DaV-Gen: End-to-End Generative Retrieval via Draft-and-Verify

Meng Zhao , **Chunmei Liu** , **Qinyong Wang**
HUJING Digital Media & Entertainment Group
{chuming.zm, meijiang.lcm, wangqinyong.wqy}@alibaba-inc.com

Abstract

Mainstream industrial information retrieval systems (e.g., search and recommendation) are usually built upon Multi-Stage Cascade Architectures (MCAs), which balance effectiveness and efficiency through a coarse-to-fine "retrieval-ranking" pipeline. However, the optimization objectives across different stages are substantially inconsistent, propagating or even amplifying the early-stage errors that ultimately degrade the quality of final results. While emerging end-to-end generative models offer a potential solution by unifying the pipeline, their online serving performance is severely hindered by the auto-regressive process inherited from the standard decoder-only structure. To bridge this gap, we introduce **DaV-Gen**, a novel unified solution designed to fundamentally refactor the paradigm for both search and recommendation via a "Draft-and-Verify" mechanism. Inspired by the process used by speculative decoding, our framework redesigns the generation task into two synergistic operations within a single model. During training, the model is concurrently optimized for both candidate drafting and fine-grained verification. This is achieved by a composite loss function that jointly trains the model on two distinct but related objectives: 1) a contrastive loss that structures the embedding space for efficient drafting, and 2) a fusion loss that combines generative likelihood with vector similarity to produce a superior verification score. This integrated training strategy equips the model with dual capabilities. At inference time, it first performs highly efficient vector-based drafting to generate a candidate set, and then verifies these candidates using the more powerful fused scoring function, thereby achieving both the speed of sparse drafting and the precision of advanced generative models within a unified, end-to-end architecture.

1 Introduction

Most large-scale information retrieval systems, including Search Engines and Recommender Systems, rely on a Multi-Stage Cascade Architecture (MCA) [Covington *et al.*, 2016; Zhou *et al.*, 2018] to balance efficacy and efficiency. However, this distinct "retrieve-and-rank" pipeline suffers from *objective inconsistency*, where misalignment between stage-specific goals leads to *error propagation*—early misses by the retriever are irreversible, fundamentally limiting final result quality [Zhang *et al.*, 2025; Hron *et al.*, 2021]. To unify this pipeline, recent end-to-end models [Deng *et al.*, 2025] represent items as discrete "Semantic IDs" [Rajput *et al.*, 2023], transforming retrieval into a sequence-to-sequence task. Despite their promise, standard Generative Information Retrieval (GenIR) models are severely hindered by their auto-regressive nature, which introduces prohibitive inference latency and lacks deterministic control over recommendation list length.

To bridge the gap between the efficiency of cascades and the unified expressiveness of generative models, we introduce **DaV-Gen**, a novel framework that refactors the paradigm via a **Draft-and-Verify** mechanism. Instead of relying on slow token-by-token generation, DaV-Gen reformulates the task into two synergistic operations within a single, end-to-end differentiable architecture (Figure 1). We systematically tackle the inherent trade-offs in current systems by addressing three fundamental design questions:

1) How to balance representational efficiency and expressiveness? (Figure 1a) Standard sparse tokens facilitate generation but often lose fine-grained semantics, while dense vectors excel at retrieval but lack structural hierarchy. Our design resolves this by synthesizing a *Hybrid Sparse-Dense Representation*. By fusing the raw continuous signals from the RQ-VAE encoder with the contextualized discrete Semantic ID, we create a robust embedding that supports high-recall vector drafting without sacrificing the granular precision needed for downstream verification.

2) How to align disparate objectives in a single model? (Figure 1b) Solving the objective inconsistency of MCAs requires more than just sharing parameters; it demands a unified optimization landscape. We construct a *Collaborative Training Strategy* where the model is co-optimized for both contrastive discrimination and generative likelihood. Crucially, a fusion-based pairwise objective acts as a bridge, ensuring that the drafting module's semantic space is perfectly aligned with the verification module's ranking preferences, forcing the two stages to reinforce rather than contradict each other.

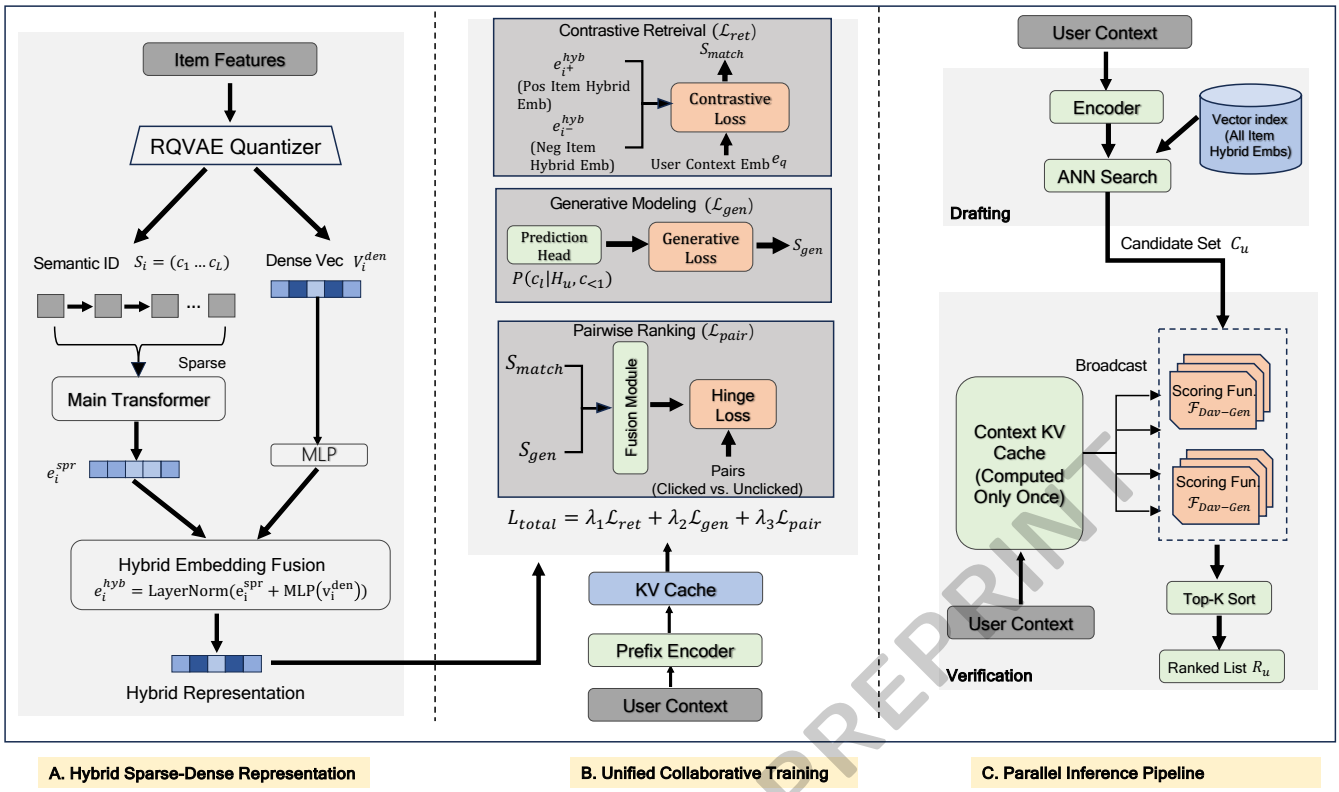

Figure 1: Overview of **DaV-Gen**. (a) **Hybrid Sparse-Dense Representation**: We construct a robust item embedding by fusing the fine-grained dense vector from the RQ-VAE encoder with the contextualized sparse Semantic ID. (b) **Unified Training Framework**: The model is jointly optimized with a composite loss (Contrastive + Generative + Pairwise) to align drafting and verification objectives within a single model. (c) **Parallel Inference Pipeline**: Adopting a "Draft-and-Verify" paradigm, DaV-Gen first drafts candidates via ANN search and then performs parallel verification in a single forward pass, significantly reducing latency compared to token-by-token generation.

3) How to break the latency bottleneck of generative models? (Figure 1c) The primary barrier to deploying GenIR is the serial dependency of auto-regressive decoding. We dismantle this barrier by introducing a *Parallel Inference Pipeline*. By decoupling candidate acquisition from scoring, DaV-Gen first drafts candidates via efficient ANN search and then verifies the entire batch in a single forward pass. This distinct architectural choice replaces the $O(L)$ complexity of sequential decoding with $O(1)$ parallel scoring, enabling real-time responsiveness.

The main contributions of this paper are summarized as follows:

- **Draft-and-Verify Paradigm:** We propose DaV-Gen, a unified framework that refactors the generative retrieval paradigm from sequential token generation to parallel verification. This architecture fundamentally resolves the latency bottleneck of auto-regressive models, achieving industrial-grade inference efficiency that significantly outperforms traditional cascaded systems, while enabling precise list-length control.
- **Hybrid Sparse-Dense Representation:** We introduce a novel item representation that fuses fine-grained dense vectors from the RQ-VAE encoder with contextualized sparse Semantic IDs. This hybrid approach bridges the gap between efficient retrieval and precise semantic understanding, enhancing both the recall of the drafting stage and the accuracy of the verification stage.
- **Unified Optimization Framework:** We design a comprehensive training pipeline driven by a composite loss function. This strategy effectively eliminates the objective inconsistency of traditional cascades by optimizing a single model to simultaneously master coarse-grained drafting and fine-grained verification within a unified vector space.

2 Methodology

In this section, we first formalize the problem of generative retrieval and describe the conventional auto-regressive approach. We then present our proposed DaV-Gen framework, detailing its comprehensive training process and efficient inference architecture.

2.1 Formalism for Generative Retrieval

Let $\mathcal{U}$ denote the set of users and $\mathcal{I}$ be the set of all items in the corpus. Both search and recommendation share the same goal: retrieving relevant items based on a user's context. We define the comprehensive user context as $\mathcal{C}_{ctx}$. In Recommendation, the context is implicit: $\mathcal{C}_{ctx} = H_u = (\mathcal{S}_u, \mathcal{P}_u)$, where

$\mathcal{S}_u = (i_1, \ldots, i_t)$ is the interaction history and $\mathcal{P}_u$ is the user profile. In Search, the context is explicit: $\mathcal{C}_{ctx} = \{q, H_u\}$, driven primarily by the query q and augmented by history H_u. For notation simplicity and consistency, we use H_u to denote this generalized context throughout the paper. The foundation of modern generative recommendation is the concept of a **Semantic ID**. Instead of using a single atomic identifier for each item, we first represent each item $i \in \mathcal{I}$ by its rich content features. A quantizer model $\mathcal{Q}$, such as the pre-trained RQ-VAE [Rajput *et al.*, 2023], maps these features to a structured Semantic ID, which is a sequence of discrete tokens (or codewords) $S_i = (c_1, c_2, \ldots, c_L)$, where each token c_j belongs to a finite codebook $\mathcal{C}$. The task is thus to model the conditional probability $P(S_{i_{t+1}}|H_u)$, predicting the Semantic ID of the next relevant item i_{t+1} given the context.

2.2 Auto-Regressive Recommendation

The mainstream approach to solving this task, as seen in models like TIGER, is purely auto-regressive. Crucially, these models model the joint probability of a user's entire interaction sequence $\mathcal{S}_u = (i_1, \ldots, i_T)$ as a strict causal chain. The training objective minimizes the negative log-likelihood accumulated over all time steps in the sequence:

$$\mathcal{L}_{\text{AR}} = -\sum_{t=1}^{T}\sum_{l=1}^{L} \log P(c_l^{(t)}|i_1, \ldots, i_{t-1}, c_{<l}^{(t)}) \qquad (1)$$

where $c_l^{(t)}$ denotes the l-th token of the item i_t at step t, and the condition $i_1, \ldots, i_{t-1}$ explicitly enforces the dependency on the entire preceding history. This formulation necessitates a sequential, item-by-item decoding process, as the prediction of step t strictly awaits the completion of step $t-1$, serving as the primary source of high inference latency.

2.3 The DaV-Gen Training Process

Our training process is a comprehensive pipeline designed to create an end-to-end model.

Progressive Knowledge Injection

The primary challenge in adopting a pre-trained language model (PLM) for generative retrieval lies in the semantic gap between the PLM's open-world knowledge and the domain-specific item representations (i.e., Semantic IDs). Since the discrete tokens constituting a Semantic ID are randomly initialized, directly optimizing the entire model can lead to catastrophic forgetting of the PLM's linguistic capabilities. To address this, we design a robust strategy that progressively injects item-specific knowledge into the backbone model via a self-supervised reconstruction task. In this task, the model is trained to bidirectionally predict an item's Semantic ID given its textual attributes and vice versa.

1. **Embedding Alignment.** In the initial stage, we focus solely on aligning the newly introduced Semantic ID tokens with the pre-existing semantic space of the PLM. We freeze all parameters of the transformer backbone, making only the embedding matrix of new tokens trainable. By projecting the textual descriptions of items into the model's latent space and optimizing the embeddings of their corresponding Semantic IDs to match these representations, we establish a coarse-grained alignment. This warm-up process ensures that the Semantic ID tokens acquire meaningful vector representations without destabilizing the carefully pre-trained weights of the base model.
2. **Deep Semantic Integration.** Once the token embeddings are preliminarily aligned and stabilized, we proceed to the second stage by unfreezing all model parameters. The pre-training task continues in a full-parameter fine-tuning regime. This phase allows the model to learn complex, non-linear interactions between the item's hierarchical structure (captured by the Semantic ID sequence) and its semantic content. It enables the backbone to deeply integrate the specific item knowledge with its general world knowledge, effectively transforming the PLM from a general-purpose text generator into a domain-expert recommender system.

Hybrid Sparse-Dense Item Representation

We draw inspiration from the COBRA framework [Yang *et al.*, 2025] to leverage the mutual reinforcement of sparse and dense signals. However, distinct from COBRA, which models them as an alternating sequence for step-by-step auto-regressive generation, we fuse them into a single unified vector. This design choice is critical for our "Draft-and-Verify" paradigm: it compresses the item's multimodal semantics into a fixed-length embedding compatible with Maximum Inner Product Search (MIPS), enabling the high-speed drafting that COBRA's sequential architecture cannot directly support.

Specifically, the hybrid embedding e_i^{hyb} has two parts:

- **Dense Content Vector (v_i^{den}):** Derived directly from the RQ-VAE's encoder output before quantization. It preserves the fine-grained, non-discrete semantic features (e.g., specific textual descriptions) often lost in tokenization.
- **Sparse Structural Vector (e_i^{spr}):** To capture the hierarchical category information of the Semantic ID S_i, we feed the token sequence into the backbone and compute e_i^{spr} via mean-pooling the final hidden states.

These components are fused via a projection layer to form the final item anchor:

$$e_i^{\text{hyb}} = \text{LayerNorm}(e_i^{\text{spr}} + \text{MLP}(v_i^{\text{den}})) \qquad (2)$$

where MLP aligns the dimensions. This representation serves as the indexable vector for drafting and the semantic anchor for verification.

Supervised Fine-Tuning (SFT) Optimization Strategy

During SFT, the model is optimized with a composite loss function. The total loss $\mathcal{L}_{\text{total}}$ is a weighted sum of three key components:

$$\mathcal{L}_{\text{total}} = \lambda_1\mathcal{L}_{\text{ret}} + \lambda_2\mathcal{L}_{\text{gen}} + \lambda_3\mathcal{L}_{\text{pair}} \qquad (3)$$

where λ_1, λ_2, and λ_3 are hyper-parameters. This composite objective equips the model with three essential skills: $\mathcal{L}_{\text{ret}}$ teaches the model to produce effective embeddings for drafting; $\mathcal{L}_{\text{gen}}$ imparts a deep contextual understanding of item sequences; and $\mathcal{L}_{\text{pair}}$ directly optimizes the final verification capability. We detail each component below.

(1) Contrastive Drafting Loss ($\mathcal{L}_{\text{ret}}$). This loss is the cornerstone of our efficient drafting stage, designed to structure the embedding space for fast and accurate candidate generation. We calculate this loss over the training batch $\mathcal{B}$. For each user context H_u in the batch, we define the candidate set $\mathcal{C}_u = \{i^+\} \cup \{i_j^-\}$ as the union of the positive item i^+ and the set of in-batch negative items. The contrastive loss is formulated as the average over the batch:

$$\mathcal{L}_{\text{ret}} = -\frac{1}{|\mathcal{B}|} \sum_{u \in \mathcal{B}} \log \frac{\exp(\text{sim}(e_q, e_{i^+}^{\text{hyb}})/\tau)}{\sum_{k \in \mathcal{C}_u} \exp(\text{sim}(e_q, e_k^{\text{hyb}})/\tau)} \tag{4}$$

where e_q is the user embedding derived from H_u, e_k^{hyb} denotes the hybrid embedding for item k, $\text{sim}(\cdot,\cdot)$ is the cosine similarity, and τ is a temperature hyper-parameter that controls the sharpness of the distribution.

This loss explicitly teaches the model the concept of semantic relevance. By rewarding the proximity of a user representation to its positive item while penalizing its proximity to numerous negative items, we compel the model to organize the high-dimensional embedding space into meaningful, semantically coherent clusters. The superiority of this approach is twofold. Firstly, it produces embeddings that are directly optimized for MIPS. This is the critical property that allows us to use highly efficient Approximate Nearest Neighbor (ANN) search libraries (e.g., FAISS) for the initial drafting stage, which directly solves the high-latency problem of purely generative models. Secondly, by learning from a diverse set of negatives, the model gains a robust understanding of item-to-item relationships, enhancing its ability to generalize and recommend novel yet relevant items, thus improving the quality of the candidate set passed to the subsequent verification phase.

(2) Generative Loss ($\mathcal{L}_{\text{gen}}$). This loss leverages a generative task to capture a deep, contextual understanding of items. Unlike standard auto-regressive recommenders that model dependencies between items, our approach treats each candidate item independently. However, we explicitly retain the auto-regressive dependency within the tokens of each item's Semantic ID $(c_1, \ldots, c_L)$ to capture its internal hierarchical structure. We formulate the objective over the entire training batch $\mathcal{B}$, where each sample consists of a user context H_u and the corresponding ground-truth positive item i. The loss is defined as the average negative log-likelihood across the batch:

$$\mathcal{L}_{\text{gen}} = -\frac{1}{|\mathcal{B}|} \sum_{(H_u, i) \in \mathcal{B}} \sum_{l=1}^{L} \log P(c_l | H_u, c_{<l}) \tag{5}$$

where $P(c_l | H_u, c_{<l})$ is the predicted probability for the l-th token of item i, conditioned on the user history H_u and the preceding tokens $c_{<l}$ of the same item.

This design extracts a verification signal that is orthogonal to simple vector similarity. While the contrastive loss answers "which items are semantically similar?", the generative loss answers "how plausible is this specific item in this user's context?". It forces the model to move beyond geometric proximity and learn the fine-grained compositional semantics of an item (as encoded by its RQ-VAE tokens) and how they align with a user's dynamic preferences. The resulting negative log-likelihood serves as a powerful generative score that enables a sophisticated and accurate final verification decision when fused with the matching score.

To efficiently compute this loss across a large batch of candidates, we implement a Broadcasted Prefix Caching mechanism. A naive implementation would necessitate concatenating the long context H_u with every candidate item, leading to redundant computations where the context is re-encoded N times. Instead, we compute the Key-Value (KV) states of H_u once per batch and broadcast them to serve as a shared memory prefix for all candidates. The model then computes the likelihood of item tokens by attending to this pre-computed cache, effectively reducing the context encoding complexity from $O(N \cdot |H_u|)$ to $O(|H_u|)$.

(3) Pairwise Ranking Loss ($\mathcal{L}_{\text{pair}}$). This loss directly optimizes the model's final verification capability using a fused score and explicit user feedback. First, we define the final score $\sigma(i)$ for a candidate item i as the output of a fusion network f_{fusion}, which takes two inputs: a matching score s_{match} and a generative score s_{gen}.

$$\sigma(i) = f_{\text{fusion}}(s_{\text{match}}(i), s_{\text{gen}}(i)) \tag{6}$$

where $s_{\text{match}}(i) = \text{sim}(e_q, e_i^{\text{hyb}})$ and $s_{\text{gen}}(i)$ is derived from the item's generative loss. The fusion network allows the model to learn the optimal, non-linear combination of these two signals. The model is then trained on this score using a pairwise hinge loss, averaged over valid pairs constructed from the batch. We construct two distinct sets of pairs:

1. Hard pairs $\mathcal{S}_{\text{hard}}$: Pairs of clicked items i and unclicked items j from the same session.
2. Calibration pairs $\mathcal{S}_{\text{rand}}$: Pairs of exposed items i and randomly sampled negative items k.

The objective is to ensure the score of a preferred item is higher than that of a non-preferred item by at least a margin m:

$$\begin{aligned} \mathcal{L}_{\text{pair}} = & \frac{1}{|\mathcal{S}_{\text{hard}}|} \sum_{(i,j) \in \mathcal{S}_{\text{hard}}} [\max(0, \sigma(j) - \sigma(i) + m)] \\ & + \frac{1}{|\mathcal{S}_{\text{rand}}|} \sum_{(i,k) \in \mathcal{S}_{\text{rand}}} [\max(0, \sigma(k) - \sigma(i) + m)] \end{aligned} \tag{7}$$

The motivation for using these two types of pairs is to create a robust and well-calibrated verifier. The first set of pairs (clicked vs. unclicked) presents the model with hard negatives. This teaches the model to make fine-grained distinctions among items that are all relevant enough to be shown to the user, directly optimizing for engagement. The second set of pairs (exposed vs. random) provides a crucial global calibration. It teaches the model that any item deemed relevant enough for exposure should be scored significantly higher than a random item from the corpus. This ensures a sensible scoring baseline and improves the overall robustness of the verification function.

2.4 Parallelized Generative Inference

Our comprehensive training process enables a highly efficient two-stage inference architecture that is well-suited for online serving.

Stage 1: Candidate Drafting

The inference process begins with a fast candidate drafting step. Given a user's context H_u, the framework first computes the corresponding query/context embedding e_q. Let $\mathcal{E}_{\mathcal{I}} = \{e_i^{\text{hyb}} | i \in \mathcal{I}\}$ be the pre-computed set of hybrid embeddings for all items in the corpus, stored in an efficient vector index. The drafting stage produces a candidate set $\mathcal{C}_u$ of size N using an Approximate Nearest Neighbor (ANN) search:

$$\mathcal{C}_u = \text{ANN}(e_q, \mathcal{E}_{\mathcal{I}}, N) \tag{8}$$

where $\text{ANN}(\cdot)$ represents the fast vector search operation. This step is non-generative and returns a fixed-size candidate list almost instantaneously.

Stage 2: Parallel Verification

The N retrieved candidates from $\mathcal{C}_u$ are then passed to our trained DaV-Gen model for scoring. Crucially, to ensure low latency, we leverage the Broadcasted Prefix Caching mechanism introduced in the training phase. Instead of concatenating the user context with each candidate repeatedly, we compute the context's KV cache once and broadcast it to initialize the attention states for all N candidates. The model then processes the tokens of all N candidates in a single parallel forward pass, conditioned on this cached prefix. Let $\mathcal{F}_{\text{DaV-Gen}}$ be the learned scoring function. For every candidate item $i \in \mathcal{C}_u$, the score is computed as:

$$\text{score}(i) = \mathcal{F}_{\text{DaV-Gen}}(i | \text{Cache}(H_u)) \quad \forall i \in \mathcal{C}_u \tag{9}$$

The candidates are then sorted in descending order based on this score to form a ranked list $\mathcal{L}_u$. This design decouples the context length from the candidate set size, ensuring that the inference latency grows only marginally with the number of verified items. The final recommendation list, $\mathcal{R}_u$, consists of the top-K items from this list:

$$\mathcal{R}_u = \text{Top-K}(\mathcal{L}_u) \tag{10}$$

This architecture directly solves the challenges of prior approaches. It sidesteps the high-latency auto-regressive bottleneck by replacing sequential generation with fast drafting and parallel verification. Furthermore, since the number of items to draft (N) and display (K) are fixed parameters, it offers precise, deterministic control over the resulting list length. Finally, because a single, unified model is trained for both drafting and verification, the objective inconsistency of classic cascaded systems is eliminated.

3 Experiments

To validate the effectiveness and generality of DaV-Gen, we conduct extensive experiments covering both recommendation and search scenarios. Our evaluation aims to answer the following research questions: **RQ1 (Model Accuracy):** Does DaV-Gen outperform state-of-the-art models on widely used benchmarks? **RQ2 (Industrial Scalability):** Can DaV-Gen handle large-scale industrial scenarios effectively compared to traditional methods? **RQ3 (Online Business Value):** Does the framework deliver real-world business improvements in an online production environment? **RQ4 (Ablation):** What is the contribution of our proposed innovative components in DaV-Gen?

3.1 Experimental Setup

Datasets

Following recent works [Wang *et al.*, 2024; Xing *et al.*, 2025], we perform experiments on three widely used recommendation benchmarks: **Amazon Beauty**, **Amazon Sports**, and **Yelp**. To verify the model's scalability in Search, we include a large-scale industrial dataset, **Ind-Search**, collected from a leading video search engine, containing over 100M items and 500M search logs. For all datasets, we adopt the leave-one-out strategy for evaluation.

Baselines

We compare DaV-Gen with representative methods:

- **Discriminative Models: SASRec** [Kang and McAuley., 2018], **BERT4Rec** [Sun *et al.*, 2019], **HGN** [Ma *et al.*, 2019], and the industrial strong baseline **MoE** (a Mixture-of-Experts based DNN model for CTR prediction) serving online.
- **Generative & Retrieval Models: TIGER** [Rajput *et al.*, 2023], **LC-Rec** [Wang *et al.*, 2024], **OneRec** [Deng *et al.*, 2025], and the state-of-the-art dense retriever **gte-qwen-7b** (fine-tuned).

Implementation Details

For fair comparison, the embedding dimension is set to 64 for all models. For generative models (TIGER, OneRec, DaV-Gen), we use a codebook of 5 layers of hierarchical quantization. DaV-Gen is trained with the Adam optimizer with a learning rate of $1e-5$ and a batch size of 256.

3.2 Performance Comparison (RQ1)

Table 1 reports the performance on public recommendation datasets. We observe the following:

1. Superiority over Discriminative Models: Generative methods generally outperform SASRec and BERT4Rec. This demonstrates that directly modeling the generation probability of semantic IDs captures collaborative signals more effectively. **2. Comparison with Unified Frameworks:** OneRec performs significantly better than TIGER and RPG, validating the effectiveness of end-to-end training models. However, DaV-Gen further surpasses OneRec. While OneRec relies on auto-regressive decoding, DaV-Gen utilizes a "Draft-and-Verify" mechanism that globally scores candidates, leading to more robust verification performance.

3.3 Evaluation on Industrial Search (RQ2)

To assess the industrial scalability of DaV-Gen, we evaluated it on the Ind-Search dataset against two strong production baselines: the fine-tuned dense retriever gte-qwen-7b and the online ranking model MoE. We report Recall@50, NDCG@10, and MRR@10.

Method	Amazon Beauty		Amazon Sports		Yelp	
	Recall@10	NDCG@10	Recall@10	NDCG@10	Recall@10	NDCG@10
SASRec	0.0624	0.0385	0.0482	0.0295	0.0681	0.0412
BERT4Rec	0.0638	0.0392	0.0495	0.0301	0.0695	0.0425
HGN	0.0705	0.0441	0.0571	0.0358	0.0738	0.0462
TIGER	0.0698	0.0435	0.0552	0.0348	0.0725	0.0455
LC-Rec	0.0715	0.0452	0.0568	0.0355	0.0742	0.0468
OneRec	<u>0.0732</u>	<u>0.0465</u>	<u>0.0588</u>	<u>0.0370</u>	<u>0.0765</u>	<u>0.0481</u>
DaV-Gen (Ours)	**0.0752**	**0.0481**	**0.0605**	**0.0382**	**0.0784**	**0.0495**
Improv.	+2.73%	+3.44%	+2.89%	+3.24%	+2.48%	+2.91%

Table 1: Performance comparison on three recommendation datasets. The best results are highlighted in **bold**, and the second best are <u>underlined</u>. Improvement is calculated relative to the strongest baseline.

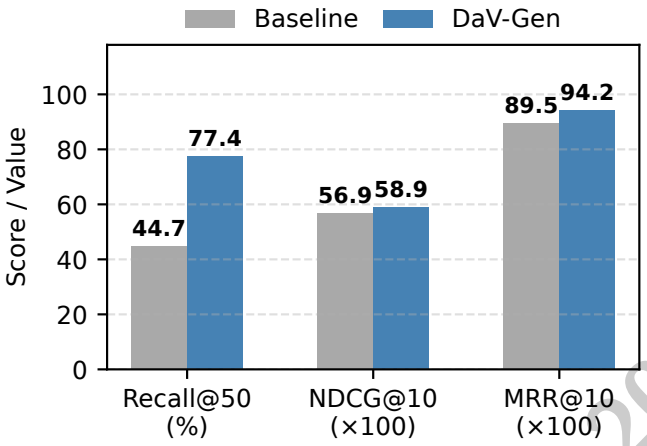

Figure 2: Performance comparison on Ind-Search. Metrics for ranking (NDCG, MRR) are scaled by $\times 100$ for visualization.

Figure 2 summarizes the comparison. **Retrieval (Recall@50):** DaV-Gen achieves a dominant 77.4%, outperforming the dense retriever baseline (44.7%) by over 30 absolute percentage points. This confirms that our Hybrid Sparse-Dense Representation effectively captures the complex semantics in video search queries that pure dense vectors miss. **Ranking (NDCG & MRR):** DaV-Gen consistently surpasses the discriminative MoE baseline. It improves NDCG@10 from 0.569 to 0.589 and boosts MRR@10 from 0.895 to 0.942. The high MRR score specifically highlights DaV-Gen's ability to place the best result at top positions, which is critical for optimizing user search experience.

3.4 Online A/B Testing (RQ3)

We deployed DaV-Gen in the production environment of a leading video provider to serve live search traffic. The experiment involved millions of users and was conducted over one week. We focus on three core business metrics reflecting user engagement, conversion efficiency and search satisfaction:

- **Avg. Time Spent per User (ATS):** The total time spent divided by the number of exposed users (ATS $= \frac{\sum \text{Time Spent}}{\text{Exposed UV}}$). This is the primary metric for user stickiness.
- **User Conversion Rate (UCVR):** The ratio of users who performed at least one video play to the total exposed users (UCVR $= \frac{\text{Play UV}}{\text{Exposed UV}}$), reflecting the feed's ability to convert exposure into consumption.
- **Avg. Successful Searches (ASS):** The number of search queries with at least one click divided by exposed users (ASS $= \frac{\text{Hit Search Volume}}{\text{Exposed UV}}$), indicating the efficiency of satisfying user intent within a session.

Metric	Improvement	Significance
Avg. Time Spent per User	+2.09%	$p < 0.05$
User Conversion Rate	+0.47%	$p < 0.05$
Avg. Successful Searches	+0.31%	$p < 0.05$

Table 2: Online A/B Testing Results

Table 2 reports the experimental results relative to the highly optimized production baseline. DaV-Gen yields a +2.09% uplift in ATS, suggesting that our fine-grained verification mechanism significantly enhances user stickiness by ensuring the most engaging content is prioritized. Furthermore, the consistent gains in UCVR (+0.47%) and ASS (+0.31%) demonstrate the model's superior capability in capturing user intent and facilitating content discovery.

Latency Analysis. Efficiency is critical for online serving. We explicitly compare the inference latency of DaV-Gen against two representative paradigms: 1) **Pure Generative Models:** Standard auto-regressive models (e.g., TIGER) suffer from prohibitive latency ($\approx$ **3s**) due to the sequential item-by-item decoding process, making them impractical for real-time search. 2) **Traditional Cascade Structure:** The current production baseline employs a complex funnel (Query Processing $\rightarrow$ Recall $\rightarrow$ Pre-Ranking $\rightarrow$ Ranking), resulting in an accumulated latency of $\approx$ **130ms**. 3) **DaV-Gen (Ours):** By adopting a unified "Draft-and-Verify" strategy and leveraging Parallel Scoring, DaV-Gen reduces the latency to $\approx$ **70ms**. This represents a 2.5$\times$ speedup compared to the

Variant	Amazon Beauty		Amazon Sports	
	Recall@10	NDCG@10	Recall@10	NDCG@10
DaV-Gen (Full)	**0.0752**	**0.0481**	**0.0605**	**0.0382**
w/o Hybrid Rep.	0.0710	0.0445	0.0569	0.0355
w/o Prog. Inject.	0.0722	0.0458	0.0580	0.0366
w/o Gen. Loss	0.0718	0.0454	0.0575	0.0362
w/o Fusion Loss	0.0728	0.0462	0.0588	0.0371

Table 3: Ablation study on Amazon Beauty and Sports datasets.

traditional cascade and is orders of magnitude faster than pure generative approaches, demonstrating superior suitability for high-concurrency industrial environments.

3.5 Ablation Study (RQ4)

To verify the robustness of our design, we extend the ablation study to two datasets: Amazon Beauty and Amazon Sports. We investigate the contribution of key components by evaluating the following variants:

- **w/o Hybrid Rep.:** Removes the dense semantic vectors, relying solely on sparse ID matching.
- **w/o Prog. Inject.:** Skips Embedding Alignment of the progressive knowledge injection, directly proceeding to the full-parameter Deep Semantic Integration.
- **w/o Gen. Loss ($\mathcal{L}_{\text{gen}}$):** Removes the token-level generative objective, training the model only with contrastive and pairwise ranking losses.
- **w/o Fusion Loss ($\mathcal{L}_{\text{pair}}$):** Removes the fusion-based pairwise ranking objective, relying solely on the sum of contrastive and generative scores for inference.

The results across both datasets, summarized in Table 3, provide consistent and critical insights:

(1) Removing the dense vectors leads to the most significant performance drop (e.g., -5.6% Recall on Beauty). This validates that sparse semantic tokens alone are insufficient; combining them with dense embeddings significantly enriches the model's expressiveness and drafting coverage. (2) The variant skipping the alignment stage (*w/o Prog. Inject.*) suffers a notable decline. This demonstrates that the "semantic gap" between the PLM and Semantic IDs requires a progressive bridge. Without the initial embedding alignment to map ID tokens into the PLM's latent space, the subsequent full-parameter learning struggles to converge optimally. (3) The removal of $\mathcal{L}_{\text{gen}}$ results in clear degradation. This proves that the token-level generative objective is not merely auxiliary; it forces the model to learn fine-grained sequential dependencies and internal item semantics that contrastive learning alone cannot capture. (4) The collaborative matching fusion loss ensures that the generation objective aligns with the verification metric. Without it, we observe a consistent drop in NDCG, confirming its role in refining the final verification order.

4 Related Works

Existing approaches in recommender systems generally fall into two paradigms: the established MCAs and the emerging GenIR. DaV-Gen bridges these worlds, integrating the efficiency of cascades with the unified expressiveness of generative models.

Discriminative Multi-stage Cascading Architectures. MCAs have long been the industrial standard, handling massive corpora by decomposing the ranking problem into sequential stages [Qin *et al.*, 2022]. The retrieval stage prioritizes high-recall candidate selection via Maximum Inner Product Search (MIPS). This evolved from early Collaborative Filtering to deep dual-encoder models like DSSM [Huang *et al.*, 2013] and YoutubeDNN [Covington *et al.*, 2016], and later to sequential models like SASRec [Kang and McAuley., 2018] which capture dynamic user preferences. The subsequent ranking stage employs complex interaction-focused architectures, such as DIN [Zhou *et al.*, 2018] and SIM [Pi *et al.*, 2020], to capture fine-grained interests. However, the strict separation between these stages creates an "objective inconsistency" problem: the retriever's training objective is often misaligned with the ranker's, leading to irreversible error propagation [Hron *et al.*, 2021].

Generative Retrieval and Ranking. GenIR reformulates IR as a unified sequence-to-sequence generation task, directly predicting target item identifiers. Early approaches like P5 [Geng *et al.*, 2022] and M6-Rec [Cui *et al.*, 2022] leveraged Pre-trained Language Models (PLMs) to process raw text tokens. While semantically rich, these methods incurred prohibitive computational costs and struggled with context limits. To address this, the field shifted towards discrete Semantic IDs derived from Residual Quantized VAEs (RQ-VAE). Models like TIGER [Rajput *et al.*, 2023] and LC-Rec [Wang *et al.*, 2024] generate these hierarchical tokens auto-regressively, significantly improving the modeling of collaborative signals. Despite their effectiveness, the sequential dependency of token-by-token decoding imposes a severe latency bottleneck [Rajput *et al.*, 2023; Wang *et al.*, 2024]. Recent research has thus pivoted to address these limitations from different angles: RPG [Hou *et al.*, 2025] proposes parallel generation to break the decoding bottleneck; TBGRecall [Liang *et al.*, 2025] and REG4Rec [Xing *et al.*, 2025] enhance the model's ability to handle complex sessions and reasoning paths; and OneRec [Deng *et al.*, 2025] focuses on structurally unifying retrieval and ranking to solve objective inconsistency. Distinct from these efforts, DaV-Gen proposes a "Draft-and-Verify" paradigm, which uniquely combines the inference speed of retrieval-based drafting with the precision of parallel generative verification.

5 Conclusion

We presented DaV-Gen, refactoring generative retrieval from sequential generation to parallel verification. Our Hybrid Sparse-Dense Representation and "Draft-and-Verify" mechanism successfully resolve objective inconsistency in cascades and decoding latency in generative models. Extensive experiments confirm its superior accuracy and efficiency. Future work will extend the modality-agnostic Dense Content Vector with visual or audio embeddings for richer multimodal fusion.

Contribution Statement

Meng Zhao and Chunmei Liu contributed equally to this work and are co-first authors. Qinyong Wang is the corresponding author.

References

[Covington *et al.*, 2016] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *RecSys*, pages 191–198, 2016.

[Cui *et al.*, 2022] Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. M6-rec: Generative pretrained language models are open-ended recommender systems. *arXiv*, 2022.

[Deng *et al.*, 2025] Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment. *arXiv*, 2025.

[Geng *et al.*, 2022] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *RecSys*, pages 299–315, 2022.

[Hou *et al.*, 2025] Yupeng Hou, Jiacheng Li, Ashley Shin, Jinsung Jeon, Abhishek Santhanam, Wei Shao, Kaveh Hassani, Ning Yao, and Julian McAuley. Generating long semantic ids in parallel for recommendation. *arXiv*, 2025.

[Hron *et al.*, 2021] Jiri Hron, Karl Krauth, Michael Jordan, and Niki Kilbertus. On component interactions in two-stage recommender systems. *NeurIPS*, 34:2744–2757, 2021.

[Huang *et al.*, 2013] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *CIKM*, pages 2333–2338, 2013.

[Kang and McAuley., 2018] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *ICDM*, page 197–206, 2018.

[Liang *et al.*, 2025] Zida Liang, Weiqiang Sun, Jian Wu, Ke Chen, Changfa Wu, Ziyang Wang, Yuning Jiang, Silu Zhou, Dunxian Huang, Yuliang Yan, et al. Tbgrecall: A generative retrieval model for e-commerce recommendation scenarios. *arXiv*, 2025.

[Ma *et al.*, 2019] Chen Ma, Peng Kang, and Xue Liu. Hierarchical gating networks for sequential recommendation. In *KDD*, pages 825–833, 2019.

[Pi *et al.*, 2020] Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *CIKM*, pages 2685–2692, 2020.

[Qin *et al.*, 2022] Jiarui Qin, Jiachen Zhu, Bo Chen, Zhirong Liu, Weiwen Liu, Ruiming Tang, Rui Zhang, Yong Yu, and Weinan Zhang. Rankflow: Joint optimization of multi-stage cascade ranking systems as flows. In *SIGIR*, pages 814–824, 2022.

[Rajput *et al.*, 2023] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. Recommender systems with generative retrieval. *NeurIPS*, 36:10299–10315, 2023.

[Sun *et al.*, 2019] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*, pages 1441–1450, 2019.

[Wang *et al.*, 2024] Wenjie Wang, Xinyu Lin, Fuli Feng, Xiangnan He, and Tat-Seng Chua. Generative recommendation on user and item graphs. In *KDD*, 2024.

[Xing *et al.*, 2025] Haibo Xing, Hao Deng, Yucheng Mao, Lingyu Mu, Jinxin Hu, Yi Xu, Hao Zhang, Jiahao Wang, Shizhun Wang, Yu Zhang, et al. Reg4rec: Reasoning-enhanced generative model for large-scale recommendation systems. *arXiv*, 2025.

[Yang *et al.*, 2025] Yuhao Yang, Zhi Ji, Zhaopeng Li, Yi Li, Zhonglin Mo, Yue Ding, Kai Chen, Zijian Zhang, Jie Li, Shuanglong Li, et al. Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations. *arXiv*, 2025.

[Zhang *et al.*, 2025] Luankang Zhang, Kenan Song, Yi Quan Lee, Wei Guo, Hao Wang, Yawen Li, Huifeng Guo, Yong Liu, Defu Lian, and Enhong Chen. Killing two birds with one stone: Unifying retrieval and ranking with a single generative recommendation model. In *SIGIR*, pages 2224–2234, 2025.

[Zhou *et al.*, 2018] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. Deep interest network for click-through rate prediction. In *KDD*, pages 1059–1068, 2018.