Towards Pareto Optimality in Popularity Debiasing for Generative Recommender Systems

Jun Yin[1]*, Bangguo Zhu[3]*, Peng Huo[4], Ruochen Liu[5], Hao Chen[6], Senzhang Wang[3]
Shirui Pan[2]†, Chengqi Zhang[1]†
[1]Hong Kong Polytechnic University, [2]Griffith University, [3]Central South University
[4]National Super Computing Center in Tianjin, [5]HKUST (Guangzhou), [6]City University of Macau
Junmay.yin@connect.polyu.hk, 8210231122@csu.edu.cn
s.pan@griffith.edu.au, Chengqi.zhang@polyu.edu.hk

Abstract

Recently, Generative Recommenders (GRs), characterized by a unified end-to-end framework, have exhibited astonishing potential in transforming the recommendation paradigm. Despite their effectiveness, we recognize that GRs are still susceptible to the long-standing issue of popularity bias that has pervaded the recommendation community. Although a few studies have attempted to extend traditional debiasing methods to GRs, their effectiveness is marginal, and the fundamental reason why GRs suffer from popularity bias remains under-explored. To bridge this gap, this study focuses on two core aspects in GRs: the optimization of generative framework and the item tokenization based on semantic index. Based on theoretical analyses, we identify that the severe popularity bias emerges from the confluence of a token-level optimization flaw and the undifferentiated property of item tokenization. Accordingly, this study develops a novel generative recommender system, called Ghost, by designing the asymmetric unlikelihood optimization and the skeleton-founded tokenization. Extensive empirical evaluations across three datasets, alongside multiple SOTA baselines, reveal that Ghost substantially alleviates popularity bias and promotes fairer recommendations, while incurring slight degradation to the overall recommendation utility.

1 Introduction

Recommender systems, due to the ability to capture user preferences by analyzing historical interactions, are essential for enhancing user experiences across platforms such as e-commerce [1], video streaming [2], and social networks [3]. Recently, generative recommenders (GRs) have emerged as a transformative paradigm [4, 5, 6, 7, 8] by replacing traditional item IDs [9, 10] with semantic indices (SIDs) [11, 12]. Driven by the adoption of large language models (LLMs) [13, 14, 15] as the backbone, the architecture of state-of-the-art (SOTA) GRs appears to become increasingly well-established, ushering in an era of rapid advancement.

However, despite their significant improvements in recommendation performance, we identify that current leading GRs remain plagued by the persistent problem of popularity bias [16, 17, 18], which has long affected the recommender systems field [19]. As illustrated in Figure 1, three state-of-the-art (SOTA) GRs, LETTER [6], LC-Rec [5], and ED^2 [7], all exhibit an extreme over-recommendation phenomenon for popular head items while struggling to accurately predict the niche tail items. In particular, Figure 1*a*) showcases the significant gap between the performance of head items and tail items, which is up to *45 times* greater for LC-Rec. Figure 1*b*) presents the number of head

*Equal contribution and co-first authors.
†Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

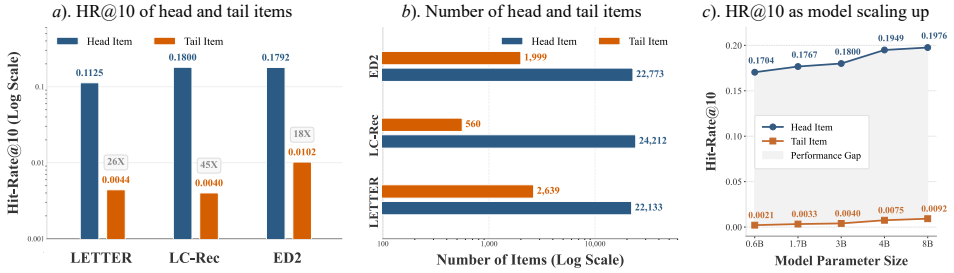

Figure 1: *a*). Comparison of Hit-Rate@10 (i.e., HR@10) between head and tail items. *b*). Comparison between the number of head and tail items in the recommendation list provided by three GRs. *c*). Tendency of HR@10 as the backbone parameters of LC-Rec scaling up.

and tail items in recommendation lists and reveals an obvious preference for popular items, which account for *more than 97 percent* of the recommendation lists. Furthermore, according to Figure 1*c*), as the backbone scale of LC-Rec increases *from 0.6B to 8B*, the popularity bias is not mitigated spontaneously. Consequently, the popularity bias in GRs precipitates a filter bubble, wherein trending content monopolizes visibility while high-quality, long-tail items are severely marginalized.

Although a few preliminary studies [19] have attempted to extend conventional methods of popularity debiasing, such as item re-weighting [20, 21] and result re-ranking [22, 23], to GRs, their effectiveness is relatively marginal. As illustrated by Figure 2, existing methods (i.e., IFair-RW and IFair-RR) [19] still struggle to achieve appropriate *Pareto Optimality* [24, 25] while collectively considering the overall recommendation performance, tail item recommendation performance, and recommendation fairness. A non-negligible distance persists from the ideal region. On the other hand, the fundamental reason why GRs suffer from severe popularity bias is still under-explored. Compared with the discriminative paradigm [26, 10, 9], two distinctive characteristics of GRs are *(i)* the end-to-end generative framework and *(ii)* the item tokenization based on SIDs. However, these two distinct aspects are completely overlooked when designing popularity debiasing methods for GRs, rendering them devoid of strategic principles.

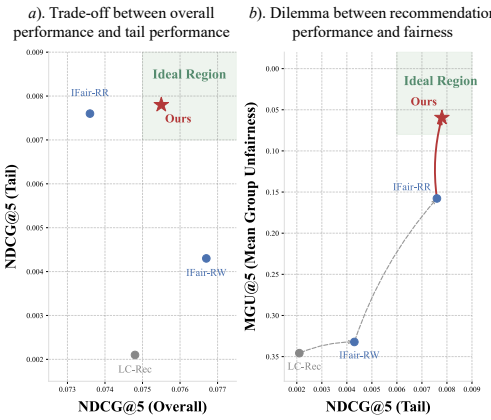

Figure 2: Limitations of current popularity debiasing methods on GRs. *a*). Trade-off between overall recommendation performance and that of tail items. *b*). Dilemma between performance and fairness of recommendation results.

To bridge the current gap, this study is anchored in the optimization dynamics of the generative framework and the SID structures under current tokenization. *Firstly*, regarding the optimization based on maximum likelihood estimation (MLE), which is widely adopted by current GRs [4, 5, 7], we start with a gradient analysis and identify that the SID tokens of tail items mostly suffer from a gradient starvation issue. Due to the heavily long-tailed training distribution [16, 27], the tail item tokens are pathologically pushed away from user preference. Hence, during the recommendation process, when head and tail items compete, the optimization flaw leads to the dominance of head item tokens. *Secondly*, this study sheds light on the fact that the current item tokenization is undifferentiated for head and tail items, without accounting for disparities in item popularity. It induces unpredictable item competition and continuously amplifies the token-level bias between head and tail items. Eventually, the undifferentiated tokenization renders the probability of tail items severely hijacked by their popular counterparts, and thus the whole GR model struggles to cast off the popularity bias.

Based on the crucial insights above, we develop a novel generative recommender named Ghost [3], which is equipped with the asymmetric unlikelihood optimization (AUO) and the skeleton-founded tokenization (SKT). In detail, to mitigate the gradient starvation issue, AUO introduces asymmetric

[3] *Ghost* denotes a **G**enerative recommender with asymmetric unlikeli**h**ood **o**ptimization and **s**keleton-founded **t**okenization.

token-level unlikelihood between the tail items and their head counterparts. By constructing a reasonable undesired collection, AUO succeeds in rescuing the ineffective gradients of tail item tokens. To inhibit the bias amplification effect caused by undifferentiated tokenization, SKT first designates the head item SIDs as the skeleton of the whole SID system. Subsequently, tail item SIDs are cultivated along the skeleton structure, with a dedicated focus on capturing the distinctiveness between tail and head items. The main contributions of this study are summarized as follows.

- This study diagnoses the troublesome issue of popularity bias in generative recommenders and identifies two fundamental factors, *(i)* the gradient starvation problem of tail item tokens and *(ii)* the bias amplification effect of undifferentiated item tokenization.
- This study develops Ghost, a novel GR that exhibits resilience to popularity bias, by designing the asymmetric unlikelihood optimization (AUO) and the skeleton-founded item tokenization (SKT). Specifically, AUO integrates effective negative feedback to calibrate the optimization of tail item tokens. Based on the skeleton structure, SKT ingeniously reduces disordered head-tail competition and characterizes the distinctiveness of tail items.
- Extensive empirical evaluations across three datasets, alongside multiple SOTA baselines, reveal that Ghost substantially alleviates popularity bias and promotes fairer recommendations, ultimately achieving the desired Pareto optimality of generative recommenders.

2 Preliminary

Problem Formulation. This study focuses on the *sequential recommendation* task [10, 5, 7], which aims to predict the next most suitable item based on the user historical behavior. Considering a system of K items $\{v_k|k=1,2,\cdots,K\}$ and J users $\{u_j|j=1,2,\cdots,J\}$, the behavior of u_j can be represented by an item sequence $h_{u_j}=[v_{k_1},v_{k_2},\cdots,v_{k_l}]$, where l is the sequence length.

Generative Recommender Systems. Existing efforts towards developing GRs [4, 5, 6, 7, 28] primarily concentrate on the paradigm based on SIDs. Generally, SID-based GRs consist of two main components, i.e., item tokenization and recommendation-oriented finetuning [5, 7]. For item tokenization, SID-based GRs usually introduce vector quantization techniques, such as VQ-VAE [11], RQ-VAE [12], and RQ-KMeans [29], to convert the continuous embeddings of items into discrete indices. Taking RQ-VAE as an example, and assuming that X_v is the item embedding and the collection $\{\mu_n^{(i)}\}_{n=1}^N$ denotes the i-th codebook within the RQ-VAE, the SID generation process can be represented as,

$$c_v^{(i)}=\arg\min_{n\in\{1,2,\ldots,N\}}\|r_v^{(i)}-\mu_n^{(i)}\|_2^2,\ r_v^{(i+1)}=r_v^{(i)}-\mu_{c_v^{(i)}}^{(i)},\text{For } i=1,2,\cdots,L, \tag{1}$$

where $r_v^{(1)}=X_v$ and L denotes the SID length. Afterwards, item v can be indexed as an SID $\Omega_v=(c_v^{(1)},c_v^{(2)},\cdots,c_v^{(L)})$. For recommendation-oriented finetuning, the sequential recommendation task can be reformulated as a language generation task based on SIDs. In particular, for historical behavior h_u and target item v with SID Ω_v, the optimization objective of SID-based GRs follows Maximum Likelihood Estimation (MLE) and usually adopts the negative log-likelihood (NLL) loss, as follows,

$$\mathcal{L}_{\text{NLL}}=-\sum\nolimits_i\log\mathcal{P}_\theta(c_v^{(i)}|h_u,c_v^{<i}), \tag{2}$$

where θ denotes the GR parameters and $c_v^{<i}$ denotes the sub-sequence of SID Ω_v before the i-th position. *This study focuses on SID-based GRs, as they are currently the most influential paradigm with the greatest performance potential.*

Popularity Bias. In the domain of recommender systems, the interaction frequency of items often follows a long-tail distribution [18, 17, 16, 30], especially a power-law distribution. Popularity bias refers to the algorithmic preference to disproportionately favor frequently interacted short-head items at the expense of highly relevant but sparse long-tail items [16]. *This study follows a common practice [27] in popularity debiasing research, which groups the items into the head set (i.e., the top 20% most popular items) and the tail set (i.e., the remaining 80% of items) based on item popularity.*

Pareto Optimality. In multi-objective optimization, a model is simultaneously evaluated across M distinct, often conflicting metrics [24]. Assuming that all objectives $\{f_m\}_{m=1}^M$ are to be minimized, a solution s is said to strictly precede another solution s', denoted as $s\prec s'$, if and only if

$\forall\, m_1, f_{m_1}(s) \leq f_{m_1}(s')$ and $\exists\, m_2, f_{m_2}(s) < f_{m_2}(s')$. A solution s^* is defined as Pareto optimal if there exists no other feasible solution s such that $s \prec s^*$. Consequently, Pareto optimality represents the optimal trade-off among the multiple evaluation metrics, indicating that no single objective can be further improved without fundamentally degrading at least one other objective [24, 25].

3 Diagnose Popularity Bias in GRs

To investigate the popularity bias in current GRs, this study starts with an analysis of *(i)* the gradient of MLE optimization and *(ii)* the SID structure under undifferentiated tokenization.

3.1 Gradient Analysis of MLE Optimization

Under the MLE objective $\mathcal{L}_{\text{NLL}}$ defined in Eq.(2), the optimization conditioned on user historical behavior h_u is governed by the Softmax derivative. Let $\mathcal{D}$ denote the training distribution of user-item interaction pairs (h_u, v), where the history h_u is encoded into a representation X_{h_u} and the target item v is tokenized as a SID $\Omega_v = (c_v^{(1)}, c_v^{(2)}, \cdots, c_v^{(L)})$.

LEMMA 1 (Gradient Starvation in MLE). *For an arbitrary SID token c, whose embedding is e_c, the expected MLE gradient update $\mathbb{E}_{\mathcal{D}}[\Delta e_c]$ over the training distribution $\mathcal{D}$ conforms to*

$$\mathbb{E}_{\mathcal{D}}[\Delta e_c] \propto \mathbb{E}_{\mathcal{D}}\left[\sum\nolimits_i \left(\mathbb{I}\{c = c_v^{(i)}\} \cdot (1 - \mathcal{P}_\theta(c|h_u, c_v^{<i})) - \mathbb{I}\{c \neq c_v^{(i)}\} \cdot \mathcal{P}_\theta(c|h_u, c_v^{<i})\right) \cdot X_{h_u}\right].$$

However, for a tail token c_{tail} that predominantly composes tail items, the expected gradient update is heavily skewed in the negative direction, formulated as,

$$\mathbb{E}_{\mathcal{D}}[\langle \Delta e_{c_{\text{tail}}}, X_{h_u} \rangle] \approx -\mathbb{E}_{\mathcal{D}}\left[\sum\nolimits_i \mathcal{P}_\theta(c_{\text{tail}}|h_u, c_v^{<i}) \cdot \|X_{h_u}\|_2^2\right] \leq 0. \tag{3}$$

In a long-tailed distribution $\mathcal{D}$, tokens composing head items frequently act as the targets, receiving massive positive gradient updates that align the embedding e_c with user preference X_{h_u}. Conversely, tokens that are specific to tail items mainly act as trivial negative samples in the denominator of the Softmax operation. This leads to the *Gradient Starvation* issue [31] for tail item tokens, in which the token embeddings are consistently pushed away from the user preference space.

3.2 SID Branching Points under Undifferentiated Tokenization

Subsequently, we further investigate how this token-level optimization flaw propagates into item-level bias during the recommendation process. Let $c_{\text{head}}^{(i)}$ and $c_{\text{tail}}^{(i)}$ be candidate tokens competing at the i-th item generation step, and step i is denoted as a branching point. Due to the asymptotically repulsive gradient updates stemming from gradient starvation, the generation probability of the tail token $c_{\text{tail}}^{(i)}$ is systematically penalized, resulting in a bias amplification factor.

COROLLARY 1 (Head Token Dominance at Branching Point). *At step i, where head and tail paths compete, the token generation probability ratio diverges from the true data distribution $\mathcal{P}_d$ by a local amplification factor $\gamma_i > 1$:*

$$\frac{\mathcal{P}_\theta(c_{\text{head}}^{(i)}|h_u, c^{<i})}{\mathcal{P}_\theta(c_{\text{tail}}^{(i)}|h_u, c^{<i})} = \gamma_i \cdot \frac{\mathcal{P}_d(c_{\text{head}}^{(i)}|h_u, c^{<i})}{\mathcal{P}_d(c_{\text{tail}}^{(i)}|h_u, c^{<i})}. \tag{4}$$

Eq.(4) implies that the generation process becomes pathologically overconfident in head tokens, regardless of the current context (i.e., h_u and $c^{(<i)}$). Furthermore, current GRs mostly tokenize items into SIDs in an undifferentiated style, which treats items identically without accounting for their inherent popularity disparities. The undifferentiated nature induces that there is no predictable structural branching point between head and tail items. Within the undifferentiated tokenization, a tail item v_{tail} shares prefixes of varying lengths with numerous popular items and is bound to encounter a sequence of branching points with head token dominance.

LEMMA 2 (Bias Amplification via Undifferentiated Tokenization). *Let $\mathcal{Z}$ (where $|\mathcal{Z}| = z \leq L$) be the set of steps along the generation process of v_{tail}, where it must compete against head candidate*

tokens. The probability of successfully navigating these intersections without being hijacked by head items is geometrically suppressed during recommendation,

$$\mathcal{P}_\theta(v_{\text{tail}}|h_u) = \prod_j \mathcal{P}_\theta(c_{\text{tail}}^{(j)}|h_u, c^{<j}) \leq (\gamma_{\min})^{-z} \cdot \prod_j \mathcal{P}_d(c_{\text{tail}}^{(j)}|h_u, c^{<j}), \tag{5}$$

where $\gamma_{min} = \min_{j\in\mathcal{Z}}(\gamma_j) > 1$. Therefore, the probability ratio of generating a competing head item over the tail item v_{tail} cascades geometrically by a factor of at least $\mathcal{O}(\gamma_{min}^z)$.

Discussion. According to the analyses above, the severe popularity bias emerges from the confluence of *(i)* gradient starvation in MLE optimization and *(ii)* bias amplification of undifferentiated tokenization. Detailed derivations are presented in Appendix G. Intuitively, under MLE optimization, tail item tokens fail to receive effective gradient signals, trapping them in gradient starvation. Then, the gradient starvation of tail item tokens results in the head token dominance at branching point. Finally, during the recommendation process, a sequence of head-dominated branching points are brought by the undifferentiated item tokenization, leading to the severe popularity bias.

4 Methodology

With a focused effort to the two diagnosed fundamental factors, the Ghost model is developed by designing the skeleton-founded item tokenization (SKT) and the asymmetric unlikelihood optimization (AUO). Procedurally, the overview of Ghost is illustrated in Figure 3. In particular, SKT subtly reduces unpredictable branching points by formulating a mechanism for SID inheritance from head to tail items. Hence, the bias amplification effect of undifferentiated tokenization is suppressed. Afterwards, based on the undesired collection of head item tokens, AUO is able to reallocate the supervision signals and thus rescue the gradient starvation issue of tail item tokens.

4.1 Semantic Information Encoding

Following the standard pipeline [4, 5, 7], the item representations are embedded based on the attached textual features by utilizing a pre-trained textual encoder f_e. For item v_k, the textual feature can be denoted as T_k, typically including the title, detailed description, brand, etc. Within the textual encoder f_e, the tokenizer first converts the textual content into token indices, and then the token embedding layer projects the token indices into continuous token embeddings. Eventually, the neural network encoder transforms the token embeddings into semantic representations X_k based on inherent semantic knowledge. The semantic information encoding can be formulated as,

$$X_k = f_e(T_k) = \text{Encoder}\left(\text{Embed}\left(\text{Tokenizer}\left(T_k\right)\right)\right). \tag{6}$$

4.2 Skeleton-Founded Item Tokenization

Current GRs [4, 5, 7] mostly adopt standard vector quantization techniques, such as RQ-VAE [12] and RQ-KMeans [29], to allocate item SIDs. These standard approaches are agnostic to item popularity disparities, where head and tail items are indiscriminately processed. Therefore, the generated SIDs contain unstructured branching points where head and tail item tokens compete. To overcome this limitation, we propose the skeleton-founded item tokenization (SKT). In detail, SKT circumvents the chaotic competition between head and tail item tokens by specifying the position of the branching point. This approach makes it possible to capture the occurring patterns of tail items without having them drowned out by their popular counterparts.

To be more specific, SKT assigns the SIDs for head and tail items asynchronously. Firstly, given the head item set $\mathcal{V}_{\text{head}}$ and the corresponding representations, SKT adopts the RQ-KMeans algorithm to generate the head item SIDs. For head item v with representation X_v, the SID generation follows,

$$c_v^{(i)} = \arg\min_{n\in\{1,2,\ldots,N\}} \|r_v^{(i)} - \mu_n^{(i)}\|_2^2,\ r_v^{(i+1)} = r_v^{(i)} - \mu_{c_v^{(i)}}^{(i)}. \text{ For } i = 1, 2, \ldots, L^h. \tag{7}$$

L^h is the SID length for head items. Initialized with $r_v^{(1)} = X_v$, the head item v is assigned SID $\Omega_v = (c_v^{(1)}, c_v^{(2)}, \ldots, c_v^{(L^h)})$. Afterwards, for the tail item v' with representation $X_{v'}$, SKT begins by retrieving the head item with the highest semantic similarity, denoted as v^*. Subsequently, the tail

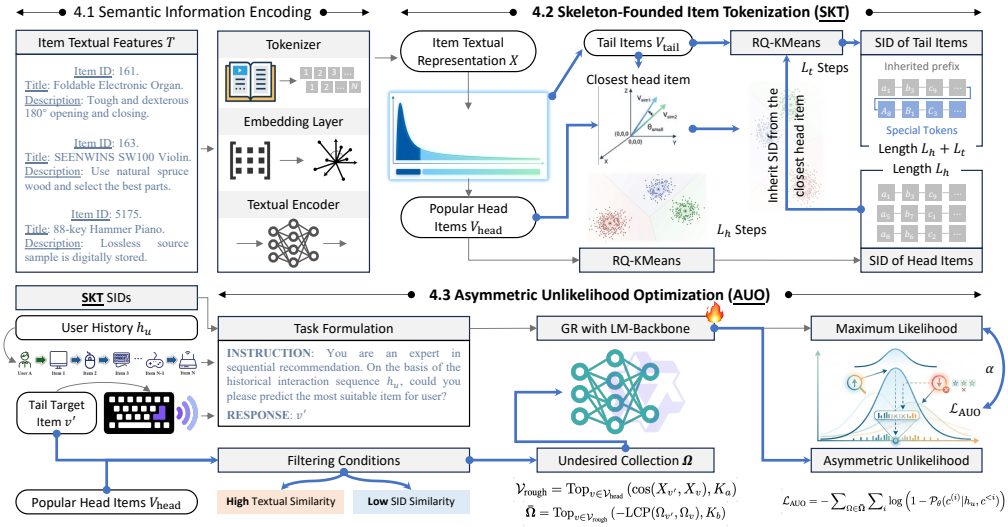

Figure 3: Overview of the Ghost model. First, textual representations are encoded based on item features. After categorizing items into head and tail sets, SKT assigns SIDs via RQ-KMeans, allowing tail items to inherit prefixes from their closest head items. At last, AUO optimizes the GR model by penalizing an undesired collection customized for each tail item, alongside standard MLE training.

item v' inherits the first L^h tokens from Ω_{v^*} and then additionally generates L^t SID tokens. Similar to Eq.(7), the additional SID generation for tail item v' is represented below,

$$c_{v'}^{(j)} = \arg\min_{n \in \{1,2,\ldots,N\}} \|r_{v'}^{(j)} - \mu_n^{(j)}\|_2^2, \; r_{v'}^{(j+1)} = r_{v'}^{(j)} - \mu_{c_{v'}^{(j)}}^{(j)}. \text{ For } j = 1, 2, \ldots, L^t. \quad (8)$$

Differing from the SID generation of head items, Eq.(8) is initialized with

$$r_{v'}^{(1)} = X_{v'} - \sum\nolimits_i \mu_{c_{v^*}^{(i)}}^{(i)},$$

and thus the tail item v' is indexed with SID $\Omega_{v'} = (c_{v^*}^{(1)}, \ldots, c_{v^*}^{(L^h)}, c_{v'}^{(1)}, \cdots, c_{v'}^{(L^t)})$.

Conceptually, the SID collection of head items $\{\Omega_v | v \in \mathcal{V}_{\text{head}}\}$ serves as the skeleton of the whole SID system and essentially tessellates the SID space into several balanced partitions. On one hand, by inheriting the tail item SID from that of the closest head item, SKT maintains the correlation between items. Therefore, the basic principle of SID-based GRs, that similar items are prone to share similar SIDs, still holds. On the other hand, SKT unifies the branching point between head and tail item at step $(L^h + 1)$, subtly reducing the bias amplification effect of undifferentiated item tokenization. Moreover, regarding the expressiveness of our SIDs, SKT captures the distinctiveness of the tail item v' compared to its closest head item v^* through L^t additional SID tokens, which elegantly leverages the strong modeling capability on head items towards tail item improvement.

4.3 Asymmetric Unlikelihood Optimization

Based on the novel SIDs generated by SKT, Ghost further designs the asymmetric unlikelihood optimization (AUO) to rectify the gradient starvation issue of tail item tokens. As analyzed in Section 3.1, the optimization of current GRs relies on MLE, where the tail items seldom serve as right answers. As a result, the optimized GR models prefer repeating head items, since they are never explicitly penalized for over-estimating statistically popular but contextually inappropriate head item tokens.

Inspired by the philosophy of unlikelihood training, AUO modulates the supervision signals by actively imposing penalties if the GR model generates notoriously undesirable tokens. Opposite to the standard NLL loss $\mathcal{L}_{\text{NLL}}$, the AUO loss $\mathcal{L}_{\text{AUO}}$ can be concisely defined as follows,

$$\mathcal{L}_{\text{AUO}} = -\sum\nolimits_{\Omega \in \bar{\boldsymbol{\Omega}}} \sum\nolimits_i \log\left(1 - \mathcal{P}_\theta(c^{(i)} | h_u, c^{<i})\right), \quad (9)$$

where $\bar{\boldsymbol{\Omega}}$ denotes the SID collection of undesired items. Notably, the AUO loss $\mathcal{L}_{\text{AUO}}$ is customized for tail items rather than the whole item set. For tail item v' with representation $X_{v'}$ and SID $\Omega_{v'}$,

Ghost first retrieves the top similar items from the popular head collection $\mathcal{V}_{\text{head}}$, which functions as a rough selection of undesired items. Then, Ghost further refines the rough candidates $\mathcal{V}_{\text{rough}}$ according to their SID distance from $\Omega_{v'}$, corresponding to a finer filtering of notoriously undesired items. Formally, the construction of $\bar{\mathbf{\Omega}}$ can be represented as follows,

$$\mathcal{V}_{\text{rough}} = \text{Top}_{v \in \mathcal{V}_{\text{head}}} \left(\cos(X_{v'}, X_v), K_a\right), \ \bar{\mathbf{\Omega}} = \text{Top}_{v \in \mathcal{V}_{\text{rough}}} \left(-\text{LCP}(\Omega_{v'}, \Omega_v), K_b\right), \tag{10}$$

where the $\text{Top}(f(\cdot,\cdot), K)$ operator returns the K objects with the highest f function value, the $\text{LCP}(\cdot,\cdot)$ operator returns the length of the *Longest Common Prefix* between the two input sequences, and K_a, K_b are two hyper-parameters controlling the candidate scale of the undesired item selection.

For tail item v', the undesired collection $\bar{\mathbf{\Omega}}_{v'}$ includes several head items whose representations are similar to $X_{v'}$, while their SIDs are divergent from $\Omega_{v'}$. Hence, AUO is able to inhibit the GR model from being hijacked by the popular head items that are similar to the ground-truth tail item in terms of semantic representation. In summary, the specialized asymmetric unlikelihood optimization is combined with MLE optimization, and the overall objective function of Ghost is defined as,

$$\mathcal{L}_{\text{All}} = \mathcal{L}_{\text{NLL}} + \alpha \cdot \mathcal{L}_{\text{AUO}}, \tag{11}$$

where α is the weighted parameter. Implementation details of Ghost are introduced in Appendix C.

4.4 Theoretical Analysis

Within SKT, since a tail item v' inherits the prefix skeleton of its closest head item v^*, the branching point of item recommendation is uniformly deferred to step $(L^h + 1)$. By establishing a singular, predictable branching point, SKT exponentially restricts the local amplification factor in LEMMA 2.

LEMMA 3 (Mitigation of Bias Amplification). *The generation probability of the tail item v' is insulated from the multi-step geometric suppression, and its deviation from the true data distribution $\mathcal{P}_d$ is governed exclusively by a localized factor,*

$$\mathcal{P}_\theta(v'|h_u) = \prod_{j=1}^{L^h+L^t} \mathcal{P}_\theta(c_{v'}^{(j)}|h_u, c^{<j}) \approx (\gamma_{EOS})^{-1} \cdot \prod_{j=1}^{L^h+L^t} \mathcal{P}_d(c_{v'}^{(j)}|h_u, c^{<j}), \tag{12}$$

where $\gamma_{EOS} \geq 1$ represents the head-dominance factor confined to the $(L^h + 1)$-th generative recommendation step, in which the tokens of tail item v' compete exclusively against the EOS token.

Contrasting Eq.(12) with Eq.(5), we can notice that by transforming a geometric suppression $\mathcal{O}(\gamma_{\text{min}}^z)$ into a single-step discrepancy $\mathcal{O}(\gamma_{EOS})$, SKT effectively halts the amplification of popularity bias during the generative recommendation process. Regarding the overall objective defined in Eq.(11), a comprehensive gradient analysis is conducted to uncover its rationale. While MLE subjects tail tokens to gradient starvation through the suppressive $-\mathcal{P}_\theta(c_{\text{tail}}^{(i)}|h_u, c^{<i}) \cdot X_{h_u}$ term, the Ghost model provides a powerful *Rescue Force* on the basis of AUO.

LEMMA 4 (Gradient Rescue based on AUO). *For false positive head tokens $c_{head}^- \in \bar{\mathbf{\Omega}}$, the gradient correctly cancels the denominator via the softmax derivative for its own direct penalty. Crucially, it also systematically absorbs the cross-penalization dynamics originating from other tokens within the undesired set. Formally, the gradient is,*

$$\Delta e_{c_{\text{head}}^-} \propto -\underbrace{(1+\alpha)\mathcal{P}_\theta(c_{\text{head}}^-) \cdot X_{h_u}}_{\text{Targeted Repulsion}} + \underbrace{\alpha \sum\nolimits_{c_j \in \bar{\mathbf{\Omega}} \setminus \{c_{\text{head}}^-\}} \frac{\mathcal{P}_\theta(c_j)\mathcal{P}_\theta(c_{\text{head}}^-)}{1 - \mathcal{P}_\theta(c_j)} \cdot X_{h_u}}_{\text{Cross-Penalization Offset}}. \tag{13}$$

For a token c_{tail} of the target tail item, the expected update gradient is reformulated as,

$$\Delta e_{c_{\text{tail}}} \propto \underbrace{-\mathcal{P}_\theta(c_{\text{tail}}) \cdot X_{h_u}}_{\text{MLE Push}} + \underbrace{\alpha \sum\nolimits_{c_j \in \bar{\mathbf{\Omega}}} \frac{\mathcal{P}_\theta(c_j)\mathcal{P}_\theta(c_{\text{tail}})}{1 - \mathcal{P}_\theta(c_j)} \cdot X_{h_u}}_{\text{AUO Rescue}}. \tag{14}$$

By actively penalizing the undesired head tokens in $\bar{\mathbf{\Omega}}$, the Jacobian of the AUO loss systematically redistributes positive probability mass to the remaining tokens. This structural rescue ensures that tail tokens receive active, rational parameter updates, preventing them from being indefinitely pushed away from the user intent space as mere trivial negative samples. Detailed derivations of LEMMA 4 and LEMMA 5 are presented in Appendix G as well.

Table 1: Performance comparison of Ghost and baselines across three datasets. The best, the runner-up, and the third-best results are highlighted in **bold**, underlined, and colored fonts.

Dataset	Metric	HR@5 ↑		HR@10 ↑		NDCG@5 ↑		NDCG@10 ↑		MGU@5 ↓	MGU@10 ↓	ARP@5 ↓	ARP@10 ↓	CNS@5 ↑	CNS@10 ↑
	Model	All	Tail	All	Tail	All	Tail	All	Tail						
Ins	LETTER	0.0593	0.0025	0.0662	0.0044	0.0530	0.0016	0.0553	0.0023	0.1965	0.1442	**181.2628**	**151.7149**	0.2457	0.2714
	LC-Rec	0.0870	0.0027	0.1046	0.0040	0.0748	0.0021	0.0804	0.0025	0.3458	0.3380	365.4217	338.5304	0.3214	0.3134
	ED^2	**0.0898**	0.0067	0.1068	0.0102	0.0765	0.0043	0.0820	0.0055	0.2551	0.2305	279.8572	237.9431	0.6067	0.6217
	Augmentation	0.0875	0.0071	0.1039	0.0105	0.0759	0.0051	0.0812	0.0061	0.2383	0.2049	271.1657	242.7466	0.6327	0.6323
	Substitution	0.0780	0.0058	0.0914	0.0084	0.0645	0.0039	0.0689	0.0048	**0.0310**	**0.0114**	253.3299	229.5070	0.5714	0.5601
	IFairLRS-RW	0.0888	0.0058	**0.1072**	0.0081	**0.0767**	0.0043	**0.0826**	0.0050	0.3322	0.3226	328.5940	299.5196	0.5007	0.4887
	IFairLRS-RR	0.0853	0.0100	0.1053	0.0110	0.0736	**0.0078**	0.0800	0.0081	0.1578	0.2330	301.2252	286.8023	0.7462	0.6286
	Ghost (Ours)	0.0864	**0.0117**	0.1017	**0.0173**	0.0755	**0.0078**	0.0805	**0.0097**	0.0596	0.0958	248.8179	209.8877	**0.8974**	**0.8694**
Arts	LETTER	0.0448	0.0062	0.0521	0.0090	0.0378	0.0040	0.0401	0.0050	0.1768	0.1283	85.6235	73.2413	0.2108	0.2693
	LC-Rec	**0.0885**	0.0188	**0.1095**	0.0269	**0.0737**	0.0133	**0.0805**	0.0159	0.2992	0.2905	135.2274	119.2569	0.5518	0.5548
	ED^2	0.0796	0.0105	0.0993	0.0172	0.0656	0.0073	0.0719	0.0095	0.2960	0.2879	148.5051	129.2274	0.3331	0.3548
	Augmentation	0.0839	0.0218	0.1021	0.0306	0.0697	0.0159	0.0756	0.0187	0.1572	0.1299	112.3944	98.8012	0.6946	0.7258
	Substitution	0.0732	0.0215	0.0915	0.0305	0.0581	0.0155	0.0640	0.0184	**0.0490**	**0.0531**	96.9228	88.0055	0.6910	0.7269
	IFairLRS-RW	0.0845	0.0138	0.1022	0.0191	0.0706	0.0102	0.0763	0.0119	0.3076	0.2925	141.4042	122.9239	0.4359	0.4270
	IFairLRS-RR	0.0827	0.0179	0.1005	0.0248	0.0696	0.0130	0.0753	0.0152	0.2255	0.2103	134.5895	117.1711	0.5447	0.5485
	Ghost (Ours)	0.0831	**0.0296**	0.1000	**0.0393**	0.0706	**0.0213**	0.0760	**0.0245**	0.1763	0.1621	**70.5222**	**65.1832**	**0.8828**	**0.8780**
Games	LETTER	0.0264	0.0063	0.0375	0.0096	0.0187	0.0044	0.0222	0.0055	0.1727	0.1396	119.9021	108.3559	0.2458	0.2652
	LC-Rec	**0.0636**	0.0148	**0.0938**	0.0244	**0.0438**	0.0091	**0.0536**	0.0122	0.2907	0.2740	164.9579	150.6237	0.5352	0.5499
	ED^2	0.0572	0.0086	0.0854	0.0150	0.0393	0.0052	0.0484	0.0072	0.3110	0.2925	188.5855	170.8846	0.3052	0.3284
	Augmentation	0.0559	0.0173	0.0822	0.0282	0.0384	0.0113	0.0469	0.0148	0.1526	0.1210	138.5779	125.0857	0.6701	0.6978
	Substitution	0.0448	0.0199	0.0683	0.0311	0.0309	0.0131	0.0384	0.0167	0.0783	0.0824	121.4722	111.5519	0.7074	0.7093
	IFairLRS-RW	0.0605	0.0134	0.0907	0.0228	0.0416	0.0087	0.0513	0.0117	0.2869	0.2710	166.8988	153.0392	0.4878	0.5103
	IFairLRS-RR	0.0581	0.0229	0.0877	0.0315	0.0402	0.0153	0.0497	0.0181	0.1051	0.1449	148.2428	141.5092	0.8073	0.7229
	Ghost (Ours)	0.0562	**0.0257**	0.0840	**0.0392**	0.0390	**0.0167**	0.0479	**0.0218**	**0.0743**	**0.0484**	**111.4375**	**106.0538**	**0.9349**	**0.9406**

5 Experiment

• **Dataset**. This study evaluates Ghost [4] and baselines on three public benchmarks extracted from the Amazon platform, i.e., *"Musical Instruments"*, *"Arts, Crafts and Sewing"*, and *"Video Games"*.

• **Baseline**. The evaluation introduces three SOTA GRs (LETTER [6], LC-Rec [5], and ED^2 [7]) as the standard baselines. IFairLRS [19], which pioneers the investigation of popularity bias in GRs, serves as a competitive baseline. In detail, *RW* and *RR* refer to the *re-weighting only* and *re-weight & re-ranking* strategies. Moreover, to provide a comprehensive comparison, this study replaces each popular head item with the most similar tail item according to a pre-defined probability. The modified interaction sequence is either appended to the original training set or substituted for the original sequence, corresponding to the baselines denoted as *Augmentation* and *Substitution* (The details of baselines are presented in Appendix C.2).

• **Metric**. Following well-established benchmarks [5, 32], this study adopts Hit-Rate (HR) and normalized discounted cumulative gain (NDCG) to evaluate the recommendation performance. Besides, this study adopts mean group unfairness (MGU) and average recommendation popularity (ARP) to quantify the fairness and the average popularity of the recommendation results [27]. To reflect the proximity to Pareto optimality, this study defines the comprehensively normalized score (CNS) as follows, based on the Min-Max normalized values of HR, NDCG, MGU, and ARP metrics,

$$\text{CNS} = (\overline{\text{HR}}_{\text{All}} + \overline{\text{HR}}_{\text{Tail}} + \overline{\text{NDCG}}_{\text{All}} + \overline{\text{NDCG}}_{\text{Tail}} + \overline{\text{MGU}} + \overline{\text{ARP}})/6. \tag{15}$$

The detailed introduction of the datasets and metrics is presented in Appendix C.

5.1 Main Result

To demonstrate the effectiveness of Ghost in mitigating popularity bias, we compare its performance against SOTA baselines across three public datasets. The results are summarized in Table 1.

In general, Ghost can effectively approach the Pareto frontier while jointly considering overall recommendation performance, tail recommendation performance, and recommendation fairness. In detail, we can draw the following three observations. ***First***, Ghost is primarily characterized by its significant performance gains in tail-item recommendation. Across three datasets, Ghost outperforms standard GRs (i.e., LETTER, LC-Rec, and ED^2) by increasing tail HR and NDCG by 63.91% and 70.66% on average. Compared to existing popularity debiasing methods, it delivers an average increase of 28.39% and 15.04% for tail HR and NDCG, with maximum improvements up to 57.24%. ***Second***, Ghost effectively suppresses the over-recommendation of head items. It lowers the MGU by

[4]The Ghost project is available at *Ghost Project*.

Table 2: Performance comparison of ablation study on *Ins* dataset. *RQK-4/6* denotes a LC-Rec model adopted 4/6-tokens SIDs provided by RQ-KMeans, respectively. *RQK-4-6* adopts 4-tokens SIDs for head items while 6-tokens SIDs for tail items, while without the inheriting process in SKT.

Model	HR@5 ↑		HR@10 ↑		NDCG@5 ↑		NDCG@10 ↑		MGU@5 ↓	MGU@10 ↓	ARP@5 ↓	ARP@10 ↓
	All	Tail	All	Tail	All	Tail	All	Tail				
Ghost	0.0864	**0.0117**	0.1017	**0.0173**	0.0755	**0.0078**	0.0805	**0.0097**	0.0596	0.0958	**248.8179**	**209.8877**
w/o AUO	0.0849	0.0111	0.1003	0.0161	0.0733	0.0075	0.0782	0.0091	**0.0035**	**0.0112**	270.4236	235.9759
w/o SKT	0.0920	0.0060	0.1102	0.0095	**0.0798**	0.0047	0.0856	0.0058	0.3211	0.3056	387.0114	357.7677
RQK-4	0.0929	0.0059	0.1116	0.0095	0.0796	0.0042	0.0856	0.0054	0.3037	0.2885	304.3177	267.9522
RQK-6	**0.0932**	0.0052	**0.1133**	0.0088	0.0797	0.0038	**0.0862**	0.0049	0.3073	0.2902	303.1651	269.7153
RQK-4-6	0.0883	0.0070	0.1080	0.0094	0.0762	0.0049	0.0826	0.0057	0.2268	0.2468	320.2727	288.6501

an average of 55.76% compared to standard GRs, and outperforms the most competitive popularity debiasing baseline, IFairLRS-RR, by further reducing the MGU by 16.68% on average (up to a maximum of 66.60%). ***Third***, with respect to overall recommendation performance, Ghost exhibits an acceptable degradation compared to standard models. Relative to the strongest baseline, LC-Rec, the overall HR and NDCG of Ghost drop by only 7.46% and 6.34% on average. For context, the performance drops among all popularity debiasing methods in Table 1 range between 2.46% and 18.90% for HR, and between 2.24% and 21.28% for NDCG. Taken together, Ghost yields an average CNS gain of 16.81%, peaking at 22.06%, which indicates its capability to substantially approach the Pareto optimal state for the GRs popularity bias issue.

5.2 Ablation Study

To investigate the contributions of the core components in Ghost, an ablation study is conducted by removing specific modules. Based on Table 2, the following insights can be drawn. ***First***, bias amplification stemming from undifferentiated tokenization constitutes the critical cause of the GRs susceptibility to popularity bias. A performance comparison between Ghost and the *w/o SKT* variant reveals that the inability to inhibit the unpredictable competition between head and tail tokens leads to a substantial decline in tail recommendation performance. The results from *RQK-4*, *RQK-6*, and *RQK-4-6* serve as additional evidence supporting this finding. ***Second***, the supervisory signals corrected by AUO must be absorbed by the corresponding learnable parameters. By comparing the performance of Ghost, *w/o AUO*, and *w/o SKT*, one can notice that the model ability to mitigate popularity bias is limited when relying solely on AUO. However, once SKT introduces additional tokens for tail items, the corrective effect provided by AUO drives the model closer to Pareto optimality.

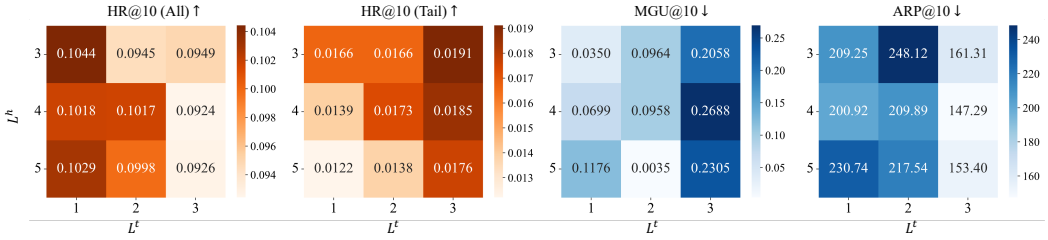

Figure 4: Analysis of SID lengths, including head length L^h and additional length L^t for tail items.

5.3 SID Length Analysis

As shown by Figure 4, we vary the skeleton length $L^h \in \{3, 4, 5\}$ and the additional tail-specific length $L^t \in \{1, 2, 3\}$. The following insights can be drawn. A larger L^t effectively mitigates popularity bias, as evidenced by the improvement in tail HR@10 and the reduction in ARP@10. However, this exploration comes at the expense of overall performance and fairness. This reveals a clear trade-off in identifier length allocation. Assigning longer sequences to tail items enhances their representational capacity and retrieval probability, but it simultaneously introduces noise that dilutes the learning of head items, thereby compromising overall performance. Furthermore, the skeleton length L^h dictates the extent to which tail items inherit semantic prefixes from head items. Forcing tail items to strictly mirror head items over too many generative steps suppresses their unique semantic identities.

Table 3: Performance comparison across different backbone scales. The best and the runner-up are highlighted in **bold**, underlined fonts.

Scale	Model	HR@5 ↑		HR@10 ↑		NDCG@5 ↑		NDCG@10 ↑		MGU@5 ↓	MGU@10 ↓	ARP@5 ↓	ARP@10 ↓
		All	Tail	All	Tail	All	Tail	All	Tail				
0.6B	LC-Rec	0.0794	0.0015	0.0983	0.0021	0.0671	0.0010	0.0731	0.0011	0.3568	0.3550	433.3062	404.2913
	IFairLRS-RW	0.0847	0.0025	**0.1030**	0.0036	0.0725	0.0018	0.0784	0.0021	0.3483	0.3431	393.1610	360.2472
	IFairLRS-RR	0.0819	0.0053	0.1010	0.0060	0.0705	0.0045	0.0766	0.0047	0.2414	0.2918	378.3699	353.8215
	Ghost	**0.0848**	**0.0101**	0.1025	**0.0148**	**0.0748**	**0.0069**	**0.0805**	**0.0084**	**0.0136**	**0.0208**	**239.9323**	**209.8331**
1.7B	LC-Rec	0.0855	0.0027	0.1025	0.0033	0.0731	0.0022	0.0786	0.0024	0.3531	0.3512	418.5129	386.3999
	IFairLRS-RW	**0.0879**	0.0038	**0.1070**	0.0057	**0.0764**	0.0026	**0.0825**	0.0032	0.3459	0.3422	373.8739	341.3491
	IFairLRS-RR	0.0847	0.0084	0.1047	0.0090	0.0734	0.0070	0.0798	0.0071	0.2052	0.2676	346.5825	326.8923
	Ghost	0.0870	**0.0110**	0.1059	**0.0174**	0.0761	**0.0081**	0.0821	**0.0102**	**0.0262**	**0.0508**	**248.1768**	**224.2784**
4B	LC-Rec	**0.0937**	0.0057	0.1112	0.0074	**0.0811**	0.0047	**0.0867**	0.0053	0.3363	0.3301	330.5470	301.0089
	IFairLRS-RW	**0.0937**	0.0067	**0.1125**	0.0097	0.0800	0.0051	0.0860	0.0061	0.3160	0.3033	293.3685	264.1469
	IFairLRS-RR	0.0884	0.0124	0.1097	0.0148	0.0764	0.0097	0.0832	0.0105	0.0871	0.1836	256.1628	247.6822
	Ghost	0.0898	**0.0136**	0.1074	**0.0190**	0.0782	**0.0101**	0.0838	**0.0118**	**0.0356**	**0.0211**	**239.5142**	**215.0566**
8B	LC-Rec	**0.0954**	0.0062	**0.1170**	0.0092	**0.0817**	0.0047	**0.0886**	0.0056	0.3233	0.3142	300.1242	268.9937
	IFairLRS-RW	0.0900	0.0066	0.1102	0.0088	0.0785	0.0048	0.0850	0.0055	0.3251	0.3177	317.3005	285.7003
	IFairLRS-RR	0.0872	0.0120	0.1076	0.0140	0.0764	0.0095	0.0830	0.0101	0.1445	0.2212	289.5438	273.9240
	Ghost	0.0907	**0.0150**	0.1109	**0.0228**	0.0787	**0.0106**	0.0851	**0.0131**	**0.0479**	**0.0410**	**207.7282**	**181.4011**

6 Scaling Pattern Analysis

To further investigate the scalability of our proposed framework, we compare the performance of Ghost against various baselines across different LLM backbone scales (ranging from 0.6B to 8B). The results are summarized in Table 3.

In general, Ghost consistently demonstrates superior popularity bias mitigation and long-tail item excavation capabilities, regardless of the underlying backbone size. In detail, we can draw the following three observations. ***First***, Ghost is primarily characterized by its robust debiasing and tail-item recommendation performance across all scales. Compared to both the standard baseline LC-Rec and the fairness-aware baselines IFairLRS, Ghost consistently achieves the highest Tail HR and Tail NDCG, alongside the lowest ARP and MGU scores. For instance, at the 8B scale, Ghost substantially increases Tail HR@10 to 0.0228 compared to 0.0092 of LC-Rec, while drastically reducing ARP@10 to 181.4011 compared to 268.9937 of LC-Rec. ***Second***, with respect to overall recommendation performance, Ghost maintains highly competitive utility. Despite aggressively promoting tail items, the overall HR and NDCG metrics of Ghost remain closely comparable to those of the standard generative recommendation models and generally surpass the IFairLRS baselines. This indicates that Ghost achieves an excellent trade-off, mitigating popularity bias without inducing unacceptable degradation in general recommendation accuracy. ***Third***, Ghost exhibits strong scalability with respect to LLM backbone capacity. As the parameter size increases from 0.6B to 8B, Ghost effectively leverages the enhanced representation and reasoning capabilities of larger models to further boost its performance. Specifically, Ghost's Tail HR@10 steadily increases from 0.0148 at 0.6B to 0.0228 at 8B, while its ARP@10 progressively drops from 209.8331 to 181.4011. This demonstrates that larger backbones consistently empower the framework to deliver increasingly diverse and fair recommendations. *Additionally*, the hyper-parameter analysis is in Appendix F.

7 Conclusion

In this paper, we theoretically demonstrate that standard MLE training and undifferentiated item tokenization of current GRs inherently cause a token-level optimization flaw and multi-step bias amplification. Accordingly, we propose Ghost equipped with asymmetric unlikelihood optimization (AUO) and skeleton-founded tokenization (SKT). AUO provides explicit negative supervision to rescue tail tokens from gradient starvation, while SKT establishes unified branching points to halt item-level bias amplification. Extensive empirical evaluations confirm that Ghost effectively breaks the filter bubble and substantially promotes fairer long-tail recommendations with slight losses to overall recommendation utility, approaching Pareto optimality in GRs popularity debiasing.

Acknowledgement

We sincerely acknowledge Mr. Chenhan Wang from Hyper.AI for the support of computing resources.

Impact Statement

This work aims to enhance the fairness and diversity of Generative Recommender Systems. By mitigating the fundamental causes of popularity bias, our proposed model effectively breaks the pervasive filter bubble. Ethically, this approach addresses the algorithmic "Matthew Effect," which typically over-represents trending items while severely marginalizing niche content. By promoting fairer long-tail recommendations without significantly compromising overall utility, this research fosters the responsible deployment of LLM-based recommendation technologies.

References

[1] Guorui Zhou, Chengru Song, Xiaoqiang Zhu, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. Deep interest network for click-through rate prediction. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[2] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the ACM Conference on Recommender Systems*, 2016.

[3] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. Deepinf: Social influence prediction with deep learning. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[4] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan H. Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2023.

[5] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, and Ming Chen. Adapting large language models by integrating collaborative semantics for recommendation. In *Proceedings of the IEEE International Conference on Data Engineering*, 2024.

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

[7] Jun Yin, Zhengxin Zeng, Mingzheng Li, Hao Yan, Chaozhuo Li, Weihao Han, Jianjin Zhang, Ruochen Liu, Hao Sun, Weiwei Deng, Feng Sun, Qi Zhang, Shirui Pan, and Senzhang Wang. Unleash llms potential for sequential recommendation by coordinating dual dynamic index mechanism. In *Proceedings of the ACM on Web Conference*, 2025.

[8] Jianyang Zhai, Zi-Feng Mai, Chang-Dong Wang, Feidiao Yang, Xiawu Zheng, Hui Li, and Yonghong Tian. Multimodal quantitative language for generative recommendation. In *Proceedings of the International Conference on Learning Representations*, 2025.

[9] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.

[10] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *Proceedings of the IEEE International Conference on Data Mining*, 2018.

[11] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2017.

[12] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[13] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633–638, 2025.

[14] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, and Faisal Azhar. Llama: Open and efficient foundation language models. 2023.

[15] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[16] Siyi Lin, Chongming Gao, Jiawei Chen, Sheng Zhou, Binbin Hu, Yan Feng, Chun Chen, and Can Wang. How do recommendation models amplify popularity bias? an analysis from the spectral perspective. In *Proceedings of the ACM International Conference on Web Search and Data Mining*, 2025.

[17] Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021.

[18] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. Popularity-opportunity bias in collaborative filtering. In *Proceedings of the ACM International Conference on Web Search and Data Mining*, 2021.

[19] Meng Jiang, Keqin Bao, Jizhi Zhang, Wenjie Wang, Zhengyi Yang, Fuli Feng, and Xiangnan He. Item-side fairness of large language model-based recommendation system. In *Proceedings of the ACM Web Conference*, 2024.

[20] Lei Wang, Chen Ma, Xian Wu, Zhaopeng Qiu, Yefeng Zheng, and Xu Chen. Causally debiased time-aware recommendation. In *Proceedings of the ACM Web Conference*, 2024.

[21] Fan Zhang and Qijie Shen. A model-agnostic popularity debias training framework for click-through rate prediction in recommender system. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023.

[22] Xinyi Li, Yifan Chen, Benjamin Pettit, and Maarten De Rijke. Personalised reranking of paper recommendations using paper content and user behavior. *ACM Transactions on Information Systems*, 37(3):1–23, 2019.

[23] Diego Carraro and Derek Bridge. Enhancing recommendation diversity by re-ranking with large language models. *ACM Transactions on Recommender Systems*, 4(2):1–40, 2025.

[24] Kaisa Miettinen. *Nonlinear multiobjective optimization*, volume 12. Springer Science & Business Media, 1999.

[25] Debabrata Mahapatra and Vaibhav Rajan. Multi-task learning with user preferences: Gradient descent with controlled ascent in pareto optimization. In *Proceedings of the International Conference on Machine Learning*, 2020.

[26] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the International Conference on World Wide Web*, 2017.

[27] Jin Li, Huilin Gu, Shoujin Wang, Qi Zhang, Shui Yu, Chen Wang, Xiwei Xu, and Fang Chen. Towards fair large language model-based recommender systems without costly retraining. In *Proceedings of the ACM Web Conference*, 2026.

[28] Xinyu Lin, Pengyuan Liu, Wenjie Wang, Yicheng Hu, Chen Xu, Fuli Feng, Qifan Wang, and Tat-Seng Chua. Bringing reasoning to generative recommendation through the lens of cascaded ranking. In *Proceedings of the ACM Web Conference*, 2026.

[29] Xinchen Luo, Jiangxia Cao, Tianyu Sun, Jinkai Yu, Rui Huang, Wei Yuan, Hezheng Lin, Yichen Zheng, Shiyao Wang, Qigen Hu, Changqing Qiu, Jiaqi Zhang, Xu Zhang, Zhiheng Yan, Jingming Zhang, Simin Zhang, Mingxing Wen, Zhaojie Liu, and Guorui Zhou. Qarm: Quantitative alignment multi-modal recommendation at kuaishou. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, 2025.

[30] Weijieying Ren, Lei Wang, Kunpeng Liu, Ruocheng Guo, Lim Ee Peng, and Yanjie Fu. Mitigating popularity bias in recommendation with unbalanced interactions: A gradient perspective. In *Proceedings of the IEEE International Conference on Data Mining*, 2022.

[31] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C. Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2021.

[32] Clark Mingxuan Ju, Liam Collins, Leonardo Neves, Bhuvesh Kumar, Louis Yufeng Wang, Tong Zhao, and Neil Shah. Generative recommendation with semantic ids: A practitioner's handbook. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, 2025.

[33] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems*, 41(3):1–39, 2023.

[34] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *Proceedings of the International Conference on Machine Learning*, 2016.

[35] Chen Gao, Yu Zheng, Wenjie Wang, Fuli Feng, Xiangnan He, and Yong Li. Causal inference in recommender systems: A survey and future directions. *ACM Transactions on Information Systems,*, 42(4):1–32, 2024.

[36] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. Causal intervention for leveraging popularity bias in recommendation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

[37] Yoseph Linde, Andres Buzo, and Robert Gray. An algorithm for vector quantizer design. *IEEE Transactions on communications*, 28(1):84–95, 1980.

[38] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021.

[39] Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the International Conference on World Wide Web*, 2016.

[40] Marco Cuturi. Sinkhorn distances: lightspeed computation of optimal transport. In *Proceedings of the International Conference on Neural Information Processing Systems*, volume 2, pages 2292–2300, 2013.

[41] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[42] Qwen Team. Qwen2.5: A party of foundation models, September 2024.

[43] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[44] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

[45] Guorui Zhou, Honghui Bao, Jiaming Huang, Jiaxin Deng, Jinghao Zhang, Junda She, Kuo Cai, Lejian Ren, Lu Ren, Qiang Luo, et al. Openonerec technical report. *arXiv preprint arXiv:2512.24762*, 2025.

[46] Chongming Gao, Ruijun Chen, Shuai Yuan, Kexin Huang, Yuanqing Yu, and Xiangnan He. Sprec: Self-play to debias llm-based recommendation. In *Proceedings of the ACM on Web Conference*, 2025.

A Notation

Table 4: Summary of Mathematical and Model Notations

Symbol	Description	Symbol	Description	Symbol	Description
K, J	Number of items and users in the system, respectively.	L	Length of SIDs.	$\mathcal{V}_{\text{head}}$	Set of popular head items.
v_k, u_j	The k-th item and j-th user.	θ	Parameters of the Generative Recommender (GR) model.	L^h	SID length assigned for head items (the skeleton length).
h_{u_j} or h_u	Historical behavior / item sequence of user u_j.	$\mathcal{P}_\theta$	Token generation probability parameterized by θ.	v'	A target tail item.
l	Length of the item sequence.	$\mathcal{P}_d$	True data distribution.	v^*	The closest head item to tail item v' based on highest semantic similarity.
T_k	Textual features (e.g., title, description) attached to item v_k.	$\mathcal{L}_{\text{NLL}}$	Negative log-likelihood loss used for Maximum Likelihood Estimation (MLE).	L^t	Additional SID tokens length specifically for tail items.
f_e	Pre-trained textual encoder.	$\mathcal{D}$	Training distribution of user-item interaction pairs.	γ_{EOS}	Head-dominance factor at the (L^h+1)-th generative step against the EOS token.
X_v, X_k	Semantic textual representation of item v or k.	X_{h_u}	Encoded representation of the user's historical behavior h_u.	$\overline{\mathbf{\Omega}}$	SID collection of roughly selected undesired head items.
$\mu_n^{(i)}$	The n-th embedding in the i-th codebook within the RQ-VAE.	e_c	Embedding of token c.	$\mathcal{V}_{\text{rough}}$	Rough candidate set of popular head items semantically similar to the tail item.
$r_v^{(i)}$	Residual embedding at step i during tokenization.	$c_{\text{head}}^{(i)}, c_{\text{tail}}^{(i)}$	Candidate head and tail tokens competing at the i-th step.	K_a, K_b	Hyper-parameters controlling the candidate scale for undesired item selection.
$c_v^{(i)}$	The i-th SID token for item v.	γ_i	Amplification factor at step i.	$\mathcal{L}_{\text{AUO}}$	Asymmetric unlikelihood optimization (AUO) loss.
Ω_v	Complete SID of item v, indexed as $(c_v^{(1)}, c_v^{(2)}, \ldots, c_v^{(L)})$.	$\mathcal{Z}$	Set of steps during tail item generation where it competes against head tokens.	$\mathcal{L}_{\text{All}}$	Overall optimization objective function of the Ghost model.
$c_v^{<i}$	Sub-sequence of SID Ω_v before the i-th position.	γ_{min}	Minimum factor across the competing steps in $\mathcal{Z}$.	α	Weighted parameter controlling the AUO loss.

B Related Work

B.1 Recommender Systems: Discriminative and Generative Paradigms

Traditional recommender systems typically formulate recommendation as a discriminative task, relying heavily on discrete item IDs to capture user preferences by analyzing historical interactions [26, 10, 9, 1, 2]. At the core of this pipeline, both users and items are projected into a shared low-dimensional latent space to learn dense embeddings, allowing the system to determine the final ranking by computing their pairwise similarities or interaction scores. To efficiently handle massive item catalogs, these systems typically employ a multi-stage pipeline, primarily consisting of candidate generation and ranking. Specifically, the candidate generation stage rapidly retrieves a coarse-grained subset of relevant items from the entire corpus, which are subsequently evaluated and sorted by a more complex ranking model to produce the final recommendations.

Recently, fueled by the adoption of large language models (LLMs) [15, 14, 13] as the underlying backbone, Generative Recommenders (GRs) have emerged as a transformative paradigm [4, 32]. Instead of modeling interaction probabilities directly, GRs replace traditional item IDs with Semantic IDs (SIDs), reframing recommendation as a unified end-to-end sequential generation process [6, 5, 7, 8]. To construct these SIDs, standard vector quantization techniques, such as VQ-VAE [11], RQ-VAE [12], and RQ-KMeans [29], are widely utilized to convert continuous item embeddings into discrete indices [5, 4]. However, these established tokenization strategies are fundamentally undifferentiated. They assign SIDs identically without accounting for inherent item popularity disparities, leading to unstructured and unpredictable branching points where head and tail item tokens compete.

B.2 Popularity Bias and Fairness in Recommendation

Traditional recommender systems frequently suffer from popularity bias, a phenomenon where a small fraction of highly interacted items disproportionately dominates algorithm exposure, leaving

the long-tail of items largely ignored [16]. This algorithmic amplification, often referred to as the *rich-get-richer* effect (a.k.a. the Matthew Effect), not only degrades user experience by failing to capture niche or diverse interests but also introduces critical fairness concerns across the platform, particularly for minority content creators who receive inequitable visibility. To mitigate these disparities, early literature primarily focused on heuristic post-processing and re-ranking techniques that explicitly boosted the exposure of long-tail items, albeit often at the expense of overall accuracy [33]. More recently, the field has gravitated toward principled statistical and causal frameworks to achieve debiased learning. Schnabel et al. utilize inverse propensity weighting (IPW) [34] to directly correct data collection biases by re-weighting user-item interactions during the model training phase. Furthermore, recent advances leveraging causal inference [35] and adversarial learning have enabled systems to explicitly disentangle genuine user preferences from popularity-driven conformity, yielding robust representations that strive to optimize accuracy while maintaining multi-sided fairness for both users and providers [36].

B.3 Vector Quantization

Vector Quantization (VQ) originated as a classical signal processing technique for data compression [37], but it has recently become a cornerstone of deep representation learning. The seminal VQ-VAE [11] successfully integrated discrete latent spaces into neural architectures, effectively resolving the posterior collapse issue common in continuous models. Building upon this, frameworks like VQGAN [38] utilized adversarial objectives to dramatically enhance reconstruction fidelity. Crucially, by mapping high-dimensional data into discrete codebook indices, VQ allows complex signals to be treated as sequential tokens for autoregressive transformers. Beyond visual and audio synthesis, this discrete tokenization paradigm has recently advanced generative recommender systems. As we mentioned above, by employing VQ techniques, such as residual quantization [29, 12], to discretize continuous item embeddings into semantic, categorical IDs, researchers have successfully reformulated recommendation as a sequence-to-sequence generation task [4, 5, 7]. This approach allows large language models to autoregressively predict next-item interactions using these discrete item tokens, seamlessly bridging traditional collaborative filtering with the powerful generative and reasoning capabilities of modern Transformer architectures.

C Experimental Detail

C.1 Dataset

The dataset statistics are presented in Table 5. The three sequential recommendation datasets originate from the Amazon Product Review dataset [39], which contains user review data from May 1996 to October 2018. Particularly, three categories for the sequential recommendation task, including *"Musical Instruments"*, *"Video Games"*, and *"Arts, Crafts and Sewing"*, are extracted and organized into individual datasets *Instruments*, *Games*, and *Arts*, respectively. Within the above datasets, each item is associated with a series of textual contents, including the item title, the detailed description, the item category, and so on. Similarly, the associated textual contents of the user entity include the user comment, the search query, and so on. Following standard procedure, inactive users/items with fewer than 5 interactions are filtered out, and the user interaction sequence is created in chronological order. In Figure 5, we present the distribution of item popularity for each of the three datasets, all of which are heavily long-talied.

Table 5: Statistics of the evaluated datasets. *Avg.L* is the average length of the user interaction sequences.

Dataset	#User	#Item	#Interaction	Sparsity	Avg.L
Instruments	24,772	9,922	206,153	99.92%	8.32
Games	50,546	16,859	452,989	99.95%	8.96
Arts	45,141	20,956	390,832	99.96%	8.66

C.2 Baseline

• **LETTER** [6], short for LEarnable Tokenizer for generaTivE Recommendation, is a learnable item tokenizer tailored for LLM-based generative recommendation. It addresses the limitations of existing ID, textual, and codebook-based identifiers by comprehensively integrating hierarchical semantics, collaborative signals, and code assignment diversity into item identifiers. Specifically, LETTER employs a RQ-VAE to encode item semantic information into hierarchical code sequences. To overcome the misalignment between semantics and collaborative signals, it introduces a contrastive

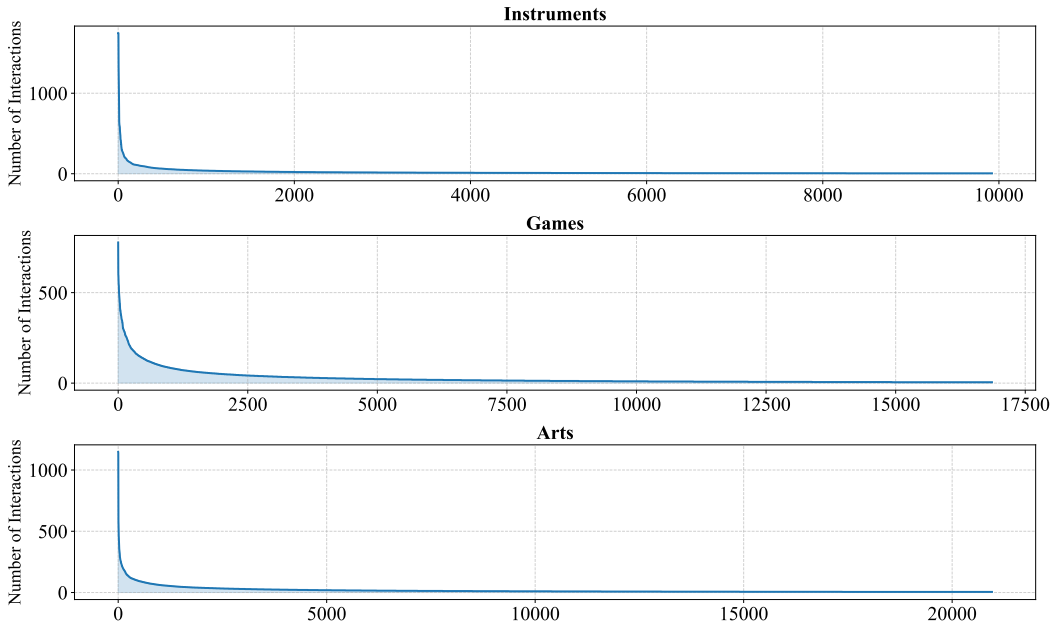

Figure 5: Long tail distribution of the item popularity.

alignment loss to align semantic quantized embeddings with collaborative filtering embeddings from well-trained models. Additionally, it applies a diversity loss based on constrained K-means clustering to regularize code embeddings, effectively mitigating code assignment and item generation biases. When instantiated on generative recommender models, LETTER further incorporates a ranking-guided generation loss to theoretically and empirically augment their top-K ranking capabilities.

• **LC-Rec** [5] is a generative large language model based sequential recommendation approach designed to bridge the semantic gap between the language semantics of LLMs and the collaborative semantics of recommender systems. To represent items effectively without vocabulary explosion, LC-Rec employs a tree-structured vector quantization method based on item text embeddings to construct discrete item indices. It further utilizes a uniform semantic mapping technique [40] during index allocation to eliminate potential index conflicts among items. Instead of relying on predefined candidate sets, LC-Rec can autoregressively generate target items from the entire item set. To achieve deep integration of language and collaborative semantics, the LLM is fine-tuned on a series of specialized semantic alignment tasks, including sequential item prediction, explicit index-language alignment, and implicit recommendation-oriented alignment.

• **ED^2**, i.e., the End-to-End Dual Dynamic recommender [7], is an LLM-based sequential recommender system that introduces a dual dynamic index mechanism. It addresses the limitations of existing LLM-based models that typically separate index generation from the sequential recommendation process and neglect user-related information. By utilizing a dual architecture with two homogeneous discrete index generators, ED^2 synchronously generates indices for both users and items, assembling index generation and sequential recommendation into a unified end-to-end LLM pipeline. To facilitate the LLM comprehension of the untrained dynamic index tokens, the model incorporates a multi-grained token regulator that establishes alignment supervision between dynamic index tokens and corresponding natural language tokens. Additionally, ED^2 leverages customized instruction tuning tasks and associated user collection data to exploit implicit high-order user-item interaction patterns based on historical behaviors.

• ***Augmentation & Substitution*** replace popular head items in the user interaction sequence with their most similar tail items according to a pre-defined replacement probability p. This straightforward principle aims to mitigate popularity bias by artificially increasing the representation of tail items during training. Specifically, for a typical long-tailed distribution where head items account for 80% of total interactions and tail items account for 20%, we set the target replacement probability $p = 0.375$. This value is derived by solving the equilibrium equation $0.8 - 0.8p = 0.2 + 0.8p$ to achieve a perfectly balanced ratio in the modified sequence. The target tail item for substitution is

typically selected based on pre-calculated item similarity. Finally, the modified interaction sequence is integrated into the training process in two different ways. ***(i)*** The *Substitution* baseline directly overwrites the original sequence, achieving a strictly balanced head-to-tail ratio. ***(ii)*** Alternatively, the *Augmentation* baseline appends the modified sequence to the original training set. By combining the original data (0.8 head and 0.2 tail) with the modified data (0.5 head and 0.5 tail), *Augmentation* yields a final head-to-tail ratio of 13:7, approximately 1.857, which significantly alleviates the initial 4:1 imbalance while strictly preserving the original interaction contexts.

• **IFairLRS** [19] is an effective framework designed to enhance the item-side fairness of large language model-based recommendation systems (LRS). The framework addresses the item-side unfairness in LRS that primarily stems from two factors: the imbalanced distribution of historical user interactions and the inherent semantic biases present within LLMs. To mitigate these issues without sacrificing recommendation accuracy, IFairLRS calibrates recommendations by deploying two specifically adapted strategies across the main stages of building an LRS. During the instruction finetuning stage, it employs a reweighting strategy that adjusts the weights of training samples based on the bias observed between the distribution of target items and historical interactions. In the post learning stage (i.e., inference), it utilizes a re-ranking strategy that incorporates a punishment term based on group unfairness (GU) to adjust the final top-K recommendations.

C.3 Metric

Following well-established benchmarks [5, 32], this study adopts Hit-Rate (HR) and normalized discounted cumulative gain (NDCG) to evaluate the recommendation performance. Formally, HR@K is measured as follows,

$$\text{HR@}K = \frac{1}{|U_{\text{test}}|} \sum_{u \in U_{\text{test}}} \mathbb{I}(\mathcal{V}_u \cap L_u^{(K)} \neq \emptyset), \tag{16}$$

where U_{test} denotes the user set for evaluation, $L_u^{(K)}$ denotes the top-K recommendation list for user u, $\mathcal{V}_u$ represents the set of ground-truth interacted items for user u in the test set, and $\mathbb{I}(\cdot)$ is an indicator function that returns 1 if the condition is true and 0 otherwise.

Furthermore, NDCG@K is defined below,

$$\text{NDCG@}K = \frac{1}{|U_{\text{test}}|} \sum_{u \in U_{\text{test}}} \frac{\text{DCG}_u\text{@}K}{\text{IDCG}_u\text{@}K}, \tag{17}$$

$$\text{DCG}_u\text{@}K = \sum_{i=1}^{K} \frac{r_{u,i}}{\log_2(i+1)}. \tag{18}$$

Here, $r_{u,i} \in \{0, 1\}$ indicates whether the i-th recommended item in $L_u^{(K)}$ is relevant, i.e., exists in $\mathcal{V}_u$. $\text{IDCG}_u\text{@}K$ represents the ideal DCG score obtained by perfectly ranking all relevant items in $\mathcal{V}_u$ at the very top of the recommendation list.

This study adopts mean group unfairness (MGU) and average recommendation popularity (ARP) to quantify the fairness and the average popularity of the recommendation results [27]. Particularly, MGU is measured as follows,

$$\text{MGU} = (\text{GU}_{\text{head}} + \text{GU}_{\text{tail}})/2, \tag{19}$$

where GU_G stands for the unfairness of group G and $G \in \{\text{head}, \text{tail}\}$.

GU_G is defined as $\text{GU}_G = \text{GR}_G - \text{GH}_G$, where GR_G and GH_G represent the popularity of group G in recommendation results and interaction history, respectively. Formally, let $\mathbf{H}$ denote the set of all user interaction sequences in the history, $\mathbf{L}$ denote the set of top-K recommendations of all users at the inference phase, and $\mathcal{G}$ denote the set of item groups. Given an item group $G \in \mathcal{G}$, we can measure the recommendation proportion of group G by

$$\text{GR}_G = \frac{\sum_{L \in \mathbf{L}} \sum_{v \in L} \mathbb{I}(v \in G)}{\sum_{G' \in \mathcal{G}} \sum_{L \in \mathbf{L}} \sum_{v \in L} \mathbb{I}(v \in G')}, \tag{20}$$

where $\mathbb{I}(v \in G)$ is an identity function:

$$\mathbb{I}(v \in G) = \begin{cases} 1, & \text{item } v \text{ belongs to group } G \\ 0, & \text{otherwise} \end{cases}. \tag{21}$$

Intuitively, GR_G calculates the recommendation proportion of group G in the top-K recommendations of all users. Accordingly, the interaction proportion of group G in the historical interaction sequences $\mathbf{H}$ can be obtained by

$$\text{GH}_G = \frac{\sum_{H \in \mathbf{H}} \sum_{v \in H} \mathbb{I}(v \in G)}{\sum_{G' \in \mathcal{G}} \sum_{H \in \mathbf{H}} \sum_{v \in H} \mathbb{I}(v \in G')}. \tag{22}$$

Furthermore, ARP is defined below,

$$\text{ARP} = \frac{1}{|U_{\text{test}}|} \sum_{u \in U_{\text{test}}} \frac{\sum_{i \in L_u} \varphi(i)}{|L_u|}, \tag{23}$$

where U_{test} denotes the user set for evaluation, L_u denotes the recommendation list for user u, and $\varphi(i)$ returns the popularity (i.e., the number of occurrences in the training set) of item i.

Moreover, in Section 5, we mentioned that CNS is based on the Min-Max normalized values of HR, NDCG, MGU, and ARP metrics. Specifically, for *higher-is-better* metrics HR and NDCG, the Min-Max normalization is defined as,

$$\bar{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}. \tag{24}$$

On the other side, for *lower-is-better* metrics MGU and ARP, the adopted Min-Max normalization is,

$$\bar{x} = \frac{x_{\max} - x}{x_{\max} - x_{\min}}. \tag{25}$$

C.4 Implementation Detail

Table 6: Hidden state dimensions of Qwen3 series.

Scale	Hidden Dimension
0.6B	1024
1.7B	2048
4B	2560
8B	4096

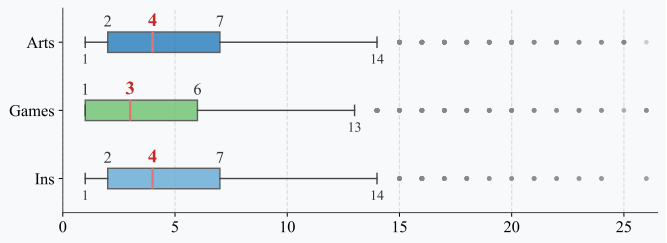

Figure 6: Number of tail items sharing the same SID prefix.

This study employs the open source large language model *Qwen* [41, 42, 43] developed by Alibaba. In particular, the experiments in Section 5, including the main result, the ablation study, and the SID length analysis, adopt *Qwen2.5-3B* as the backbone model for all the evaluated GR models (i.e, Ghost, standard GRs, and GRs popularity debiasing methods). For the scaling pattern analysis, we adopt the fancy *Qwen3 series*, and the backbone scale ranges from 0.6B to 8B. The hidden state dimension of Qwen2.5-3B is 2,048. The hidden state dimensions of the Qwen3 series are listed in Table 6. The original size of all the Qwen vocabulary is 151,936.

Within the skeleton-founded item tokenization (SKT), a series of linear layers [4096, 2048, 1024, 512, 256, 128, 64] is adopted to gradually reduce the Qwen representation into a 32-dimensional space for RQ-Kmeans. Within each RQ-Kmeans iteration, the number of clustering centers is 256 for both head items and tail items. For head items and tail items, the lengths of SIDs generated by SKT are distinct. The length of the head item SIDs exactly equals L^h, while that of the tail item SIDs is $L^h + L^t$. Hence, the SIDs generated by SKT are essentially indices with variable lengths. Figure 6 presents a box plot illustrating the distribution of the number of head prefixes inherited by tail items across three datasets. The visualization highlights the central tendency, variance, and outliers within each domain. The data indicate that the Ins and Arts datasets share identical descriptive statistics, both demonstrating a median value of 4 and an interquartile range (IQR) spanning from 2 to 7. However, the Games dataset exhibits slightly lower overall values, featuring a median of 3 and an IQR ranging from 1 to 6. *For the asymmetric unlikelihood optimization (AUO)*, the default values of K_a, K_b controlling the candidate scale of the undesired item selection are set to 200 and 5. Therefore, for each tail item v', the cardinality of the undesired collection $\mathbf{\Omega}_{v'}$ is 5.

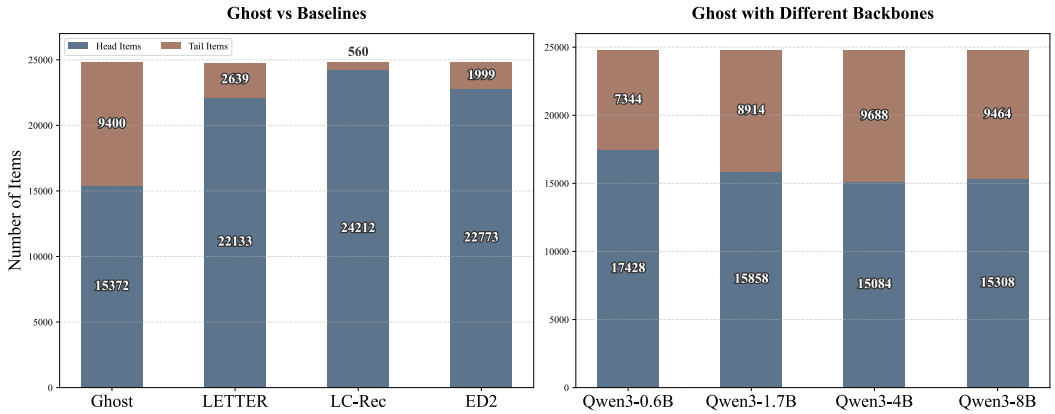

Figure 7: Numbers of head and tail items in the recommendation results provided by *(left)* Ghost and baseline models, and *(right)* Ghost with different backbones.

Regarding the Ghost training phase, we adopt the AdamW optimizer [44] with a learning rate of 3×10^{-5}. In Section F, we conduct an analysis of the undesired collection size and the learning rate. All the experiments are completed on a machine with 8 *NVIDIA A100 Tensor Core 80GB* GPUs.

D Head-to-Tail Ratio Analysis

In this section, we present the ratio of head items to tail items in the recommendation results provided by each GR model. Figure 7 illustrates the exposure distribution of head and tail items retrieved by Ghost and various baselines. While existing generative baselines like LETTER, LC-Rec, and ED2 exhibit severe popularity bias by disproportionately favoring head items. For example, LC-Rec retrieves 24,212 head items versus only 560 tail items. Ghost significantly alleviates this discrepancy, achieving a balanced distribution of 15,372 head and 9,400 tail items. Furthermore, evaluating Ghost across different Qwen3 backbone scales reveals a positive correlation between LLM capacity and long-tail item excavation. Upgrading the backbone from 0.6B to 4B parameters steadily increases the retrieval of tail items from 7,344 to 9,688, demonstrating that the enhanced representation capabilities of larger models effectively empower the framework to deliver diverse, balanced recommendations without overfitting to mainstream trends.

E Equal-sized Grouping based on Item Popularity

To provide a more fine-grained understanding of where the performance improvements originate, we analyze the recommendation results across different popularity segments. Figure 8 details the performance comparison on the Ins dataset, breaking down the item space into five equal-sized groups sorted by popularity (from the most popular 0-20 segment to the least popular 80-100 segment), alongside overall accuracy and debiasing metrics. In general, the fine-grained results confirm that Ghost successfully shifts exposure from over-recommended head items to underexposed mid-tail and tail items without sacrificing general utility. Specifically, we can draw the following three insights:

First, Ghost demonstrates an exceptional capability to excavate and accurately recommend items across the broad long-tail distribution. For the intermediate and long-tail segments (specifically the 20-40, 40-60, and 60-80 groups), Ghost consistently achieves the highest Hit@10 and NDCG@10 scores among all evaluated models. For example, in the 20-40 group, Ghost reaches a Hit@10 of 0.0273, substantially outperforming the standard LC-Rec baseline (0.0070) and the strongest fairness-aware baseline, IFair-RR (0.0226). This proves the model's targeted effectiveness in surfacing relevant, less-mainstream content. ***Second***, Ghost effectively suppresses the over-exposure of head items, leading to significantly enhanced recommendation fairness. This is clearly evidenced by the steep reductions in Average Recommendation Popularity (ARP) and Monopoly Gini of Users (MGU) metrics. Ghost lowers the ARP@10 to 212.73 and MGU@10 to 0.0462, representing a massive drop compared to LC-Rec's 322.14 and 0.1079, respectively. While the heuristic "Replace" baseline

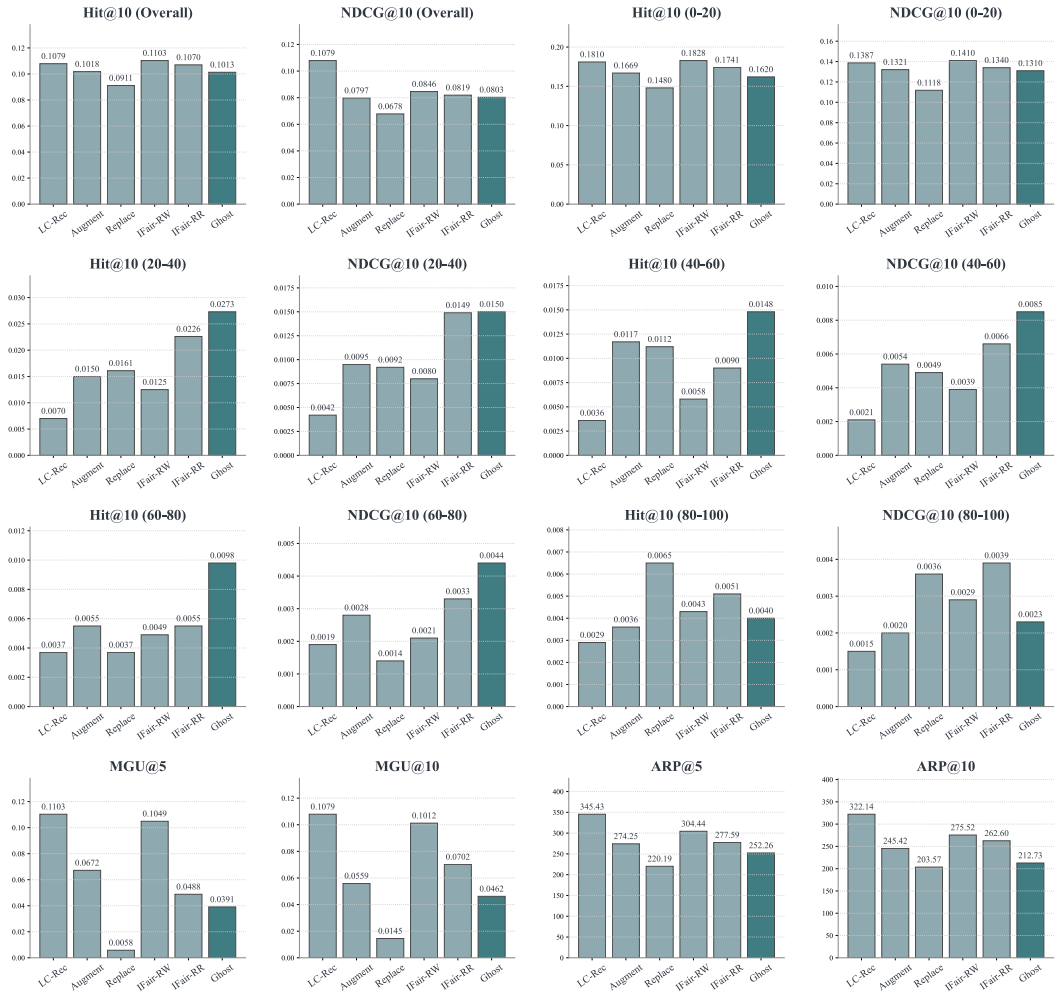

Figure 8: Performance comparison of each equal-sized grouping on Ins dataset.

achieves slightly lower ARP and MGU scores, it does so by indiscriminately swapping items, which severely damages recommendation quality.

F Hyper-parameter Analysis

F.1 AUO Weight α

Here, we investigate the impact of the weight parameter α of AUO. Figure 9 investigates the sensitivity of the Ghost model to the AUO weight α on the Ins dataset. The results indicate that α plays a critical role in balancing recommendation accuracy and popularity bias mitigation. Specifically, $\alpha = 0.1$ emerges as the optimal setting, achieving the highest Overall NDCG@10 and Tail NDCG@10 while maintaining highly competitive Overall and Tail HR@10. Crucially, at this optimal point, the ARP@10 experiences a local drop, confirming the model's capability to effectively surface less popular items without sacrificing accuracy. Conversely, increasing the weight beyond 0.1 (e.g., 0.2 to 0.5) consistently degrades both overall and tail-specific accuracy metrics, significantly diminishes MGU@10, and steadily increases ARP@10. This clear trend demonstrates that an excessively large AUO weight forces the model to over-prioritize mainstream items, thereby exacerbating popularity bias and compromising both long-tail excavation and overall recommendation quality.

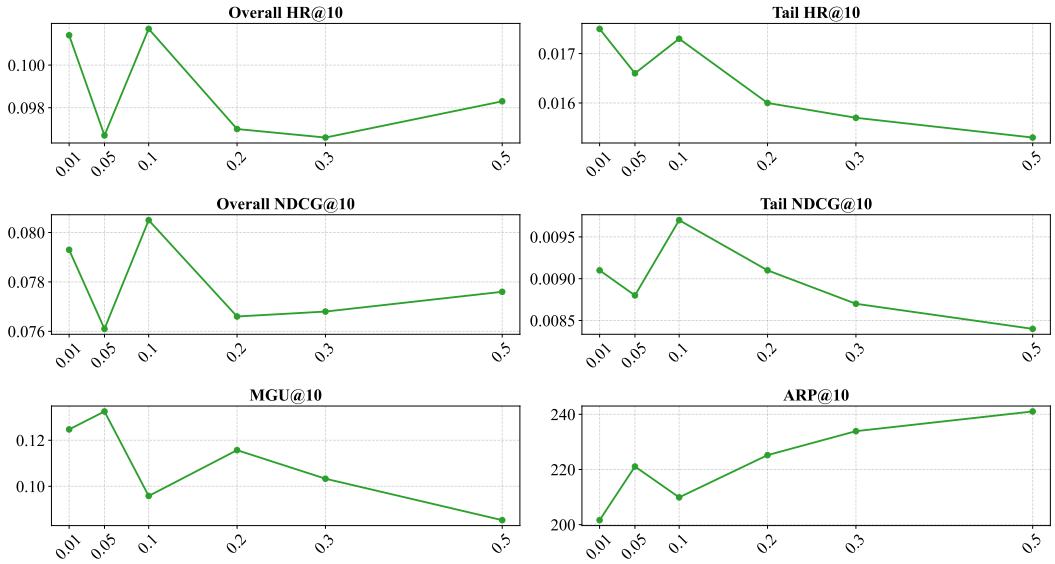

Figure 9: Tendency of Ghost performance on Ins dataset, under different AUO weights α. The x-axis denotes the values of α, and the y-axis is the metric values.

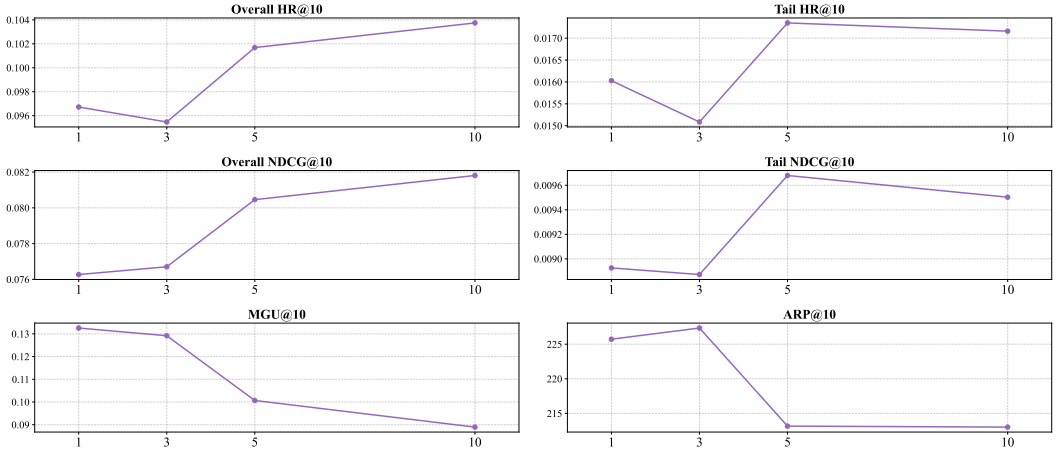

Figure 10: Tendency of Ghost performance on Ins dataset, under different undesired collection sizes. The x-axis denotes the size of undesired collection, and the y-axis is the metric values.

F.2 Size of Undesired Collection $\bar{\Omega}$

Here, we investigate the impact of the undesired collection size. Figure 10 investigates the impact of the undesired collection size $|\bar{\Omega}|$ on the performance of the Ghost model using the Ins dataset. The empirical results demonstrate that the magnitude of $|\bar{\Omega}|$ critically influences the trade-off between recommendation accuracy and popularity bias mitigation. Specifically, setting the collection size to 5 emerges as the optimal configuration; at this point, the model achieves peak effectiveness in excavating long-tail items, evidenced by the highest Tail HR@10 and Tail NDCG@10, alongside a sharp and favorable decline in ARP@10. Concurrently, overall accuracy metrics (Overall HR@10 and NDCG@10) experience substantial gains compared to smaller sizes. However, expanding the undesired collection further to 10 yields diminishing returns: while overall accuracy marginally increases, tail-specific metrics slightly degrade, and user coverage (MGU@10) continues a steady, negative decline. This indicates that a moderate undesired collection size effectively provides sufficient contrast against over-recommended popular items to surface relevant tail items, whereas an excessively large collection may introduce noise that compromises tail item retrieval and overall recommendation diversity across different users.

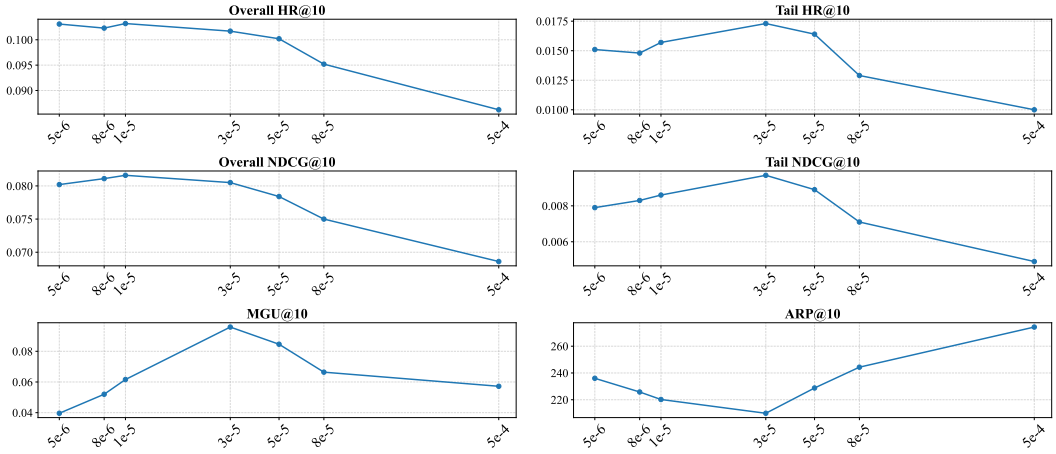

Figure 11: Tendency of Ghost performance on Ins dataset, under different learning rates. The x-axis denotes learning rate, and the y-axis is the metric values.

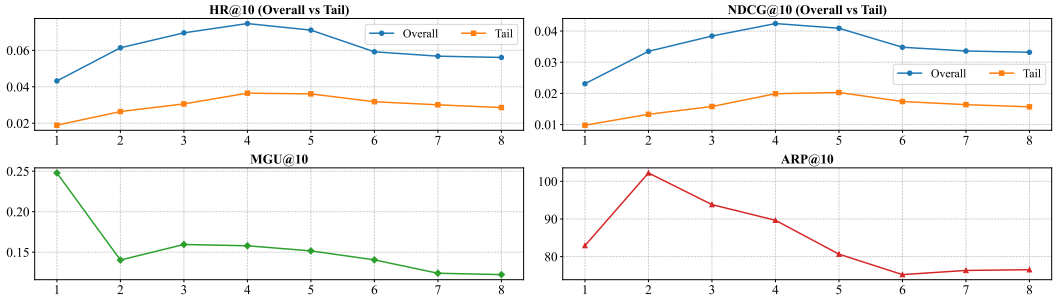

Figure 12: Tendency of Ghost performance on Games dataset, under different optimization epochs. The x-axis denotes the epoch values, and the y-axis is the metric values.

F.3 Learning rate

Here, we investigate the impact of the learning rate during Ghost optimization. Figure 11 illustrates the sensitivity of the Ghost model to varying learning rates on the Ins dataset, revealing its crucial role in balancing overall recommendation accuracy with popularity bias mitigation. The empirical trends identify 3×10^{-5} as the optimal learning rate configuration. At this specific point, the model attains peak effectiveness in long-tail item excavation, evidenced by the highest Tail HR@10 and Tail NDCG@10, while simultaneously maximizing user coverage (MGU@10). Crucially, the ARP@10 reaches its global minimum at this setting, indicating a strong suppression of popularity bias without severely compromising the highly competitive overall accuracy metrics (Overall HR@10 and NDCG@10, which peak slightly earlier around 1×10^{-5}). Conversely, employing an excessively large learning rate (e.g., 5×10^{-4}) yields detrimental effects across all dimensions: overall and tail-specific accuracies sharply degrade, user coverage diminishes, and ARP@10 spikes significantly. This demonstrates that an overly aggressive learning rate destabilizes the debiasing mechanism, causing the model to collapse back into disproportionately favoring mainstream items.

F.4 Epoch

Here, we investigate the impact of the optimization epoch. Figure 12 visualizes the impact of the number of optimization epochs on the Ghost model performance using the Games dataset. The empirical trajectories indicate that determining an appropriate stopping point is crucial for balancing recommendation accuracy with popularity bias mitigation. Specifically, epoch 4 emerges as the optimal training duration, where both overall and tail-specific accuracy metrics (HR@10 and NDCG@10) simultaneously achieve their peak performance. During the initial training phase (epochs 1 and 2), the model exhibits suboptimal accuracy and a sharp spike in Average Recommendation Popularity (ARP@10), despite an initially high user coverage (MGU@10) that quickly drops. As

optimization progresses towards epoch 4, the model effectively converges, significantly enhancing long-tail item retrieval while steadily driving down ARP@10. Conversely, prolonged training (epochs 6 through 8) evidently leads to overfitting; this over-optimization results in a consistent degradation across both overall and tail accuracies, a continued decline in MGU@10, and yields no further meaningful benefits for popularity debiasing. Therefore, maintaining a moderate number of optimization epochs is essential to maximize the framework's capacity to deliver accurate, balanced, and diverse recommendations.

G Theoretical Derivations and Justifications

This section provides the rigorous theoretical foundations and detailed mathematical proofs for the lemmas and corollaries presented in the main text regarding the Ghost model. Specifically, it elucidates the gradient starvation issue inherent in Maximum Likelihood Estimation (MLE), the localized bias amplification effect induced by undifferentiated tokenization, and the mathematical mechanisms through which Skeleton-Founded Tokenization (SKT) and Asymmetric Unlikelihood Optimization (AUO) structurally mitigate popularity bias.

G.1 Prerequisites and Theoretical Foundations

To establish a rigorous theoretical framework for diagnosing popularity bias in Generative Recommenders (GRs), we first formalize three fundamental prerequisites regarding the data distribution and the architecture of Semantic Index (SID) based models.

- **Prerequisite 1 (Long-tail Distribution)**. In the sequential recommendation training dataset $\mathcal{D}$, the frequency of user-item interactions exhibits a heavily skewed, long-tailed distribution (e.g., a power-law distribution). Consequently, the occurrence probability of items in the head set $\mathcal{V}_{\text{head}}$ (the top 20% most popular items) is vastly greater than that of items in the tail set $\mathcal{V}_{\text{tail}}$ (the remaining 80%), i.e., $\mathbb{P}_{\mathcal{D}}(v \in \mathcal{V}_{\text{head}}) \gg \mathbb{P}_{\mathcal{D}}(v \in \mathcal{V}_{\text{tail}})$.
- **Prerequisite 2 (Inner Product Mapping)**. Current SOTA GR architectures formulate the next-token prediction task by computing generative logits via the inner product of the user historical behavior representation X_{h_u} and the candidate token embedding e_c. The generation probability is formalized via a Softmax operation across the token vocabulary,

$$\mathcal{P}_\theta(c|h_u) = \frac{\exp(\langle e_c, X_{h_u} \rangle)}{\sum_{c'} \exp(\langle e_{c'}, X_{h_u} \rangle)} \tag{26}$$

- **Assumption 1 (Identical Token Space under Undifferentiated Tokenization)**. Current vector quantization techniques (e.g., RQ-VAE, RQ-KMeans) process head and tail items indiscriminately, meaning they share and compete within the identical representation space. Let c_{tail} denote a token structurally exclusive to tail items. Due to the heavy-tailed interaction distribution established in Prerequisite 1, the mathematical expectation of c_{tail} acting as a ground-truth target token within the training distribution $\mathcal{D}$ asymptotically approaches zero.

G.2 Detailed Derivations of LEMMA 1 and COROLLARY 1

Proof of LEMMA 1 (Gradient Starvation in MLE).

Basically, the model is optimized using standard Maximum Likelihood Estimation, specifically minimizing the Negative Log-Likelihood loss over the sequential tokens of the target item SID.

For an arbitrary generation step i and target item v, the instantaneous loss is

$$\mathcal{L}_{\text{NLL}}^{(i)} = -\log \mathcal{P}_\theta(c_v^{(i)}|h_u, c_v^{<i}).$$

According to the standard derivative properties of the Softmax function in Eq.(26), the partial derivative of the prediction probability with respect to an arbitrary candidate token embedding e_c yields a Jacobian formulation that separates into target and non-target cases. Consequently, the gradient of the NLL loss with respect to e_c is,

$$\frac{\partial \mathcal{L}_{\text{NLL}}^{(i)}}{\partial e_c} = \left(\mathcal{P}_\theta(c|h_u, c_v^{<i}) - \mathbb{I}\{c = c_v^{(i)}\} \right) X_{h_u} \tag{27}$$

In gradient descent, the parameter update direction opposes the computed gradient, denoted as $\Delta e_c \propto -\frac{\partial \mathcal{L}}{\partial e_c}$. Taking the mathematical expectation over the entire training distribution $\mathcal{D}$, we obtain,

$$\mathbb{E}_{\mathcal{D}}[\Delta e_c] \propto \mathbb{E}_{\mathcal{D}}\left[\sum_i \left(\mathbb{I}\{c = c_v^{(i)}\} - \mathcal{P}_\theta(c|h_u, c_v^{<i})\right) X_{h_u}\right] \tag{28}$$

Now, consider a tail-specific token c_{tail}. Based on Prerequisite 1 and Assumptions 1, its probability of being sampled as the ground-truth target token within any batch drawn from $\mathcal{D}$ is negligible, $\mathbb{P}_{\mathcal{D}}(c_{\text{tail}} = c_v^{(i)}) \approx 0$. Consequently, the indicator function term $\mathbb{I}\{c_{\text{tail}} = c_v^{(i)}\}$ reliably vanishes. By projecting the expected gradient update onto the user preference vector X_{h_u}, we isolate the token alignment trajectory, formulated as,

$$\mathbb{E}_{\mathcal{D}}[\langle \Delta e_{c_{\text{tail}}}, X_{h_u} \rangle] \approx -\mathbb{E}_{\mathcal{D}}\left[\sum_i \mathcal{P}_\theta(c_{\text{tail}}|h_u, c_v^{<i}) \cdot \|X_{h_u}\|_2^2\right] \leq 0 \tag{29}$$

Conclusion. Eq.(29) rigorously proves that tail tokens consistently act as trivial negative samples within the Softmax denominator during MLE optimization. Lacking positive reinforcement from the indicator function, their embeddings $e_{c_{\text{tail}}}$ are pathologically pushed away from the user intent space, trapping them in a state of Gradient Starvation [31].

Proof of COROLLARY 1 (Head Token Dominance at Branching Point).

Building upon LEMMA 1, the parameter space is assumed to have converged to a skewed state where head tokens have received massive positive updates due to high interaction frequencies, i.e., $\mathbb{I}\{c_{\text{head}} = c_v^{(i)}\} = 1$ frequently, while tail tokens are starved.

This optimization skew manifests in the representation space as an extreme disparity in inner products, $\langle e_{c_{\text{head}}}, X_{h_u} \rangle \gg \langle e_{c_{\text{tail}}}, X_{h_u} \rangle$. Let i denote an arbitrary generative branching point where head and tail candidate tokens structurally compete. The predicted probability ratio for these tokens is given by,

$$\frac{\mathcal{P}_\theta(c_{\text{head}}^{(i)}|h_u, c^{<i})}{\mathcal{P}_\theta(c_{\text{tail}}^{(i)}|h_u, c^{<i})} = \frac{\exp(\langle e_{c_{\text{head}}^{(i)}}, X_{h_u} \rangle)}{\exp(\langle e_{c_{\text{tail}}^{(i)}}, X_{h_u} \rangle)} = \exp\left(\langle e_{c_{\text{head}}^{(i)}} - e_{c_{\text{tail}}^{(i)}}, X_{h_u} \rangle\right) \tag{30}$$

Because the gradient updates heavily favor the head token, the exponential term in Eq.(30) is pathologically amplified. This causes the predictive distribution $\mathcal{P}_\theta$ to diverge significantly from the true underlying data distribution $\mathcal{P}_d$. We formally define this local divergence as the amplification factor γ_i as follows,

$$\gamma_i = \frac{\mathcal{P}_\theta(c_{\text{head}}^{(i)}|h_u, c^{<i})/\mathcal{P}_\theta(c_{\text{tail}}^{(i)}|h_u, c^{<i})}{\mathcal{P}_d(c_{\text{head}}^{(i)}|h_u, c^{<i})/\mathcal{P}_d(c_{\text{tail}}^{(i)}|h_u, c^{<i})} > 1 \tag{31}$$

Eq.(31) demonstrates the inevitable probability dominance of head tokens, establishing that the generative process becomes overconfident in predicting head tokens, regardless of context.

G.3 Detailed Derivations of LEMMA 2 and LEMMA 3

Proof of LEMMA 2 (Bias Amplification via Undifferentiated Tokenization).

Basically, the current item tokenization method assigns Semantic Indices (SIDs) of length L indiscriminately, treating all items equally without accounting for popularity.

Under undifferentiated tokenization, a long-tail item v_{tail} shares prefixes of varying lengths with numerous head items. Consequently, generating v_{tail} requires decoding a sequence of tokens where it must repeatedly survive competition against popular items.

Suppose there exists a set $\mathcal{Z}$ containing z unpredictable branching points ($|\mathcal{Z}| = z \leq L$) where tail tokens structurally compete against head tokens. Based on the chain rule of autoregressive sequence generation and the dominance established in Corollary 1, each time the generation navigates a competitive branching point $j \in Z$, the relative generation probability of the tail token is suppressed by at least a factor of γ_j,

$$\mathcal{P}_\theta(c_{\text{tail}}^{(j)}|h_u, c^{<j}) \leq \gamma_j^{-1} \cdot \mathcal{P}_d(c_{\text{tail}}^{(j)}|h_u, c^{<j}) \tag{32}$$

Over the entire generation sequence of v_{tail}, we multiply the conditional probabilities. The suppressions at branching points accumulate geometrically,

$$\begin{aligned}\mathcal{P}_\theta(v_{\text{tail}}|h_u) &= \prod_{j=1}^{L} \mathcal{P}_\theta(c_{\text{tail}}^{(j)}|h_u, c^{<j}) \\ &\leq \left(\prod_{j\in Z} \gamma_j^{-1}\right) \prod_{j=1}^{L} \mathcal{P}_d(c_{\text{tail}}^{(j)}|h_u, c^{<j}) \\ &\leq (\gamma_{\text{min}})^{-z} \prod_{j=1}^{L} \mathcal{P}_d(c_{\text{tail}}^{(j)}|h_u, c^{<j}) \end{aligned} \tag{33}$$

where $\gamma_{\text{min}} = \min_{j\in\mathcal{Z}}(\gamma_j) > 1$.

Conclusion. This derivation establishes that undifferentiated tokenization cascades localized, token-level gradient bias into a macroscopic, geometric $\mathcal{O}(\gamma_{\text{min}}^z)$ probability suppression for tail items, fully explaining their severe marginalization in recommendation lists.

Proof of LEMMA 3 (Mitigation of Bias Amplification via SKT).

The Skeleton-Founded Tokenization (SKT) mechanism asynchronously defines SIDs. The head items dictate the L^h-length skeleton of the SID space. A tail item v' is forced to explicitly inherit the first L^h tokens from its semantically closest head item v^*, and subsequently generates L^t specific tokens to characterize its distinctiveness.

During the initial generation steps $j \in [1, L^h]$, the tail item v' and the head item v^* share an identical prefix trajectory ($c_{v'}^{(j)} = c_{v^*}^{(j)}$). Consequently, no probability divergence or chaotic token competition occurs between them. The unstructured branching points that comprised set $\mathcal{Z}$ in LEMMA 2 are entirely eliminated. The genuine structural divergence is uniformly deferred to the single, predictable step $(L^h + 1)$. At this exact locus, the head item outputs an End-Of-Sequence (EOS) token, whereas the tail item generates the first token of its distinct semantic prefix. Because the competition is now strictly restricted to this single step, the cardinality of the branching set z is explicitly limited to 1.

The geometric suppression series detailed in Eq.(33) therefore collapses into a singular, localized deviation bounded by the head-dominance factor at the EOS step (γ_{EOS}):

$$\mathcal{P}_\theta(v'|h_u) = \prod_{j=1}^{L^h+L^t} \mathcal{P}_\theta(c_{v'}^{(j)}|h_u, c^{<j}) \approx (\gamma_{EOS})^{-1} \prod_{j=1}^{L^h+L^t} \mathcal{P}_d(c_{v'}^{(j)}|h_u, c^{<j}) \tag{34}$$

Conclusion. This rigorously proves that SKT effectively halts the Markovian amplification chain of popularity bias. By establishing a unified structural branching point, it transforms a multi-step geometric suppression $\mathcal{O}(\gamma_{\text{min}}^z)$ into an insulated, single-step discrepancy $\mathcal{O}(\gamma_{EOS})$.

G.4 Detailed Derivations of LEMMA 4

Proof of LEMMA 4 (Gradient Rescue based on AUO).

The Ghost model incorporates Asymmetric Unlikelihood Optimization (AUO) to actively penalize a dynamically generated set of undesired tokens $\bar{\mathbf{\Omega}}$. For a target tail item v', $\bar{\mathbf{\Omega}}$ consists of SIDs from head items that share high textual similarity with v' but possess divergent SID structures, acting as deceptive popular distractions.

The AUO loss function introduces an explicit penalty for generating tokens in $\bar{\mathbf{\Omega}}$,

$$\mathcal{L}_{\text{AUO}} = -\sum_{c\in\bar{\mathbf{\Omega}}} \log(1 - \mathcal{P}_\theta(c)) \tag{35}$$

To derive the partial derivative with respect to an arbitrary candidate embedding e_k, let the logit be defined as $z_k = \langle e_k, X_{h_u}\rangle$. We first utilize the Jacobian matrix of the Softmax function for individual output probabilities,

- For the diagonal term ($i = k$),

$$\frac{\partial \mathcal{P}_\theta(c_i)}{\partial z_k} = \mathcal{P}_\theta(c_i)(1 - \mathcal{P}_\theta(c_i))$$

- For the off-diagonal terms ($i \neq k$),

$$\frac{\partial \mathcal{P}_\theta(c_i)}{\partial z_k} = -\mathcal{P}_\theta(c_i)\mathcal{P}_\theta(c_k)$$

Applying the chain rule to the AUO objective yields:

$$\frac{\partial \mathcal{L}_{\text{AUO}}}{\partial z_k} = \sum_{c_i \in \bar{\mathbf{\Omega}}} \frac{1}{1 - \mathcal{P}_\theta(c_i)} \frac{\partial \mathcal{P}_\theta(c_i)}{\partial z_k} \tag{36}$$

Scenario 1: For a false positive head token $c^-_{\text{head}} \in \bar{\mathbf{\Omega}}$ (Active Suppression).

Here, the embedding of interest $k = c^-_{\text{head}}$ belongs to the actively penalized set. We separate the summation in Eq.(36) into terms where $i = k$ and $i \neq k$,

$$\frac{\partial \mathcal{L}_{\text{AUO}}}{\partial z^-_{\text{head}}} = \frac{\mathcal{P}_\theta(c^-_{\text{head}})(1 - \mathcal{P}_\theta(c^-_{\text{head}}))}{1 - \mathcal{P}_\theta(c^-_{\text{head}})} + \sum_{c_i \in \bar{\mathbf{\Omega}} \setminus \{c^-_{\text{head}}\}} \frac{-\mathcal{P}_\theta(c_i)\mathcal{P}_\theta(c^-_{\text{head}})}{1 - \mathcal{P}_\theta(c_i)} \tag{37}$$

Simplifying the first term, we get,

$$\frac{\partial \mathcal{L}_{\text{AUO}}}{\partial z^-_{\text{head}}} = \mathcal{P}_\theta(c^-_{\text{head}}) - \sum_{c_i \neq c^-_{\text{head}}} \frac{\mathcal{P}_\theta(c_i)\mathcal{P}_\theta(c^-_{\text{head}})}{1 - \mathcal{P}_\theta(c_i)} \tag{38}$$

The overall objective is a weighted combination of MLE and AUO ($\mathcal{L} = \mathcal{L}_{NLL} + \alpha\mathcal{L}_{\text{AUO}}$). Incorporating the standard MLE base gradient (which negatively updates c^-_{head} as it is a false negative, not the target), the overall parameter update direction becomes,

$$\Delta e_{c^-_{\text{head}}} \propto -(1 + \alpha)\mathcal{P}_\theta(c^-_{\text{head}})X_{h_u} + \alpha \sum_{c_j \in \bar{\mathbf{\Omega}} \setminus \{c^-_{\text{head}}\}} \frac{\mathcal{P}_\theta(c_j)\mathcal{P}_\theta(c^-_{\text{head}})}{1 - \mathcal{P}_\theta(c_j)} X_{h_u} \tag{39}$$

This explicitly shows the direct penalty via the softmax derivative.

Scenario 2: For a target tail token $c_{\text{tail}} \notin \bar{\mathbf{\Omega}}$ (Cross-Penalization Rescue).

Here, the embedding of interest $k = c_{\text{tail}}$ corresponds to our ground-truth tail item, which is specifically excluded from the explicitly penalized head set $\bar{\mathbf{\Omega}}$. Therefore, the condition $i \neq k$ strictly holds for *all* terms in the summation of Eq.(36),

$$\frac{\partial \mathcal{L}_{\text{AUO}}}{\partial z_{\text{tail}}} = \sum_{c_j \in \bar{\mathbf{\Omega}}} \frac{-\mathcal{P}_\theta(c_j)\mathcal{P}_\theta(c_{\text{tail}})}{1 - \mathcal{P}_\theta(c_j)} \tag{40}$$

In gradient descent, the specific parameter update contribution derived from AUO towards this tail token acts in the opposing direction ($-\alpha \cdot \partial \mathcal{L}_{\text{AUO}} / \partial z_{\text{tail}}$),

$$\Delta e^{\text{AUO}}_{c_{\text{tail}}} \propto +\alpha \sum_{c_j \in \bar{\mathbf{\Omega}}} \frac{\mathcal{P}_\theta(c_j)\mathcal{P}_\theta(c_{\text{tail}})}{1 - \mathcal{P}_\theta(c_j)} X_{h_u} \tag{41}$$

Merging this positive AUO rescue term with the suppressive MLE base term established in LEMMA 1 rigorously completes the derivation:

$$\Delta e_{c_{\text{tail}}} \propto -\mathcal{P}_\theta(c_{\text{tail}})X_{h_u} + \alpha \sum_{c_j \in \bar{\mathbf{\Omega}}} \frac{\mathcal{P}_\theta(c_j)\mathcal{P}_\theta(c_{\text{tail}})}{1 - \mathcal{P}_\theta(c_j)} X_{h_u} \tag{42}$$

Conclusion. This Jacobian analysis illuminates how the AUO loss function systematically transfigures the penalization of notorious head tokens into a positive, structural rescue force for long-tail tokens. By redistributing probability mass, this mechanism actively counteracts and dismantles the gradient starvation trap established in LEMMA 1, ensuring tail tokens receive rational parameter updates.

H Limitations and Future Work

• **Supervised Finetuning**. While this study provides a comprehensive diagnosis and mitigation strategy for popularity bias in GRs, the scope of our investigation is currently constrained to the supervised fine-tuning (SFT) paradigm. Specifically, our theoretical analysis of the gradient starvation issue and the subsequent design of the AUO are fundamentally grounded in the MLE framework. Recent advancements in generative modeling have increasingly adopted reinforcement learning (RL) techniques to align models with complex objectives [45, 46]. The specific mechanisms by which popularity bias manifests in RL-based GRs, and whether our proposed skeleton-founded tokenization and unlikelihood penalties remain effective under reward-driven optimization, have not yet been analyzed and represent a critical direction for future research.

• **Pre-computed Undesired Collection**. The undesired collection utilized in the AUO module relies on a static, pre-computed approach. As detailed in our methodology, the undesired items are identified by retrieving head items with high initial semantic similarity but divergent SIDs relative to the target tail item. Currently, this collection is established prior to training and is not dynamically adjusted as the model's parameters and internal representations evolve during the optimization process. While dynamically updating the undesired collection at each epoch could theoretically provide more precise and adaptive negative supervision, the static approach was deliberately adopted as a pragmatic trade-off to preserve computational efficiency and prevent prohibitive training overhead.

I Large Language Models Usage

In this work, large language models (LLMs) were used solely in an auxiliary capacity. Specifically, they assisted in improving the precision of the prose by identifying and correcting grammatical mistakes, refining word choices, and suggesting appropriate color schemes for figures. All research ideas, methodological developments, experiments, and the main body of the manuscript were independently conceived, carried out, and written by the authors.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction of this article accurately summarize the core contributions, and all the claims are fully supported by the theoretical analysis and experimental results presented in the text.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In Section H, we discuss the limitations of this work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Regarding the theoretical results in main text, we provide detailed derivations alongside explanations in Section G.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We elaborate the implementation details, including the datasets, metrics, and model configurations in Section C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Our whole repository is available at *Ghost Project*.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: In Section C.4, we introduce the details of experimental setting.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: From Section **??** to Section F, we conduct extensive experiments to demonstrate the effectiveness of our method.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the compute resource in Section C.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We make sure that this study conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The impact statement is discussed in Section 7.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: [N/A]

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Models and datasets used in this study are all public and open-source assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: [N/A]

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: [N/A]

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: [N/A]

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We declare the usage of LLM in Section I.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.