GenPlugin: A Plug-and-Play Framework for Long-Tail Generative Recommendation with Exposure Bias Mitigation

Kun Yang[1], Siyao Zheng[2], Tianyi Li[1], Xiaodong Li[1], and Hui Li[1(✉)]

[1] Key Laboratory of Multimedia Trusted Perception and Efficient Computing, Ministry of Education of China, Xiamen University, Xiamen, China
{kyang,litianyi}@stu.xmu.edu.cn, {xdli,hui}@xmu.edu.cn
[2] School of Informatics, Xiamen University, Xiamen, China
zhengsiyao@stu.xmu.edu.cn

Abstract. Generative recommendation (GenRec) offers LLM integration, reduces embedding costs, and eliminates per-candidate scoring, attracting great attention. Despite its promising performance, this study reveals that it suffers from generation exposure bias and poor long-tail item generalization, two critical limitations overlooked by prior works on GenRec. To address these, we propose GenPlugin, a plug-and-play framework featuring a dual-encoder, shared-decoder architecture. During pre-training, it aligns language and ID views via contrastive learning, harmonizing item representations across two complementary views. Besides, GenPlugin uses a novel training strategy that probabilistically substitutes ground-truth item ID tokens with predictions from the language-semantics encoder, alleviating exposure bias. To improve long-tail generative recommendation, we propose a retrieval-based data augmentation mechanism. It fine-tunes the decoder of GenPlugin to endow GenPlugin with the ability to use relevant users w.r.t. contexts or collaborative information to augment the generation of item ID tokens in long-tail recommendation scenarios. We have plugged GenPlugin into several representative GenRec models and the extensive experiments demonstrate that GenPlugin can notably mitigate generation exposure bias during item ID generation while significantly improving the quality of long-tail item recommendation. The implementation is available at: https://github.com/KDEGroup/GenPlugin.

Keywords: Recommendation system · Sequential Recommendation · Generative Recommendation

1 Introduction

Recommender system (RecSys) is essential to numerous online services (e.g., Amazon, TikTok and YouTube) [19,22]. Traditionally, RecSys models rely on

K. Yang and S. Zheng—Contribute equally to this paper.

H. Jung et al. (Eds.): DASFAA 2026, LNCS 16535, pp. 504–519, 2026.
https://doi.org/10.1007/978-981-92-0363-5_31

the ID-based paradigm where user behaviors are formatted into item identifiers (item IDs) [8]. In this paradigm, items are uniquely identified by pre-defined numerical or string-based IDs (e.g., "1234"). For instance, deep learning based collaborative filtering methods [1] assign an embedding to each item ID and update these embeddings iteratively. During recommendation, RecSys calculates each candidate item's ranking score and offers items with the highest scores as the recommendation. Despite its widespread adoption over the years, the ID-based paradigm suffers from several limitations. Notably, it struggles with semantic gaps when integrating large language models (LLMs), as item IDs lack explicit, rich semantic information [36]. Additionally, handling a large number of items incurs high storage and processing costs of embeddings [8].

Generative recommendation (GenRec), a new recommendation paradigm, has recently gained substantial attention [8,12,18,36]. GenRec represents each item using a unique token sequence during the item creation phase. Each token (i.e., ID token) encodes semantic meaning and is designed to enhance LLMs' understanding of the underlying semantics. The number of ID tokens required is significantly smaller than the total number of items and GenRec can use finite tokens to represent almost infinite items [8], largely reducing the embedding cost. In the generative recommendation phase, GenRec directly generates the IDs of recommended items without the need to compute individual recommendation scores for each candidate item.

While GenRec has achieved notable progress, our empirical study in Sect. 2 reveals two challenges that hinder its practical deployment:

- **P1: Generation Exposure Bias.** Existing GenRec methods exhibit a critical limitation stemming from the discrepancy between training and inference phases. The generation is conditioned on ground-truth item ID tokens, meaning that the first $t-1$ ID tokens are given as input to generate the t-th ID token. However, during inference (the generative recommendation stage), predictions rely entirely on previously generated tokens, which may contain errors from prior steps. The gap between training and inference, termed generation exposure bias, leads to substantial discrepancies between generated and ground-truth item ID token sequences, a similar phenomenon to exposure bias in text generation [26]. *Notably, this limitation remains underexplored in existing GenRec studies.*
- **P2: Inadequate Generalization to Long-Tail Item Recommendation.** GenRec is expected to generalize to long-tail item recommendation [18], a challenging task in RecSys where items with few interactions suffer from low-quality recommendation [11]. The reason is that, for infrequent items, GenRec can understand their characteristics through their generative item ID tokens that are encoded with item semantics and are shared among different items. However, we find that, despite the potential of shared semantic representations to bridge data sparsity, existing GenRec methods struggle to handle long-tail item recommendation, indicating that solely relying on generative item IDs is insufficient for handling this task.

To combat the above problems, we propose GenPlugin, a plug-and-play framework for long-tail generative recommendation with exposure bias mitigation. The contributions of this work are:

- We introduce GenPlugin as a modular plugin featuring a dual-encoder, shared-decoder architecture. This design integrates contrastive semantics alignments during pre-training to harmonize item representations across two complementary views: language view (original item textual features) and ID view (generative item IDs from GenRec).
- To address **P1**, GenPlugin uses a novel training strategy that probabilistically substitutes ground-truth item ID tokens with predictions from the language-semantics encoder. By dynamically blending these signals during training, the model learns to robustly generate item IDs even when prior steps contain errors, effectively mitigating exposure bias.
- To alleviate **P2**, we propose a retrieval-based data augmentation mechanism used in GenPlugin. It fine-tunes the decoder of GenPlugin to endow GenPlugin with the ability to use relevant users w.r.t. contexts or collaborative information to augment the generation of item ID tokens in long-tail recommendation scenarios.
- We have plugged GenPlugin into several representative GenRec models. Extensive experiments demonstrate that GenPlugin can mitigate generation exposure bias during item ID generation while significantly improving the quality of long-tail item recommendation, validating GenPlugin's ability to serve as an easy-to-deploy, performance-enhancing plugin for GenRec.

2 Empirical Study of Existing GenRec

We take a representative GenRec method TIGER [18] as an example to illustrate the existence of generation exposure bias and poor long-tail item recommendation in current GenRec.

First, we report Hit Rate@10 (H@10) of TIGER [18] on six datasets in Table 1. In Table 1, TIGER w/ DS indicates enhancing TIGER with our proposed dual-encoder, shared-decoder plugin (Sect. 3.1) that adopts semantic-substitution guidance to mitigate generation exposure bias (Sect. 3.2). We can observe that, directly using ground-truth item ID tokens to guide model training (i.e., TIGER) significantly lags behind TIGER w/ DS that uses exposure bias mitigation strategy, showing that generation exposure bias indeed exists in GenRec and it negatively affects the quality of recommendation.

We further analyze the performance of TIGER on long-tail item recommendation. Figure 1 illustrates the recommendation performance w.r.t. different item groups on Amazon Beauty and items are binned into different groups according to their interaction counts. From Fig. 1, we can see that long-tail distribution exists: a large portion of items has only a few interactions and TIGER performs poorly on these groups. The empirical study shows that GenRec does not perform well on long-tail items, meaning that a substantial number of items cannot receive satisfactory recommendations (Fig. 1).

Table 1. H@10 of TIGER on six datasets. TIGER w/ DS indicates enhancing TIGER with exposure bias mitigation.

	TIGER	TIGER w/ DS
Beauty	0.0609	0.0790
Toys	0.0518	0.0824
Sports	0.0356	0.0467
Instruments	0.1211	0.1398
Arts	0.1225	0.1430
ML-100k	0.0901	0.1294

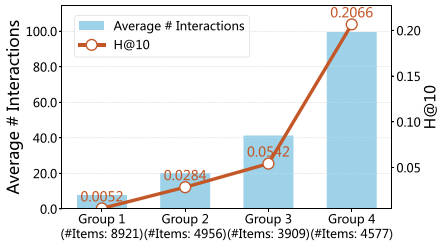

Fig. 1. Performance of TIGER on Amazon Beauty w.r.t. different item groups.

The above empirical study reveals the limitations of current GenRec and motivates us to design GenPlugin for addressing generation exposure bias and improving coverage for long-tail items.

3 Our Method GenPlugin

Figure 2 provides an overview of GenPlugin. GenPlugin adopts a dual-encoder, shared-decoder architecture. During pre-training, the two encoders integrate contrastive semantics alignment to better understand the generative item IDs produced by GenRec across two complementary views: language view and ID view (Sect. 3.1). The shared decoder is strengthened with a semantic-substitution guidance mechanism to alleviate exposure bias (Sect. 3.2). To improve the quality of long-tail item recommendation, we further fine-tune GenPlugin to endow it with the capability to utilize retrieved, relevant data as augmentation to bridge data sparsity, the key reason causing the low-quality long-tail item recommendation, so that GenPlugin can utilize auxiliary data to augment generative recommendation during the inference stage (Sect. 3.3).

3.1 Dual Encoders with Contrastive Semantics Alignments

For semantic encoding, GenPlugin adopts dual encoders, one for language semantics and the other for ID semantics. Each encoder are standard Transformers encoder [25]. The motivation for choosing a dual-encoder design is that generative ID tokens are out-of-vocabulary (OOV) tokens (e.g., "A_10" and "B_4") which are difficult for direct understanding, and using a language-semantics encoder to assist with the encoding process can help GenPlugin better capture the underlying ID semantics given the associated language descriptions.

The inputs to dual encoders are based on user historical interaction sequences (user sequences for short): $S_u = \{i_{u,1}, \cdots, i_{u,m}\}$ where $i_{u,j}$ indicates the j-th item interacted by user u and m is the maximum number of items in the user sequences. Each user sequence contains items interacted by the corresponding user in chronological order. The overall encoding process is actually *multi-view learning* of each user sequence.

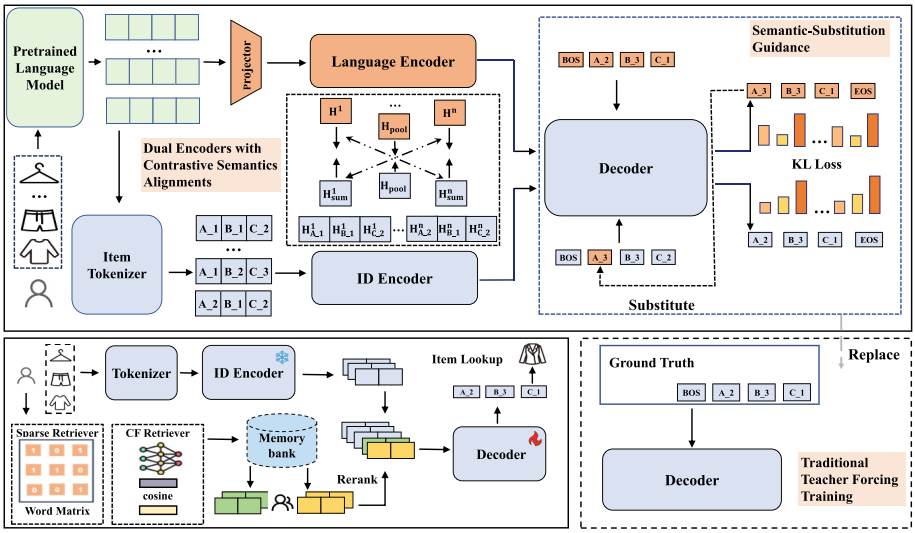

Fig. 2. Overview of GenPlugin

Language-Semantics Encoder. The inputs to the language-semantics encoder are the textual view of user sequences. We adopt LLM as the embedding extractor and pass the textual features of each item in a user sequence through LLM to get item embeddings as the inputs.

As the embedding dimensionalities of different LLMs vary, the extracted item embedding $\mathbf{e}_{\mathrm{i}}$ for the item i is further projected to $\hat{\mathbf{e}}_i$ in a new space through a two-layer feedforward neural network. Note that we freeze LLM and only update the parameters of the projection network.

Then, the user sequence, $\hat{\mathbf{E}}_u = \{\hat{\mathbf{e}}_{u,1}, \cdots, \hat{\mathbf{e}}_{u,m}\}$, represented by the projected embeddings of each item, is passed through the language-semantics encoder, resulting in the language representation of S_u: $H_u = \{\mathbf{h}_{u,1}, \cdots, \mathbf{h}_{u,m}\}$, where $\mathbf{h}_{u,j}$ is the encoded representation for the j-th item in S_u.

ID-Semantics Encoder. The inputs to the ID-semantics encoder are the ID view of user sequences, i.e., generative IDs. The user sequence is represented by ID tokens of each item in the sequence: $T_u = \{t_{u,1}^{(1)}, \cdots, t_{u,1}^{(k)}, \cdots, t_{u,m}^{(1)}, \cdots, t_{u,m}^{(k)}\}$, where k is the number of tokens in an item ID and $t_{u,j}^{(r)}$ indicates the r-th ID token of j-th item interacted by user u.

Then, T_u is passed through the ID-semantics encoder and the output is the ID representation of S_u: $C_u = \{\mathbf{c}_{u,1}^{(1)}, \cdots, \mathbf{c}_{u,1}^{(k)}, \cdots, \mathbf{c}_{u,m}^{(1)}, \cdots, \mathbf{c}_{u,m}^{(k)}\}$, where $\mathbf{c}_{u,j}^{(r)}$ indicated the encoded ID token representation for the r-th ID token of the j-th item in S_u.

Contrastive Semantic Alignments. As language-semantics encoder and ID-semantics encoder are encoding the same data from two different views, aligning

them for mutual benefit can further improve the encoded semantics. For this purpose, we design two contrastive cross-view semantic alignment tasks: contrastive item-level alignment and user preference alignment.

Contrastive item-level alignment aligns different views of each item. For each item, GenPlugin sums its ID token embeddings from ID-semantics encoder as its ID representation and aligns it with item language representation from language-semantics encoder via a contrastive loss:

$$\ell(\mathbf{u}, \mathbf{v}) = \sum_i \log \frac{\exp\big(\mathrm{sim}(\mathbf{u}_i, \mathbf{v}_i)/\tau\big)}{\sum_{j \in \mathrm{neg}(i)} \exp\big(\mathrm{sim}(\mathbf{u}_i, \mathbf{v}_j)/\tau\big)} \tag{1}$$

$$\mathcal{L}_{\mathrm{item}} = \ell(\mathbf{h_i}, \mathbf{g_i}) + \ell(\mathbf{g_i}, \mathbf{h_i}) \tag{2}$$

where $\mathbf{h}_i$ and $\mathbf{g}_i = \mathrm{sum}\big(\{\mathbf{c}_i^{(1)} \cdots \mathbf{c}_i^{(g)}\}\big)$ indicate the language representation and the ID representation of item i, respectively. $sim(*, *)$ denotes the inner product operation and τ is a temperature hyperparameter. $\mathrm{Neg}(i)$ indicates the negative items of item i (i.e., all items in user sequences from the same batch).

Contrastive user-preference alignment aligns user preferences learned through two encoders. For the same user sequence S_u, the user preference patterns extracted from different views should be close to each other. For each user sequence, GenPlugin performs average pooling on the outputs of the two encoders, H_u and C_u, and gets $\mathbf{p}_u$ and $\mathbf{q}_u$ as the user's preference representation for language view and ID view, respectively. $\mathbf{p}$ and $\mathbf{q}$ are further aligned through a contrastive loss similar to Eq. 2:

$$\mathcal{L}_{\mathrm{user}} = \ell(\mathbf{p_u}, \mathbf{q_u}) + \ell(\mathbf{q_u}, \mathbf{p_u}) \tag{3}$$

Optimizing Eq. 2 and Eq. 3 enhances the complementarity between the language and ID views, yielding better representations for GenPlugin and improving the subsequent retrieval-augmented generative recommendation.

3.2 Shared Decoder with Semantic-Substitution Guidance

To address the inherent exposure bias in generative recommendation caused by Teacher Forcing, a naive solution can involve a two-pass decoding process [15]. The first pass generates predictive probabilities, which are then used to guide the second decoding pass by replacing the ground-truth ID tokens. However, the output of the first decoding pass merely serves as a substitute for the ground truth, offering limited additional benefit to the learning process.

Hence, we propose a Shared Decoder with Semantic-Substitution Guidance method. This approach achieves efficient bias mitigation through the collaborative training of Semantic view and an ID view on the foundation of a shared decoder. The core of this method lies in a dual-level synergy mechanism: first, probabilistic semantic-substitution guidance at the decoder's input; second, forced alignment of predictive distributions at the model's objective end.

Specifically, we first design a *Probabilistic Semantic-Substitution Guidance* mechanism to break the rigid constraints of traditional teacher forcing. At each

decoding timestep for the ID view, the model follows the conventional teacher forcing path with a probability of p_1, using the ground-truth ID as input. Concurrently, with a probability of $1 - p_1$, it switches to a semantic guidance path.

Along this guidance path, we further introduce a token-level random substitution strategy: each ground-truth ID token in the sequence has a probability of $1 - p_2$ of being replaced by its corresponding predicted token from the language view, thereby generating a "Semantic Substitute". To efficiently generate this guidance signal, we only perform a local softmax normalization on the top-q candidates with the highest probabilities in the language view's output logits. We then compute their weighted average embedding. This lightweight design avoids the computational overhead of a full-vocabulary calculation while efficiently condensing key semantic information into a single guidance vector, which is then injected into the shared decoder to guide the generation process of the ID view.

Besides input guidance, we employ Distributional Alignment at the objective layer to foster a consensus in semantic understanding between the two views. We begin by sharpening the original logit distributions of both views using a temperature coefficient ϕ to accentuate the influence of high-probability tokens. The sharpened probability distribution $P_{i,l}(c)$ is defined as:

$$P_{i,l}(c) = \frac{\exp(z_c/\phi)}{\sum_k \exp(z_k/\phi)} \tag{4}$$

where z_c is the logit value for class c. Subsequently, we minimize the discrepancy between the predictive distributions of the two views using a symmetric Kullback-Leibler (KL) Divergence loss function, thereby promoting bidirectional alignment:

$$\mathcal{L}_{\text{kl}} = \sum_i \sum_{l=1}^{k} \left(D_{\text{kl}}(P_{i,l}^{\text{lan}} \parallel P_{i,l}^{\text{id}}) + D_{\text{kl}}(P_{i,l}^{\text{id}} \parallel P_{i,l}^{\text{lan}}) \right) \tag{5}$$

where $P_{i,l}^{\text{lan}}$ and $P_{i,l}^{\text{id}}$ represent the temperature-adjusted probability distributions (vectors over the vocabulary) for the l-th token in the item ID of i, derived from the language and ID views, respectively. This design establishes a Mutual Distillation mechanism: the language view's soft predictions guide the ID view, and the ID view's outputs in turn constrain the language view. This bidirectional alignment promotes mutual learning between the two views and helps reduce the exposure bias introduced by teacher forcing.

3.3 Retrieval-Augmented Generative Recommendation

After pre-training, GenPlugin acquires the capability to offer generative recommendations. We design a retrieval-augmented generative recommendation mechanism that employs retrieval-based data augmentation to address data scarcity, the primary factor behind its suboptimal performance on long-tail items. The mechanism mainly consists of a dual-path retrieval component and a re-ranking with ID semantics component.

In the dual-path retrieval component, to augment the generative recommendation, we design two complementary retrieval strategies to retrieve more supplementary data for a user sequence:

- **Content-Aware Retrieval.** We construct a pseudo-document for each user by concatenating the textual features (e.g., titles and descriptions) of the items that the user has interacted with. The pseudo-document reflects the user's preferences in the language space. For a target user u, we apply BM25 [20] to retrieve the top-z users whose pseudo-documents are most similar to u's pseudo-document.
- **Collaborative Information Retrieval.** While content-aware retrieval returns similar users in the language space, the retrieval results lack the guidance of collaborative signals. To complement this, we train SASRec [4], a sequential recommender, to encode the sequential user-item interaction patterns. Given the user sequence S_u for a user u, SASRec applies self-attention to model the temporal dynamics and outputs a latent representation $\mathbf{g}_u$ that reflects u's collaborative profile. For a target user, we retrieve the top-z most similar users based on cosine similarity among users' collaborative profiles.

The dual-path retrieval module returns $2z$ users that are similar to the target user of the given user sequence, either in language space or collaborative representation space.

However, the retrieval list from dual-path retrieval may include noisy matches. To refine the augmentation list, we design a re-ranking component to re-rank the retrieval results according to ID semantics, i.e., the learned semantic space for generative IDs (i.e., $\mathbf{q}$ in Eq. 3). Specifically, we compute cosine similarity scores between the target user's representation and each representation of the user in the retrieval list and sort them accordingly. After re-ranking, we obtain a set of v users with the largest similarities to the target user, and they can be used as data augmentation to enhance generative recommendation. Note that, to preserve diverse signals, users retrieved by both content-aware retrieval and collaborative information retrieval are always retained in the final list, regardless of their similarity scores w.r.t. the target user.

Fine-Tuning and Inference. Given that the encoder has a time complexity of $\mathcal{O}(md^2 + m^2d)$, where m denotes the user sequence length and d is the representation dimensionality, directly concatenating the interaction histories of v users in the re-ranking list with that of the target user as the input to encoder will incur approximately v times higher computational overhead.

To address this issue, we cache the preference representations ($\mathbf{q}$ in Eq. 3) generated for users during the re-ranking stage. This design effectively avoids redundant computation in the encoder caused by multi-user input. During fine-tuning, we first freeze the encoder to ensure that each user's representation remains consistent with its pre-trained version, thereby maintaining cross-stage alignment. For each target user, we encode the corresponding user sequence using the ID-semantics encoder, the same operation as in re-ranking. At the same time, we retrieve the preference representations of similar users from the cache. These

Table 2. Statistics of the preprocessed datasets.

Dataset	#Users	#Items			#Inters.
		Total	Head	Tail	
Beauty	22,363	12,101	2,420	9,681	198,360
Toys	19,412	11,924	2,384	9,540	167,526
Sports	35,598	18,357	3,671	14,686	296,175
Instruments	17,112	6,250	1,250	5,000	136,226
Arts	22,171	9,416	1,883	7,533	174,079
MovieLens-100k	943	1,682	337	1,345	100,000

representations are then concatenated with the target user's representation, and the results are fed into the decoder for fine-tuning.

The retrieval augmentation process for the inference stage follows the same procedure as the fine-tuning stage, except that model parameters are not updated during inference.

3.4 Putting All Together

When training GenPlugin together with the underlying GenRec, the overall loss function is as follows:

$$\mathcal{L} = \mathcal{L}^{\text{lan}}(\{\mathbf{E}\}) + \mathcal{L}^{\text{id}}(\{\mathbf{T}\}) + \lambda_1 \mathcal{L}_{\text{item}} + \lambda_2 \mathcal{L}_{\text{user}} + \lambda_3 \mathcal{L}_{\text{kl}}, \tag{6}$$

where λ_1, λ_2 and λ_3 are loss weights. $\mathcal{L}^{\text{lan}}(\{\mathbf{E}\})$ indicates the training loss for the language view where the inputs $\{\mathbf{E}\}$ are item embeddings generated by the embedding extractor (LLM). $\mathcal{L}^{\text{id}}(\{T\})$ denotes the training loss for the ID view where the inputs $\{T\}$ are user sequences represented by item ID tokens.

4 Experiment

4.1 Experiment Settings

Dataset. We conduct experiments on six public datasets. Five of them come from the Amazon Product Review corpus [16], namely Beauty, Toys and Games (Toys), Sports and Outdoors (Sports), Musical Instruments (Instruments), and Arts. The sixth dataset is MovieLens-100k (ML-100k). We follow prior work on generative recommendation [18,27,34] for data preprocessing. To analyze long-tail recommendation performance, we split items into head and tail groups according to popularity following [11]. Summary statistics for all datasets are reported in Table 2.

Evaluation Metrics. We evaluate using Hit Rate (H@k) and Normalized Discounted Cumulative Gain (N@k), and report results at cutoffs $k \in \{5, 10\}$.

Table 3. Overall performance. "++" indicates the method is enhanced with GenPlugin. The block indicates the statistically significant improvements (i.e., two-sided t-test with $p < 0.05$) over the backbone.

Dataset	Metric	Classical					Generative					
		GRU4Rec	SASRec	Bert4Rec	S^3-Rec	LLM-ESR	TIGER	TIGER++	LETTER	LETTER++	MQL4GRec	MQL4GRec++
Beauty	H@5	0.0412	0.0350	0.0303	0.0356	0.0406	0.0390	**0.0557**	0.0419	**0.0568**	0.0446	**0.0544**
	N@5	0.0264	0.0207	0.0198	0.0212	0.0259	0.0258	**0.0369**	0.0281	**0.0393**	0.0292	**0.0373**
	H@10	0.0590	0.0536	0.0486	0.0614	0.0731	0.0604	**0.0815**	0.0664	**0.0839**	0.0702	**0.0809**
	N@10	0.0301	0.0278	0.0236	0.0309	0.0337	0.0323	**0.0463**	0.0363	**0.0482**	0.0375	**0.0454**
Toys	H@5	0.0363	0.0345	0.0246	0.0355	0.0412	0.0318	**0.0560**	0.0379	**0.0581**	0.0421	**0.0503**
	N@5	0.0246	0.0224	0.0163	0.0232	0.0249	0.0208	**0.0367**	0.0250	**0.0379**	0.0257	**0.0327**
	H@10	0.0570	0.0531	0.0426	0.0545	0.0719	0.0521	**0.0840**	0.0609	**0.0885**	0.0637	**0.0779**
	N@10	0.0321	0.0276	0.0194	0.0285	0.0347	0.0276	**0.0475**	0.0325	**0.0478**	0.0328	**0.0415**
Sports	H@5	0.0223	0.0150	0.0147	0.0236	0.0218	0.0225	**0.0319**	0.0230	**0.0344**	0.0233	**0.0253**
	N@5	0.0148	0.0078	0.0088	0.0154	0.0141	0.0144	**0.0206**	0.0146	**0.0227**	0.0152	**0.0165**
	H@10	0.0359	0.0260	0.0263	0.0365	0.0354	0.0357	**0.0490**	0.0380	**0.0513**	0.0390	**0.0415**
	N@10	0.0187	0.0113	0.0122	0.0192	0.0164	0.0187	**0.0263**	0.0194	**0.0283**	0.0197	**0.0218**
Instruments	H@5	0.0955	0.0913	0.0832	0.0921	0.0987	0.1014	**0.1164**	0.1041	**0.1174**	0.1086	**0.1129**
	N@5	0.0767	0.0724	0.0658	0.0687	0.0791	0.0897	**0.1004**	0.0916	**0.1000**	0.0929	**0.0979**
	H@10	0.1188	0.1201	0.1075	0.1115	0.1321	0.1214	**0.1412**	0.1289	**0.1443**	0.1278	**0.1389**
	N@10	0.0846	0.0815	0.0726	0.0741	0.0912	0.0953	**0.1075**	0.0992	**0.1100**	0.1007	**0.1058**
Arts	H@5	0.0826	0.0946	0.0727	0.0759	0.0992	0.0956	**0.1173**	0.0953	**0.1131**	0.1068	**0.1119**
	N@5	0.0613	0.0683	0.0524	0.0543	0.0743	0.0768	**0.0945**	0.0773	**0.0910**	0.0862	**0.0910**
	H@10	0.1101	0.1256	0.0945	0.1042	0.1341	0.1223	**0.1445**	0.1232	**0.1460**	0.1351	**0.1446**
	N@10	0.0702	0.0728	0.0597	0.0642	0.0786	0.0853	**0.1028**	0.0854	**0.1012**	0.0960	**0.1010**
ML-100k	H@5	0.0636	0.0721	0.0551	0.0782	0.0804	0.0509	**0.0785**	0.0615	**0.0922**	0.0658	**0.0859**
	N@5	0.0391	0.0423	0.0303	0.0493	0.0488	0.0320	**0.0514**	0.0386	**0.0576**	0.0400	**0.0501**
	H@10	0.1230	0.1166	0.1007	0.1271	0.1248	0.0901	**0.1336**	0.1208	**0.1484**	0.1188	**0.1463**
	N@10	0.0580	0.0566	0.0449	0.0548	0.0606	0.0449	**0.0690**	0.0575	**0.0759**	0.0566	**0.0696**

Backbones and Baselines. We select three representative generative recommendation (GenRec) models as backbones: TIGER [18]: It introduces codebook-based item IDs based on RQ-VAE. LETTER [27]: Through a set of regularization techniques, LETTER integrates collaborative filtering signals into item ID. MQL4GRec [34]: Unifies multimodal items via a quantitative language to bridge domains and enhance generative recommendation. Note that our goal is to demonstrate the effectiveness of GenPlugin when plugged into GenRec. To reduce the experimental cost of pre-training, we utilize the multi-modal GenRec MQL4GRec without pre-training.

We further include five traditional recommendation models: GRU4Rec [2]: Built on Recurrent Neural Networks, this approach leverages Gated Recurrent Units to encode user interaction sequences. SASRec [4]: Utilizing a unidirectional Transformer encoder, SASRec represents users by the final item in their interaction sequence. BERT4Rec [23]: This framework employs bidirectional self-attention for masked item prediction, enhancing sequential recommendation. S^3Rec [37]: By incorporating four auxiliary self-supervised tasks, S3Rec learns richer item-attribute relationships in sequential data. LLM-ESR [11]: It addresses the long-tail problem by simultaneously leveraging collaborative signals and semantic information through the dual-view modeling and self-distillation.

Implementation Details. We generate textual representations by encoding the item title and description with LLaMA2 [24], where the mean pooling of the hidden states from the last layer is utilized as the final embedding. We obtain

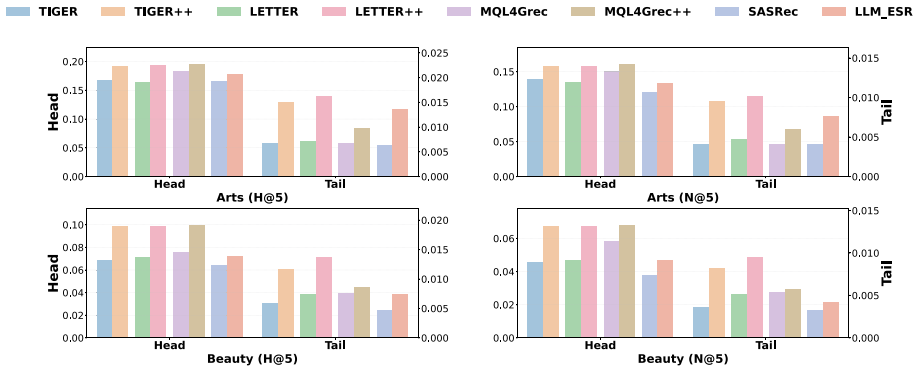

Fig. 3. Performance comparison on head and tail items.

item visual representations by following MQL4GRec [34] to encode item images. Each encoder and decoder in GenPlugin has 4 Transformer layers with 6 self-attention heads. The dimensionality of each head is 64. We set the FFN dimension to 1024 and the token embedding dimension to 128. The maximum item sequence length is 20 and the batch size is 512. We tune the hyperparameters by monitoring the validation loss, where an early stopping criterion with a patience of 20 is adopted to prevent over-training and facilitate better generalization.

4.2 Experimental Results

Overall Performance. The overall results are presented in Table 3, from which we have the following observations: GenRec methods exhibit superior performance compared to traditional approaches. This advantage primarily stems from the fact that generative methods, in constructing IDs, utilize both semantic information and a hierarchical structuring process. When enhanced with GenPlugin, all GenRec models achieve significant improvements. For instance, on the Beauty dataset, GenPlugin improves the TIGER backbone by 34.9% in H@10 and 43.3% in N@10. Furthermore, we observe that as the performance of the underlying GenRec improves, the enhancements brought by GenPlugin generally increase. Specifically, LETTER++ achieves 1.9% and 6.5% gains in H@5 and N@5 over TIGER++ on Beauty dataset.

Effect on Long-Tail Item Recommendation. Figure 3 compares performance on the head and tail item groups for the Arts and Beauty datasets. Across both groups and on both H@5 and N@5, GenPlugin consistently improves all three GenRec backbones and surpasses the traditional long-tail mitigation method LLM-ESR for both head and tail items. For head items, our method outperforms each backbone on all datasets, showing it is effective even for popular items. For tail items, it yields large gains; for instance, on Arts, TIGER++ improves H@5 by 54% over TIGER, highlighting strong long-tail performance.

Table 4. The ablation study on the Toys dataset and Arts dataset with LETTER as the backbone model. Results are reported in terms of H@10 and N@10.

Model	Toys						Arts					
	Overall		Head		Tail		Overall		Head		Tail	
GenPlugin	**0.0889**	**0.0478**	**0.1453**	**0.0808**	**0.0415**	**0.0212**	**0.1463**	**0.1025**	**0.2417**	**0.1736**	**0.0257**	**0.0132**
w/o DSA	0.0702	0.0366	0.1197	0.0629	0.0281	0.0147	0.1412	0.0962	0.2373	0.1649	0.0214	0.0104
w/o SSG	0.0797	0.0432	0.1305	0.0733	0.0361	0.0174	0.1417	0.0980	0.2391	0.1686	0.0212	0.0105
w/o RAR	0.0832	0.0442	0.1409	0.0797	0.0375	0.0180	0.1426	0.1004	0.2406	0.1724	0.0209	0.0103

Ablation Study. As shown in Table 4, removing any single component leads to consistent and interpretable drops: (*i*) cross-view alignment (DSA) is foundational, producing the largest across-the-board gains; (*ii*) semantic-substitution guidance (SSG) stabilizes the generative pipeline by mitigating exposure bias; and (*iii*) retrieval augmentation (RAR) primarily improves long-tail performance.

Together, DSA and SSG underpin our generative pipeline. DSA strengthens cross-view representations by aligning language and ID views, while SSG reduces train–inference mismatch by substituting teacher-forced ID tokens with high-confidence semantic tokens and applying bidirectional KL at the output layer. Removing either module leads to pronounced overall performance drops across datasets. Using Toys as a representative case, discarding DSA reduces Overall H@10 by 19.9%, and removing SSG lowers it by 10.3%. These results indicate that DSA supplies the core cross-view semantics and SSG stabilizes optimization; both are essential for strong overall accuracy. RAR performs dual-path retrieval (content-aware and collaborative), re-ranks candidates in the ID semantics space, and injects a few high-quality neighbor signals into the decoder to strengthen sparse regions. Removing RAR disproportionately harms the tail: on Toys, Tail H@10 drops by 9.6% versus 3.0% on Head. This gap confirms RAR's role as a tail specialist, complementing the global benefits brought by DSA and SSG.

Analysis of Hyper-parameters. Taking TIGER as an example for underlying GenRec and Toys as the test dataset, we further investigate the effects of hyper-parameters on GenPlugin and the results are reported in Fig. 4.

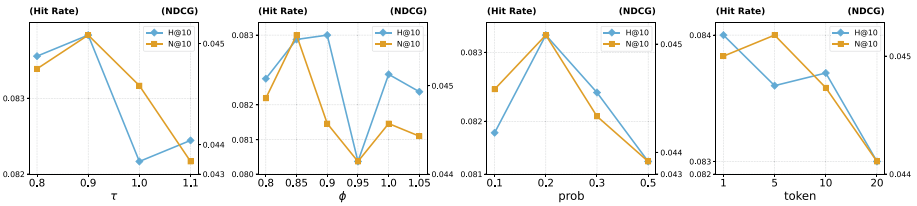

Fig. 4. Performance of TIGER++ over different hyper-parameters on Toys.

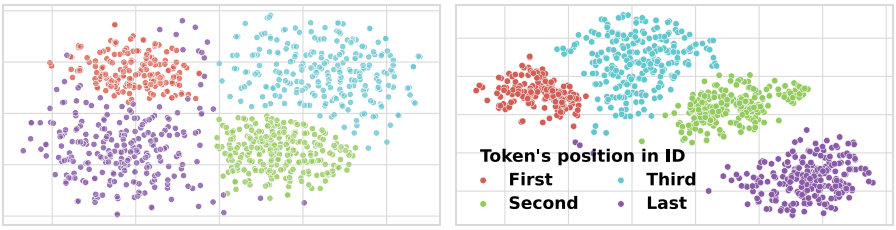

Fig. 5. OOV Token Representations Visualization.

The hyperparameter τ controls the sharpness of the contrastive loss. Moderate values yield the best performance. A too low τ causes the model to overemphasize a few negative samples, while a too high τ overly smooths distinctions and weakens discriminative power. The hyperparameter ϕ adjusts the softness of the KL divergence. Smaller ϕ values create harder, more peaked targets, whereas larger ϕ values produce softer distributions that capture richer but noisier relations. Performance peaks before ϕ is 0.9, beyond which the alignment quality declines. The hyperparameter Replacement Probability (prob) performs best around 0.2; a moderate replacement rate enhances robustness by introducing controlled noise, while higher rates lead to instability and degraded performance. Finally, the hyperparameter q which specifies the number of fusion tokens achieves optimal results around five tokens: a few highly relevant tokens improve recommendation quality, but too many introduce noise and reduce effectiveness.

Visualization. Figure 5 visualizes the representation space of added out-of-vocabulary tokens using t-SNE [14] with LETTER as the underlying GenRec on Toys. We can see that our proposed SSG method produces more tightly clustered representations at each position compared to using only teacher forcing. This clustering leads to a more robust and consistent semantic space, effectively improving the model's capacity to mitigate exposure bias.

5 Related Work

5.1 Generative Recommendation (GenRec)

GenRec overcomes limitations of ID-based RecSys by leveraging generative models to directly generate recommendations without explicit candidate scoring [8]. Pioneering methods like TIGER [18] and LC-Rec [36] employ vector quantization (e.g., RQ-VAE [33]) to convert item features into discrete tokens, enabling LLM-driven recommendation generation through token sequence modeling. This paradigm shift has spurred extensive research across three directions:

– Collaborative Semantics Alignment: Works bridge collaborative and language semantics via instruction tuning [5,21,36] or contrastive learning [3,27–29], aligning behavioral patterns with language representations.

- Stage Unification: Methods simplify multi-stage pipelines by integrating ID creation and recommendation. STORE [13] uses a unified LLM framework, TTDS [32] employs dual codebooks for joint user-item quantization, LIGER [31] unified generative and dense retrieval and ETEGRec [9] couples stages via sequence-item alignment.
- Multi-modal Extensions: Recent efforts enable cross-modal understanding through techniques like dual-aligned quantization [6], graph-enhanced RQ-VAE [10], behavior-semantic collaboration [28], multimodal preference discerner enhanced recommendation [17] and modality-agnostic vocabulary translation [34], preserving synergies between heterogeneous features.

5.2 Retrieval-Augmented Recommendation

Retrieval-augmented generation (RAG) enables dynamic incorporation of external knowledge to improve accuracy and interoperability and its idea has been applied in several RecSys models. For instance, RUEL [30] leverages behavior-sequence retrieval with item-level attention to refine recommendations. Ada-Retrieval [7] employs adaptive multi-round retrieval to iteratively update user/item representations with contextual information. RaSeRec [35] introduces a two-stage framework where cross-attention dynamically fuses retrieved user-item representations, achieving better performance. These methods highlight RAG's effectiveness in bridging generative capabilities with precise knowledge retrieval for improving recommendation quality.

6 Conclusion

In this paper, we present GenPlugin, a plug-and-play framework for generative recommendation that addresses exposure bias and long-tail challenges. GenPlugin adopts a dual-encoder, shared-decoder architecture, aligning language and ID semantics via contrastive pretraining. To mitigate exposure bias, it introduces a probabilistic ID replacement strategy during training. For better long-tail generalization, it incorporates retrieval-based augmentation to enhance recommendation. Through the experiments, we verify the effectiveness and flexibility of our GenPlugin.

Acknowledgments. This work was supported by National Natural Science Foundation of China (No. 62572410), Natural Science Foundation of Xiamen, China (No. 3502Z202471028). Xiaodong Li was supported by Xiamen Science and Technology Project (No. 3502Z202571028) and the Fundamental Research Funds for the Central Universities, Xiamen University (No. 20720250171).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.: Neural collaborative filtering. In: WWW, pp. 173–182 (2017)
2. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: ICLR (2016)
3. Hong, M., et al.: EAGER-LLM: enhancing large language models as recommenders through exogenous behavior-semantic integration. In: WWW, pp. 2754–2762 (2025)
4. Kang, W., McAuley, J.J.: Self-attentive sequential recommendation. In: ICDM, pp. 197–206 (2018)
5. Kim, T., Yoon, S., Kang, S., Yeo, J., Lee, D.: SC-Rec: enhancing generative retrieval with self-consistent reranking for sequential recommendation. arXiv Preprint arXiv:2408.08686 (2024)
6. Li, K., et al.: BBQRec: behavior-bind quantization for multi-modal sequential recommendation. arXiv Preprint arXiv:2504.06636 (2025)
7. Li, L., Lian, J., Zhou, X., Xie, X.: Ada-retrieval: an adaptive multi-round retrieval paradigm for sequential recommendations. In: AAAI, pp. 8670–8678 (2024)
8. Li, L., Zhang, Y., Liu, D., Chen, L.: Large language models for generative recommendation: a survey and visionary discussions. In: COLING, pp. 10146–10159 (2024)
9. Liu, E., Zheng, B., Ling, C., Hu, L., Li, H., Zhao, W.X.: End-to-end learnable item tokenization for generative recommendation. arXiv Preprint arXiv:2409.05546 (2024)
10. Liu, H., Wei, Y., Song, X., Guan, W., Li, Y., Nie, L.: MMGRec: multimodal generative recommendation with transformer model. arXiv Preprint arXiv:2404.16555 (2024)
11. Liu, Q., et al.: LLM-ESR: large language models enhancement for long-tailed sequential recommendation. In: NeurIPS (2024)
12. Liu, Q., et al.: Vector quantization for recommender systems: a review and outlook. arXiv Preprint arXiv:2405.03110 (2024)
13. Liu, Q., Zhu, J., Fan, L., Zhao, Z., Wu, X.: STORE: streamlining semantic tokenization and generative recommendation with a single LLM. arXiv Preprint arXiv:2409.07276 (2024)
14. Maaten, L.V.D., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. 2579–2605 (2008)
15. Mihaylova, T., Martins, A.F.T.: Scheduled sampling for transformers. In: Annual Meeting of the Association for Computational Linguistics (ACL) (2019)
16. Ni, J., Li, J., McAuley, J.J.: Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In: EMNLP-IJCNLP, pp. 188–197 (2019)
17. Paischer, F., et al.: Preference discerning with LLM-enhanced generative retrieval. arXiv Preprint arXiv:2412.08604 (2024)
18. Rajput, S., et al.: Recommender systems with generative retrieval. In: NeurIPS (2023)
19. Ricci, F., Rokach, L., Shapira, B. (eds.): Recommender Systems Handbook. Springer, USA (2022)
20. Robertson, S.E., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond. Found. Trends Inf. Retr. **3**(4), 333–389 (2009)
21. Shao, M., Huang, H., Peng, Q., Liu, H.: ULMRec: user-centric large language model for sequential recommendation. arXiv Preprint arXiv:2412.05543 (2024)

22. Smith, B., Linden, G.: Two decades of recommender systems at amazon.com. IEEE Internet Comput. **21**(3), 12–18 (2017)
23. Sun, F., et al.: BERT4Rec: sequential recommendation with bidirectional encoder representations from transformer. In: CIKM, pp. 1441–1450 (2019)
24. Touvron, H., et al.: LLaMA 2: open foundation and fine-tuned chat models. arXiv Preprint arXiv:2307.09288 (2023)
25. Vaswani, A., et al.: Attention is all you need. In: NIPS, pp. 5998–6008 (2017)
26. Wang, C., Sennrich, R.: On exposure bias, hallucination and domain shift in neural machine translation. In: ACL, pp. 3544–3552 (2020)
27. Wang, W., et al.: Learnable item tokenization for generative recommendation. In: CIKM, pp. 2400–2409 (2024)
28. Wang, Y., et al.: EAGER: two-stream generative recommender with behavior-semantic collaboration. In: KDD, pp. 3245–3254 (2024)
29. Wang, Y., et al.: Content-based collaborative generation for recommender systems. In: CIKM, pp. 2420–2430 (2024)
30. Wu, N., Gong, M., Shou, L., Pei, J., Jiang, D.: RUEL: retrieval-augmented user representation with edge browser logs for sequential recommendation. In: CIKM, pp. 4871–4878 (2023)
31. Yang, L., et al.: Unifying generative and dense retrieval for sequential recommendation. arXiv Preprint arXiv:2411.18814 (2024)
32. Yin, J., et al.: Unleash LLMs potential for recommendation by coordinating twin-tower dynamic semantic token generator. arXiv Preprint arXiv:2409.09253 (2024)
33. Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., Tagliasacchi, M.: SoundStream: an end-to-end neural audio codec. IEEE ACM Trans. Audio Speech Lang. Process. **30**, 495–507 (2022)
34. Zhai, J., et al.: Multimodal quantitative language for generative recommendation. In: ICLR (2025)
35. Zhao, X., et al.: RaSeRec: retrieval-augmented sequential recommendation. arXiv Preprint arXiv:2412.18378 (2024)
36. Zheng, B., et al.: Adapting large language models by integrating collaborative semantics for recommendation. In: ICDE, pp. 1435–1448 (2024)
37. Zhou, K., et al.: S3-Rec: self-supervised learning for sequential recommendation with mutual information maximization. In: CIKM, pp. 1893–1902 (2020)