Discrimination Is Generation: Unifying Ranking and Retrieval from a Tokenizer Perspective

Shuli Wang*
Meituan
Chengdu, China
wangshuli03@meituan.com

Junwei Yin
Meituan
Chengdu, China
yinjunwei03@meituan.com

Changhao Li
Meituan
Chengdu, China
lichanghao@meituan.com

Senjie Kou
Meituan
Chengdu, China
kousenjie@meituan.com

Chi Wang
Meituan
Chengdu, China
wangchi06@meituan.com

Yinqiu Huang
Meituan
Chengdu, China
huangyinqiu@meituan.com

Yinhua Zhu
Meituan
Chengdu, China
zhuyinhua@meituan.com

Haitao Wang
Meituan
Chengdu, China
wanghaitao13@meituan.com

Xingxing Wang
Meituan
Beijing, China
wangxingxing04@meituan.com

Abstract

Semantic IDs (SIDs) define the generation space of generative recommendation and directly determine its personalization ceiling. However, existing tokenizers are trained independently with retrieval objectives, leaving personalization signals fully decoupled from the SID construction process—a fundamental gap that causes generative retrieval to persistently lag behind discriminative ranking. In this paper, we rethink the essence of SIDs: *ranking seeks argmax in item space while retrieval seeks argmax in token space; both are the same problem solved at different granularities.* Based on this insight, we propose DIG (**D**iscrimination **I**s **G**eneration), which embeds the tokenizer inside a discriminative ranking model for end-to-end training—the ranker naturally becomes a retrieval model, yielding two models from a single training run. DIG is organized around a *feature assignment taxonomy*: item-intrinsic static features are encoded into SIDs, user-item cross features (u2i) implicitly drive codebook boundaries toward recommendation decision boundaries during training, and an MLP_{u2t} distillation module approximates u2i at the token level for inference. Experiments on three public benchmarks and two industrial datasets demonstrate that DIG simultaneously improves ranking, retrieval, and unified retrieval-ranking quality.

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

generative recommendation, semantic ID, discriminative training, unified retrieval-ranking, tokenizer

ACM Reference Format:
Shuli Wang, Junwei Yin, Changhao Li, Senjie Kou, Chi Wang, Yinqiu Huang, Yinhua Zhu, Haitao Wang, and Xingxing Wang. 2026. Discrimination Is Generation: Unifying Ranking and Retrieval from a Tokenizer Perspective. In . ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

*Corresponding author.

Conference'17, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Generative recommendation (GR) quantizes items into discrete Semantic ID (SID) sequences and performs full-corpus argmax via beam search—complexity decoupled from the item catalog size, fundamentally breaking the computational bottleneck of traditional retrieval funnels [7, 11]. Building on the SID+NTP framework pioneered by TIGER [11], subsequent work has converged on a multi-stage pipeline: the tokenizer is trained independently, and the generative model is aligned downstream. Quality improvements have come through collaborative signal alignment [4, 7, 17, 25], end-to-end SID learning [4, 9], and differentiable quantization [13].

Despite this progress, a critical deficiency persists: **existing SIDs are severely under-personalized.** Current SID methods fall into two categories: *pure semantic tokenization* (e.g., TIGER—pretrained language model + RQ-VAE quantization) and *retrieval-aligned tokenization* (e.g., LETTER, DAS, DOS—dual-tower contrastive losses driving quantization). Both encode only static item attributes; user-item cross features ($\mathbf{c}_{u,v}$)—the very signals that give discriminative rankers their fine-grained personalization power—*never participate in codebook construction.* The consequence: the same item receives an identical SID regardless of whether it faces a highly matched or completely mismatched user. These personalization signals are precisely the core source of discriminative ranking ability—a structural ceiling that has kept generative retrieval persistently trailing discriminative ranking.

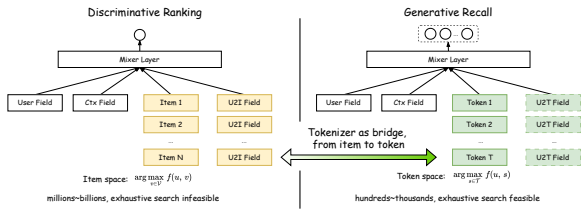

Figure 1: Comparison of existing generative retrieval pipelines vs. DIG. Existing tokenizers are trained with retrieval objectives (reconstruction or contrastive loss), fully decoupled from discriminative ranking signals. DIG embeds the tokenizer inside the ranker for end-to-end joint training, releasing generative retrieval capability already latent in the discriminative model.

The root cause is architectural: **tokenizers are trained with retrieval objectives and fully decoupled from discriminative signals.** Retrieval objectives only require that related items be close in embedding space; they impose no fine-grained personalization distinctions. The two-stage pipeline ensures that discriminative gradients can *never* flow back into the codebook, and personalization signals can *never* participate in quantization. Two recent directions attempt to close the gap—aligning tokenizers with downstream generation objectives [4, 9] and incorporating semantic tokens into rankers [2, 8, 20]—but both still optimize within the retrieval paradigm and neither injects discriminative ranking gradients back into SID construction. Both paths converge on a more fundamental question: **can the objective of discriminative ranking directly drive SID codebook construction, enabling the same token set to serve both generative retrieval and discriminative ranking?**

We answer this by rethinking the essence of SIDs. Both tasks solve the same optimization: ranking finds the highest-scoring item in *item space* while retrieval finds the optimal path in *token space*. The SID sits exactly at the intersection of both spaces—not merely a preprocessing step for generative recommendation, but a **bridge between discriminative and generative recommendation**. If SID construction is driven directly by a discriminative objective, the same token representation can simultaneously carry ranking discriminability and retrieval generativity. In other words, **generative retrieval capability does not require a separate generative model—it is already latent inside the discriminative ranker, and the tokenizer's role is to release it.**

Based on this insight, we propose **DIG** (**D**iscrimination **I**s **G**eneration). DIG embeds the tokenizer inside a discriminative ranking model for end-to-end joint training—the ranker naturally becomes a retrieval model, yielding two models from a single training run.

To make this bridge structurally sound, DIG is organized around a **feature assignment taxonomy** as its central axis, systematically resolving how three categories of features are handled across training and inference. **Unified Tokenizer** (§3.2): the VQ encoder takes only item-side static features; an offline Balanced K-Means tree guarantees zero-collision stability across model versions; SIDs are decoupled from SID embeddings—SIDs handle addressing while SID embeddings carry semantic expression end-to-end, resolving the fundamental trade-off between discrimination and semantic sharing. **Unified Training** (§3.3): ranking and retrieval share a single parameter set driven by a five-part unified loss; u2i cross features are concatenated into the scoring MLP and implicitly drive codebook boundaries toward recommendation decision boundaries end-to-end; MLP_{u2t} uses the ranker's u2i modeling capacity as teacher signal to learn user-conditioned u2t representations, substantially improving personalization resolution on the retrieval path compared to statistical averages. **Unified Inference** (§3.4): MLP_{u2t} generates u2t online to recover the third feature type; the ranking model directly executes beam search for retrieval—the unified paradigm follows naturally.

Contributions. We summarize our contributions as follows:

- We rethink the essence of SIDs and propose a novel *unified* paradigm: the tokenizer bridges item space and token space, and discriminative ranking already contains generative retrieval capability—no extra model is needed; the tokenizer releases it.
- We present DIG, a systematic implementation centered on the feature assignment taxonomy, integrating tokenizer design, unified training, and inference alignment to inject discriminative signals end-to-end into SID construction, with MLP_{u2t} personalized distillation bridging the feature gap between ranking and retrieval.
- We conduct extensive experiments on five datasets (three public + two industrial). DIG consistently outperforms state-of-the-art generative retrieval baselines on recall metrics and simultaneously improves ranking AUC, with particularly strong gains in sparse and u2i-rich scenarios.

2 Related Work

2.1 Generative Recommendation and Semantic IDs

Generative recommendation quantizes items into discrete token sequences and models user preferences auto-regressively. TIGER [11] established the foundational SID+NTP two-stage framework: offline, Sentence-T5 embeddings are quantized via RQ-VAE [18] into SIDs; online, a Transformer auto-regressively generates the target item's token sequence. Subsequent works improve SID quality primarily by introducing collaborative signals into the quantization process. LETTER [7] aligns collaborative vectors with quantized semantic vectors through contrastive learning, producing more uniform codebook distributions. CoST [16], DAS [25], and DOS [17] adopt user-item dual-tower structures driven by collaborative labels as auxiliary supervision. CoFiRec [15] fuses semantic SIDs with collaborative IDs into coarse-to-fine hierarchical token representations. All share a fundamental limitation: training signals come from retrieval-side contrastive learning, and the quantization process remains entirely decoupled from discriminative ranking objectives—u2i cross features $\mathbf{c}_{u,v}$ never participate in codebook construction.

ETEGRec [4] takes a step further by jointly training the tokenizer and generative model in an alternating frozen fashion, aligning them via NTP loss. ReSID [9] redesigns the tokenizer from an information-theoretic perspective, explicitly critiquing ETEGRec's *self-referential problem*—SIDs are simultaneously training targets and model inputs, causing optimization instability when NTP loss

adjusts SIDs end-to-end. It is important to note that ReSID's critique targets "using NTP loss to adjust SIDs"; DIG's gradient comes from a discriminative BCE loss that predicts user clicks, not SID sequences, so the self-referential problem does not apply. MTGRec [3] enhances generative model robustness via multi-tokenizer data augmentation. Differentiable SID [13] connects SID tasks to NTP via Gumbel-Softmax, allowing NTP gradients to flow back into SID assignment. S^2GR [24] inserts inference tokens before each SID layer, providing a chain-of-thought mechanism for SID generation. Both adjust SIDs from a *downstream NTP perspective*; DIG reshapes the codebook from an *upstream discriminative perspective*—the directions are opposite and complementary.

2.2 Unifying Retrieval and Ranking

Unifying retrieval and ranking into a single system is a long-standing research direction, and recent work has approached it primarily through generative models. HSTU [19] restructures recommendation as a sequence transduction task at trillion-parameter scale, verifying scaling laws for generative recommendation. OneRec [10] directly generates entire recommendation sessions using an encoder-decoder with MoE, replacing the traditional multi-stage cascade pipeline with a single generative system. GPR [22] unifies ad retrieval and ranking as an end-to-end generation task via heterogeneous hierarchical decoders. UniGRF [23], SynerGen [1], UniRec [12], OneRanker [28], and UGRF [21] each explore how generative models carry both retrieval and ranking simultaneously, via multi-task losses, hierarchical decoding, or DPO post-training. All *start from a generative model* and embed ranking capability within it. OneMall [6] takes a different approach, bridging two independent models via reinforcement learning where the ranker provides reward signals to the generative retriever.

Fundamental distinction of DIG. All of the above start from generative models to build unified frameworks; ranking capability is embedded into generation. DIG takes the *opposite* direction: starting from a discriminative ranker and extending its scoring space to full-corpus retrieval through the tokenizer. The ranker structure is unchanged; the tokenizer—acting as a bridge between item space and token space—endows the ranker with native retrieval capability. This path fully preserves the discriminative ranker's accumulated advantages (u2i cross features, high-order feature interactions), achieves architecture simplicity through a single training run, and requires neither a generative paradigm nor RL bridging. Generative capability is not designed in—it is *released* from the discriminative ranker.

2.3 Semantic Tokenization for Ranking

The idea of semantic tokenization has recently migrated from generative retrieval into discriminative ranking, motivated primarily by storage and compute efficiency. UIST [20] simultaneously tokenizes items and users, achieving ∼200× storage compression via hierarchical mixed inference; its tokenizer is trained independently with reconstruction error, decoupled from the ranking objective. STORE [2] proposes Semantic Tokenization + orthogonal rotation + Efficient Attention, rotating low-cardinality static features into the SID embedding space to enhance feature interactions, achieving +2.71% online CTR. TRM [8] replaces item ID embeddings with semantic tokens in search and ranking at scale, reducing sparse storage by 33% and achieving +0.85% AUC. These works demonstrate that semantic tokens can replace traditional item ID embeddings without hurting ranking quality—an important empirical foundation. However, they share the same structural limitation as retrieval-side SID methods: the tokenizer is an independent preprocessing tool, u2i cross features $\mathbf{c}_{u,v}$ never participate in tokenizer construction, and the codebook boundaries reflect content similarity rather than recommendation decision boundaries.

DIG differs from all of the above in three fundamental ways: (1) **Training signal**: discriminative BCE loss drives the tokenizer end-to-end, rather than reconstruction error or contrastive loss; (2) **Role of u2i features**: $\mathbf{c}_{u,v}$ implicitly drives codebook boundaries toward recommendation decision boundaries during training, making SID partitions reflect user preference rather than semantic similarity contours; (3) **System architecture**: DIG proposes the unified paradigm where online retrieval directly reuses the ranking model via beam search, fundamentally eliminating the semantic gap of traditional dual-system pipelines.

3 Methodology

We first present the problem formulation and DIG's overall framework (§3.1), establishing the feature assignment taxonomy as the central design axis. We then detail the Unified Tokenizer (§3.2), Unified Training (§3.3), and Unified Inference (§3.4).

3.1 Problem Formulation and DIG Framework

Notation. Let $\mathcal{V}$ be the item set and $\mathcal{U}$ the user set. Each item $v \in \mathcal{V}$ has static content features $\mathbf{x}_v^s$ (category, brand, region, etc.). Each user $u \in \mathcal{U}$ has user features $\mathbf{x}_u$ (profile, behavior history) and context features **ctx** (time, location, scene). For a user-item pair (u, v), **u2i** cross features $\mathbf{c}_{u,v}$ (user's historical CTR/CVR on this item) represent the core information advantage of ranking over retrieval. At retrieval time, $\mathbf{c}_{u,v}$ is unavailable (the target item is unknown); DIG introduces $\mathbf{u2t} \triangleq \mathbf{u2t}^{(l)}$, the token-level aggregation of $\mathbf{c}_{u,v}$ within bucket s_l, as a retrieval-time approximation. The naming *u2i* (user-to-item) and *u2t* (user-to-token) reflects the coarsening from item granularity to token granularity.

Semantic ID (SID). A SID is a mapping from items to discrete token sequences: $\phi : \mathcal{V} \to \mathcal{T}^L$, where $\mathcal{T} = \{1, \ldots, K\}$ is the codebook vocabulary and L is the number of quantization layers. Item v's SID is denoted $\phi(v) = (s_1^v, \ldots, s_L^v)$.

Core insight. Discriminative ranking finds $\arg\max_{v \in \mathcal{V}} f(u, v)$ in item space; generative retrieval finds $\arg\max_{(s_1,\ldots,s_L)} g(u, s_1, \ldots, s_L)$ in token space. Both solve the same optimization at different granularities. The tokenizer ϕ sits at the intersection: if SID construction is driven by the discriminative objective, the same token representation simultaneously carries ranking discriminability and retrieval generativity. *Generative retrieval capability is already latent inside the discriminative ranker—the tokenizer's role is to release it.*

Feature assignment taxonomy. The central challenge for the unified paradigm is handling features that ranking has but retrieval does not—specifically the u2i cross features $\mathbf{c}_{u,v}$, which encode user-item interaction history and are the core source of discriminative ranking's personalization advantage. These features cannot be statically encoded into SIDs (since the same item maps to different $\mathbf{c}_{u,v}$

Table 1: Feature assignment taxonomy in DIG. $\mathbf{x}_v^s$: item-side features; $\mathbf{e}_{\text{sid}}^{(1:l)}$: SID embedding prefix; u2i $\triangleq \mathbf{c}_{u,v}$: item-level cross features; u2t$^{(l)}$: token-level aggregation of $\mathbf{c}_{u,v}$ within bucket s_l.

Type	Symbols	In SID?	Training-side handling	Inference-side handling
Type-I: item-side	$\mathbf{x}_v^s$	✓	VQ encoder input $\rightarrow \mathbf{e}_v \rightarrow \mathbf{e}_{\text{sid}}^{(1:L)}$	Offline SID lookup table
Type-II: request-level	$\mathbf{x}_u$, **ctx**	×	Ranking loss condition	Beam search decoder condition
Type-III: struct. lossy	$\mathbf{c}_{u,v}$	×	Implicit codebook shaping via batch **u2t**$^{(1:l)}$	MLP_{u2t} distillation $\rightarrow$ **û2t**$^{(1:l)}$

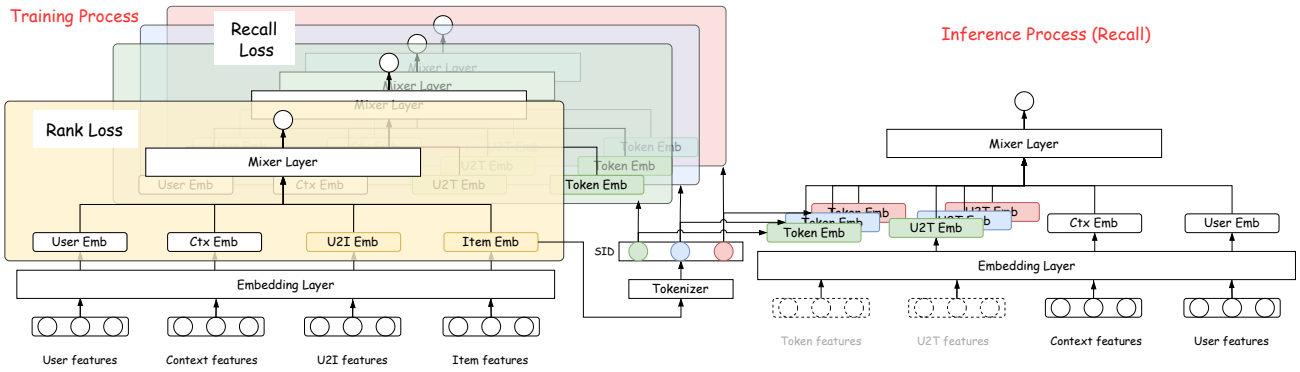

Figure 2: Overall architecture of DIG. The tokenizer is embedded inside the DIN+DCNv2+MoE ranker. Three feature streams are handled by the feature assignment taxonomy: Type-I static features shape SIDs offline; Type-II request-level features serve as ranking/decoder conditions; Type-III u2i cross features implicitly shape codebook boundaries during training and are approximated by MLP_{u2t} at inference.

for different users), yet they must somehow be preserved to avoid degrading retrieval quality. As shown in Table 1, DIG partitions all features into three types, and the handling of each type is the organizing principle behind the tokenizer design (§3.2), training (§3.3), and inference (§3.4).

3.2 Unified Tokenizer: SID Construction and Stability

VQ encoder. The VQ encoder takes only item-side features $\mathbf{x}_v^s$ as input, producing a continuous item-side embedding:

$$\mathbf{e}_v = \text{Enc}(\mathbf{x}_v^s). \tag{1}$$

Restricting the encoder to time-invariant features guarantees **SID stability**: the same item always maps to the same token sequence across model versions, preserving the SID-to-item inverted index.

Residual quantization and prefix accumulation. DIG applies residual quantization (RQ) to $\mathbf{e}_v$, encoding residuals layer by layer over L layers:

$$s_l^v = \arg\min_k \|\mathbf{r}_{l-1} - \mathbf{c}_{l,k}\|_2, \tag{2}$$

$$\hat{\mathbf{e}}_v^{(l)} = \mathbf{c}_{l,s_l^v}, \quad \mathbf{r}_l = \mathbf{r}_{l-1} - \hat{\mathbf{e}}_v^{(l)}, \tag{3}$$

$$\hat{\mathbf{e}}_v^{(1:l)} = \textstyle\sum_{i=1}^{l} \hat{\mathbf{e}}_v^{(i)}, \tag{4}$$

where $\mathbf{r}_0 = \mathbf{e}_v$ is the initial residual; $s_l^v \in \mathcal{T}$ is the assigned code index at layer l; $\mathbf{c}_{l,k}$ is the k-th codebook vector; and $\hat{\mathbf{e}}_v^{(1:l)}$ is the prefix-accumulated quantized representation used as the item's coarse-to-fine approximation. Codebook vectors are updated via EMA: $\mathbf{c}_{l,k} \leftarrow \alpha\, \mathbf{c}_{l,k} + (1-\alpha)\, \bar{\mathbf{e}}_{l,k}$. The residual structure builds a coarse-to-fine hierarchy: $\hat{\mathbf{e}}_v^{(1:l)}$ approximates the full representation with monotonically shrinking error as l grows—the structural basis for layer-wise beam search.

SID–embedding decoupling. Existing methods use codebook vector accumulations $\hat{\mathbf{e}}_v^{(1:L)}$ directly as item representations, coupling the *addressing* and *semantic expression* roles of SIDs into a single set of codebook vectors. This coupling creates a fundamental trade-off: finer SID partitions improve addressing precision but prevent semantic sharing across tokens; coarser partitions allow sharing but reduce discriminability. Furthermore, since codebook vectors are updated via non-differentiable argmin assignment, methods that rely on them for scoring must use STE or Gumbel-Softmax to approximate gradients through the quantization step.

DIG decouples the two roles into two separate parameter sets with different update rules:

- **Codebook vectors** $\mathbf{c}_{l,k}$: handle *addressing only* (Eq. 2–3), updated via EMA. Do not participate in scoring.
- **SID embeddings** $\mathbf{e}_{l,k}^{\text{sid}}$: handle *semantic expression*, updated end-to-end by the discriminative loss. Do not affect SID assignment.

The SID embedding prefix is:

$$\mathbf{e}_{\text{sid}}^{(1:l)} = \sum_{i=1}^{l} \mathbf{e}_{i,s_i^v}^{\text{sid}}, \tag{5}$$

where $\mathbf{e}^{\text{sid}}_{i,k}$ is the learnable embedding for code k at layer i, driven end-to-end by the discriminative loss. Because SID embeddings are standard learnable parameters independent of the argmin quantization, discriminative gradients flow through them directly—no STE approximation is needed. This also resolves the addressing–sharing trade-off: SID partitions can be made arbitrarily fine for zero-collision addressing, while SID embeddings freely learn cross-token semantic sharing. Critically, each SID embedding $\mathbf{e}^{\text{sid}}_{l,k}$ is defined per feature field and shares the same dimensionality as the ranking item feature embeddings, so $\mathbf{e}^{(1:l)}_{\text{sid}}$ is a drop-in replacement for $\mathbf{e}_v$ in the ranking MLP—the retrieval path reuses the ranking MLP without any structural modification. Space alignment between codebook geometry and SID embeddings is maintained by $\mathcal{L}_{\text{sem}}$, which anchors the codebook to the encoder output by requiring the codebook vector accumulation $\hat{\mathbf{e}}^{(1:L)}_v$ to reconstruct $\mathbf{e}_v$. Ranking and retrieval share the same Mixer (encompassing all modules whose parameters interact with the target item/token representation, including the scoring MLP, Target-Attention, and MoE), with $\mathbf{e}_u$ processed through a shared BN and the remaining inputs through path-specific BNs:

$$\hat{y}_{\text{rank}} = \sigma\big(\text{Mixer}([\text{BN}_u(\mathbf{e}_u);\ \text{BN}_v(\mathbf{e}_v);\ \text{BN}_{\text{u2i}}(\mathbf{c}_{u,v})])\big), \tag{6}$$

$$\hat{y}^{(l)}_{\text{recall}} = \sigma\Big(\text{Mixer}([\text{BN}_u(\mathbf{e}_u);\ \text{BN}_{\text{sid}}(\mathbf{e}^{(1:l)}_{\text{sid}});\ \text{BN}_{\text{u2t}}(\mathbf{u2t}^{(1:l)})])\Big), \tag{7}$$

where BN_u is shared across both paths; BN_v, BN_{u2i}, BN_{sid}, BN_{u2t} are feature-specific BNs named after the feature they normalize.

Low-collision design. Since SIDs are updated end-to-end during training, strict offline zero-collision guarantees no longer apply. DIG instead controls collision rate through two complementary mechanisms: (1) **Zero-collision initialization**: codebooks are initialized via an offline Balanced K-Means tree that partitions items into equal-size clusters with zero initial collisions, providing a stable starting point for end-to-end training; (2) **Large SID space**: with $L = 4, K = 256$, the SID space supports $256^4 \approx 4\text{B}$ unique tuples—orders of magnitude larger than any realistic catalog—making post-training collisions negligible in practice.

3.3 Unified Training: Joint Loss

DIG inserts a RQ quantizer after the item embedding layer of a standard DIN+DCNv2+MoE ranker; all other structures are unchanged. The Mixer takes three inputs: item-side embedding $\mathbf{e}_v$, user representation $\mathbf{e}_u$, and u2i cross features $\mathbf{c}_{u,v}$.

Unified training objective. All five losses are optimized jointly end-to-end:

$$\mathcal{L} = \mathcal{L}_{\text{rank}} + \mathcal{L}_{\text{recall}} + \lambda_1\mathcal{L}_{\text{commit}} + \lambda_2\mathcal{L}_{\text{sem}} + \lambda_3\mathcal{L}_{\text{u2t}}. \tag{8}$$

$\mathcal{L}_{\text{rank}}$ is the standard ranking BCE loss that drives the whole system toward the recommendation objective:

$$\mathcal{L}_{\text{rank}} = \text{BCE}(\hat{y}_{\text{rank}},\ y)\,. \tag{9}$$

$\mathcal{L}_{\text{recall}}$ provides layer-wise supervision aligned with beam search expansion:

$$\mathcal{L}_{\text{recall}} = \frac{1}{L}\sum_{l=1}^{L}\text{BCE}\Big(\hat{y}^{(l)}_{\text{recall}},\ y\Big)\,. \tag{10}$$

$\mathcal{L}_{\text{commit}}$ prevents the encoder output from drifting away from codebook entries:

$$\mathcal{L}_{\text{commit}} = \sum_{l=1}^{L}\|\text{sg}[\mathbf{c}_{l,s^v_l}] - \mathbf{r}_{l-1}\|_2^2, \tag{11}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator and $\mathbf{r}_{l-1}$ is the residual entering layer l.

$\mathcal{L}_{\text{sem}}$ reconstructs $\mathbf{e}_v$ from the quantized representation to prevent codebook collapse:

$$\mathcal{L}_{\text{sem}} = \|\mathbf{e}_v - \hat{\mathbf{e}}^{(1:L)}_v\|_2^2, \tag{12}$$

where $\hat{\mathbf{e}}^{(1:L)}_v$ is the full-depth prefix accumulation of codebook vectors (Eq. 4).

$\mathcal{L}_{\text{u2t}}$ is the MLP_{u2t} distillation loss defined below.

Implicit u2i shaping of the codebook (Type-III, training). Although $\mathbf{c}_{u,v}$ cannot be statically encoded into SIDs, it can indirectly drive codebook boundaries toward recommendation decision boundaries through batch aggregation. At each training step, items in the same token bucket s_l within a mini-batch $\mathcal{B}$ aggregate their u2i features into a token-level statistic:

$$\mathbf{u2t}^{(l)} = \frac{1}{|\mathcal{V}^{(l),\mathcal{B}}_{s_l}|}\sum_{v'\in\mathcal{V}^{(l),\mathcal{B}}_{s_l}}\mathbf{c}_{u,v'}, \tag{13}$$

where $\mathcal{V}^{(l),\mathcal{B}}_{s_l}$ is the set of items in bucket s_l at layer l within the current mini-batch, and $\mathbf{c}_{u,v'}$ is the u2i cross feature of user u for item v'. Via $\mathcal{L}_{\text{recall}}$, the quantizer is implicitly rewarded for clustering items with similar u2i signals into the same bucket. **SID boundaries thus reflect recommendation decision boundaries, not merely content similarity contours.**

MLP_{u2t} distillation (Type-III, training). Bucket-averaged $\mathbf{u2t}^{(l)}$ loses within-bucket personalization. MLP_{u2t} recovers it by learning a user-conditioned approximation, supervised by the batch statistic:

$$\hat{\mathbf{u2t}}^{(l)}(u) = \text{MLP}^{(l)}_{\text{u2t}}\Big([\mathbf{e}_u;\ \mathbf{e}^{(1:l)}_{\text{sid}}]\Big), \tag{14}$$

$$\mathcal{L}_{\text{u2t}} = \frac{1}{L}\sum_{l=1}^{L}\|\hat{\mathbf{u2t}}^{(l)}(u) - \text{sg}[\bar{\mathbf{u2t}}^{(l)}(u)]\|_2^2, \tag{15}$$

where $\hat{\mathbf{u2t}}^{(l)}(u)$ is the personalized u2t prediction from $\text{MLP}^{(l)}_{\text{u2t}}$ (one lightweight MLP per layer); $\bar{\mathbf{u2t}}^{(l)}(u)$ is the batch-mean u2t statistic (Eq. 14) serving as the teacher signal; and $\text{sg}[\cdot]$ stops gradients from flowing into the teacher. Inputs $\mathbf{e}_u$ and $\mathbf{e}^{(1:l)}_{\text{sid}}$ are both available online at inference, so MLP_{u2t} requires no offline feature tables. Its output dimension equals d_c (dimension of $\mathbf{c}_{u,v}$), so $\hat{\mathbf{u2t}}^{(1:l)}$ can directly replace $\mathbf{c}_{u,v}$ in the shared MLP at retrieval time.

3.4 Unified Inference: Feature Alignment and Beam Search

Online retrieval (beam search). At step l, all K tokens at layer l are scored with the shared Mixer, substituting $\mathbf{c}_{u,v}$ with the online MLP_{u2t} output:

$$\hat{y}^{(l)}_{\text{recall}} = \sigma\Big(\text{Mixer}([\text{BN}_u(\mathbf{e}_u);\ \text{BN}_{\text{sid}}(\mathbf{e}^{(1:l)}_{\text{sid}});\ \text{BN}_{\text{u2t}}(\hat{\mathbf{u2t}}^{(1:l)}(u))])\Big)\,. \tag{16}$$

After L steps, candidates are recovered from the SID-to-item inverted index built offline.

Online ranking. Retrieved candidates enter the ranker with full u2i features:

$$\hat{y}_{\text{rank}} = \sigma\big(\text{Mixer}([\text{BN}_u(\mathbf{e}_u);\, \text{BN}_v(\mathbf{e}_v);\, \text{BN}_{\text{u2i}}(\mathbf{c}_{u,v})])\big)\,. \tag{17}$$

Training-inference symmetry. Both retrieval and ranking paths share the same Mixer; the only differences are item-side representation granularity ($\mathbf{e}_{\text{sid}}^{(1:l)}$ at retrieval vs. $\mathbf{e}_v$ at ranking) and cross-feature granularity ($\hat{\mathbf{u2t}}^{(1:l)}$ vs. $\mathbf{c}_{u,v}$). This symmetry fundamentally eliminates the semantic gap of traditional dual-system pipelines: the retrieval search objective is exactly aligned with the ranking optimization target.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate on five datasets spanning extreme sparsity ranges:

- **KuaiRec-Small** [5]: dense interaction matrix (1,411 users, 3,327 items, 99.6% density). Positive label: watch_ratio≥0.7.
- **KuaiRec-Big** [5]: sparse matrix (7,176 users, 10,728 items, 16.3% density).
- **Taobao Ad** [27]: real-world ad click logs (∼26.6M ranking samples, ∼0.003% density); u2i: per-category/brand CTR, impression counts, purchase counts.
- **Meituan-Large**: industrial dataset (∼20M users, ∼500K items, ∼7.5M interactions, ∼0.0008% density) with rich u2i cross features.
- **Meituan-Small**: smaller Meituan business (∼12K users, ∼40K items, ∼1.5M interactions, ∼0.31% density, ∼0.016% positive rate).

All samples are split strictly by time: each user's last click as test, second-to-last as validation, and all prior samples as training. *Ranking samples*: every exposure record with $(\mathbf{x}_u, \text{hist}, \mathbf{x}_v^s, \mathbf{c}_{u,v}, y)$; u2i features use prefix-accumulated statistics (no leakage). *Retrieval samples*: positive-only (y=1), full-corpus random negatives; evaluation via beam search over the full item corpus, reported as Recall@K / NDCG@K. KuaiRec provides a stricter *retrieval-native* benchmark (full interaction matrix; test targets may never have been exposed by a ranker), while Taobao and Meituan are sourced from ranking exposure logs where the candidate pool is pre-filtered upstream.

Backbone. DIG embeds the tokenizer into a DIN+DCNv2+MoE ranker: DIN [26] models user history via attention; DCNv2 [14] captures high-order feature interactions; MoE provides multi-expert capacity for diverse user groups. The tokenizer is a plug-in component after the item embedding layer, leaving all backbone structures intact.

Baselines. We compare against five representative generative retrieval methods, all using the same NTP generative model for fair comparison: (1) **TIGER** [11]: RQ-VAE reconstruction loss, NTP generation; (2) **LETTER** [7]: dual-tower contrastive learning for quantization alignment; (3) **DAS** [25] and (4) **DOS** [17]: variants of collaborative-signal-driven quantization; (5) **ETEGRec** [4]: end-to-end tokenizer-generator joint training.

u2i features. Taobao: per-category/brand CTR, impression and purchase counts (6 dims). KuaiRec: per-user `watch_ratio` and `like_rate`. Taobao/Meituan (exposure-log-based) have meaningful u2i for every training sample; KuaiRec-Big (full matrix) has 83.7% pairs without prior interaction, making u2i features zero or degenerate for these pairs.

Implementation. VQ encoder: 4-layer Transformer on item-side static features. RQ quantizer: $L = 4$ layers, $K = 256$ codebook entries per layer, embedding dim $d = 64$. MLP_{u2t}: one 2-layer MLP per layer, hidden dim 64, output dim aligned with u2i cross feature dim. Training: batch size 2048, Adam optimizer, lr $= 1 \times 10^{-3}$, $\lambda_1 = 0.25$, $\lambda_2 = 0.1$, $\lambda_3 = 0.1$. With $L = 4$ layers and $K = 256$ codebook entries, the u2t batch statistics are stable at all layers: layer 1 has ∼180 active buckets/batch (∼11 items/bucket); layers 2–4 progressively narrow but the cumulative prefix $\mathbf{u2t}^{(1:l)}$ aggregates across all preceding layers, so even shallow buckets at deeper layers receive a meaningful signal. All experiments run on 8×A100 GPUs.

4.2 Retrieval Performance

RQ1: Can DIG surpass generative retrieval baselines? The core claim of DIG is that injecting discriminative signals into SID construction produces better retrieval than tokenizers trained with retrieval-only objectives. We verify this by comparing DIG against five representative generative retrieval baselines across all five datasets. Table 2 reports Recall@10 and NDCG@10.

DIG achieves the best retrieval quality across all five datasets and all metrics. Three observations stand out:

(1) Consistent superiority over all baselines, regardless of SID construction strategy. Existing tokenizers—whether trained with semantic reconstruction (TIGER) or retrieval-aligned contrastive losses (LETTER, DAS, DOS)—encode only static item attributes. DIG's discriminative SID aligns codebook boundaries with user preference decision boundaries via end-to-end BCE loss; beam search therefore explores a token space that reflects actual recommendation utility rather than content similarity, closing the semantic gap that plagues all NTP-based baselines.

(2) Gains are consistent across all density regimes, with distinct underlying sources. On KuaiRec-Small (99.6% density), NTP baselines already have access to strong user histories, yet DIG still achieves +52.0% R@10. This gain stems from the u2t feature mechanism and end-to-end discriminative training: even in dense scenarios, u2i signals $\mathbf{c}_{u,v}$ drive codebook boundaries toward recommendation decision boundaries, and MLP_{u2t} provides personalized token-level approximations that NTP models fundamentally lack. On Taobao (∼0.003%), NTP baselines suffer additional degradation from sequence starvation (average click history length <5 on Taobao), causing near-complete collapse in weaker baselines (e.g., TIGER R@10=0.0001). The large relative gains on Taobao (+171% over ETEGRec) and Meituan-Small (+220%) partly reflect baseline weakness in sparse settings; the absolute gains over the best baseline (+0.0101 and +0.0077) confirm consistent improvement. DIG's discriminative signal comes from ranking-side exposure samples and is independent of SID click sequence length, making it natively robust to short sequences and cold-start users.

(3) Largest gains on industrial datasets combining sparsity with rich u2i features. Meituan-Small exhibits extreme interaction sparsity (∼0.016% positive rate) yet provides rich u2i cross features from the industrial ranking pipeline. This combination delivers the highest gain across all datasets (+220% R@10): sparsity

Table 2: Main retrieval results (Recall@10 / NDCG@10). Best in bold. KuaiRec datasets are retrieval-native benchmarks (full interaction matrix); Taobao and Meituan are sourced from ranking exposure logs.

Method	Taobao		KuaiRec-S		KuaiRec-B		MT-Large		MT-Small	
	R@10	N@10	R@10	N@10	R@10	N@10	R@10	N@10	R@10	N@10
TIGER	0.0001	0.0020	0.0052	0.0081	0.0030	0.0045	0.0018	0.0041	0.0015	0.0033
LETTER	0.0001	0.0027	0.0063	0.0109	0.0059	0.0060	0.0023	0.0051	0.0018	0.0039
DAS	0.0003	0.0054	0.0116	0.0216	0.0102	0.0119	0.0029	0.0063	0.0021	0.0048
DOS	0.0005	0.0106	0.0213	0.0426	0.0121	0.0234	0.0037	0.0082	0.0031	0.0067
ETEGRec	0.0007	0.0126	0.0275	0.0504	0.0153	0.0277	0.0041	0.0089	0.0035	0.0074
DIG (ours)	**0.0019**	**0.0341**	**0.0418**	**0.1189**	**0.0254**	**0.0463**	**0.0097**	**0.0213**	**0.0112**	**0.0248**
Improv.	*+171%*	*+171%*	*+52.0%*	*+136.1%*	*+66.0%*	*+67.1%*	*+136.6%*	*+139.3%*	*+220%*	*+235.1%*

renders NTP baselines most vulnerable (short sequences, poor generalization), while rich u2i features maximize the quality of DIG's discriminative codebook signal—the two effects amplify each other.

4.3 Ranking Performance

RQ2: Does end-to-end training hurt ranking? Embedding the tokenizer inside the ranker introduces retrieval-path gradients that could interfere with the ranking objective. We check whether this joint training hurts ranking quality by comparing DIG against the rank-only model (recall_loss_weight=0), which has the tokenizer but disables all retrieval-path gradients. Table 3 reports ranking AUC across five datasets.

Table 3: Ranking AUC comparison. Δ = DIG − Base. Base is the rank-only model (recall_loss_weight=0).

Method	Taobao	KuaiRec-S	KuaiRec-B	MT-Large	MT-Small
Base	0.6225	0.8428	0.8240	0.9058	0.6521
DIG	**0.6402**	**0.8565**	**0.8304**	**0.9071**	**0.6726**
Δ	**+0.0177**	**+0.0137**	+0.0064	+0.0013	**+0.0205**

End-to-end joint training consistently improves ranking quality across all five datasets.

The primary goal of the joint training design is to not hurt ranking while enabling retrieval; the ranking results universally exceed this bar.

Finding 1: DIG improves ranking across all datasets. Gains range from +0.0013 (Meituan-Large) to +0.0205 (Meituan-Small). On Taobao, the gain reaches +0.0177; on KuaiRec-Small, +0.0137; on KuaiRec-Big, +0.0064. These gains come as a byproduct of the SID embedding parameter-sharing mechanism: u2i-driven codebook boundaries cluster items with similar behavior preferences, enriching SID embeddings with collaborative signals that benefit both retrieval and ranking.

Finding 2: Larger gains on datasets with richer u2i features. The two largest gains occur on Meituan-Small (+0.0205) and Taobao (+0.0177), followed by KuaiRec-Small (+0.0137) and KuaiRec-Big (+0.0064)—all datasets where meaningful u2i signals are available. This pattern confirms that the ranking bonus is not a side-effect of retrieval-path regularization, but a direct consequence of discriminative SID boundaries encoding user preference: the same codebook boundaries that drive retrieval quality also provide structured collaborative regularization to the ranking MLP.

Finding 3: Structural mechanism behind the ranking bonus. Two complementary channels contribute: (1) u2i-driven codebook boundaries provide collaborative regularization via $\mathcal{L}_{\text{sem}}$ coupling between codebook geometry and SID embeddings; (2) layer-wise $\mathcal{L}_{\text{recall}}$ supervision encourages a hierarchical embedding structure that sharpens item discrimination at each prefix depth. Both channels are absent in rank-only training (recall_loss_weight=0), explaining why Base consistently falls below DIG.

4.4 Unified Retrieval-Ranking

RQ3: Is unified-model candidate quality better than independent retrieval? The unified paradigm uses the same ranking MLP for both retrieval and ranking, with MLP_{u2t} approximating u2i features at token granularity during retrieval. The key question is whether this approximation preserves enough scoring quality to make the retrieval path useful—and whether the resulting retrieval-ranking AUC gap is acceptable. Table 4 quantifies this gap across datasets.

Table 4: Unified retrieval-ranking AUC gap. The gap measures how much the retrieval path's scoring ability lags behind the full ranking path.

Dataset	Rank AUC	Recall AUC	Gap (Δ)
Taobao	0.6402	0.6115	−0.0287
KuaiRec-S	0.8565	0.8431	−0.0134
KuaiRec-B	0.8304	0.8132	−0.0172
MT-Large	0.9071	0.8459	−0.0612
MT-Small	0.6726	0.6483	−0.0243

Finding 1: The retrieval-ranking AUC gap correlates with u2i feature complexity, not interaction density. On KuaiRec datasets, u2i features are simple (watch_ratio, like_rate), and MLP_{u2t} approximates them accurately—gaps are small (−0.0134 on KuaiRec-S, −0.0172 on KuaiRec-B). On Taobao and Meituan-Large, u2i features are high-dimensional (per-category/brand CTR, impression counts, purchase counts), making the token-level approximation harder and widening the gap (−0.0287 and −0.0612 respectively).

This gap is an inherent cost of the unified-model design: MLP_{u2t} approximates item-level $\mathbf{c}_{u,v}$ at token granularity, and approximation error grows with u2i feature complexity.

4.5 Ablation Study

DIG's design involves three key components: (1) end-to-end discriminative gradients reshaping the codebook, (2) training-side u2i signals implicitly shaping bucket boundaries, and (3) MLP_{u2t} providing personalized u2t at inference. We isolate each component to quantify its individual contribution and verify the necessity of the full design.

Ablation 1: Discriminative gradient necessity (E2E vs. two-stage). We replace DIG's end-to-end training with a two-stage baseline (Fixed SID): SIDs are pre-generated offline using the same Balanced K-Means tree initialization as DIG, then fixed throughout training—the VQ encoder is frozen and codebook boundaries do not change. This isolates the sole variable of whether discriminative gradients are allowed to reshape the codebook, evaluated across all three public datasets.

Table 5: Ablation 1: Effect of discriminative gradient on retrieval and ranking across three public datasets.

Dataset	Config	Rank AUC	Recall AUC	Recall@10
Taobao	DIG (E2E)	**0.6402**	**0.6115**	**0.0019**
	Fixed SID	0.6355	0.6089	0.0014
	Δ	+0.0047	+0.0026	+35.7%
KuaiRec-S	DIG (E2E)	**0.8565**	**0.8431**	**0.0418**
	Fixed SID	0.8544	0.7562	0.0275
	Δ	+0.0021	**+0.0869**	**+52.0%**
KuaiRec-B	DIG (E2E)	**0.8304**	**0.8132**	**0.0254**
	Fixed SID	0.8308	0.7451	0.0008
	Δ	−0.0004	**+0.0681**	**+212%**

Across all three datasets, fixing the tokenizer consistently degrades Recall AUC (KuaiRec-S: −0.0869; KuaiRec-B: −0.0681; Taobao: −0.0026), while Rank AUC remains virtually identical ($|\Delta| \leq 0.002$). The Recall AUC gaps are especially pronounced on KuaiRec datasets, where the fixed tokenizer groups items by content geometry rather than user preference—beam search explores semantically coherent but preference-irrelevant regions. The near-zero ranking cost across all datasets confirms that discriminative gradients reshape the codebook without interfering with the ranking MLP, which still receives full u2i features regardless. **Discriminative gradient is the fundamental source of DIG's retrieval quality, with negligible cost to ranking.**

Ablation 2: Independent contributions of training-side u2i and inference-side MLP_{u2t}. Orthogonal ablation on Taobao:

Table 6: Ablation 2: Orthogonal ablation on Taobao (R@10 / N@10).

Config	R@10	N@10
Full DIG	0.0019	0.0341
w/o training-side u2i	0.0014 (−26.3%)	0.0253 (−25.8%)
w/o inference MLP_{u2t}	0.0016 (−15.8%)	0.0290 (−14.9%)
w/o both	0.0010 (−47.4%)	0.0183 (−46.3%)

The complementary effect confirms mutual reinforcement: removing both causes −47.4%, exceeding the sum of individual drops (−26.3%+−15.8%=42.1%). Training-side u2i drives codebook boundaries toward recommendation decision boundaries, which in turn improves MLP_{u2t} distillation signal quality—the two components amplify each other rather than independently stacking.

Ablation 3: MLP_{u2t} vs. statistical mean u2t. On Taobao: MLP_{u2t} (full DIG) achieves R@10=0.0019, while statistical mean u2t yields R@10=0.0015 (−21.1%). MLP_{u2t}'s advantage is most prominent on *sparse tokens and long-tail users*: for low-activity users, statistical mean u2t is highly unstable, while MLP_{u2t} recovers stable estimates through model generalization.

Ablation 4: Necessity of layer-wise supervision. Replacing layer-wise $\mathcal{L}_{\text{recall}}$ with only final-layer supervision ($l = L$) causes retrieval metrics to drop significantly (−18.4% R@10) while ranking AUC remains nearly unchanged (≈ 0). This precisely separates two mechanisms: layer-wise supervision's value is *eliminating training-inference discrepancy* (intermediate prefixes become out-of-distribution inputs without supervision), not providing ranking regularization. DIG's positive ranking effect comes from structural improvement of discriminative SID embeddings, independent of layer-wise supervision.

4.6 Sparse Scenario Stability Analysis

DIG relies on u2i signals to drive codebook boundaries toward recommendation decision boundaries (via $\mathcal{L}_{\text{recall}}$) and to supervise MLP_{u2t} distillation (via batch u2t statistics). A natural concern is whether DIG remains stable when u2i features are sparse or degenerate—a common situation in real-world deployments where not all user-item pairs have interaction history. We systematically analyze stability across five datasets with varying u2i completeness (Table 7).

Table 7: Sparse scenario stability. ΔAUC = DIG − Base ranking AUC.

Dataset	Density	u2i Completeness	ΔAUC	Stability
KuaiRec-S	99.6%	Full	+0.0137	Stable
KuaiRec-B	16.3%	Sparse	+0.0064	Stable
Taobao	0.003%	Full (ranking logs)	+0.0177	Stable
MT-Large	0.0008%	Full*	+0.0013	Stable
MT-Small	0.31%	Full*	+0.0205	Stable

* Meituan datasets use hundreds of u2i cross features sourced directly from real industrial ranking pipelines, providing full coverage over all exposed user-item pairs.

Finding 1: Training sample construction determines u2i completeness, not interaction matrix density. Taobao's interaction matrix density is only 0.003%, yet every training sample carries complete u2i features because the data comes from ranking exposure logs—the ranker only scores user-item pairs that already have interaction history. KuaiRec-Big has 16.3% matrix density, but its training set is built from the full interaction matrix including unobserved pairs—83.7% of training samples have zero or degenerate u2i values. Despite this, DIG still achieves +0.0064 AUC gain on

KuaiRec-Big, suggesting that even partial u2i coverage is sufficient for discriminative SID boundaries to provide ranking benefit.

Finding 2: $\mathcal{L}_{\text{recall}}$ is the primary training signal for SID embeddings. The reason is structural: the ranking path uses full item-side embedding $\mathbf{e}_v$, so $\mathcal{L}_{\text{rank}}$'s gradient flows mainly through $\mathbf{e}_v$ and $\mathbf{e}_u$—SID embeddings receive only a weak indirect signal via $\mathcal{L}_{\text{sem}}$. $\mathcal{L}_{\text{recall}}$ directly supervises the SID prefix $\mathbf{e}_{\text{sid}}^{(1:l)}$ at every layer, making SID embeddings its primary optimization target. Setting recall_loss_weight to 0 causes SID embeddings to degrade and ranking AUC to fall, confirming that $\mathcal{L}_{\text{recall}}$ is irreplaceable.

Conclusion. DIG is stable across all five datasets regardless of interaction matrix density. The key driver is the quality of u2i signals in ranking exposure logs; DIG leverages these signals to reshape codebook boundaries, delivering consistent ranking gains even in sparse interaction scenarios. For industrial deployment, **u2i feature coverage rate should be evaluated first; for low-coverage scenarios, configuring a stronger MLP_{u2t} further improves distillation quality**.

5 Conclusion

We proposed DIG, a unified paradigm that bridges discriminative ranking and generative retrieval by embedding the tokenizer inside the ranker for end-to-end joint training. The key insight—ranking and retrieval are the same optimization at different granularities—motivates a feature assignment taxonomy that governs all three design components: the tokenizer decouples SID addressing from semantic expression via independent SID embeddings; a five-part unified loss drives both paths jointly, with u2i signals driving codebook boundaries toward recommendation decision boundaries and MLP_{u2t} bridging the feature gap at inference; and the shared ranking MLP executes beam search directly, eliminating the semantic gap of traditional dual-system pipelines. Experiments on five datasets confirm that DIG simultaneously improves retrieval, ranking, and unified-model candidate quality, with especially strong gains in sparse and u2i-rich scenarios.

References

[1] Hao Chen et al. 2024. SynerGen-VL: Towards Synergistic Image Understanding and Generation with Vision Experts and Token Folding. *arXiv preprint arXiv:2412.09604* (2024).

[2] Weilin Chen, Jingzhou Tang, Jinbo Lin, Jingen Lin, Pengyu Wei, Bin Liu, and Wei Wang. 2025. STORE: Semantic Tokenization with Orthogonal Rotation for Efficient Recommendation. *arXiv preprint arXiv:2501.10239* (2025).

[3] Jiangxia Du, Xiaojun Chen, Bin Liao, and Shen Zhu. 2024. MTGRec: Multi-Tokenization Graph Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[4] Jiangxia Du, Jingping Song, Bin Liao, Shen Zhu, and Runze Xie. 2025. ETEGRec: End-to-End Tokenizer-Encoder-Generator for Generative Recommendation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[5] Chongming Gao, Shijun Li, Yuan Zhang, Jiawei Chen, Biao Li, Wenqiang Lei, Peng Jiang, and Xiangnan He. 2022. KuaiRec: A Fully-Observed Dataset and Insights for Evaluating Recommender Systems. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 540–550.

[6] Qijiong Jia, Jianguo Liu, Jiaxin Wan, Bo Wang, Feilong Yang, and Wenbo Chen. 2024. OneMall: RL-Enhanced Unified One-Stop Shopping Assistant. *arXiv preprint arXiv:2407.12837* (2024).

[7] Hanbing Li, Pengyu Xiao, Mingliang Xu, Yongquan Liu, Shen Fan, Ruiming Wang, and Zhen Li. 2024. LETTER: Linking Collaborative and Language Embeddings for Tokenization in Recommender Systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1401–1410.

[8] Zhe Li, Mingyang Zhao, Keyu Wang, Xinlei Hu, Zhiying Wang, and Chenfei Gu. 2026. TRM: Semantic Token Replacement for Online Search and Recommendation. *arXiv preprint arXiv:2603.14088* (2026).

[9] Liangcai Lin, Mingming Liu, Ruihao Hu, Xiaoquan Liu, Zhaocheng Qiu, and Weidong Wen. 2024. ReSID: Towards Semantically Aligned Semantic IDs for Generative Recommendation. *arXiv preprint arXiv:2404.02848* (2024).

[10] Jiaxin Qiu, Jiashu Ma, Shengyu Guan, Tao He, Weiming Chen, Zhenguo Liu, and Feng Zheng. 2024. OneRec: Unifying Retrieve and Rank with Generative Recommender and Iterative Preference Alignment. *arXiv preprint arXiv:2501.18253* (2024).

[11] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan H Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q Tran, Jonah Samost, et al. 2023. Recommender Systems with Generative Retrieval. In *Advances in Neural Information Processing Systems*, Vol. 36.

[12] Ruiyang Ren et al. 2024. UniRec: A Unified Model for Sequential Recommendation. *arXiv preprint arXiv:2402.01330* (2024).

[13] Zihua Si, Zhongxiang Liu, Jiale Chen, Liang Pang, Jiafeng Sun, and Xueqi Cheng. 2024. Differentiable Semantic IDs for Generative Recommendation. In *Proceedings of the ACM Web Conference 2024*.

[14] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. DCN V2: Improved Deep & Cross Network and Practical Lessons for Web-scale Learning to Rank Systems. In *Proceedings of the Web Conference 2021*. 1785–1797.

[15] Zheng Wang, Jiajun Yang, Peijia Lin, Zhuoran Liu, and Hao Wang. 2024. CoFiRec: Coarse-to-Fine Tokenization for Generative Recommendation. *arXiv preprint arXiv:2402.11705* (2024).

[16] Haoran Yang, Pan Li, Chen Gao, Wenjun Fu, Depeng Jin, and Yong Li. 2024. CoST: Contrastive Quantization based Semantic Tokenization for Generative Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[17] Junwei Yin, Senjie Kou, Changhao Li, Shuli Wang, Xue Wei, Yinqiu Huang, Yinhua Zhu, Haitao Wang, and Xingxing Wang. 2026. DOS: Dual-Flow Orthogonal Semantic IDs for Recommendation. In *Companion Proceedings of the ACM Web Conference 2026*.

[18] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. 2021. SoundStream: An End-to-End Neural Audio Codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 30, 495–507.

[19] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, et al. 2024. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. In *Proceedings of the 41st International Conference on Machine Learning*.

[20] Hao Zhang, Runze Xie, Bin Liao, Sen Xu, Zhenhua Liu, and Yong Ge. 2024. UIST: Unified Item Semantic Tokenization for Better Retrieval and Ranking. *arXiv preprint arXiv:2405.09170* (2024).

[21] Luankang Zhang, Kenan Song, Yi Quan Lee, Wei Guo, Hao Wang, et al. 2025. Killing Two Birds with One Stone: Unifying Retrieval and Ranking with a Single Generative Recommendation Model. *arXiv preprint arXiv:2504.16454* (2025).

[22] Wenhao Zhang, Sichun Li, Shenghan Sun, Jianguo Rong, and Jun Zhao. 2024. GPR: Generative Page Recommendation for Large-Scale Online Advertising. *arXiv preprint arXiv:2403.02476* (2024).

[23] Hao Zhao, Zhuoran Liu, Jie Zheng, Rui Sun, and Jiliang Tang. 2024. UniGRF: A Unified Framework for Generative Recommendation with Retrieval and Ranking Fusion. *arXiv preprint arXiv:2405.10638* (2024).

[24] Jiaxin Zheng, Zhiqiang Li, Shanshan Feng, Tingting Liu, and Hao Yin. 2024. S^2GR: Semantic-Structured Generative Recommendation with Discrete Semantic Tokens. *arXiv preprint arXiv:2406.12033* (2024).

[25] Zihua Zheng, Xiaoquan Liu, Yanan Gu, Biao Liu, and Xiao Wang. 2024. DAS: Discriminability-oriented Semantic Token Assignment for Generative Recommendation. In *Proceedings of the ACM Web Conference 2024*.

[26] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1059–1068.

[27] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning Tree-based Deep Model for Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1079–1088.

[28] Wentao Zhu et al. 2024. OneRanker: Towards Universal Ranking for E-commerce Recommendation. *arXiv preprint arXiv:2407.09647* (2024).