Semantics Meet Signals: Dual Codebook Representationl Learning for Generative Recommendation

Zheng Hui[1], **Xiaokai Wei**[2], **Reza Shirkavand**[3], **Chen Wang**[2],
Weizhi Zhang[4], **Alejandro Peláez**[2], **Michelle Gong**[2]
[1]University of Cambridge [2]Roblox Corporation
[3]University of Maryland [4]University of Illinois Chicago
zh2483@columbia.edu {xwei, cwang, apelaez, mgong}@roblox.com
rezashkv@cs.umd.edu, wzhan42@uic.edu

Abstract

Generative recommendation has recently emerged as a powerful paradigm that unifies retrieval and generation, representing items as discrete semantic tokens and enabling flexible sequence modeling with autoregressive models. Despite its success, existing approaches rely on a single, uniform codebook to encode all items, overlooking the inherent imbalance between popular items rich in collaborative signals and long-tail items that depend on semantic understanding. We argue that this uniform treatment limits representational efficiency and hinders generalization. To address this, we introduce FlexCode, a popularity-aware framework that adaptively allocates a fixed token budget between a collaborative filtering (CF) codebook and a semantic codebook. A lightweight MoE dynamically balances CF-specific precision and semantic generalization, while an alignment and smoothness objective maintains coherence across the popularity spectrum. We perform experiments on both public and industrial-scale datasets, showing that FlexCode consistently outperform strong baselines. FlexCode provides a new mechanism for token representation in generative recommenders, achieving stronger accuracy and tail robustness, and offering a new perspective on balancing memorization and generalization in token-based recommendation models.

1 Introduction

Recommender systems are a central component of modern information platforms, enabling personalized ranking and retrieval in e-commerce, streaming media, and social feeds (Gomez-Uribe and Hunt, 2016; Wang et al., 2025; Hui et al., 2025). Traditional recommendation methods (Koren et al., 2009; He et al., 2017; Sun et al., 2019; Geng et al., 2022) learn latent representations from observed user–item interactions. These models are effective for frequently interacted (head) items because they are able to memorize fine-grained co-occurrence structure. However, they exhibit systematic degradation on long-tail and cold-start items, where interaction data is sparse (Zhang et al., 2021; Liu et al., 2023). This reflects an inherent memorization bias: collaborative filtering captures relational specificity but lacks semantic generalization. Semantic-ID based Generative recommendation (Rajput et al., 2023; Wang et al., 2024) has recently emerged as a compelling alternative paradigm. Rather than treating recommendation as score prediction over fixed item IDs, generative recommenders map items into discrete semantic tokens using vector quantization or learned codebooks, and then train an autoregressive model to generate future items in sequence. This formulation offers several advantages. It unifies recommendation with sequence generation, making it compatible with large language models and multilingual or multimodal supervision. It also supports better generalization to unseen items, since tokenization can draw on textual, visual, or metadata semantics rather than relying solely on collaborative history (Lv et al., 2024).

Despite these benefits, current generative recommenders suffer from two fundamental limitations that prevent them from fully replacing traditional CF systems in practice. The first limitation is representation entanglement (Shirkavand et al., 2025; Fang et al., 2025). Most existing approaches adopt a single, shared codebook for all items, and attempt to embed both semantic content information and collaborative interaction information into the same quantized space. This forces the model to compress heterogeneous factors of variation into a single representation. As a consequence, the learned item tokens are often neither semantically pure nor collaboration-accurate. The same codeword is expected to simultaneously reflect textual or visual meaning and encode high-order co-consumption structure. This coupling leads to interference. For head items, the semantic signal can dilute highly

localized collaborative patterns. For tail items, the collaborative signal can dominate even when it is statistically unreliable, which harms cold-start behavior. In short, entanglement in a single codebook induces representational collapse: neither factor is modeled optimally (Baykal et al., 2024). The second limitation is static capacity allocation (Zhang et al., 2024). Existing semantic identifier and unified ID-plus-semantic methods allocate a fixed representational budget to each item. Every item is assigned the same number and type of codebook tokens, regardless of its popularity and data regime. This is in tension with the long-tailed nature of recommender data (Klimashevskaia et al., 2024). Head items have abundant interaction data and therefore benefit from high-capacity collaborative representations. Tail items, on the other hand, lack interaction support and rely more strongly on semantic evidence such as text descriptions, visual attributes, or structured metadata. Treating these two classes of items identically wastes capacity in one regime and starves it in the other. In particular, it leads to overfitting on the head while under-representing the tail.

We view these two limitations as a more general problem in generative recommendation: adaptive capacity allocation. Given a fixed token budget per item, how should a system decide how much of that capacity to devote to collaborative specificity versus semantic generalization, and how can it avoid destructive interference between the two? To address these challenges, we propose **FlexCode**, a popularity-aware generative recommendation framework that explicitly disentangles collaborative and semantic channels and dynamically allocates representational capacity between them.

FlexCode introduces two separate codebooks: a collaborative codebook $\mathbf{C}_{\mathrm{CF}}$ that captures high-order interaction structure, and a semantic codebook $\mathbf{C}_{\mathrm{SEM}}$ that captures modality-grounded meaning. Our framework uses a lightweight mixture-of-experts (MoE) gating mechanism that conditions on item statistics such as popularity or interaction sparsity, and selects how much capacity to allocate to each codebook under a fixed overall token budget. Intuitively, head items are routed toward collaborative experts and receive more tokens from $\mathbf{C}_{\mathrm{CF}}$, while tail items are routed toward semantic experts and receive more tokens from $\mathbf{C}_{\mathrm{SEM}}$. In addition to disentangling representation sources and adapting capacity, **FlexCode** introduces two regularization objectives to ensure stability and coherence. Together, these components allow **FlexCode** to interpolate smoothly between collaborative-style and semantic-style item encodings, rather than forcing a hard dichotomy. Conceptually, this approach operationalizes the intuition that memorization and generalization are not mutually exclusive, but should be emphasized differently for different regions of the item distribution.

We evaluate **FlexCode** on both publicly available benchmarks and large-scale industrial datasets. Across all settings, our method outperforms Item-ID based, Semantic-ID and unified representation baselines on standard ranking metrics. Our main contributions are as follows:

- We identify two structural limitations of existing generative recommenders: representation entanglement in a single shared codebook and static capacity allocation that ignores item popularity and data sparsity. We formalize these limitations as instances of an adaptive capacity allocation problem.
- We propose **FlexCode**, a framework that factorizes item representations into a collaborative codebook and a semantic codebook, and uses a popularity-aware mixture-of-experts gate to allocate a fixed token budget between them on a per-item basis.
- We demonstrate consistent improvements over recent Semantic-ID and unified-ID baselines on both public and industrial datasets, and show that **FlexCode** yields better head–tail balance and superior robustness under tight token budgets.

2 Background and Preliminaries

2.1 Generative Recommendation Framework

Let $\mathcal{U}$ denote the set of users and $\mathcal{I}$ the set of items. For each user $u \in \mathcal{U}$, we observe an interaction sequence

$$\mathbf{s}_u = [i_1, i_2, \ldots, i_T],$$

where $i_t \in \mathcal{I}$ represents the t-th item interacted with by user u. The goal of recommendation is to predict the next item i_{T+1} given the prefix sequence $\mathbf{s}_u$.

In *generative recommendation*, this problem is reformulated as a conditional sequence generation task. Each item i_t is first mapped into one or more

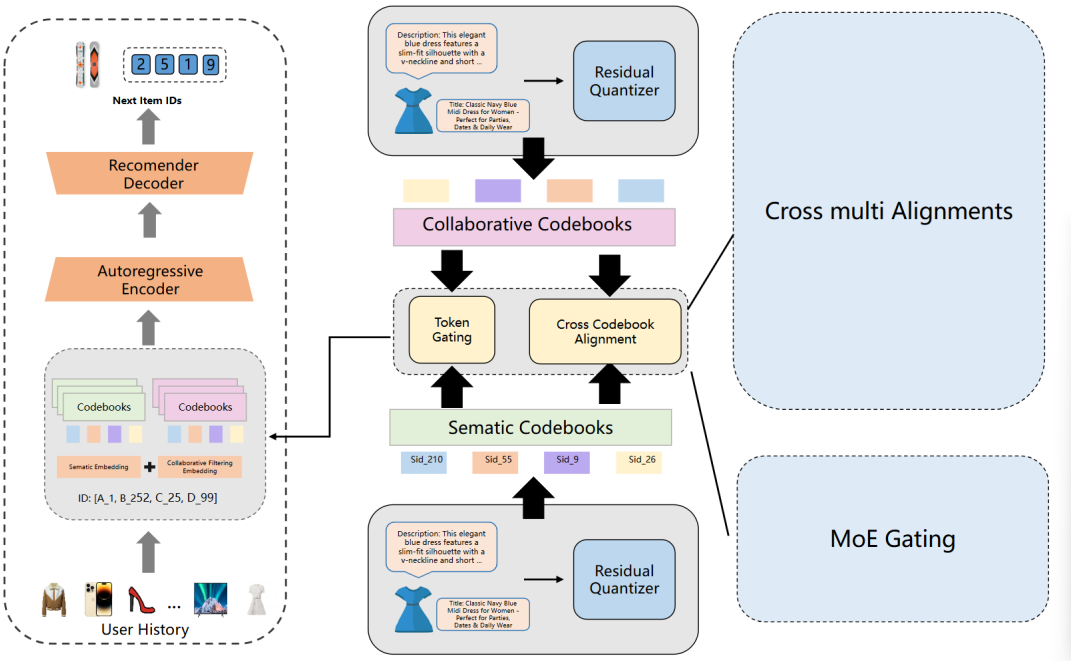


Figure 1: Overview of the **FlexCode** framework for generative recommendation. Each item is encoded by a dual codebook with collaborative and semantic codebooks, aligned via a cross-codebook contrastive objective. A popularity-aware Mixture-of-Experts (MoE) router adaptively allocates the budget between codebooks, and an autoregressive Transformer is trained on the resulting sequences to generate items.

discrete tokens through a vector-quantized codebook,

$$\mathbf{z}_{i_t} = [z_{i_t}^1, z_{i_t}^2, \ldots, z_{i_t}^L],$$

where each $z_{i_t}^l \in \{1, \ldots, K\}$ indexes one entry in a learnable codebook $\mathbf{C} \in \mathbb{R}^{K \times d}$, L is the number of tokens per item, and d is the embedding dimension. An autoregressive model p_θ (e.g., a Transformer decoder) is then trained to maximize the likelihood of the item sequence in tokenized form:

$$\mathcal{L}_{\text{gen}} = -\sum_{u \in \mathcal{U}} \sum_{t=1}^{T} \log p_\theta(\mathbf{z}_{i_t} \mid \mathbf{z}_{i_{<t}}).$$

This formulation allows recommendation to be cast as a language modeling problem, enabling the use of large pre-trained generative models.

2.2 Collaborative and Semantic Representations

Traditional collaborative filtering (CF) models learn an embedding matrix $\mathbf{E}_{\text{CF}} \in \mathbb{R}^{|\mathcal{I}| \times d}$ directly from user–item interactions. Given user embedding $\mathbf{e}_u$ and item embedding $\mathbf{e}_i$, the relevance score is typically computed as

$$r(u, i) = \mathbf{e}_u^\top \mathbf{e}_i,$$

and optimized with implicit feedback losses such as Bayesian Personalized Ranking (BPR).

By contrast, semantic encoders (e.g., textual or visual models) map content features $\mathbf{x}_i$ into a representation

$$\mathbf{e}_{\text{sem}}(i) = f_{\text{sem}}(\mathbf{x}_i),$$

capturing modality-derived meaning independent of user interactions. While semantic representations generalize better to cold-start items, they lack collaborative precision.

Generative recommenders based on semantic identifiers (Semantic-IDs) (Rajput et al., 2023) quantize these embeddings into discrete tokens via vector quantization:

$$\hat{z}_i = \arg\min_k \|\mathbf{e}_{\text{sem}}(i) - \mathbf{C}_k\|_2^2,$$

producing token indices that are used for sequence generation. However, existing methods employ a single codebook $\mathbf{C}$ for all items, mixing collaborative and semantic signals implicitly. This *uniform codebook assumption* limits representational flexibility and ignores item popularity heterogeneity.

2.3 Problem Formulation: Capacity Allocation

Let $\mathbf{C}_{\text{CF}}$ and $\mathbf{C}_{\text{SEM}}$ denote the collaborative and semantic codebooks, respectively. Given a fixed

total token budget L, we define a function

$$L_{\text{CF}}(i) + L_{\text{SEM}}(i) = L,$$

where $L_{\text{CF}}(i)$ and $L_{\text{SEM}}(i)$ determine how many tokens of item i are drawn from each codebook. The objective of capacity allocation is to learn an adaptive mapping

$$g(i) = \sigma(w \cdot \log(\text{pop}(i)) + b),$$

where $\text{pop}(i)$ is the empirical popularity (interaction frequency) of item i, and $g(i) \in [0, 1]$ determines the proportion of CF vs. semantic tokens. Accordingly,

$$L_{\text{CF}}(i) = \lfloor g(i) \cdot L \rfloor, \quad L_{\text{SEM}}(i) = L - L_{\text{CF}}(i).$$

For head items (high $\text{pop}(i)$), $g(i)$ approaches 1, allocating more CF tokens; for tail items, $g(i)$ decreases, emphasizing semantic tokens. This defines an *adaptive dual-codebook encoding* that allocates capacity according to item popularity.

The resulting generation objective combines both token sources:

$$\mathcal{L}_{\text{gen}}^{\text{dual}} = -\sum_{u \in \mathcal{U}} \sum_{t=1}^{T} \log p_\theta\left([\mathbf{z}_{i_t}^{\text{CF}}, \mathbf{z}_{i_t}^{\text{SEM}}] \mid \mathbf{z}_{i_{<t}}\right).$$

This formulation generalizes both pure CF-based tokenization (when $g(i) = 1$) and pure semantic tokenization (when $g(i) = 0$), and lays the foundation for the proposed *Dynamic Dual-Codebook Learning* framework.

3 Methodology

In this section, we elaborate on the proposed **FlexCode** framework. Our methodology addresses the challenges of representation entanglement and static capacity allocation in existing generative recommendation systems by introducing a novel dual codebook architecture, popularity-aware token allocation, and a robust autoregressive generation scheme. The overall framework is depicted in Figure 1.

3.1 Dual Codebook Construction

To address the representation entanglement issue, **FlexCode** introduces two specialized codebooks for each item: a Semantic Codebook (SC) for capturing modality-specific meaning and a Collaborative Codebook (CC) for encoding high-order interaction patterns.

3.1.1 Semantic Codebook Learning (SCL)

The Semantic Codebook aims to distill the inherent meaning of an item from its associated textual and categorical metadata. For each item i, we first concatenate its descriptive attributes such as `brand`, `category`, `price`, and `title` into a unified text string. This string is then fed into a powerful pre-trained text embedding model to obtain a dense semantic embedding $\mathbf{e}_i^{\text{sem}} \in \mathbb{R}^{d_{\text{sem}}}$.

To convert these continuous semantic embeddings into a discrete codebook, we employ a Residual Quantization Variational Autoencoder (RQ-VAE) (Lee et al., 2022). For each $\mathbf{e}_i^{\text{sem}}$, the RQ-VAE encodes it into a sequence of L_{sem} discrete code vectors, denoted as $\mathbf{c}_i^{\text{sem}} = [\mathbf{c}_{i,1}^{\text{sem}}, \ldots, \mathbf{c}_{i,L_{\text{sem}}}^{\text{sem}}]$, where each $\mathbf{c}_{i,j}^{\text{sem}}$ is an index to an entry in a shared semantic codebook $\mathcal{V}_{\text{sem}}$. The total reconstruction loss for the semantic codebook is defined as:

$$\mathcal{L}_{\text{SCL}} = \mathbb{E}_{\mathbf{e}_i^{\text{sem}} \sim D_{\text{sem}}} \left[\|\mathbf{e}_i^{\text{sem}} - \text{Decode}(\mathbf{c}_i^{\text{sem}})\|_2^2 \right] + \sum_{j=1}^{L_{\text{sem}}} \mathcal{L}_{\text{vq}}\left(\mathbf{z}_{i,j-1}^{\text{sem}}, \mathbf{q}_{i,j}^{\text{sem}}\right) \quad (1)$$

where $\text{Decode}(\cdot)$ reconstructs the embedding from its quantized representation, and $\mathcal{L}_{\text{vq}}$ is the standard VQ-VAE loss. The output of this stage for item i is its semantic codebook $\mathbf{C}_i^{\text{SEM}}$.

3.1.2 Collaborative Codebook Learning (CCL)

The Collaborative Codebook is designed to capture high-order co-purchase or co-view patterns. To obtain a rich collaborative embedding, we use a SASRec-like architecture (Kang and McAuley, 2018) to derive a context-aware collaborative embedding $\mathbf{e}_i^{\text{col}} \in \mathbb{R}^{d_{\text{col}}}$ for each item i. Similar to the semantic codebook, we apply an RQ-VAE to discretize these collaborative embeddings into a sequence of L_{col} discrete code vectors, $\mathbf{c}_i^{\text{col}}$, from a collaborative codebook $\mathcal{V}_{\text{col}}$. The loss function $\mathcal{L}_{\text{CCL}}$ is defined analogously to $\mathcal{L}_{\text{SCL}}$:

$$\mathcal{L}_{\text{CCL}} = \mathbb{E}_{\mathbf{e}_i^{\text{col}} \sim D_{\text{col}}} \left[\left\| \mathbf{e}_i^{\text{col}} - \text{Decode}\left(\mathbf{c}_i^{\text{col}}\right) \right\|_2^2 \right] + \sum_{j=1}^{L_{\text{col}}} \mathcal{L}_{\text{vq}}\left(\mathbf{z}_{i,j-1}^{\text{col}}, \mathbf{q}_{i,j}^{\text{col}}\right) \quad (2)$$

The output for item i is its collaborative codebook $\mathbf{C}_i^{\text{COL}}$.

3.2 Cross-Codebook Alignment (CCA)

While the two codebooks capture different aspects of an item, they must not drift into unrelated representation spaces. To ensure coherence, we introduce a Cross-Codebook Alignment (CCA) objective. This objective operates on the *reconstructed* embeddings, $\tilde{\mathbf{e}}_i^{\text{sem}} = \text{Decode}(\mathbf{C}_i^{\text{SEM}})$ and $\tilde{\mathbf{e}}_i^{\text{col}} = \text{Decode}(\mathbf{C}_i^{\text{COL}})$. Aligning the reconstructed vectors ensures that the alignment loss acts on the information that is actually preserved by the codebooks, forcing them to learn quantized representations that are not only aligned but also decodable into a coherent shared space.

Let $P_{\text{sem}}(\cdot)$ and $P_{\text{col}}(\cdot)$ be projection heads that map these reconstructed embeddings into a shared latent space. The alignment loss $\mathcal{L}_{\text{CCA}}$ is defined using an InfoNCE objective:

$$\mathcal{L}_{\text{CCA}} = -\log \frac{\exp\big(\text{sim}(P_{\text{sem}}(\tilde{\mathbf{e}}_i^{\text{sem}}), P_{\text{col}}(\tilde{\mathbf{e}}_i^{\text{col}}))/\tau\big)}{\sum_{j\in\mathcal{I}} \exp\big(\text{sim}(P_{\text{sem}}(\tilde{\mathbf{e}}_i^{\text{sem}}), P_{\text{col}}(\tilde{\mathbf{e}}_j^{\text{col}}))/\tau\big)} \tag{3}$$

where $\text{sim}(\cdot,\cdot)$ is cosine similarity and τ is a temperature parameter. This loss pulls representations of the same item together while pushing others apart. Furthermore, this contrastive pressure serves as a powerful regularizer, encouraging more uniform usage of codebook entries and mitigating the common issue of codebook collapse in VQ-based models.

3.3 Popularity-Aware Token Allocation (PATA)

To adaptively allocate a fixed token budget L_{total} between the codebooks, we introduce a lightweight Mixture-of-Experts (MoE) router. The goal is to allocate more collaborative capacity to popular (head) items and more semantic capacity to sparse (tail) items.

For each item i, we construct a feature vector $\mathbf{x}_i = [\log(1+f_i), \text{age}_i, \text{sparsity}_i, \text{uncertainty}_i]$, where f_i is the normalized interaction frequency, age_i is the time since the item was introduced, sparsity_i is the inverse of its interaction density, and uncertainty_i is the variance of its embedding during training. While a simple sigmoid on popularity could provide a monotonic allocation, our refined MLP-based router $\phi(\cdot)$ captures more complex, non-linear interactions between these features, allowing for more nuanced allocation decisions.

The router, a shallow MLP, processes $\mathbf{x}_i$ to produce logits $[z_i^{\text{col}}, z_i^{\text{sem}}]$. A temperature-scaled softmax then yields the routing probabilities $\boldsymbol{\pi}_i$, from which we define the collaborative allocation ratio $\alpha_i = \pi_i^{\text{col}}$. This ratio determines the soft number of tokens for each codebook:

$$\bar{L}_i^{\text{col}} = \alpha_i L_{\text{total}}, \quad \bar{L}_i^{\text{sem}} = (1-\alpha_i) L_{\text{total}}. \tag{4}$$

To maintain differentiability, we use sigmoid-based masks to softly select tokens during training:

$$m_k^{\text{col}}(\alpha_i) = \sigma\left(\frac{\bar{L}_i^{\text{col}} - (k-\frac{1}{2})}{\tau_m}\right), \tag{5}$$

$$m_k^{\text{sem}}(\alpha_i) = \sigma\left(\frac{\bar{L}_i^{\text{sem}} - (k-\frac{1}{2})}{\tau_m}\right), \tag{6}$$

where the hyperparameter τ_m controls the smoothness of the soft-to-hard transition. We set $\tau_m = 0.1$ and found the model to be robust to small variations of this value. The final combined representation $\mathbf{C}_i$ concatenates the masked codebooks. At inference, the allocation is discretized via rounding. To ensure router stability, we use stratified load-balancing ($\mathcal{L}_{\text{lb}}$) within popularity bands and a local smoothness regularizer ($\mathcal{L}_{\text{smooth}}$) on the allocation ratios α_i.

3.4 Autoregressive Generation (ARG)

Given a user's historical sequence of items $S_u = (i_1, \ldots, i_{T-1})$, we retrieve their corresponding combined codebooks $(\mathbf{C}_{i_1}, \ldots, \mathbf{C}_{i_{T-1}})$. An autoregressive Transformer model is trained to maximize the likelihood of the next item's codebook $\mathbf{C}_{i_T}$. The loss is the standard cross-entropy over the sequence of predicted tokens:

$$\mathcal{L}_{\text{ARG}} = -\sum_{t=1}^{N_{\text{seq}}-1} \sum_{k=1}^{L_{\text{total}}} \log P(\mathbf{c}_{i_{t+1},k} \,|\, \mathbf{C}_{S_u^{<t+1}}, \mathbf{c}_{i_{t+1},<k}) \tag{7}$$

where $\mathbf{C}_{S_u^{<t+1}}$ represents the context from previous items and $\mathbf{c}_{i_{t+1},<k}$ represents previously generated tokens for the current target item.

3.5 Overall Objective and Training

The entire **FlexCode** framework is trained jointly in an end-to-end manner. The final objective function is a weighted sum of all components: the codebook reconstruction losses, the cross-codebook alignment loss, the autoregressive generation loss, and the routing regularization losses.

$$\begin{aligned}\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{SCL}} + \mathcal{L}_{\text{CCL}} + \lambda_{\text{CCA}}\mathcal{L}_{\text{CCA}} \\ + \lambda_{\text{ARG}}\mathcal{L}_{\text{ARG}} + \lambda_{\text{lb}}\mathcal{L}_{\text{lb}} + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}}\end{aligned} \tag{8}$$

where $\lambda_{(\cdot)}$ are hyperparameters that balance the contribution of each term. This end-to-end training allows the codebooks, the router, and the autoregressive model to co-adapt, leading to a more effective and cohesive final system.

4 Experiments

In this section, we empirically demonstrate the effectiveness and efficiency of the proposed method **FlexCode**.

4.1 Experimental Setup

4.1.1 Datasets

To ensure a comprehensive evaluation of our framework across different domains and data characteristics, we conduct experiments on three widely used public benchmarks and a large-scale proprietary industrial dataset. Following prior work (He et al., 2017; Sun et al., 2019; Zhou et al., 2020), we treat each user's chronological interaction history as an ordered sequence. We adopt the standard *leave-last-out* evaluation protocol (Kang and McAuley, 2018), using the last item for testing and the second-to-last for validation. Following common practice, we apply the 5-core filtering strategy.

Public Benchmarks We evaluate **FlexCode** on three public datasets: **Amazon-Beauty** (McAuley et al., 2015), a dense, medium-scale e-commerce dataset; **Amazon-Sports and Outdoors** (McAuley et al., 2015), a larger and sparser e-commerce dataset; and **KuaiRand-1K** (Gao et al., 2022), a large-scale short-video recommendation dataset with rich side information. Detailed statistics are in Table 2.

Proprietary Industrial Dataset To further examine the effectiveness of FlexCode under production-scale conditions, we additionally train and evaluate our model on an in-house industrial dataset collected from a commercial platform. The dataset contains tens of millions of users and their interaction sequences over tens of thousands of items, recorded over a period exceeding one year. We use the last day of logged interactions for evaluation to reflect an online inference scenario. Due to confidentiality agreements, we report only approximate, rounded statistics in Table 2.

4.1.2 Baselines

Item ID-based methods We include widely used sequence models that learn item representations directly from user–item interaction histories: Caser (Tang and Wang, 2018), GRU4Rec (Hidasi et al., 2015), HGN (Ma et al., 2019), BERT4Rec (Sun et al., 2019), SASRec (Kang and McAuley, 2018), Recformer (Li et al., 2023), and S3-Rec (Zhou et al., 2020). **Semantic ID-based methods** We further compare with recent approaches that tokenize or quantize item representations into discrete semantic identifiers: VQRec (Hou et al., 2023), TIGER (Rajput et al., 2023), LC-Rec (Zheng et al., 2024), COBRA (Yang et al., 2025) and Unified Representation Learning (Lin et al., 2025).

4.1.3 Evaluation Metrics

We assess model performance using **Recall@K** and **NDCG@K**, with $K \in \{5, 10\}$, following prior work (Rajput et al., 2023).

4.2 Main Results on Public Benchmarks

Table 1 presents the overall performance comparison on the three public benchmarks, where **FlexCode** consistently outperforms all baseline models across all metrics. The advantage is most pronounced on the larger and sparser datasets. On Amazon-Sports, **FlexCode** achieves a Recall@10 of 0.0471, a 5.3% relative improvement over the strongest semantic baseline, URL. This performance gap widens on the large-scale KuaiRand dataset, where **FlexCode** attains an NDCG@10 of 0.0632, representing an 8.0% improvement over URL and demonstrating our model's robustness to diverse data characteristics. This consistent superiority stems from its architecture, which directly addresses the limitations of prior methods. Unlike Item ID-based models such as SASRec, which lack a mechanism for semantic generalization on sparse data, **FlexCode** utilizes a dedicated semantic codebook. At the same time, it avoids the representation entanglement common in single-codebook generative models like URL by explicitly disentangling collaborative and semantic signals into separate codebooks.

The value of this design is further substantiated by the performance of our model's ablated variants. The **FlexCode-Fix** version, which uses a static 50/50 token split, already outperforms most baselines, confirming the inherent benefit of the disentangled dual-codebook structure. However, the full **FlexCode** model consistently outperforms **FlexCode-Fix**, isolating the critical contribution of the popularity-aware token allocation (PATA)

Table 1: Overall comparison of **FlexCode** and baseline models on three datasets. Best results are in **bold**, and second-best are underlined. All improvements are statistically significant with $p < 0.01$.

Model	Beauty				Sports and Outdoors				KuaiRand			
	R@5	N@5	R@10	N@10	R@5	N@5	R@10	N@10	R@5	N@5	R@10	N@10
Item ID-based methods												
Caser	0.0205	0.0131	0.0347	0.0176	0.0116	0.0072	0.0194	0.0097	0.0074	0.0068	0.0118	0.0095
GRU4Rec	0.0164	0.0099	0.0283	0.0137	0.0129	0.0086	0.0204	0.0110	0.0298	0.0217	0.0383	0.0245
HGN	0.0325	0.0206	0.0512	0.0266	0.0189	0.0120	0.0313	0.0159	0.0297	0.0169	0.0354	0.0219
BERT4Rec	0.0203	0.0124	0.0347	0.0170	0.0115	0.0075	0.0191	0.0099	0.0185	0.0196	0.0217	0.0236
SASRec	0.0387	0.0249	0.0605	0.0318	0.0233	0.0154	0.0350	0.0192	0.0332	0.0338	0.0405	0.0372
S^3-Rec	0.0387	0.0244	0.0647	0.0327	0.0251	0.0161	0.0385	0.0204	–	–	–	–
Recformer	0.0379	0.0257	0.0589	0.0321	0.0249	0.0154	0.0370	0.0201	–	–	–	–
Semantic ID-based methods												
VQ-Rec	0.0457	0.0317	0.0664	0.0383	0.0208	0.0144	0.0300	0.0173	0.0513	0.0354	0.0589	0.0412
TIGER	0.0454	0.0321	0.0648	0.0384	0.0264	0.0181	0.0400	0.0225	0.0557	0.0383	0.0624	0.0445
LC-Rec	0.0478	0.0329	0.0679	0.0389	0.0268	0.0177	0.0412	0.0221	0.0622	0.0403	0.0684	0.0497
COBRA	0.0537	0.0395	0.0725	0.0456	0.0306	0.0215	0.0434	0.0257	–	–	–	–
URL	0.0553	0.0410	0.0736	0.0471	0.0305	0.0218	0.0449	0.0273	0.0654	0.0481	0.0778	0.0585
FlexCode-SID only	0.0510	0.0375	0.0689	0.0433	0.0291	0.0204	0.0412	0.0244	0.0523	0.0461	0.0759	0.0517
FlexCode-CF only	0.0360	0.0232	0.0563	0.0296	0.0217	0.0143	0.0326	0.0179	0.0309	0.0314	0.0377	0.0346
FlexCode-Fix	0.0531	0.0394	0.0707	0.0452	0.0293	0.0209	0.0431	0.0262	0.0628	0.0462	0.0747	0.0562
FlexCode (ours)	**0.0578**	**0.0415**	**0.0769**	**0.0483**	**0.0329**	**0.0232**	**0.0471**	**0.0275**	**0.0709**	**0.0524**	**0.0825**	**0.0632**

Table 2: Statistics of public benchmark datasets after preprocessing.

Dataset	#Users	#Items	#Interactions	Avg. Seq.
Beauty	22,363	12,101	198,360	8.87
Sports	35,598	18,357	296,175	8.32
KuaiRand	1,000	3.6M	11M	11.71
Proprietary	1.5M+	1M+	45M+	28.0

mechanism. For example, on KuaiRand, the dynamic allocation of PATA lifts NDCG@10 from 0.0562 to 0.0632, a relative gain of 12.5% over the fixed-split model. This confirms that both architectural components—disentanglement and adaptive allocation—are essential to achieving state-of-the-art performance.

4.3 Further Analysis on Industrial Dataset

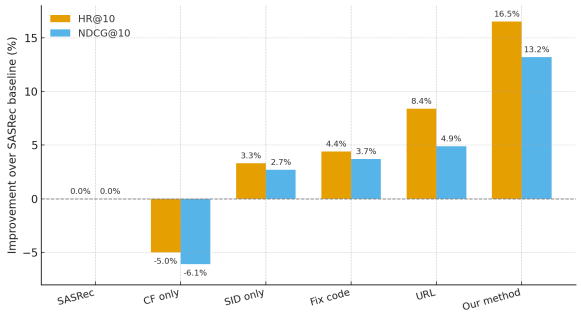
After establishing **FlexCode**'s overall superiority, we conduct analyses on our proprietary industrial dataset to examine its robustness and adaptability under production-scale conditions.

4.3.1 Overall Industrial Performance

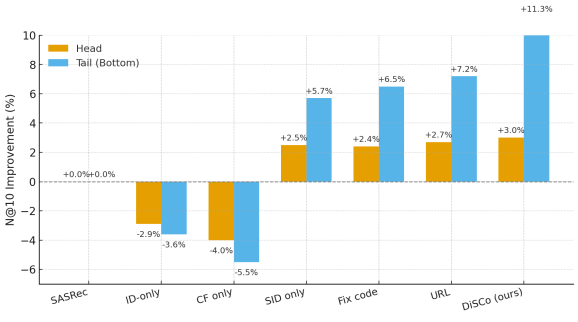
Figure 2a shows the performance improvement of generative models relative to SASRec baseline. While a pure collaborative generative model (CF only) degrades performance by over 5% and advanced unified models like URL provide a moderate 4.9% lift in NDCG@10, **FlexCode** delivers a 13.2% improvement in NDCG@10 and a 16.5% improvement in HR@10. This large margin of victory on a massive, noisy, and production-level dataset confirms that the architectural principles of **FlexCode** are not only theoretically sound but also scalable and highly effective in a real-world industrial setting.

4.3.2 Cold-Start and Long-Tail Item Performance

A central claim of this work is that adaptive allocation is critical for balancing memorization for popular items and generalization for rare items. We test this hypothesis by measuring NDCG@10 on the **Head** and **Bottom(Tail)** items from our industrial dataset, which has a highly skewed long-tail distribution. The results in Figure 2b provide direct evidence for our claim. Baselines exhibit a clear performance trade-off: the CF-only model improves head performance but degrades tail performance by 5.5%, whereas the SID-only model improves the tail by 5.7% at the cost of weaker head performance. **FlexCode** resolves this conflict. For head items, its MoE router correctly allocates more tokens to the collaborative codebook, enabling fine-grained memorization and yielding a 3.0% NDCG@10 improvement. For tail items, the router shifts capacity to the semantic codebook to enable generalization from content features, resulting in an 11.3% NDCG@10 improvement. This result is the largest gain on tail items by a wide margin and is of high practical importance, as improving long-tail discovery is a primary goal of industrial recommender systems. By dynamically



(a) Overall performance improvement of various models over the SASRec baseline on the industrial dataset.



(b) NDCG@10 improvement on Head vs. Tail items.

Figure 2: Performance evaluation on the large-scale industrial dataset.

allocating its representational budget, **FlexCode** demonstrates superior performance across the entire item popularity spectrum.

5 Ablation Study

We conduct ablation experiments on the KuaiRand dataset to analyze the contribution of each component in **FlexCode**. Table 3 summarizes the results. Removing the dual-codebook structure by using only collaborative IDs (*CID Only*) or semantic IDs (*SID Only*) leads to a significant performance drop, showing that separating and combining collaborative and semantic representations is critical. Disabling the MoE gating network and using a fixed 50/50 split (*w/o MoE Gating*) reduces adaptability to item popularity and harms performance. Finally, excluding the alignment loss (*w/o Alignment Loss*) causes a drop in performance, confirming its importance for maintaining cross-codebook coherence.

Table 3: Ablation study on the KuaiRand dataset. Each row removes or modifies a key component of **FlexCode**.

Model Variant	Recall@10	NDCG@10
FlexCode (Full)	**0.0825**	**0.0632**
FlexCode (CID Only)	0.0405	0.0372
FlexCode (SID Only)	0.0511	0.0401
w/o MoE Gating (Fixed Split)	0.0791	0.0598
w/o Alignment Loss	0.0809	0.0615

Token Budget Sensitivity. We further analyze **FlexCode**'s robustness under varying token budgets $L \in \{3, 4, 5, 6\}$. As shown in Table 4, **FlexCode** consistently achieves strong performance even with reduced token capacity. This demonstrates that its adaptive capacity reallocation allows for more efficient use of limited representational resources, a key advantage for real-world deployment.

Table 4: Token budget sensitivity on the KuaiRand dataset (NDCG@10).

Model	L = 3	L = 4	L = 5	L = 6
FlexCode (SID Only)	0.0401	0.0415	0.0418	0.0420
FlexCode (CID Only)	0.0372	0.0389	0.0395	0.0397
FlexCode-Fix (50/50 Split)	0.0598	0.0615	0.0619	0.0621
FlexCode (ours)	**0.0632**	**0.0685**	**0.0691**	**0.0693**

Table 5: Hyper-parameter sensitivity analysis on the KuaiRand dataset.

Parameter Setting	K	d	λ_{align}	λ_{smooth}	NDCG@10
Default (Base)	**512**	**64**	**0.1**	**0.01**	**0.0632**
K Variation	256	64	0.1	0.01	0.0603
	1024	64	0.1	0.01	0.0635
d Variation	512	32	0.1	0.01	0.0611
	512	128	0.1	0.01	0.0639
λ_{align} *Variation*	512	64	0.01	0.01	0.0618
	512	64	0.5	0.01	0.0625
	512	64	1.0	0.01	0.0609
λ_{smooth} *Variation*	512	64	0.1	0.001	0.0621
	512	64	0.1	0.05	0.0628
	512	64	0.1	0.1	0.0615

Hyper-Parameter Analysis. We examine the effect of key hyper-parameters in Table 5. **FlexCode** is generally stable across a broad range of settings. Larger codebooks (K) and dimensions (d) offer slight improvements until saturation. Performance is robust to moderate changes in the regularization weights (λ_{align}, λ_{smooth}), suggesting that the model is not overly sensitive to precise tuning.

6 Conclusion

In this work, we approached generative recommendation from the perspective of *adaptive capacity allocation*, introducing FLEXCODE, a dual-codebook framework with popularity-aware routing that improves both overall accuracy and long-tail performance. More broadly, our results suggest several

promising directions for future work. One avenue is to extend adaptive capacity allocation to richer multi-modal item descriptors and to user-side modeling, enabling joint reasoning over user and item codebooks. Another is to integrate FLEXCODE with large language models and online learning pipelines, studying how popularity-aware tokenization interacts with exploration, temporal drift, and calibration in real-world deployments. It is also important to examine fairness and exposure of long-tail content under different routing strategies, and to develop theoretical tools for understanding when and why dual-codebook architectures provide benefits over unified tokenizations. We hope that viewing generative recommendation through the lens of dual codebooks and adaptive capacity allocation will inspire further work on balancing memorization and generalization in token-based recommender systems.

References

Gulcin Baykal, Melih Kandemir, and Gozde Unal. 2024. Edvae: Mitigating codebook collapse with evidential discrete variational autoencoders. *Pattern Recognition*, 156:110792.

Dengzhao Fang, Jingtong Gao, Chengcheng Zhu, Yu Li, Xiangyu Zhao, and Yi Chang. 2025. Hid-vae: Interpretable generative recommendation via hierarchical and disentangled semantic ids. *arXiv preprint arXiv:2508.04618*.

Chongming Gao, Shijun Li, Yuan Zhang, Jiawei Chen, Biao Li, Wenqiang Lei, Peng Jiang, and Xiangnan He. 2022. Kuairand: An unbiased sequential recommendation dataset with randomly exposed videos. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pages 3953–3957.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. 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.

Carlos A. Gomez-Uribe and Neil Hunt. 2016. The netflix recommender system: Algorithms, business value, and innovation. *ACM Trans. Manage. Inf. Syst.*, 6(4).

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. 2023. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of the ACM Web Conference 2023*, pages 1162–1171.

Zheng Hui, Xiaokai Wei, Yexi Jiang, Kevin Gao, Chen Wang, Frank Ong, Se-eun Yoon, Rachit Pareek, and Michelle Gong. 2025. Matcha: Can multi-agent collaboration build a trustworthy conversational recommender? *arXiv preprint arXiv:2504.20094*.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE.

Anastasiia Klimashevskaia, Dietmar Jannach, Mehdi Elahi, and Christoph Trattner. 2024. A survey on popularity bias in recommender systems. *User Modeling and User-Adapted Interaction*, 34(5):1777–1834.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11523–11532.

Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023. Text is all you need: Learning language representations for sequential recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1258–1267.

Guanyu Lin, Zhigang Hua, Tao Feng, Shuang Yang, Bo Long, and Jiaxuan You. 2025. Unified semantic and id representation learning for deep recommenders. *arXiv preprint arXiv:2502.16474*.

Qidong Liu, Fan Yan, Xiangyu Zhao, Zhaocheng Du, Huifeng Guo, Ruiming Tang, and Feng Tian. 2023. Diffusion augmentation for sequential recommendation. In *Proceedings of the 32nd ACM International conference on information and knowledge management*, pages 1576–1586.

Zheqi Lv, Shaoxuan He, Tianyu Zhan, Shengyu Zhang, Wenqiao Zhang, Jingyuan Chen, Zhou Zhao, and Fei Wu. 2024. Semantic codebook learning for dynamic recommendation models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 9611–9620.

Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 825–833.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. 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.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, and 1 others. 2023. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36:10299–10315.

Reza Shirkavand, Xiaokai Wei, Chen Wang, Zheng Hui, Heng Huang, and Michelle Gong. 2025. Catalog-native llm: Speaking item-id dialect with less entanglement for recommendation. *arXiv preprint arXiv:2510.05125*.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450.

Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 565–573.

Chen Wang, Xiaokai Wei, Yexi Jiang, Frank Ong, Kevin Gao, Xiao Yu, Zheng Hui, Se-eun Yoon, Philip Yu, and Michelle Gong. 2025. Solving the content gap in roblox game recommendations: Llm-based profile generation and reranking. *arXiv preprint arXiv:2502.06802*.

Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2400–2409.

Yuhao Yang, Zhi Ji, Zhaopeng Li, Yi Li, Zhonglin Mo, Yue Ding, Kai Chen, Zijian Zhang, Jie Li, Shuanglong Li, and 1 others. 2025. Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations. *arXiv preprint arXiv:2503.02453*.

Taolin Zhang, Junwei Pan, Jinpeng Wang, Yaohua Zha, Tao Dai, Bin Chen, Ruisheng Luo, Xiaoxiang Deng, Yuan Wang, Ming Yue, and 1 others. 2024. Towards scalable semantic representation for recommendation. *arXiv preprint arXiv:2410.09560*.

Yin Zhang, Derek Zhiyuan Cheng, Tiansheng Yao, Xinyang Yi, Lichan Hong, and Ed H Chi. 2021. A model of two tales: Dual transfer learning framework for improved long-tail item recommendation. In *Proceedings of the web conference 2021*, pages 2220–2231.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. 2024. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1435–1448. IEEE.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1893–1902.