carry semantics but may not map cleanly and uniquely to catalog items [2, 6, 18]. Codebook-based Semantic IDs provide a compact, compositional alternative [20]. VQ-VAE provides the underlying discretization mechanism by assigning encoder outputs to entries in a learned codebook [19]. Residual quantization successively encodes the remaining approximation error, producing a multi-level code sequence without requiring one extremely large codebook [8]. Adding residual levels can refine the representation, but it also lengthens the token sequence that the downstream generative model must predict. TIGER adopts this construction to quantize content embeddings into hierarchical SIDs for generative retrieval [13]. Related work in generative retrieval learns discrete document identifiers end to end or organizes item identifiers in a balanced semantic tree [16, 17].

Recommendation-aligned tokenization. In recommendation, SID quality depends on more than reconstruction fidelity. LC-Rec learns vector-quantized item indices with uniform semantic mapping and uses alignment tasks to integrate language and collaborative semantics into an LLM-based recommender [27]. LETTER regularizes an RQ-VAE tokenizer with objectives for hierarchical semantics, collaborative alignment, and code-assignment diversity [20]. ETEGRec further couples the tokenizer and recommender through recommendation-oriented alignment objectives and alternating end-to-end optimization [9]. Other studies refine initial item tokens using feedback from the generative model, augment pretraining with multiple RQ-VAE identifiers, or replace reconstruction with contrastive multimodal alignment [2, 26, 28]. DIGER makes SID learning differentiable so that recommendation gradients can update the tokenizer, while using uncertainty-controlled exploration to mitigate codebook collapse [4]. These methods establish item tokenization as a recommendation-aware learning problem rather than a purely semantic compression step.

Codebook capacity. Work on discrete generative modeling has also explored how to increase token capacity without relying exclusively on deeper code stacks. Finite scalar quantization constructs a large implicit codebook as a Cartesian product of scalar levels and avoids the learned-codebook collapse observed in conventional VQ [11]. Lookup-free quantization similarly supports large discrete vocabularies without an embedding lookup and has been used to build expressive tokenizers for autoregressive generation [25]. Although these techniques use different quantizers and were developed outside recommendation, they show that discrete capacity can be increased along the vocabulary dimension rather than only through deeper code stacks. This width–depth trade-off is central to the SID architecture studied in this work.

2.2 SID Generation and Constrained Decoding

Autoregressive SID generation. Once items have been tokenized, a generative recommender predicts the code sequence of the target item from the user’s interaction history. Hierarchical SIDs are commonly generated from left to right, with each prediction conditioned on the preceding codes [13, 16, 28]. Because only a small subset of all possible code sequences corresponds to catalog items, inference typically uses beam search together with catalog-derived constraints to retain valid candidates. This procedure couples the cost of inference to SID length: additional codes require additional model predictions and expand the sequence of search decisions [7].

Parallel and unordered generation. RPG removes the fixed ordering among SID tokens, predicts a long SID in parallel, and uses graph-guided decoding to exclude invalid identifiers [7]. LLaDA-Rec replaces left-to-right decoding with discrete diffusion and introduces an adapted beam-search procedure for parallel SID generation [14]. REG4Rec constructs multiple unordered semantic tokens with a mixture-of-experts-based parallel quantization codebook and uses them to support diverse reasoning paths, followed by consistency-oriented pruning [22]. These methods weaken token-level sequential dependence, but they still require additional mechanisms to ensure valid and reliable item predictions.

2.3 Dynamic Semantic IDs and Streaming Item Indexing

Streaming clustering. Conventional item tokenization assumes a fixed corpus and a stationary embedding distribution. Streaming settings violate both assumptions: new items arrive over time, while the relative prevalence and centroids of existing clusters may shift. Online clustering addresses continual arrival by assigning each observation and updating cluster centers incrementally [1]. Forgetting factors further reduce the influence of older observations, allowing cluster statistics and quality estimates to reflect evolving structure [12]. For item indexing, however, adaptation must be balanced against assignment stability, because changing an item’s SID also changes the discrete target consumed by downstream models.

Adaptive recommendation indexing. Recent recommender systems have begun to model this temporal dimension explicitly. SSRLive argues that static SIDs cannot represent rapidly changing live-room content and generates both static and dynamic SIDs within a unified generative–discriminative architecture [15]. MERGE targets skewed and non-stationary item distributions in large-scale streaming recommendation by constructing clusters adaptively, monitoring cluster occupancy, and forming hierarchical indices through fine-to-coarse merging [23].

2.4 Evaluation of Semantic ID Codebooks

Codebook quality metrics. Evaluation of vector quantizers commonly combines reconstruction fidelity with statistics of discrete-code usage. VQ-VAE evaluates the quality of the reconstructed input and the learned discrete latent representation, while subsequent work treats codebook collapse and inactive codes as separate diagnostic concerns [11, 19]. In generative recommendation, LETTER analyzes code-assignment and code-embedding distributions, relating imbalanced assignments to generation bias [20]. DRQ provides a more explicit diagnostic framework based on expected codeword overlap and effective codebook capacity. Its industrial case study jointly examines latent geometry, robustness under perturbation, codebook utilization, reconstruction fidelity, and behavior-aware retrieval, illustrating that these dimensions need not improve together [21].

Task-level and temporal evaluation. Existing studies typically construct a codebook and assign SIDs before training a generative recommender, and then report downstream ranking metrics such as Recall and NDCG [2, 9, 13, 20, 28]. This task-level protocol directly measures recommendation performance, but it offers limited evidence about which property of the SID construction method accounts for a performance difference. A separate line of streaming-clustering research develops online cluster-validity indices that update quality statistics as data arrive, enabling the monitoring of evolving cluster structures [12]. Existing evaluation practices therefore span codebook quality metrics, downstream recommendation accuracy, and temporal cluster validity, but an integrated evaluation protocol tailored to both the static quality and temporal behavior of recommendation SIDs remains underexplored.

3 Method

3.1 Problem Formulation and Overview

Let $\mathcal{I}_t$ denote the item corpus available on day t . Each item $i \in \mathcal{I}_t$ is represented by a precomputed embedding $\mathbf{e}_i \in \mathbb{R}^d$, a stable item key $\mathbf{m}_i$, and an exposure count $p_i^{(t)}$. For semantic quantization, we use the ℓ_2 -normalized item embedding

$$\mathbf{x}_i = \frac{\mathbf{e}_i}{\|\mathbf{e}_i\|_2} \in \mathbb{R}^d. \quad (1)$$

The stable item key $\mathbf{m}_i$ is used only by the disambiguation rule described below.

Our framework produces a two-token SID

$$\mathbf{s}_i^{(t)} = (s_{i,\text{sem}}^{(t)}, s_{i,\text{dis}}), \quad (2)$$

where $s_{i,\text{sem}}^{(t)}$ is selected from a large semantic codebook $\mathcal{C}^{(t)} = \{\mathbf{c}_k^{(t)}\}_{k=1}^{K_s}$ and $s_{i,\text{dis}}$ is selected from a disambiguation vocabulary of size K_d . Throughout this paper, these two positions are referred to as **SID1** (semantic) and **SID2** (disambiguation), respectively. The method has three components. First, a single large semantic codebook replaces the two semantic residual-quantization levels in the original three-token SID while the existing disambiguation token is retained. Second, exposure-aware weights and temporally regularized center updates adapt the semantic codebook to a changing item distribution. Third, a codebook-level offline evaluation framework measures representation quality, semantic-code utilization, cluster load, full-SID collision, and temporal stability before downstream recommendation training.

Figure 2 summarizes initial construction and online updating. The upper panel learns the single large semantic codebook with exposure-weighted k -means and appends the deterministic disambiguation code. The lower panel updates only the semantic component. SID2 remains fixed across codebook versions.

3.2 Single-Level Semantic Codebook with Disambiguation

From two semantic tokens to one. The original three-token architecture represents item i as $(s_i^{(1)}, s_i^{(2)}, s_{i,\text{dis}})$. Given semantic codebooks $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$, residual quantization assigns the first code from $\mathbf{x}_i$ and the second from the remaining residual:

$$\begin{aligned} s_i^{(1)} &= \arg \min_k \|\mathbf{x}_i - \mathbf{c}_k^{(1)}\|_2^2, & \mathbf{r}_i^{(1)} &= \mathbf{x}_i - \mathbf{c}_{s_i^{(1)}}^{(1)}, \\ s_i^{(2)} &= \arg \min_k \|\mathbf{r}_i^{(1)} - \mathbf{c}_k^{(2)}\|_2^2, & \hat{\mathbf{x}}_i^{\text{RQ}} &= \mathbf{c}_{s_i^{(1)}}^{(1)} + \mathbf{c}_{s_i^{(2)}}^{(2)}. \end{aligned} \quad (3)$$

We instead concentrate the semantic capacity in one codebook. For the normalized item embedding $\mathbf{x}_i$ and semantic codebook $\mathcal{C}^{(t)}$, the assignment is

$$s_{i,\text{sem}}^{(t)} = \arg \min_{k \in \{1, \dots, K_s\}} \|\mathbf{x}_i - \mathbf{c}_k^{(t)}\|_2^2, \quad \hat{\mathbf{x}}_i^{(t)} = \mathbf{c}_{s_{i,\text{sem}}^{(t)}}^{(t)}. \quad (4)$$

The large vocabulary increases the capacity of the single semantic token without reintroducing another autoregressive position.

Disambiguation token. Semantic quantization may assign multiple items to the same semantic code. We therefore retain the final disambiguation layer of the deployed system. Let g be a fixed assignment rule over the stable item key:

$$s_{i,\text{dis}} = g(\mathbf{m}_i) \in \{1, \dots, K_d\}. \quad (5)$$

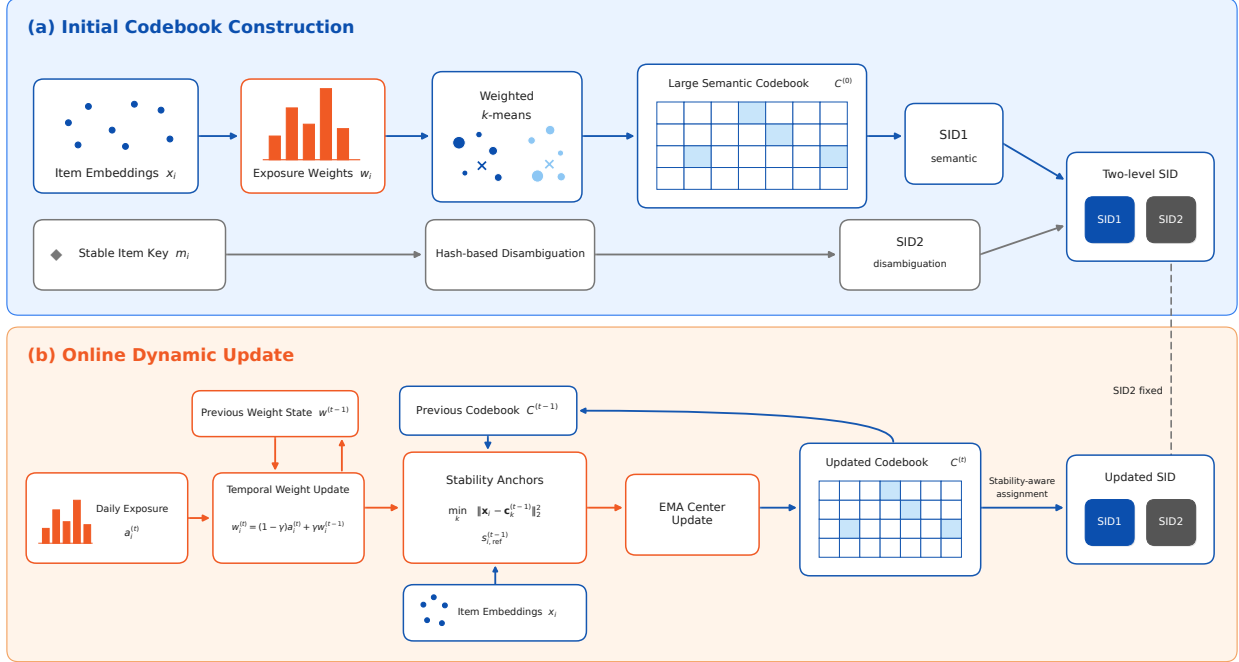


Figure 2: Overview of the proposed single-level large semantic codebook. Initial construction combines exposure-weighted semantic quantization (SID1) with deterministic collaborative disambiguation (SID2). Online updates maintain temporal exposure weights, penalize exposure-weighted SID1 switches, and update active centers while keeping SID2 unchanged.

The rule is deterministic and held fixed across codebook updates. In the industrial implementation, $g(\mathbf{m}_i)$ is a stable hash over the item-associated key tuple $\mathbf{m}_i$. Consequently, an unchanged item receives the same disambiguation token even when its semantic assignment is updated. This token is used to separate items sharing a semantic code and does not participate in semantic reconstruction.

Autoregressive generation. For a user history $\mathcal{H}$, the proposed target factorizes as

$$P(\mathbf{s}_i \mid \mathcal{H}) = P(s_{i,\text{sem}} \mid \mathcal{H})P(s_{i,\text{dis}} \mid \mathcal{H}, s_{i,\text{sem}}). \quad (6)$$

The original architecture requires an additional conditional factor for the second semantic token. Our design therefore reduces the number of autoregressive predictions from three to two while preserving the disambiguation stage.

3.3 Exposure-Aware Codebook Learning

Item frequency in industrial traffic is strongly skewed. An unweighted objective assigns the same importance to every item, regardless of its exposure frequency. The resulting partition therefore reflects the catalog distribution rather than the traffic distribution and may allocate insufficient representational capacity to frequently exposed items. To account for traffic while preventing head items from dominating the optimization, we compress the raw daily exposure count as follows:

$$a_i^{(t)} = 1 + \log_{10}(\max\{p_i^{(t)}, 1\}). \quad (7)$$

For a static training window, the resulting exposure weight is used in the weighted quantization objective

$$\min_{\mathcal{C}, \{s_i\}} \sum_i w_i \|\mathbf{x}_i - \mathbf{c}_{s_i}\|_2^2. \quad (8)$$

The initial semantic codebook is trained with weighted k -means. During weighted k -means++ initialization, let $D_i^2 = \min_{\mathbf{c} \in \mathcal{C}_{\text{sel}}} \|\mathbf{x}_i - \mathbf{c}\|_2^2$ denote the squared distance to the nearest selected center. Let I_{next} denote the item index selected as the next center. It is sampled according to

$$P(I_{\text{next}} = i \mid \mathcal{C}_{\text{sel}}) = \frac{w_i D_i^2}{\sum_j w_j D_j^2}. \quad (9)$$

3.4 Exposure-Aware Dynamic Codebook Update

Temporal exposure weights. We maintain one stateful weight for every item observed recently. For an item present both in the historical state and on day t , its weight is updated as

$$w_i^{(t)} = \gamma w_i^{(t-1)} + (1 - \gamma) a_i^{(t)}, \quad 0 < \gamma < 1. \quad (10)$$

A newly observed item is initialized with $w_i^{(t)} = a_i^{(t)}$. If a historical item is absent on day t , its weight decays as $w_i^{(t)} = \gamma w_i^{(t-1)}$. Items whose decayed weights fall below a small threshold ε are removed from the maintained state. This mechanism retains recent exposure history while gradually forgetting items that no longer receive traffic.

Stability anchors. To discourage unnecessary SID changes, we use the assignments under the previous codebook $\mathcal{C}^{(t-1)}$ as stability anchors for the day- t update. We encode each current embedding with the previous codebook:

$$s_{i,\text{ref}}^{(t-1)} = \arg \min_k \|\mathbf{x}_i - \mathbf{c}_k^{(t-1)}\|_2^2. \quad (11)$$

These reference assignments define the previous partition used to estimate the current daily centers and provide the anchors for the switching penalty below.

Center update and final assignment. For center k , let $A_k^{(t)} = \{i : s_{i,\text{ref}}^{(t-1)} = k\}$ be its reference-partition items on day t . We compute the exposure-weighted daily center

$$\bar{\mathbf{c}}_k^{(t)} = \frac{\sum_{i \in A_k^{(t)}} w_i^{(t)} \mathbf{x}_i}{\sum_{i \in A_k^{(t)}} w_i^{(t)}}. \quad (12)$$

For a center receiving at least one assignment, the updated value is

$$\mathbf{c}_k^{(t)} = (1 - \alpha) \mathbf{c}_k^{(t-1)} + \alpha \bar{\mathbf{c}}_k^{(t)}, \quad 0 < \alpha \leq 1. \quad (13)$$

An inactive center remains unchanged, $\mathbf{c}_k^{(t)} = \mathbf{c}_k^{(t-1)}$. We then compute the published semantic assignment against the updated centers with an exposure-weighted switching penalty:

$$s_{i,\text{sem}}^{(t)} = \arg \min_k \left[\|\mathbf{x}_i - \mathbf{c}_k^{(t)}\|_2^2 + \lambda w_i^{(t)} \mathbb{I}(k \neq s_{i,\text{ref}}^{(t-1)}) \right]. \quad (14)$$

Here, $\lambda \geq 0$ controls the adaptation–stability trade-off. The penalty discourages SID changes for high-exposure items while still allowing reassignment when it provides a sufficiently large reduction in quantization distance. This stability is important in streaming training because abruptly reassigning a high-exposure item changes many discrete supervision targets at once. The resulting discontinuity can destabilize gradient updates and hinder model optimization. We update the codebook once per day and publish the new semantic assignments together with the unchanged disambiguation rule.

Algorithm 1 summarizes one online update interval.

Algorithm 1 Exposure-aware dynamic semantic-codebook update

Require: $\mathcal{I}_t$ with $\{\mathbf{x}_i, p_i^{(t)}, \mathbf{m}_i\}$, $\mathcal{C}^{(t-1)}$, $\{w_i^{(t-1)}\}$, and $\gamma, \lambda, \alpha, \varepsilon$

Ensure: $\mathcal{C}^{(t)}$ and $\{(\text{SID1}_i, \text{SID2}_i)\}$

```
1: for all items in the union of the current corpus and weight state do
2:   if  $i \in \mathcal{I}_t$  then
3:     Compute  $a_i^{(t)}$  and update or initialize  $w_i^{(t)}$ 
4:   else
5:     Set  $w_i^{(t)} \leftarrow \gamma w_i^{(t-1)}$ 
6:   end if
7:   Remove  $i$  from the weight state if  $w_i^{(t)} < \varepsilon$ 
8: end for
9: for all  $i \in \mathcal{I}_t$  do
10:   Encode  $\mathbf{x}_i$  with  $\mathcal{C}^{(t-1)}$  to obtain  $s_{i,\text{ref}}^{(t-1)}$ 
11: end for
12: Form  $A_k^{(t)} = \{i \in \mathcal{I}_t : s_{i,\text{ref}}^{(t-1)} = k\}$  for all  $k$ 
13: for  $k = 1$  to  $K_s$  do
14:   if  $A_k^{(t)} \neq \emptyset$  then
15:     Compute  $\bar{\mathbf{c}}_k^{(t)}$  and update  $\mathbf{c}_k^{(t)}$  using Eqs. (12)–(13)
16:   else
17:     Set  $\mathbf{c}_k^{(t)} \leftarrow \mathbf{c}_k^{(t-1)}$ 
18:   end if
19: end for
20: for all  $i \in \mathcal{I}_t$  do
21:   Assign semantic SID1  $s_{i,\text{sem}}^{(t)}$  using Eq. (14)
22: end for
23: Publish  $\mathcal{C}^{(t)}$  and  $(\text{SID1}, \text{SID2}) = (s_{i,\text{sem}}^{(t)}, g(\mathbf{m}_i))$ 
```

3.5 Offline Codebook Evaluation Framework

Training a downstream generative recommender remains the definitive evaluation of recommendation quality, but it is unnecessarily expensive for every intermediate codebook candidate. We therefore evaluate codebooks first using a set of offline metrics. Let $\mathcal{E}_t$ be the evaluation items for day t and $N_t = |\mathcal{E}_t|$.

Representation quality. We measure the cosine similarity between each normalized item embedding and its reconstruction:

$$R_{\text{rec}} = \frac{1}{N_t} \sum_{i \in \mathcal{E}_t} \cos(\mathbf{x}_i, \hat{\mathbf{x}}_i). \quad (15)$$

Here, $\hat{\mathbf{x}}_i$ is the selected semantic center. For a residual-quantization baseline, it is the sum of the selected semantic centers. In addition to the mean, we report the 10th percentile (P10) and median of item-level reconstruction similarity to characterize its lower tail and central tendency.

Utilization and cluster load. For semantic level ℓ with vocabulary size K_ℓ , let $q_i^{(\ell)}$ be the code assigned to item i . Its utilization is

$$U_\ell = \frac{|\{q_i^{(\ell)} : i \in \mathcal{E}_t\}|}{K_\ell}. \quad (16)$$

For load statistics, let σ_i denote the complete semantic group of item i , namely one semantic code for the proposed structure and the tuple of semantic codes for a residual-quantization baseline. For each occupied group k , its item load is

$$n_k = \sum_{i \in \mathcal{E}_t} \mathbb{I}(\sigma_i = k). \quad (17)$$

The 95th percentile and maximum of $\{n_k\}$ identify crowded semantic groups, while the coefficient of variation (CV) measures overall load imbalance, with a smaller CV indicating a more even distribution. When a dataset provides meaningful content-type labels, we also report the type composition within each cluster. This analysis is omitted for datasets without such annotations.

Full-SID collision. Let $\mathcal{G}_z$ be the set of distinct items assigned to complete SID $z = (s_{\text{sem}}, s_{\text{dis}})$, and let $\mathcal{Z}$ be the occupied SIDs. We report both the fraction of occupied SIDs that collide and the fraction of items involved in a collision:

$$R_{\text{SID}} = \frac{\sum_{z \in \mathcal{Z}} \mathbb{I}(|\mathcal{G}_z| > 1)}{|\mathcal{Z}|}, \quad R_{\text{item}} = \frac{\sum_{z \in \mathcal{Z}} |\mathcal{G}_z| \mathbb{I}(|\mathcal{G}_z| > 1)}{N_t}. \quad (18)$$

Temporal stability. To isolate codebook change from feature change, the old and new codebooks encode the same target-day embeddings. Denote the resulting semantic codes by q_i^{old} and q_i^{new} . The item-level and exposure-weighted change rates are

$$R_{\text{chg}} = \frac{1}{N_t} \sum_{i \in \mathcal{E}_t} \mathbb{I}(q_i^{\text{old}} \neq q_i^{\text{new}}), \quad R_{\text{chg}}^{\text{PV}} = \frac{\sum_{i \in \mathcal{E}_t} p_i^{(t)} \mathbb{I}(q_i^{\text{old}} \neq q_i^{\text{new}})}{\sum_{i \in \mathcal{E}_t} p_i^{(t)}}. \quad (19)$$

These offline measurements are used to screen codebook candidates. Downstream recommendation experiments are still required to establish their task-level effectiveness.

4 Experiments

4.1 Experimental Setup

Datasets. We use Amazon Reviews 2014 (Beauty) [10] and the KuaiRec 2.0 Big Matrix [5]. Their statistics are summarized in Table 1. Amazon Beauty is used for the controlled static S3–S2 comparison. KuaiRec provides dense timestamped viewing logs and supports both a static comparison with exposure-weighted S2 (PV-S2) and a separate fixed-date comparison between static and dynamic PV-S2.

Table 1: Statistics of the public datasets used in our experiments.

Dataset	Users	Items	Interactions	Temporal coverage
Amazon Beauty (5-core)	22,363	12,101	198,502	May 1996–July 2014
KuaiRec 2.0 Big Matrix	7,176	10,728	12,530,806	28 observed days

Data preprocessing and evaluation. Interactions are ordered chronologically for each user, and KuaiRec retains viewing events with `watch_ratio` ≥ 1.0 from users with at least five retained events. Amazon and the first KuaiRec comparison use leave-two-out evaluation. The last interaction is used for testing, the penultimate interaction for validation, and earlier interactions for training. All KuaiRec viewing events define the PV weights used by the leave-two-out PV-S2 baseline. Because users’ final interactions occur on different calendar dates, leave-two-out does not provide a common temporal boundary between codebook updating and evaluation. The second KuaiRec comparison therefore uses the disjoint fixed-date evaluation described in Section 4.3.2. The two sets of results are reported separately.

Codebook sizes. On both Amazon Beauty and KuaiRec, S3 uses level sizes [256, 256, 256], whereas S2, PV-S2, and Dynamic PV-S2 use [1024, 512]. The final level is a balanced disambiguation code. PV weighting and dynamic updates change how the semantic codebook is fitted, not its dimensions.

Recommendation models. We prioritize the OneRec-V1 and OneRec-V2 public-data adaptations [3, 29], and additionally report TIGER [13], RPG [7], SEATER [16], and COBRA [24]. These models cover autoregressive, parallel, tree-constrained, and sparse–dense cascaded generative recommendation. OneRec is evaluated at nominal scales from 0.1B to 0.7B. The exact configurations are listed in Appendix B. The public-data adaptations exclude production-only features and preference-alignment stages. Within each paired comparison, we keep the data split, recommender architecture, optimization settings, and seed fixed and change only the SID artifact.

Metrics and repeated runs. Recommendation quality is measured by Recall@ K and NDCG@ K for $K \in \{5, 10\}$. Main public-data results use the same three seeds for every paired codebook variant and report the mean and sample standard deviation. Codebook evaluation follows Section 3.5.

4.2 Static Two-Level versus Three-Level SIDs on Amazon Beauty

The complete OneRec-V1/V2 sweeps are summarized in Figure 3. Their numerical results are provided in Appendix C. A line plot is used because model scale is ordered. Shaded regions show the sample standard deviation across seeds. Table 2 reports the remaining models retained in the current comparison.

Across all seven model scales, S2 improves mean Recall@10 over S3 by 5.0% for OneRec-V1 and 8.7% for OneRec-V2. The corresponding mean NDCG@10 gains are 4.1% and 8.5%. S2 is better at

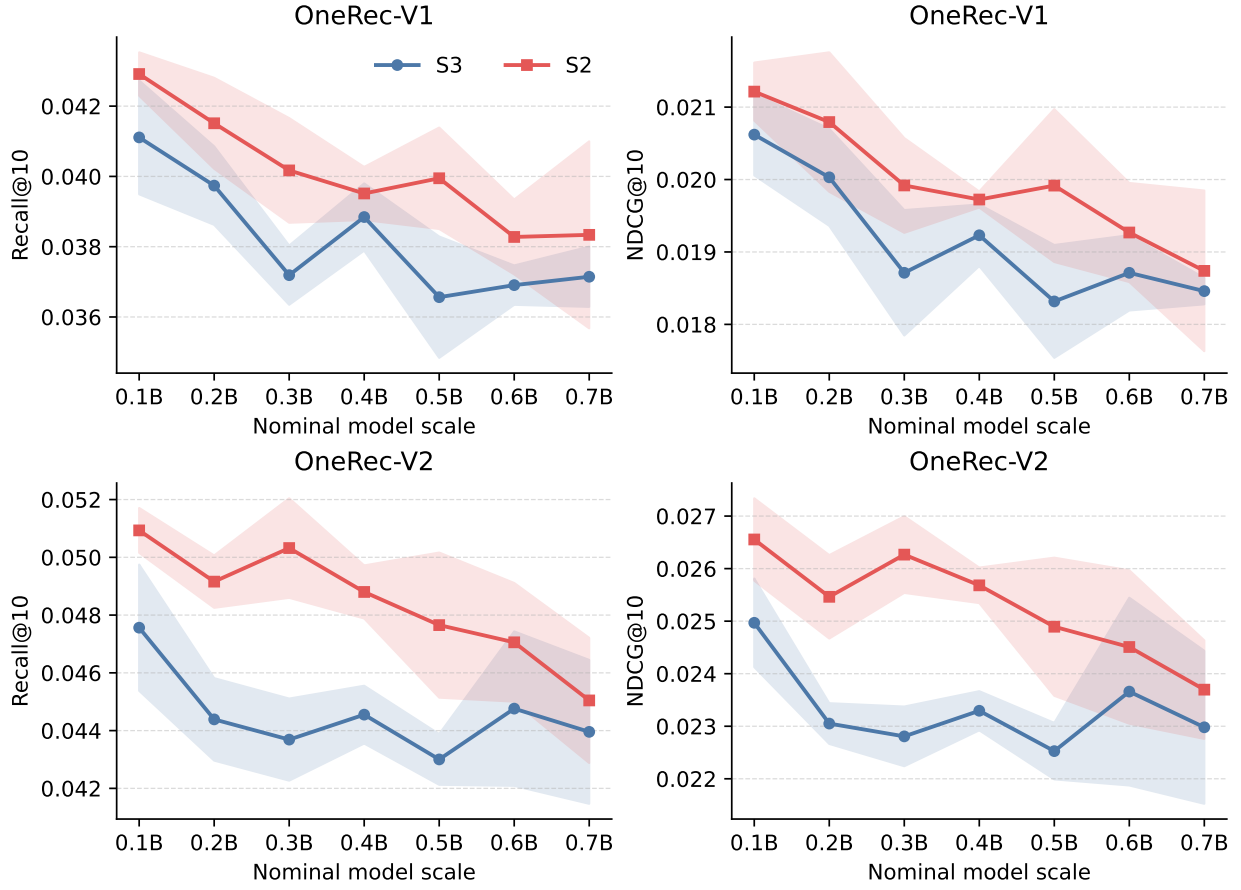


Figure 3: OneRec-V1/V2 scaling results on Amazon Reviews 2014 (Beauty). Curves and shaded regions denote the mean and sample standard deviation over seeds $\{7, 42, 2024\}$, respectively.

Table 2: Recommendation performance on Amazon Reviews 2014 (Beauty). Results are mean $\pm$ sample standard deviation over seeds $\{7, 42, 2024\}$.

Model	SID	Recall@5	Recall@10	NDCG@5	NDCG@10
TIGER	S3	0.0369 ± 0.0078	0.0520 ± 0.0136	0.0253 ± 0.0051	0.0302 ± 0.0070
TIGER	S2	0.0328 ± 0.0009	0.0463 ± 0.0008	0.0230 ± 0.0010	0.0273 ± 0.0010
SEATER	S3	0.0380 ± 0.0014	0.0562 ± 0.0014	0.0256 ± 0.0009	0.0314 ± 0.0010
SEATER	S2	0.0385 ± 0.0002	0.0584 ± 0.0008	0.0258 ± 0.0005	0.0323 ± 0.0006
RPG	S3	0.0402 ± 0.0004	0.0569 ± 0.0008	0.0284 ± 0.0003	0.0337 ± 0.0005
RPG	S2	0.0441 ± 0.0009	0.0646 ± 0.0006	0.0310 ± 0.0007	0.0375 ± 0.0006
COBRA	S3	0.0062 ± 0.0003	0.0102 ± 0.0001	0.0040 ± 0.0001	0.0053 ± 0.0000
COBRA	S2	0.0103 ± 0.0013	0.0159 ± 0.0014	0.0069 ± 0.0010	0.0086 ± 0.0010

every evaluated scale for both OneRec variants on Recall@10. Among the other recommenders, S2 improves Recall@10 for SEATER, RPG, and COBRA, while TIGER performs better with S3. These results indicate that removing one semantic level generally preserves or improves recommendation quality, although the outcome remains model dependent.

4.3 Static and Dynamic Codebooks on KuaiRec

4.3.1 Static Comparison under Leave-Two-Out

Offline codebook quality. Table 3 reports offline codebook quality for the leave-two-out comparison. For S3, reconstruction uses the sum of the two selected semantic centers, whereas S2 and PV-S2 use one semantic center. S2 achieves the highest mean reconstruction cosine similarity, followed by S3 and PV-S2, and every semantic level retains at least 94.92% utilization. Among the single-level variants, PV-S2 reduces the maximum group load from 376 to 248 and the load CV from 1.9938 to 1.6390.

Table 3: Offline codebook quality on KuaiRec.

Metric	S3	S2	PV-S2
Mean reconstruction cosine similarity	0.9958	0.9962	0.9939
Reconstruction P10 / median	0.9879 / 0.9991	0.9879 / 0.9996	0.9823 / 0.9995
	L1: 100.00%		
Semantic-code utilization	L2: 100.00%	L1: 97.56%	L1: 94.92%
Semantic-group load P95 / maximum / CV	9.0 / 256 / 3.1685	26.1 / 376 / 1.9938	34.4 / 248 / 1.6390

Recommendation performance. The complete OneRec-V1/V2 sweeps over the three static SID variants are shown in Figure 4. Their numerical values are provided in Appendix D. As in the Amazon comparison, curves report the three-seed mean and shaded regions show the sample standard deviation. Table 4 reports the remaining models.

For OneRec-V1, averaging over seven model scales, S2 and PV-S2 improve Recall@10 over S3 by 8.8% and 7.0%, respectively, with corresponding NDCG@10 gains of 5.1% and 4.8%. For OneRec-V2, S2 and PV-S2 improve Recall@10 by 7.1% and 6.9%, and NDCG@10 by 3.8% and 4.8%, respectively. Both two-level variants also improve Recall@10 for every non-OneRec model in Table 4. Neither S2 nor PV-S2 dominates uniformly: exposure weighting produces the best Recall@10 for TIGER, SEATER, RPG, and COBRA, but the preferred variant varies across OneRec scales and ranking metrics.

4.3.2 Fixed-Date Static–Dynamic Comparison

Figure 5 summarizes the fixed-date evaluation setup. The static arm trains with $\mathcal{C}_1$ on targets from July 12 through September 3. The dynamic arm updates $\mathcal{C}_1$ to $\mathcal{C}_2$ using traffic from August 9 through September 3, then fine-tunes from the matched static checkpoint on $\mathcal{C}_2$ -encoded targets over the same period. Both arms validate on September 4 and test on September 5. The validation and test days are excluded from codebook fitting.

Figure 6 compares the matched static and dynamic arms across OneRec scales. The exact numerical results are provided in Appendix E. Table 5 reports the completed three-seed comparisons for the remaining recommenders.

Averaged across model scales, Dynamic C2 improves OneRec-V1 Recall@10 by 1.4% and NDCG@10 by 7.0% relative to Static C1. For OneRec-V2, the corresponding gains are 2.7%

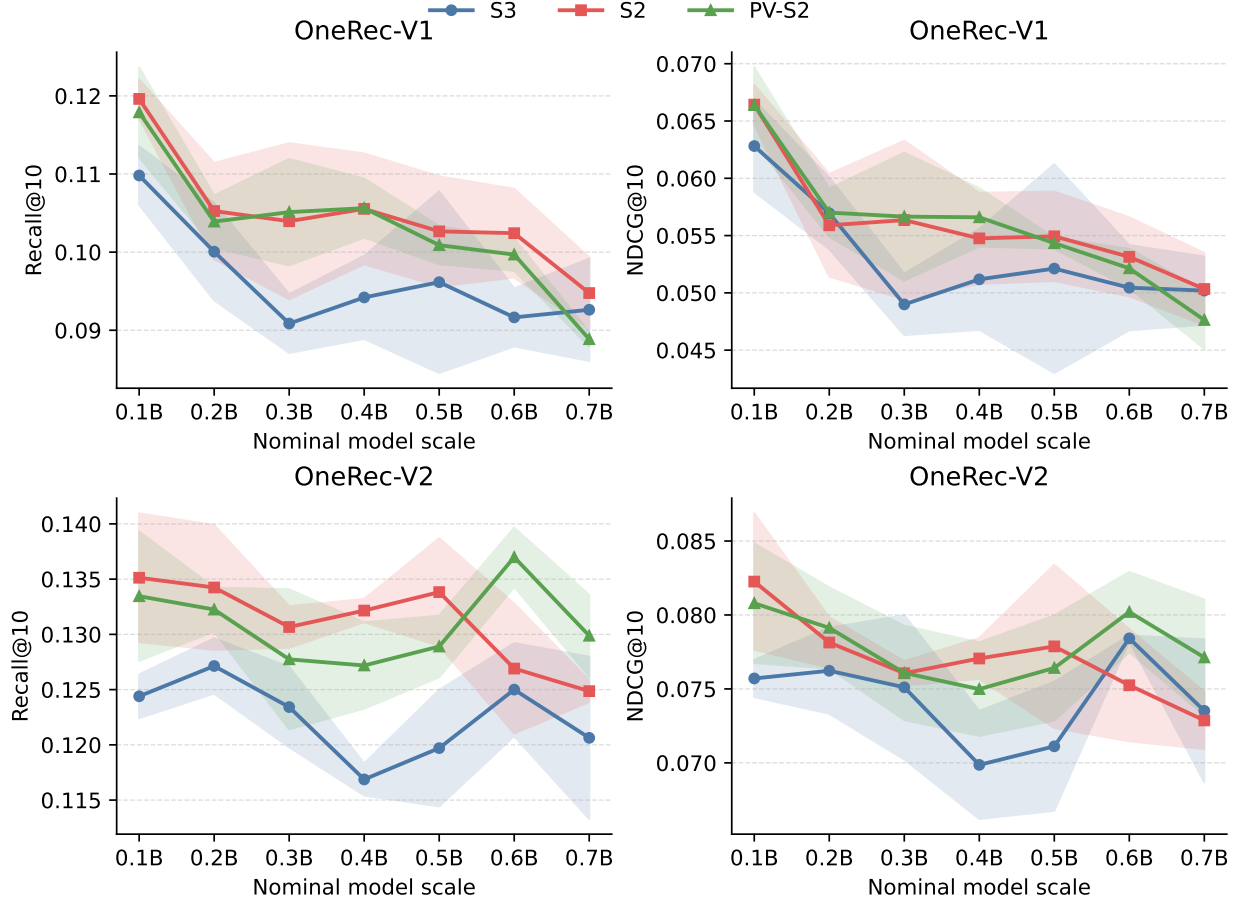


Figure 4: OneRec-V1/V2 scaling results on KuaiRec under S3, S2, and PV-S2. Curves and shaded regions denote the mean and sample standard deviation over seeds {7, 42, 2024}, respectively.

Table 4: Recommendation performance of the completed SID comparisons on KuaiRec. Results are mean $\pm$ sample standard deviation over seeds $\{7, 42, 2024\}$.

Model	SID	Recall@5	Recall@10	NDCG@5	NDCG@10
TIGER	S3	0.1057 ± 0.0032	0.1429 ± 0.0019	0.0740 ± 0.0010	0.0860 ± 0.0006
TIGER	S2	0.1094 ± 0.0022	0.1483 ± 0.0012	0.0761 ± 0.0027	0.0886 ± 0.0023
TIGER	PV-S2	0.1087 ± 0.0028	0.1492 ± 0.0024	0.0742 ± 0.0030	0.0873 ± 0.0028
SEATER	S3	0.1032 ± 0.0019	0.1374 ± 0.0007	0.0697 ± 0.0028	0.0808 ± 0.0022
SEATER	S2	0.1077 ± 0.0018	0.1473 ± 0.0040	0.0729 ± 0.0019	0.0858 ± 0.0002
SEATER	PV-S2	0.1069 ± 0.0022	0.1478 ± 0.0028	0.0717 ± 0.0022	0.0849 ± 0.0024
RPG	S3	0.0060 ± 0.0004	0.0082 ± 0.0010	0.0046 ± 0.0003	0.0053 ± 0.0003
RPG	S2	0.0060 ± 0.0004	0.0110 ± 0.0020	0.0037 ± 0.0002	0.0053 ± 0.0006
RPG	PV-S2	0.0109 ± 0.0014	0.0172 ± 0.0017	0.0066 ± 0.0005	0.0087 ± 0.0008
COBRA	S3	0.0998 ± 0.0008	0.1367 ± 0.0011	0.0693 ± 0.0008	0.0812 ± 0.0005
COBRA	S2	0.1088 ± 0.0010	0.1484 ± 0.0034	0.0754 ± 0.0003	0.0881 ± 0.0009
COBRA	PV-S2	0.1075 ± 0.0037	0.1486 ± 0.0030	0.0749 ± 0.0016	0.0882 ± 0.0014

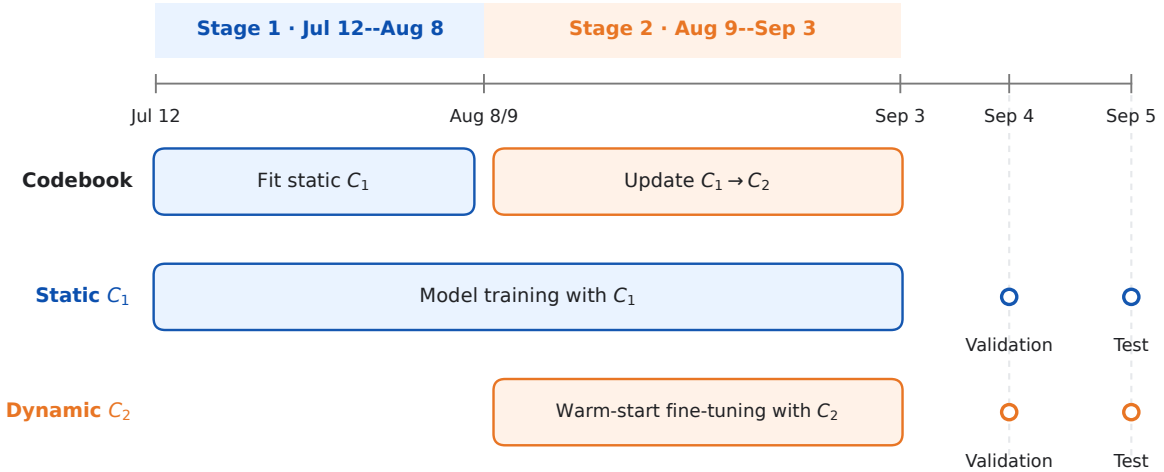


Figure 5: Fixed-date evaluation setup for the static-dynamic comparison on KuaiRec. Bars indicate the data windows used for codebook fitting or updating and recommender training. Both arms use the first positive interaction per user on September 4 for validation and on September 5 for testing.

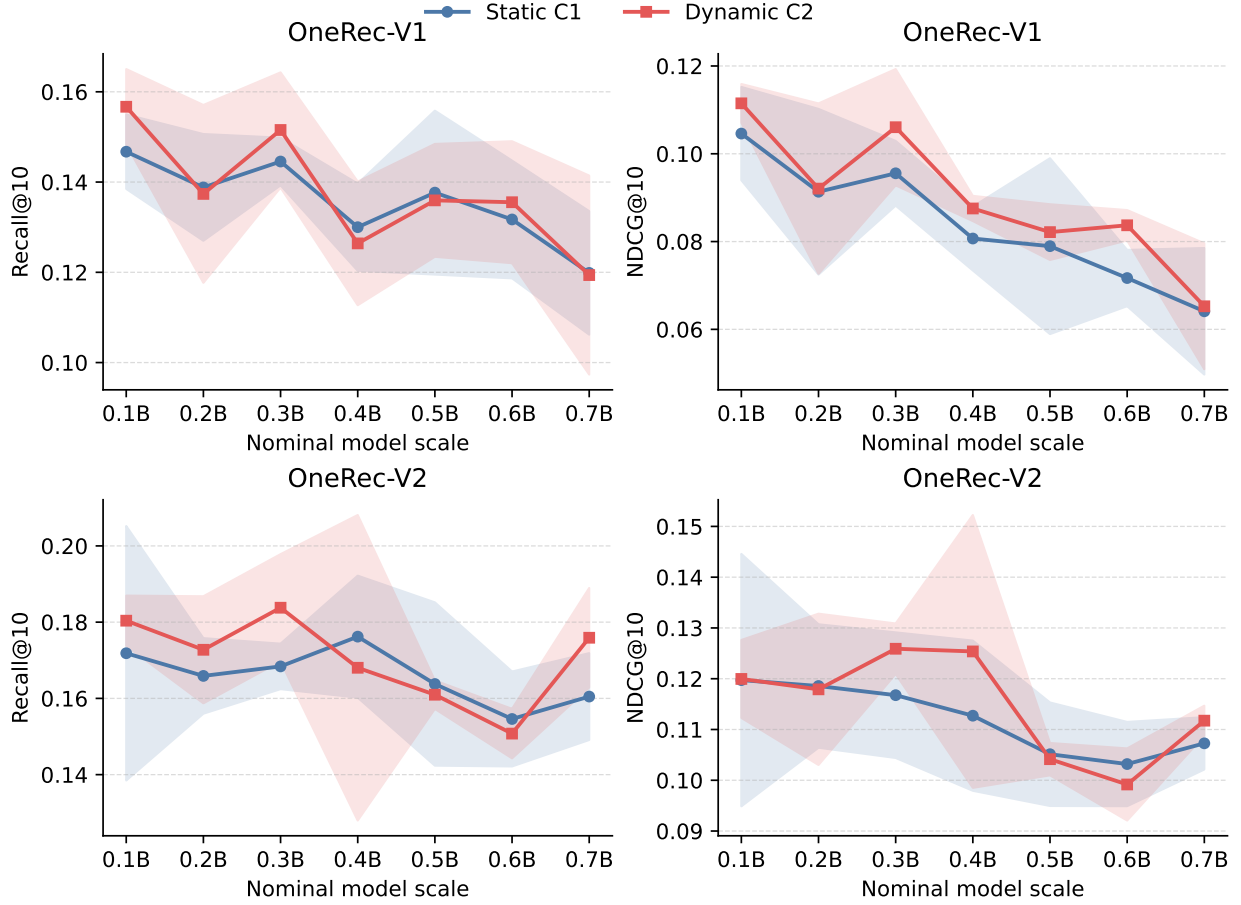


Figure 6: OneRec-V1/V2 results under the fixed-date KuaiRec comparison. Curves and shaded regions denote the mean and sample standard deviation over seeds $\{7, 42, 2024\}$, respectively. Static C1 uses the first-stage PV-S2 codebook, while Dynamic C2 uses the updated codebook and matched warm-start fine-tuning.

Table 5: Fixed-date static–dynamic recommendation performance on KuaiRec. Results are mean $\pm$ sample standard deviation over seeds $\{7, 42, 2024\}$.

Model	Codebook	Recall@5	Recall@10	NDCG@5	NDCG@10
TIGER	Static C1	0.0075 ± 0.0025	0.0092 ± 0.0011	0.0048 ± 0.0009	0.0053 ± 0.0004
TIGER	Dynamic C2	0.0496 ± 0.0775	0.0532 ± 0.0812	0.0372 ± 0.0584	0.0383 ± 0.0596
SEATER	Static C1	0.1484 ± 0.0014	0.1788 ± 0.0033	0.1161 ± 0.0034	0.1260 ± 0.0031
SEATER	Dynamic C2	0.1570 ± 0.0106	0.1890 ± 0.0090	0.1215 ± 0.0066	0.1331 ± 0.0068
RPG	Static C1	0.0356 ± 0.0238	0.0669 ± 0.0312	0.0206 ± 0.0151	0.0307 ± 0.0164
RPG	Dynamic C2	0.0465 ± 0.0136	0.0688 ± 0.0105	0.0335 ± 0.0073	0.0392 ± 0.0073
COBRA	Static C1	0.0704 ± 0.0086	0.1116 ± 0.0103	0.0402 ± 0.0072	0.0551 ± 0.0050
COBRA	Dynamic C2	0.0892 ± 0.0061	0.1242 ± 0.0037	0.0558 ± 0.0030	0.0651 ± 0.0027

and 2.7%. Dynamic C2 also improves mean Recall@10 by 5.7% for SEATER and 2.8% for RPG. Although the gains are not uniform at every OneRec scale, the aggregate results are positive for both OneRec variants, SEATER, and RPG.

Relative to C1, C2 changes SID1 for 46 of 10,728 items, corresponding to an item-level change rate of 0.4288%. The change rate is 0.6418% among items active during the second stage and 0.4383% when weighted by second-stage PV. The mean and maximum center displacements are 0.004054 and 0.050005, respectively. These measurements indicate that the update adapts the semantic partition while keeping SID1 assignments largely stable.

4.4 Industrial Codebook Evaluation

We evaluate the industrial codebooks on seven consecutive daily snapshots. For S2, reconstruction uses its single semantic center. For S3, we report both its first semantic level and the cumulative reconstruction from its two residual semantic levels. Table 6 shows the first and last daily means. Static S2 slightly exceeds the cumulative two-level semantic reconstruction of S3 on both endpoints. Static PV-S2 provides a further gain. Dynamic PV-S2 shares the same initialization and Day 1 value as static PV-S2, but reaches 0.826723 on Day 7 instead of declining with the static codebook.

Table 6: Mean reconstruction similarity on the first and seventh industrial evaluation days.

Codebook representation	Day 1	Day 7	Change
Static S3, first semantic level	0.758357	0.754451	-0.003906
Static S3, two semantic levels	0.802241	0.798868	-0.003373
Static S2	0.805676	0.800899	-0.004777
Static PV-S2	0.821336	0.814508	-0.006828
Dynamic PV-S2	0.821336	0.826723	+0.005387

4.5 Industrial Online and Serving Evaluation

We deployed the two-level SID system in a five-day online A/B test serving 2.5% of production traffic against the deployed three-level SID baseline. The primary consumption metric increased by 0.792%. Across the decoder, LazyAR, and MTP serving architectures, estimated autoregressive-decoding FLOPs decreased by 47.93%–48.70%. Single-card QPS increased by 47.0% for decoder, 28.57% for LazyAR, and 33.3% for MTP. The estimate includes the one-time cross-attention key/value projections, beam-expanded decoder computation, and vocabulary projection. Appendix A provides the symbolic derivation and separately characterizes KV-cache access. Together, the reduced decoding cost and measured throughput gains demonstrate the serving benefit of the shorter SID across decoding architectures.

5 Conclusion

This paper revisits Semantic ID codebook design for large-scale generative recommendation. Motivated by the sparse conditional use of the second semantic level in a deployed three-level SID, we replace the multi-level residual semantic representation with one large semantic codebook and retain a separate collaborative code only for disambiguation. The resulting semantic-plus-disambiguation SID concentrates semantic capacity in a single token and reduces the number of autoregressive predictions from three to two. To accommodate an evolving online item corpus, we further develop

an exposure-aware dynamic codebook that combines temporally decayed exposure weights, stability-aware assignment, and moving-average center updates. Finally, we introduce an offline evaluation framework covering reconstruction quality, code utilization and load, full-SID collisions, and temporal stability, enabling codebook candidates to be screened before costly downstream model training. Across the two public datasets, the proposed architecture improves mean Recall@10 by 5.0%–8.8% and mean NDCG@10 by 4.1%–5.1% for OneRec-V1. The corresponding gains for OneRec-V2 are 7.1%–8.7% and 3.8%–8.5%. Under the fixed-date KuaiRec evaluation, dynamic updating further improves Recall@10 by 1.4% and 2.7%, and NDCG@10 by 7.0% and 2.7%, for OneRec-V1 and OneRec-V2, respectively. In industrial codebook evaluation, the dynamic codebook improves reconstruction similarity by 1.50% while changing only 0.4288% of item-level SID1 assignments in the fixed-date KuaiRec update. Across the decoder, LazyAR, and MTP serving architectures, the shorter SID reduces estimated autoregressive-decoding FLOPs by 47.93%–48.70% and raises single-card QPS by 28.57%–47.0%. In a five-day online A/B test serving 2.5% of production traffic, the two-level SID improves the primary consumption metric by 0.792%. Together, these components provide a practical framework for simplifying, updating, and evaluating SID codebooks in industrial generative recommendation.

References

- [1] Robi Bhattacharjee, Jacob Imola, Michal Moshkovitz, and Sanjoy Dasgupta. Online k -means clustering on arbitrary data streams. In *Proceedings of the 34th International Conference on Algorithmic Learning Theory*, volume 201 of *Proceedings of Machine Learning Research*, pages 204–236, 2023.
- [2] Runjin Chen, Mingxuan Ju, Ngoc Bui, Dimosthenis Antypas, Stanley Cai, Xiaopeng Wu, Leonardo Neves, Zhangyang Wang, Neil Shah, and Tong Zhao. Enhancing item tokenization for generative recommendation through self-improvement. *arXiv preprint arXiv:2412.17171*, 2024.
- [3] Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. OneRec: Unifying retrieve and rank with generative recommender and iterative preference alignment. *arXiv preprint arXiv:2502.18965*, 2025.
- [4] Junchen Fu, Xuri Ge, Alexandros Karatzoglou, Ioannis Arapakis, Suzan Verberne, Joemon M. Jose, and Zhaochun Ren. Differentiable semantic id for generative recommendation. *arXiv preprint arXiv:2601.19711*, 2026.
- [5] Chongming Gao, Shijun Li, Wenqiang Lei, Jiawei Chen, Biao Li, Peng Jiang, Xiangnan He, Jiaxin Mao, and Tat-Seng Chua. KuaiRec: A fully-observed dataset and insights for evaluating recommender systems. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, pages 540–550, 2022. doi: 10.1145/3511808.3557220.
- [6] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (RLP): A unified pretrain, personalized prompt & predict paradigm (P5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 299–315, 2022. doi: 10.1145/3523227.3546767.
- [7] Yupeng Hou, Jiacheng Li, Ashley Shin, Jinsung Jeon, Abhishek Santhanam, Wei Shao, Kaveh Hassani, Ning Yao, and Julian J. McAuley. Generating long semantic ids in parallel for recommendation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 956–966, 2025. doi: 10.1145/3711896.3736979.

- [8] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532, 2022.
- [9] Enze Liu, Bowen Zheng, Cheng Ling, Lantao Hu, Han Li, and Wayne Xin Zhao. Generative recommender with end-to-end learnable item tokenization. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2025. doi: 10.1145/3726302.3729989.
- [10] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52, 2015. doi: 10.1145/2766462.2767755.
- [11] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: VQ-VAE made simple. In *International Conference on Learning Representations*, 2024.
- [12] Masud Moshtaghi, James C. Bezdek, Sarah M. Erfani, Christopher Leckie, and James Bailey. Online cluster validity indices for performance monitoring of streaming data clustering. *International Journal of Intelligent Systems*, 34(4):541–563, 2019. doi: 10.1002/int.22064.
- [13] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. In *Advances in Neural Information Processing Systems*, volume 36, pages 10299–10315, 2023.
- [14] Teng Shi, Chenglei Shen, Weijie Yu, Shen Nie, Chongxuan Li, Xiao Zhang, Ming He, Yan Han, and Jun Xu. LLaDA-Rec: Discrete diffusion for parallel semantic id generation in generative recommendation. *arXiv preprint arXiv:2511.06254*, 2025.
- [15] Teng Shi, Zhaoheng Li, Yuanhang Qu, Yi Liu, Lixiang Lai, and Yuning Jiang. SSRLive: Live streaming recommendation with dynamic semantic id. *arXiv preprint arXiv:2606.06970*, 2026.
- [16] Zihua Si, Zhongxiang Sun, Jiale Chen, Guozhang Chen, Xiaoxue Zang, Kai Zheng, Yang Song, Xiao Zhang, Jun Xu, and Kun Gai. Generative retrieval with semantic tree-structured identifiers and contrastive learning. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 154–163, 2024. doi: 10.1145/3673791.3698408.
- [17] Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten de Rijke, and Zhaochun Ren. Learning to tokenize for generative retrieval. In *Advances in Neural Information Processing Systems*, volume 36, pages 46345–46361, 2023.
- [18] Yi Tay, Vinh Q. Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. Transformer memory as a differentiable search index. In *Advances in Neural Information Processing Systems*, volume 35, pages 21831–21843, 2022.
- [19] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

- [20] Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2400–2409, 2024. doi: 10.1145/3627673.3679569.
- [21] Xuesi Wang, Junjie Wang, Ziliang Wang, Weijie Bian, and Guanxing Zhang. Decoupled residual quantization for robust semantic ids in recommendation. *arXiv preprint arXiv:2606.01844*, 2026.
- [22] Haibo Xing, Hao Deng, Yucheng Mao, Lingyu Mu, Jinxin Hu, Yi Xu, Hao Zhang, Jiahao Wang, Shizhun Wang, Yu Zhang, Xiaoyi Zeng, and Jing Zhang. REG4Rec: Reasoning-enhanced generative model for large-scale recommendation systems. *arXiv preprint arXiv:2508.15308*, 2025.
- [23] Jing Yan, Yimeng Bai, Zongyu Liu, Yahui Liu, Junwei Wang, Jingze Huang, Haoda Li, Sihao Ding, Shaohui Ruan, and Yang Zhang. MERGE: Next-generation item indexing paradigm for large-scale streaming recommendation. *arXiv preprint arXiv:2601.20199*, 2026.
- [24] Yuhao Yang, Zhi Ji, Zhaopeng Li, Yi Li, Zhonglin Mo, Yue Ding, Kai Chen, Zijian Zhang, Jie Li, Shuanglong Li, and Lin Liu. Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations. *arXiv preprint arXiv:2503.02453*, 2025.
- [25] Lijun Yu, José Lezama, Nitesh Bharadwaj Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G. Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David A. Ross, and Lu Jiang. Language model beats diffusion—tokenizer is key to visual generation. In *International Conference on Learning Representations*, 2024.
- [26] Penglong Zhai, Yifang Yuan, Fanyi Di, Jie Li, Yue Liu, Chen Li, Jie Huang, Sicong Wang, Yao Xu, and Xin Li. A simple contrastive framework of item tokenization for generative recommendation. *arXiv preprint arXiv:2506.16683*, 2025.
- [27] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering*, pages 1435–1448, 2024. doi: 10.1109/ICDE60146.2024.00118.
- [28] Bowen Zheng, Enze Liu, Zhongfu Chen, Zhongrui Ma, Yue Wang, Wayne Xin Zhao, and Ji-Rong Wen. Pre-training generative recommender with multi-identifier item tokenization. *arXiv preprint arXiv:2504.04400*, 2025.
- [29] Guorui Zhou, Hengrui Hu, Hongtao Cheng, Huanjie Wang, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, et al. OneRec-V2 technical report. *arXiv preprint arXiv:2508.20900*, 2025.

A Inference-Cost Derivation

We count one multiplication and one addition as two floating-point operations. Thus, multiplying $\mathbf{A} \in \mathbb{R}^{m \times k}$ by $\mathbf{B} \in \mathbb{R}^{k \times n}$ requires approximately

$$F_{\text{matmul}}(m, k, n) = 2mkn \quad (20)$$

FLOPs. Lower-order element-wise operations are omitted.

Let L denote the number of decoder layers, d the model width, h the feed-forward width, and N the cross-attention context length. At decoding step j , let b_j be the number of beams entering the decoder and V_j the output vocabulary size. The cross-attention keys and values are projected once per layer and then cached, giving the shared cost

$$F_{\text{cross-KV}} = 4LNd^2. \quad (21)$$

For one decoding step, the cross-attention query and output projections cost $4b_jLd^2$, attention against the cached context costs $4b_jLNd$, and the associated SwiGLU feed-forward block costs $6b_jLdh$. With self-attention KV caching, only the current token is projected. Its query-key-value and output projections cost $8b_jLd^2$, attention over the cached prefix costs $4b_jLjd$, and the second feed-forward block costs $6b_jLdh$. The vocabulary projection adds $2b_jdV_j$. Hence, the step cost is

$$C_j = b_jL \left[12d^2 + 12dh + 4d(N + j) \right] + 2b_jdV_j. \quad (22)$$

For an SID containing q generated tokens, the decoder-core cost is

$$F_q = F_{\text{cross-KV}} + \sum_{j=1}^q C_j. \quad (23)$$

After removing the final autoregressive level, it becomes

$$F_{q-1} = F_{\text{cross-KV}} + \sum_{j=1}^{q-1} C'_j, \quad (24)$$

where C'_j uses the beam count and vocabulary size of the shortened SID. The reported decoder-core reduction is $1 - F_{q-1}/F_q$. Computation shared by the two SID structures is excluded from both terms.

KV-cache movement is reported separately because memory access is not a floating-point operation. Let ρ denote the bytes per cached element. The cross-attention cache occupies $2\rho LNd$ bytes, and its logical reads across beam search are

$$T_{\text{cross},q} = 2\rho LNd \sum_{j=1}^q b_j. \quad (25)$$

The corresponding quantity for the shortened SID replaces q and b_j with its reduced decoding schedule. Self-attention cache reads scale as $2\rho Ld \sum_j b_j j$, while cache gathering after beam selection adds implementation-dependent memory traffic. These quantities are logical accesses rather than measured device-memory transactions, since kernel tiling and on-chip reuse are hardware dependent.

B OneRec Public-Data Model Configurations

Tables 7 and 8 report the exact architecture settings used for the nominal OneRec scales. These are public-data adaptations of the corresponding architectures rather than parameter-exact reproductions of the production systems.

C OneRec Results on Amazon Beauty

Tables 9 and 10 provide the exact values underlying Figure 3. Each entry is the mean $\pm$ sample standard deviation over seeds $\{7, 42, 2024\}$.

Table 7: OneRec-V1 public-data configurations. All variants use 24 experts and activate two experts per token.

Scale	d_{model}	Enc. layers	Dec. layers	Heads	Enc. FFN	Expert FFN
0.1B	512	6	6	8	2,048	256
0.2B	640	8	8	10	2,560	320
0.3B	768	8	10	12	3,072	320
0.4B	896	8	8	14	3,584	448
0.5B	1,024	8	8	16	4,096	512
0.6B	1,024	10	10	16	4,096	512
0.7B	1,152	9	9	18	4,608	576

Table 8: OneRec-V2 public-data configurations.

Scale	d_{model}	Layers	Attention heads	KV heads	FFN
0.1B	768	12	12	4	2,048
0.2B	1,024	16	16	4	2,048
0.3B	1,024	20	16	4	3,072
0.4B	1,280	18	20	4	3,072
0.5B	1,280	20	20	4	4,096
0.6B	1,280	24	20	4	4,096
0.7B	1,536	22	24	4	4,096

Table 9: OneRec-V1 results on Amazon Reviews 2014 (Beauty).

Scale	SID	Recall@5	Recall@10	NDCG@5	NDCG@10
0.1B	S3	0.0243 ± 0.0004	0.0411 ± 0.0016	0.0152 ± 0.0002	0.0206 ± 0.0006
0.1B	S2	0.0253 ± 0.0005	0.0429 ± 0.0006	0.0156 ± 0.0003	0.0212 ± 0.0004
0.2B	S3	0.0241 ± 0.0008	0.0397 ± 0.0011	0.0150 ± 0.0006	0.0200 ± 0.0007
0.2B	S2	0.0252 ± 0.0017	0.0415 ± 0.0013	0.0155 ± 0.0011	0.0208 ± 0.0010
0.3B	S3	0.0219 ± 0.0014	0.0372 ± 0.0008	0.0138 ± 0.0010	0.0187 ± 0.0009
0.3B	S2	0.0241 ± 0.0002	0.0402 ± 0.0015	0.0148 ± 0.0001	0.0199 ± 0.0007
0.4B	S3	0.0236 ± 0.0002	0.0388 ± 0.0010	0.0143 ± 0.0004	0.0192 ± 0.0004
0.4B	S2	0.0235 ± 0.0001	0.0395 ± 0.0008	0.0146 ± 0.0001	0.0197 ± 0.0001
0.5B	S3	0.0220 ± 0.0017	0.0366 ± 0.0017	0.0136 ± 0.0008	0.0183 ± 0.0008
0.5B	S2	0.0232 ± 0.0013	0.0399 ± 0.0014	0.0145 ± 0.0010	0.0199 ± 0.0011
0.6B	S3	0.0226 ± 0.0009	0.0369 ± 0.0006	0.0141 ± 0.0006	0.0187 ± 0.0005
0.6B	S2	0.0237 ± 0.0007	0.0383 ± 0.0011	0.0146 ± 0.0006	0.0193 ± 0.0007
0.7B	S3	0.0222 ± 0.0007	0.0371 ± 0.0009	0.0137 ± 0.0004	0.0185 ± 0.0002
0.7B	S2	0.0223 ± 0.0018	0.0383 ± 0.0027	0.0136 ± 0.0009	0.0187 ± 0.0011

Table 10: OneRec-V2 results on Amazon Reviews 2014 (Beauty).

Scale	SID	Recall@5	Recall@10	NDCG@5	NDCG@10
0.1B	S3	0.0308 ± 0.0013	0.0476 ± 0.0022	0.0196 ± 0.0006	0.0250 ± 0.0008
0.1B	S2	0.0315 ± 0.0014	0.0509 ± 0.0008	0.0203 ± 0.0010	0.0266 ± 0.0008
0.2B	S3	0.0280 ± 0.0003	0.0444 ± 0.0014	0.0178 ± 0.0004	0.0231 ± 0.0004
0.2B	S2	0.0301 ± 0.0014	0.0492 ± 0.0009	0.0193 ± 0.0010	0.0255 ± 0.0008
0.3B	S3	0.0272 ± 0.0007	0.0437 ± 0.0014	0.0175 ± 0.0003	0.0228 ± 0.0006
0.3B	S2	0.0316 ± 0.0008	0.0503 ± 0.0017	0.0202 ± 0.0004	0.0263 ± 0.0007
0.4B	S3	0.0280 ± 0.0009	0.0446 ± 0.0010	0.0180 ± 0.0007	0.0233 ± 0.0004
0.4B	S2	0.0310 ± 0.0008	0.0488 ± 0.0009	0.0200 ± 0.0003	0.0257 ± 0.0003
0.5B	S3	0.0268 ± 0.0010	0.0430 ± 0.0009	0.0173 ± 0.0005	0.0225 ± 0.0005
0.5B	S2	0.0299 ± 0.0016	0.0477 ± 0.0025	0.0192 ± 0.0012	0.0249 ± 0.0013
0.6B	S3	0.0287 ± 0.0030	0.0448 ± 0.0027	0.0185 ± 0.0019	0.0237 ± 0.0018
0.6B	S2	0.0296 ± 0.0015	0.0471 ± 0.0021	0.0189 ± 0.0011	0.0245 ± 0.0015
0.7B	S3	0.0278 ± 0.0017	0.0440 ± 0.0025	0.0177 ± 0.0012	0.0230 ± 0.0015
0.7B	S2	0.0277 ± 0.0012	0.0450 ± 0.0022	0.0181 ± 0.0007	0.0237 ± 0.0009

D OneRec Results on KuaiRec

Tables 11 and 12 provide the exact values underlying Figure 4. Each entry is the mean $\pm$ sample standard deviation over seeds $\{7, 42, 2024\}$.

E OneRec Fixed-Date Static–Dynamic Results on KuaiRec

Tables 13 and 14 provide the exact values underlying Figure 6. Each entry is the mean $\pm$ sample standard deviation over seeds $\{7, 42, 2024\}$.

Table 11: OneRec-V1 results on KuaiRec.

Scale	SID	Recall@5	Recall@10	NDCG@5	NDCG@10
0.1B	S3	0.0826 ± 0.0025	0.1098 ± 0.0037	0.0539 ± 0.0036	0.0628 ± 0.0040
0.1B	S2	0.0854 ± 0.0037	0.1196 ± 0.0025	0.0553 ± 0.0021	0.0664 ± 0.0017
0.1B	PV-S2	0.0846 ± 0.0047	0.1179 ± 0.0057	0.0556 ± 0.0030	0.0664 ± 0.0033
0.2B	S3	0.0733 ± 0.0057	0.1001 ± 0.0062	0.0483 ± 0.0029	0.0569 ± 0.0031
0.2B	S2	0.0735 ± 0.0122	0.1053 ± 0.0062	0.0456 ± 0.0065	0.0559 ± 0.0045
0.2B	PV-S2	0.0719 ± 0.0033	0.1039 ± 0.0033	0.0466 ± 0.0021	0.0570 ± 0.0021
0.3B	S3	0.0625 ± 0.0026	0.0909 ± 0.0037	0.0397 ± 0.0024	0.0490 ± 0.0027
0.3B	S2	0.0728 ± 0.0108	0.1040 ± 0.0100	0.0463 ± 0.0072	0.0564 ± 0.0069
0.3B	PV-S2	0.0754 ± 0.0072	0.1051 ± 0.0068	0.0470 ± 0.0057	0.0567 ± 0.0056
0.4B	S3	0.0687 ± 0.0063	0.0942 ± 0.0053	0.0429 ± 0.0047	0.0512 ± 0.0044
0.4B	S2	0.0704 ± 0.0076	0.1055 ± 0.0071	0.0433 ± 0.0042	0.0548 ± 0.0040
0.4B	PV-S2	0.0746 ± 0.0043	0.1056 ± 0.0038	0.0465 ± 0.0027	0.0566 ± 0.0025
0.5B	S3	0.0654 ± 0.0116	0.0962 ± 0.0116	0.0421 ± 0.0091	0.0521 ± 0.0091
0.5B	S2	0.0696 ± 0.0047	0.1027 ± 0.0070	0.0442 ± 0.0030	0.0549 ± 0.0039
0.5B	PV-S2	0.0687 ± 0.0017	0.1009 ± 0.0024	0.0439 ± 0.0005	0.0543 ± 0.0005
0.6B	S3	0.0662 ± 0.0057	0.0916 ± 0.0037	0.0422 ± 0.0045	0.0505 ± 0.0037
0.6B	S2	0.0675 ± 0.0044	0.1024 ± 0.0057	0.0418 ± 0.0031	0.0531 ± 0.0034
0.6B	PV-S2	0.0668 ± 0.0033	0.0997 ± 0.0021	0.0415 ± 0.0022	0.0521 ± 0.0017
0.7B	S3	0.0652 ± 0.0029	0.0926 ± 0.0065	0.0412 ± 0.0017	0.0502 ± 0.0030
0.7B	S2	0.0641 ± 0.0047	0.0948 ± 0.0046	0.0404 ± 0.0032	0.0503 ± 0.0031
0.7B	PV-S2	0.0593 ± 0.0012	0.0889 ± 0.0009	0.0380 ± 0.0026	0.0476 ± 0.0025

Table 12: OneRec-V2 results on KuaiRec.

Scale	SID	Recall@5	Recall@10	NDCG@5	NDCG@10
0.1B	S3	0.0931 ± 0.0024	0.1244 ± 0.0020	0.0656 ± 0.0013	0.0757 ± 0.0013
0.1B	S2	0.1014 ± 0.0067	0.1351 ± 0.0058	0.0713 ± 0.0049	0.0823 ± 0.0046
0.1B	PV-S2	0.0977 ± 0.0055	0.1335 ± 0.0058	0.0692 ± 0.0038	0.0808 ± 0.0040
0.2B	S3	0.0964 ± 0.0036	0.1271 ± 0.0025	0.0663 ± 0.0037	0.0762 ± 0.0029
0.2B	S2	0.0968 ± 0.0069	0.1342 ± 0.0056	0.0659 ± 0.0020	0.0781 ± 0.0017
0.2B	PV-S2	0.0966 ± 0.0036	0.1322 ± 0.0020	0.0676 ± 0.0032	0.0791 ± 0.0027
0.3B	S3	0.0935 ± 0.0068	0.1234 ± 0.0036	0.0654 ± 0.0060	0.0751 ± 0.0049
0.3B	S2	0.0956 ± 0.0019	0.1307 ± 0.0019	0.0647 ± 0.0006	0.0761 ± 0.0008
0.3B	PV-S2	0.0949 ± 0.0055	0.1277 ± 0.0063	0.0654 ± 0.0029	0.0761 ± 0.0032
0.4B	S3	0.0896 ± 0.0047	0.1169 ± 0.0015	0.0610 ± 0.0048	0.0699 ± 0.0037
0.4B	S2	0.0955 ± 0.0030	0.1322 ± 0.0010	0.0652 ± 0.0026	0.0771 ± 0.0014
0.4B	PV-S2	0.0924 ± 0.0085	0.1272 ± 0.0039	0.0637 ± 0.0047	0.0750 ± 0.0032
0.5B	S3	0.0893 ± 0.0070	0.1197 ± 0.0053	0.0612 ± 0.0049	0.0711 ± 0.0044
0.5B	S2	0.0985 ± 0.0054	0.1338 ± 0.0049	0.0664 ± 0.0057	0.0779 ± 0.0055
0.5B	PV-S2	0.0950 ± 0.0056	0.1289 ± 0.0028	0.0654 ± 0.0046	0.0764 ± 0.0035
0.6B	S3	0.0965 ± 0.0010	0.1250 ± 0.0042	0.0692 ± 0.0015	0.0784 ± 0.0002
0.6B	S2	0.0955 ± 0.0047	0.1269 ± 0.0059	0.0651 ± 0.0034	0.0752 ± 0.0038
0.6B	PV-S2	0.0975 ± 0.0034	0.1370 ± 0.0027	0.0674 ± 0.0036	0.0802 ± 0.0027
0.7B	S3	0.0916 ± 0.0039	0.1206 ± 0.0073	0.0641 ± 0.0040	0.0735 ± 0.0048
0.7B	S2	0.0897 ± 0.0030	0.1249 ± 0.0010	0.0614 ± 0.0024	0.0729 ± 0.0019
0.7B	PV-S2	0.0960 ± 0.0027	0.1299 ± 0.0037	0.0662 ± 0.0037	0.0771 ± 0.0039

Table 13: OneRec-V1 results in the fixed-date KuaiRec comparison.

Scale	Codebook	Recall@5	Recall@10	NDCG@5	NDCG@10
0.1B	Static C1	0.1268 ± 0.0076	0.1467 ± 0.0082	0.0972 ± 0.0089	0.1046 ± 0.0106
0.1B	Dynamic C2	0.1328 ± 0.0104	0.1567 ± 0.0084	0.1028 ± 0.0056	0.1115 ± 0.0044
0.2B	Static C1	0.1138 ± 0.0170	0.1388 ± 0.0118	0.0832 ± 0.0208	0.0913 ± 0.0189
0.2B	Dynamic C2	0.1102 ± 0.0206	0.1374 ± 0.0197	0.0832 ± 0.0197	0.0921 ± 0.0194
0.3B	Static C1	0.1198 ± 0.0069	0.1445 ± 0.0053	0.0875 ± 0.0078	0.0956 ± 0.0074
0.3B	Dynamic C2	0.1212 ± 0.0111	0.1515 ± 0.0128	0.0959 ± 0.0126	0.1061 ± 0.0132
0.4B	Static C1	0.0994 ± 0.0075	0.1300 ± 0.0097	0.0707 ± 0.0065	0.0807 ± 0.0073
0.4B	Dynamic C2	0.1003 ± 0.0124	0.1264 ± 0.0137	0.0791 ± 0.0029	0.0875 ± 0.0029
0.5B	Static C1	0.1088 ± 0.0126	0.1377 ± 0.0182	0.0685 ± 0.0196	0.0790 ± 0.0200
0.5B	Dynamic C2	0.1049 ± 0.0077	0.1359 ± 0.0125	0.0721 ± 0.0047	0.0822 ± 0.0063
0.6B	Static C1	0.1024 ± 0.0202	0.1317 ± 0.0130	0.0621 ± 0.0090	0.0717 ± 0.0064
0.6B	Dynamic C2	0.1092 ± 0.0186	0.1355 ± 0.0135	0.0746 ± 0.0038	0.0837 ± 0.0035
0.7B	Static C1	0.0918 ± 0.0156	0.1199 ± 0.0137	0.0539 ± 0.0161	0.0641 ± 0.0144
0.7B	Dynamic C2	0.0937 ± 0.0173	0.1194 ± 0.0220	0.0578 ± 0.0130	0.0653 ± 0.0143

Table 14: OneRec-V2 results in the fixed-date KuaiRec comparison.

Scale	Codebook	Recall@5	Recall@10	NDCG@5	NDCG@10
0.1B	Static C1	0.1406 ± 0.0284	0.1718 ± 0.0334	0.1095 ± 0.0235	0.1197 ± 0.0249
0.1B	Dynamic C2	0.1432 ± 0.0066	0.1804 ± 0.0066	0.1078 ± 0.0080	0.1200 ± 0.0077
0.2B	Static C1	0.1366 ± 0.0131	0.1659 ± 0.0099	0.1090 ± 0.0131	0.1186 ± 0.0122
0.2B	Dynamic C2	0.1386 ± 0.0176	0.1728 ± 0.0140	0.1067 ± 0.0160	0.1179 ± 0.0149
0.3B	Static C1	0.1391 ± 0.0061	0.1684 ± 0.0060	0.1072 ± 0.0123	0.1168 ± 0.0124
0.3B	Dynamic C2	0.1465 ± 0.0074	0.1838 ± 0.0140	0.1137 ± 0.0026	0.1259 ± 0.0050
0.4B	Static C1	0.1404 ± 0.0148	0.1762 ± 0.0160	0.1010 ± 0.0147	0.1127 ± 0.0148
0.4B	Dynamic C2	0.1424 ± 0.0276	0.1680 ± 0.0401	0.1149 ± 0.0265	0.1254 ± 0.0269
0.5B	Static C1	0.1260 ± 0.0141	0.1638 ± 0.0214	0.0928 ± 0.0079	0.1051 ± 0.0102
0.5B	Dynamic C2	0.1272 ± 0.0042	0.1610 ± 0.0037	0.0921 ± 0.0022	0.1042 ± 0.0032
0.6B	Static C1	0.1248 ± 0.0085	0.1546 ± 0.0124	0.0934 ± 0.0070	0.1032 ± 0.0083
0.6B	Dynamic C2	0.1217 ± 0.0074	0.1508 ± 0.0065	0.0886 ± 0.0062	0.0992 ± 0.0071
0.7B	Static C1	0.1299 ± 0.0078	0.1605 ± 0.0114	0.0981 ± 0.0059	0.1073 ± 0.0052
0.7B	Dynamic C2	0.1364 ± 0.0065	0.1759 ± 0.0130	0.0970 ± 0.0087	0.1117 ± 0.0030