Mitigating Knowledge Bias in LLM-based Sequential Recommendation through Decoding-Time Alignment

Anonymous ACL submission

Abstract

The integration of large language models (LLMs) into recommender systems has garnered attention due to their ability to leverage rich semantic representations. However, these models suffer from a limitation: **knowledge bias**—a systematic tendency to over-recommend items dominated by pretraining knowledge, regardless of actual user interests. Although recent studies have focused on fine-tuning LLMs to better align with user interaction data, our investigation reveals that such fine-tuning fails to fundamentally correct this deep-seated bias. In this work, we present the first in-depth analysis of knowledge bias in LLM-based recommendation. Empirically, we observe that this bias persists across different backbones, even under large-scale fine-tuning. To address this, we propose **ReKnow**, a novel decoding-time alignment method to mitigate knowledge bias. Specifically, we quantify LLMs' knowledge towards biased items, then realign output probabilities to match the target data distribution. For validation, we provide both theoretical justifications and empirical results, including evaluation on two datasets, demonstrating that ReKnow enhances the quality and diversity of LLM-based sequential recommendation. Source code[1] is provided to support reproducibility.

1 Introduction

Recent advancements in Large Language Models (LLMs) have sparked growing interest in their integration into recommender systems, owing to their capacity to encode extensive knowledge bases (Ai et al., 2023; Wu et al., 2024). Among these efforts, generative LLM-based sequential recommendation has demonstrated significant potential (Bao et al., 2023a,b; Zhang et al., 2023; Di Palma et al., 2023; Deldjoo, 2024). In this framework, the model utilizes user's historical interactions as input to predict the next item in the sequence, as depicted in Figure 1.

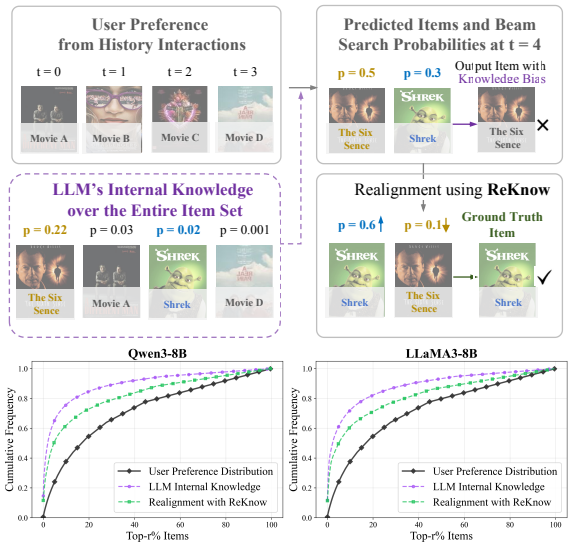

Figure 1: LLM-based sequential recommendation suffers from inherent knowledge bias that distort predictions beyond user interests. ReKnow addresses this by adjusting item probabilities through decoding-time alignment, effectively realigning the predicted cumulative frequency curve to closely match the ground truth user preference distribution.

However, despite their expressive power, LLM-based recommendation exhibits a systematic limitation: **knowledge bias**, i.e., the tendency to over-recommend items dominated by the model's pretraining knowledge, often at odds with actual user interests. As illustrated in Figure 1, even when a user's interaction history strongly suggests a preference for a specific item (e.g., "Shrek"), the model still assigns a significantly higher prediction probability to "The Sixth Sense" according to its internal knowledge. On a broader scale, this discrepancy manifests as a severe shift where the LLM's prediction distribution consistently deviates from the user preference curve.

[1] https://anonymous.4open.science/r/ReKnow-B17E

Recent efforts have focused on fine-tuning LLMs to better align with user interaction data (Jiang et al., 2024; Bao et al., 2023a). However, as shown in Section 2.2, increasing the fine-tuning data size does little to mitigate the model's knowledge bias. In parallel, several recent studies have explored bias mitigation in LLM-based recommender systems, such as addressing popularity bias (Di Palma et al., 2023) and provider fairness (Deldjoo, 2024; Kolb, 2024; Spurlock et al., 2024), yet these approaches do not explicitly target *knowledge bias*. Furthermore, these debiasing strategies fall short in two aspects: they do not explicitly quantify the model's bias, nor do they define an unbiased target to guide correction. For example, reprompting methods (Spurlock et al., 2024; Deldjoo, 2024) aim to reduce popularity bias by manually instructing the model to recommend less popular items. However, they lack a formal quantification of the model's bias, since the model's implicit notion of "popularity" may not align with the true item frequency. Another approach adjusts the LLM's recommendation using traditional recommendation models (Bao et al., 2024), but traditional models may introduce their own biases. These limitations make the debiasing process difficult to control.

In this work, we make the first systematic investigation into the origin and persistence of *knowledge bias* in LLM-based recommender systems. Our study reveals that such bias is an outcome of the model's pretraining process, and it persists even after large-scale fine-tuning. These limitations raise a critical question: *How can we systematically detect and correct knowledge bias in a controllable way?* Our key contributions are summarized as follows:

- ***Observations.*** Through controlled experiments in Section 2, we identify two critical characteristics of knowledge bias: (1) it manifests as distinct item preferences across various architectures and model sizes, and (2) it persists despite exponential increases in fine-tuning data. This phenomenon not only *reduces recommendation accuracy* by skewing predictions distributions but also *diminishes diversity* by overly favoring a limited set of items.
- ***Method.*** We propose a novel method, **ReKnow** (**Re**alignment at decoding-time to alleviate **Know**ledge bias). It guides the model's predictions using a target distribution that represents the true user interests. In practice, we approximate the target distribution with the observed distribution of items in the training set. We provide an alignment method that adjusts the LLM predictions to align with the target distribution, which involves three key steps: first, identifying biased items using a small calibration set; second, quantifying the model's inherent knowledge through a context-free prompt; and third, adjusting the model's predictions toward the target distribution with alignment weights at decoding time.
- ***Validation***. We provide both *empirical* and *theoretical* validation of ReKnow. Our empirical analysis visualizes how alignment weights correlate with the actual bias patterns, while theoretical study justifies our alignment approach. Extensive experiments on two representative backbones and datasets further demonstrate that ReKnow offers a lightweight and generalizable solutions to mitigate knowledge bias while consistently improving both recommendation accuracy and diversity.

2 Knowledge Bias in LLM-based Recommendation

In this section, we investigate the nature of knowledge bias and its persistence against fine-tuning.

2.1 The Nature and Universality

Backbone-Dependent Bias Patterns. We first delineate the nature of knowledge bias and verify its prevalence across different models. Distinct from traditional popularity bias (Klimashevskaia et al., 2024; Abdollahpouri et al., 2019), which stems from long-tail distributions in training data (i.e., data-driven), knowledge bias is rooted in the LLM's parametric memory (i.e., model-driven). As illustrated in Figure 2, distinct backbones exhibit divergent preferences even when fine-tuned on the identical dataset (see Appendix D for detailed settings). Specifically, Qwen3-8B tends to over-generate "Star Wars: Episode IV", while LLaMA3-8B exhibits a strong inclination towards "The Bourne Identity". Since the training data is held constant, this discrepancy confirms that the bias originates from the unique pre-training knowledge of each backbone rather than the data distribution.

Universality. To quantify knowledge bias, we visualize the Cumulative Distribution Function

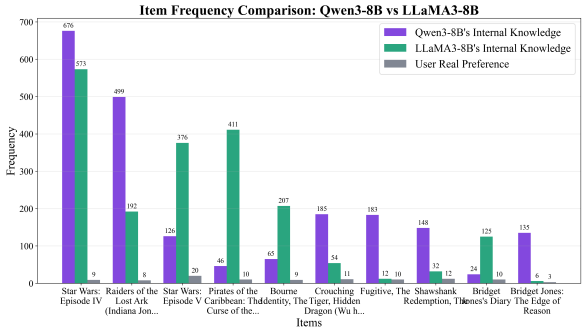

Figure 2: Comparison of item frequency.

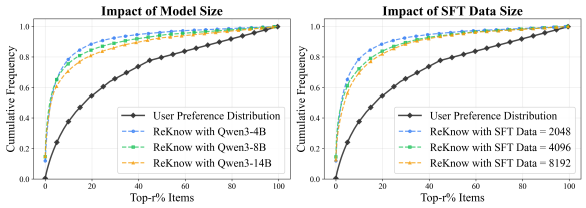

Figure 3: The CDF curves demonstrate that neither scaling the model size (Left) nor increasing the SFT data size (Right) effectively bridges the gap between model predictions and user real preferences.

(CDF) in Figure 3 and Figure 1. We independently sort the items by their frequency in descending order. The x-axis represents the top-$r\%$ of these ranked items, while the y-axis denotes the cumulative frequency. Consequently, a larger deviation from the user preference distribution curve indicates a more pronounced knowledge bias. As shown in Figure 1, across diverse backbones (Qwen3-8B and LLaMA3-8B), the *LLM Internal Knowledge* curve ascends significantly steeper than the *User Preference Distribution*. Furthermore, the left subplot of Figure 2 demonstrates that this phenomenon remains prevalent across varying model parameter scales (from Qwen3-4B to 14B), with larger models exhibiting slightly reduced bias.

2.2 Persistence Against Fine-Tuning

A natural question is whether increasing the scale of Supervised Fine-Tuning (SFT). To explore this, we conduct a controlled experiment by fine-tuning Qwen3-4B on datasets of varying sizes: 2048, 4096, and 8192 samples. As shown in Figure 3, the distributional curves for different data sizes maintains a large gap from the user preference curve. This indicates that standard SFT is insufficient to alleviate the deep-seated priors established during pre-training. These observations confirm that knowledge bias is a persistent, model-intrinsic challenge that requires a dedicated decoding-time alignment strategy for mitigation.

3 Proposed Method: ReKnow

Overview. This section presents our proposed framework, ReKnow. We first explain how LLMs act as sequential recommenders, then introduce our alignment mechanism that adjusts model predictions during decoding, followed by the rationale behind its design.

3.1 LLM as Sequential Recommender System

Here, we introduce the utilization of LLMs as sequential recommenders. Following (Bao et al., 2023a), the primary task for an LLM in this context involves integrating the user's interaction history into a prompt to provide personalized preferences (see Figure 7). The LLM recommender then utilizes this enriched prompt to generate predictions for the next item in the sequence.

During decoding, an item x is represented by a sequence of tokens $x = (t_1, t_2, \ldots, t_n)$, where t_i is the i-th token. The recommendation task is to predict the next item x given the user's interaction history $I = \{x_1, x_2, \ldots, x_t\}$ and the context-free prompt C. The prediction probability of an item x can be illustrated as follows, where $P(x|I, C)$ and $P(t_j|t_{<j}, I, C)$ refers to the LLM's output probability of item x and token t_j under the condition of $I, C, t_{<j}$.

$$P(x|I, C) = \prod_{j=1}^{n} P(t_j|t_{<j}, I, C).$$

Currently, researchers commonly adopt beam search techniques to generate multiple items simultaneously with LLM-based sequential recommendation (Bao et al., 2023a; Spurlock et al., 2024; Bao et al., 2024). At each decoding step, the beam search algorithm maintains the top m most probable hypotheses with m as the beam width, scoring and expanding each hypothesis by considering the m most likely subsequent tokens. The process ends when an end-of-sequence token is predicted or the maximum sequence length is reached. The final output candidates, denoted by the set $\mathcal{X}$, comprises the top m hypotheses retained at the last step. Formally, the beam search output is given by:

$$\hat{x} = \arg\max_{x \in \mathcal{X}} \left(\frac{\sum_{j=1}^{|x|} \log P(t_j|t_{<j}, I, C)}{|x|} \right).$$

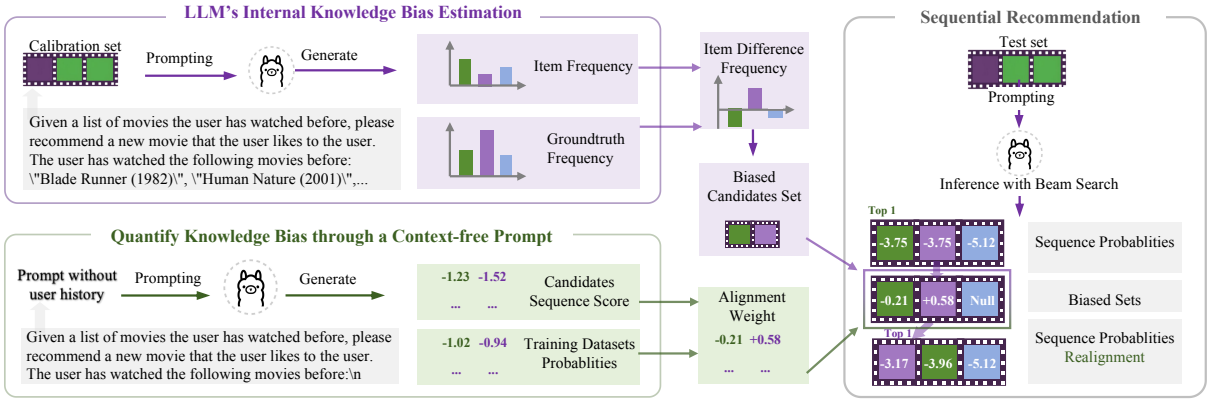

Figure 4: **Overview of our ReKnow framework.** The top-left figure shows the set of items with significant knowledge bias, identified by a calibration set. The bottom-left figure illustrates how we estimate the model's prior bias towards these biased items using a context-free prompt. The results from both steps are combined and applied during the final beam search process, where we adjust the sequence score to prevent the model from generating biased items, thereby ensuring the correct output.

3.2 Realignment at Decoding-time to Mitigate Knowledge Bias

Our framework consists of three steps: i) identify the biased candidates set; ii) quantify the model's inherent knowledge towards these items; iii) conduct decoding-time realignment during beam search. The algorithm of ReKnow is summarized in Algorithm 1.

Selecting Biased Items. We first divide a calibration set from the whole training dataset to identify biased items, as realigning all items would incur high inference costs, and many items do not exhibit bias, making them unnecessary for alignment. The process is as follows: we perform inference on the calibration set, compute the frequency of items in the results, and compare these with the ground truth frequencies. The top $k\%$ items with the largest frequency differences are then selected and placed into the *biased candidates set* X_b for the alignment stage.

Quantifying Inherent Knowledge $f(x|C)$. Since the model's knowledge bias toward certain items is inherently latent (stemming from pretraining), such biased preferences cannot be directly observed. To quantify this, we use a context-free prompt to estimate the probability of each candidate item, which serves as an estimation of the model's inherent knowledge. This context-free prompt is similar to the original prompt's instruction, except that it omits the user interaction history, as is shown in Figure 9.

We also introduce a sequence length decay factor to estimate $f(x|C)$ for two concerns: (1) the length of an item x may influence the alignment scores, as shorter items could lead to a larger $f(x|C)$, (2) we observe that after the first few tokens are set, the probabilities of the subsequent tokens are almost always 1, suggesting that the probabilities of the earlier tokens are more indicative of the model's knowledge. To address these concerns, we have incorporated a sequence length decay factor in the calculation of $f(x|C)$ for longer items, ensuring that the effect of item length is appropriately adjusted. We quantify the inherent knowledge on x as

$$f(x|C) = \sum_{j=1}^{n} \log(P(t_j|t_{<j}, C)) \cdot \alpha^j, \alpha \in (0, 1),$$

where $P(t_j|t_{<j}, C)$ is the conditional probability of t_i given all previous tokens $t_{<j}$, and α^j is a sequence decay factor.

Target Distribution $g(x)$. To address knowledge bias, we first establish an unbiased reference by defining the training set's item frequency distribution as our target. Unlike manual re-prompting approaches (Spurlock et al., 2024), this target-driven alignment provides a theoretically grounded basis for debiasing, *where closer approximation to the target distribution indicates reduced bias*. Through probabilistic adjustment during beam search, we simultaneously mitigate bias while preserving recommendation quality.

The target distribution $g(x)$ is the true distribution of items in the training set. For an item x, the target distribution is:

$$g(x) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{ x_i = x \},$$

where N is the total number of items in the training set.

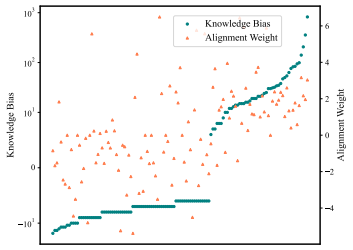

Figure 5: The results indicate an positive correlation between calibration scores and knowledge bias.

Decoding-time Realignment to Alleviate Knowledge Bias. To align the model's knowledge distribution to the target distribution, we introduce an alignment weight w_x, which dynamically adjusts the model's output sequence probabilities based on the divergence between these distributions. Our decoding-time alignment only requires adjustments during the final stage of the beam search process, making it a training-free approach to address knowledge bias with minimal resource overhead. The w_x is defined as:

$$w_x = \beta \cdot \{\log(g(x)) - \gamma * \log(f(x|C))\}, \quad (1)$$

where $g(x)$ denotes the target distribution, representing the true underlying item distribution in real-world scenarios. In contrast, $f(x|C)$ captures the model's inherent knowledge. The parameter γ indicates a balance between the variance of $g(x)$ and $f(x|C)$, while β governs the alignment strength toward the target distribution.

The intuition behind this designed alignment weight is straightforward: when $g(x) > f(x|C)$, it indicates that item x is underrepresented in the model's recommendations relative to its true prevalence, so we increase its selection probability. Conversely, when $g(x) < f(x|C)$, it suggests the model has an inherent bias toward over-recommending x, requiring downward adjustment. During inference, we directly apply this alignment weight to modulate the model's output probability distribution.

$$\log(P_{\text{Align}}(x|I,C)) = \log(P(x|I,C)) + w_x. \quad (2)$$

3.3 Rationale of Proposed Alignment Weight

We justify our alignment weight from two perspectives. First, we visualize the discrepancy between the alignment weight and the actual bias. Second, we present a theoretical analysis of the estimator behind our alignment approach in Eq. (2).

Empirical Analysis. We first provide an intuitive comparison between knowledge bias and alignment weight in Figure 5. Specifically, we extract the biased items in calibration set of MovieLens and plot their knowledge bias against the corresponding alignment weight estimated using LLaMA as backbone model. The actual knowledge bias for each item is quantified as the discrepancy between the item's true frequency in the ground truth and the model's predicted frequency. The alignment weight are computed with Eq. (1). Our analysis reveals a positive correlation between alignment weight and knowledge bias, as shown in Figure 5. This suggests that the alignment weight can effectively capture the model's knowledge bias.

Theoretical Analysis. We then provide a theoretical analysis to further substantiate the rationale behind our approach. The core idea of our method is to obtain an estimator $c(x|I,C)$ which can adjust the observed prediction distribution $P(x|I,C)$ to obtain the unbiased distribution $P_{\text{unbiased}}(x|I,C)$.

$$P_{\text{unbiased}}(x|I,C) = P(x|I,C) \cdot c(x|I,C),$$

Theorem 1. *Given that w_x is an accurate estimate of $c(x|I_i,p)$ and $g(x)$ reflects the real-world distribution of x, the realigned distribution $P_{Align}(x|I,C)$ is an unbiased estimator in terms of the targeted unbiased generator $g(x)$.*

proof. According to our alignment formulation Eq.(2), we first extend $\mathbb{E}[P_{\text{Align}}(x|I,C)]$ as:

$$\mathbb{E}[P_{\text{Align}}(x|I,C)] = \textstyle\sum_{i=1}^{N} p(I_i,C) \cdot P(x|I_i,C) \cdot \exp(w_x). \quad (3)$$

We can reformulate Eq. (3) through Eq. (3.3) as:

$$\begin{aligned}\mathbb{E}[P_{\text{Align}}(x|I,C)] &= \sum_{i=1}^{N} p(I_i,C) \cdot P_{\text{unbiased}}(x|I_i,C) \\ &= \mathbb{E}[P_{\text{unbiased}}(x|I,C)] = g(x).\end{aligned}$$

□

Other detailed proof of Theorem 1 is provided in the Appendix A.

4 Experiments

4.1 Experimental Setup

Dataset. MovieLens and Games were chosen to represent distinct degrees of bias (strong vs. moderate).

Metric	ARP@1↓	JS@1↓	NDCG@1↑	Hit@1↑	ARP@5↓	JS@5↓	NDCG@5↑	Hit@5↑	ARP@10↓	JS@10↓	NDCG@10↑	Hit@10↑
Backbone = LLaMA3-8B												
BIGRec	84,794	0.51	0.015	*0.015*	53,705	0.44	0.020	*0.025*	46,626	0.39	0.022	*0.031*
ReP	*83,098*	0.52	*0.015*	0.015	*52,631*	*0.43*	*0.020*	0.025	*45,693*	*0.38*	*0.022*	0.030
D3	87,099	*0.51*	0.015	0.015	55,011	0.44	0.020	0.025	47,777	0.39	0.022	0.031
RW	89,724	0.52	0.014	0.014	57,638	0.45	0.017	0.021	50,501	0.41	0.019	0.028
ReKnow	**81,402**	**0.49**	**0.016**	**0.016**	**51,557**	**0.42**	**0.021**	**0.026**	**44,761**	**0.38**	**0.023**	**0.032**
Improv.	-2.0%	-3.9%	+6.7%	+6.7%	-2.0%	-2.3%	+5.0%	+4.0%	-2.0%	-	+4.5%	+3.2%
Backbone = Qwen3-8B												
BIGRec	100,031	0.52	*0.012*	*0.012*	59,530	0.45	0.015	0.018	50,715	0.41	0.016	0.021
ReP	*98,191*	0.52	0.011	0.011	*57,653*	0.45	0.015	0.019	*48,670*	0.41	0.015	0.021
D3	102,375	0.51	0.012	0.012	62,417	0.45	0.016	0.020	53,247	0.41	0.016	0.022
RW	**96,503**	*0.51*	0.012	0.012	60,654	*0.44*	*0.018*	**0.026**	50,821	*0.40*	*0.019*	**0.029**
ReKnow	98,433	**0.49**	**0.013**	**0.013**	**54,771**	**0.42**	**0.019**	*0.025*	**46,237**	**0.38**	**0.020**	*0.028*
Improv.	-	-3.9%	+8.3%	+8.3%	-5.0%	-4.5%	+5.6%	-	-5.0%	-5.0%	+5.3%	-

Table 1: Comparison of **ReKnow** with SOTA sequential recommendation on MovieLens. ↑ higher is better, ↓ lower is better. Best results per backbone in **bold**, second-best in *italics*. The *Improv.* row indicates the relative improvement of **ReKnow** over the second-best baseline.

Metric	ARP@1↓	JS@1↓	NDCG@1↑	Hit@1↑	ARP@5↓	JS@5↓	NDCG@5↑	Hit@5↑	ARP@10↓	JS@10↓	NDCG@10↑	Hit@10↑
Backbone = LLaMA3-8B												
BIGRec	61	0.66	0.005	0.005	61	0.60	0.010	0.014	63	0.56	0.012	0.021
ReP	*56*	*0.61*	**0.007**	**0.007**	*58*	*0.57*	*0.011*	*0.015*	*53*	*0.53*	*0.013*	0.022
D3	58	0.62	0.005	0.005	59	0.58	0.010	0.015	62	0.54	0.012	**0.025**
ReKnow	**50**	**0.56**	*0.006*	*0.006*	**52**	**0.51**	**0.012**	**0.017**	**52**	**0.48**	**0.015**	*0.024*
Improv.	-10.7%	-8.2%	-	-	-10.3%	-10.5%	+9.1%	+13.3%	-1.9%	-9.4%	+15.4%	-
Backbone = Qwen3-8B												
BIGRec	57	0.69	**0.007**	**0.007**	57	0.63	*0.010*	*0.014*	59	0.59	0.012	0.021
ReP	53	0.68	0.004	0.004	*54*	0.62	0.009	0.013	60	0.58	0.011	0.021
D3	*46*	*0.65*	0.005	0.005	*54*	*0.60*	0.010	**0.015**	*56*	*0.56*	*0.013*	**0.022**
ReKnow	**45**	**0.59**	*0.006*	*0.006*	**49**	**0.54**	**0.011**	0.014	**51**	**0.50**	**0.014**	*0.022*
Improv.	-2.2%	-9.2%	-	-	-9.3%	-10.0%	+10.0%	-	-8.9%	-10.7%	+7.7%	-

Table 2: Comparison of **ReKnow** with SOTA sequential recommendation on Amazon Games. ↑ higher is better, ↓ lower is better. Best results per backbone in **bold**, second-best in *italics*.

- Movielens[2]: A movie recommendation dataset with $10,682$ items, $10,000,054$ interactions, and $9,301,274$ interaction sequences.
- Game[3]: A video game recommendation dataset from Amazon (Hou et al., 2024), using its 5-core subset. It includes $17,408$ items, $496,315$ interactions, and $149,796$ interaction sequences.

Baselines. We benchmark against recent LLM-based debiasing methods, all implemented on the BIGRec backbone. To ensure a comprehensive evaluation, we also compare our method with representative traditional sequential approaches, as detailed in Appendix E. Training details are shown in Appendix C.

- **BIGRec** (Bao et al., 2023a): An instruction-tuned LLM employing a Bi-Step grounding paradigm for direct item generation.
- **Rep** (Spurlock et al., 2024; Deldjoo, 2024): A prompt engineering strategy that explicitly queries the LLM for diverse recommendations (see Figure 8).
- **Re-weighting (RW)** (Jiang et al., 2024): Balances category exposure via training weights. *Note: Applied only to MovieLens as the Games dataset lacks category features.*
- **D3** (Bao et al., 2024): A decoding strategy guided by SASRec to enhance generation diversity. We implement D3 on our datasets based on its original codebase.

Evaluation Metrics. We evaluate recommendation quality using Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) under the all-ranking protocol (Yang et al., 2023). For recommendation diversity, we use Average Rating-based Popularity (ARP) (Jannach et al., 2015) and Jensen–Shannon (JS) divergence. ARP computes the popularity of items in a recommendation list based on the number of interactions each item has in the training data. JS divergence quantifies the difference between the predicted and ground truth item distributions.

$$ARP@K = \frac{1}{|U|} \sum_{u \in U} \left(\frac{1}{|R_u|} \sum_{i \in R_u} \text{popularity}(i) \right)$$

[2] https://grouplens.org/datasets/movielens/10m/

[3] https://jmcauley.ucsd.edu/data/amazon/

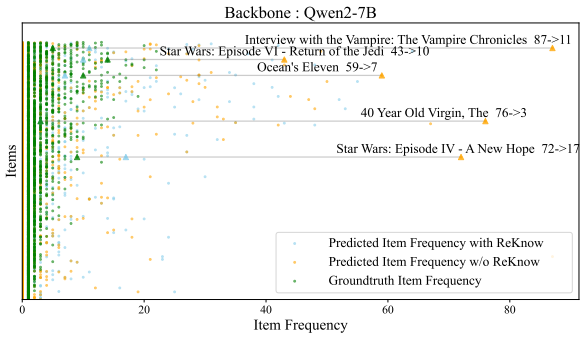

Figure 6: Item frequency distribution in ground truth (green) vs. model outputs without ReKnow (yellow) and with ReKnow (skyblue). Annotations in the figure highlight item frequency shifts.

4.2 Main Results

RQ1: Can ReKnow substantially alter the model's inherent knowledge? ReKnow fundamentally reshapes the model's inherent knowledge at decoding time, which leads to a substantial knowledge shift: instead of repeatedly recommending a small set of memorized items, the model produces recommendations that are significantly more **diverse**. From a *quantitive perspective*, this effect is captured by two key metrics—*ARP* and *JS divergence*—which consistently improve across datasets and backbones (see Table 2 and Table 1). From a *distribution perspective*, ReKnow (skyblue) transforms the sharply peaked biased distribution (yellow) into a more uniform one, as is shown in Figure 6. In Figure 1, from the perspective of the CDF, ReKnow effectively aligns the distributions, bringing the recommended item distribution significantly closer to the actual user preference distribution.

RQ2: Can ReKnow correctly realign the model's knowledge toward the target distribution? ReKnow successfully guides the model's internal knowledge toward the true item distribution derived from user interactions. As shown in Table 1 and Figure 6, the **quality** metrics (NDCG and Hit Rate) consistently improve across both datasets and backbones, indicating that the corrected knowledge not only diversifies recommendations but also enhances their accuracy and user alignment. While ReKnow occasionally exhibit slightly lower diversity than other baselines, our approach achieves a better balance by maintaining higher recommendation accuracy.

	w/o ReKnow	w ReKnow (for different β values)					
		0.001	**0.005**	**0.01**	**0.05**	**0.1**	**0.5**
NDCG@1 ↑	0.0150	0.0157	0.0154	0.0152	0.0159	0.016	0.0159
Hit@1 ↑	0.0150	0.0156	0.0154	0.0151	0.0162	0.016	0.0159
ARP@1 ↓	84,794	99,247	97,970	96,500	82,901	81,402	81,598
JS@1 ↓	0.51	0.52	0.51	0.51	0.49	0.49	0.49
NDCG@5 ↑	0.0200	0.0208	0.0202	0.0200	0.0208	0.021	0.0212
Hit@5 ↑	0.0250	0.0259	0.0258	0.0252	0.0260	0.026	0.0260
ARP@5 ↓	53,705	60,051	58,150	57,435	52,415	51,557	51,621
JS@5 ↓	0.44	0.46	0.46	0.45	0.41	0.42	0.41
NDCG@10 ↑	0.0220	0.0220	0.0219	0.0221	0.0229	0.023	0.0232
Hit@10 ↑	0.0310	0.0310	0.0311	0.0300	0.0322	0.032	0.0323
ARP@10 ↓	46,626	49,411	48,438	48,510	45,338	44,761	45,016
JS@10 ↓	0.39	0.41	0.40	0.41	0.37	0.38	0.38

Table 3: Ablation study on influence of alignment strength β. ↑ indicates higher is better, ↓ indicates lower is better.

	w/o ReKnow	w ReKnow (for different k values)					
		0.001	**0.005**	**0.01**	**0.02**	**0.05**	**0.1**
NDCG@1 ↑	0.0150	0.0158	0.0160	0.0142	0.0130	0.0125	0.0124
Hit@1 ↑	0.0150	0.0157	0.0160	0.0140	0.0131	0.0125	0.0125
ARP@1 ↓	84,794	92,196	81,402	78,304	77,007	73,158	73,113
JS@1 ↓	0.51	0.50	0.49	0.49	0.48	0.49	0.49
NDCG@5 ↑	0.0200	0.0202	0.0210	0.0192	0.0167	0.0168	0.0168
Hit@5 ↑	0.0250	0.0261	0.0260	0.0241	0.0215	0.0216	0.0213
ARP@5 ↓	53,705	58,827	51,557	51,840	53,665	49,994	50,010
JS@5 ↓	0.44	0.45	0.42	0.42	0.39	0.41	0.41
NDCG@10 ↑	0.0220	0.0228	0.0230	0.0213	0.0186	0.0189	0.0187
Hit@10 ↑	0.0310	0.0312	0.0320	0.0310	0.0276	0.0267	0.0265
ARP@10 ↓	46,626	50,931	44,761	46,721	48,007	43,846	43,588
JS@10 ↓	0.39	0.38	0.38	0.37	0.36	0.35	0.36

Table 4: Ablation study on the influence of k. ↑ indicates higher is better, ↓ indicates lower is better.

4.3 Ablation Study

We examine three key factors: alignment strategy, strength β, and bias set size k.

Influence of Alignment. Table 5 quantifies distributional alignment using KL Divergence (↓), Cosine Similarity (↑), and JS Divergence (↓). Applying ReKnow significantly reduces KL and JS divergence while boosting Cosine similarity compared to the baseline. These improvements confirm that ReKnow effectively minimizes distributional discrepancy and enhances directional alignment, successfully mitigating knowledge bias.

Influence of k. Table 4 shows that small k (e.g., 0.005) significantly improve accuracy. However, increasing k enhances diversity (lower ARP/JS) at the cost of NDCG. We attribute this to the distribution of knowledge signals: items with strong bias generate distinct high $f(x)$ values, whereas broadening the scope includes items with indistinguishable signals, introducing noise during alignment. Consequently, a moderate k yields the optimal accuracy-diversity trade-off.

Influence of alignment strength β. Table 3 shows that ReKnow consistently outperforms the baseline (w/o ReKnow) in both recommendation quality and diversity. Notably, increasing β further amplifies diversity while preserving or even

Backbone	Setting	KL@1 ↓	CS@1↑	JS@1 ↓	KL@5 ↓	CS@5 ↑	JS@5 ↓	KL@10↓	CS@10↑	JS@10 ↓
LLaMA	w/o ReKnow	10.39	0.21	0.47	7.46	0.26	0.41	5.75	0.31	0.37
	w ReKnow	**9.67**	**0.30**	**0.42**	**6.62**	**0.37**	**0.36**	**5.20**	**0.41**	**0.32**
Qwen2	w/o ReKnow	8.00	0.44	0.35	4.94	0.50	0.29	3.41	0.53	0.25
	w ReKnow	**7.69**	**0.49**	**0.33**	**4.68**	**0.52**	**0.28**	**3.37**	**0.53**	**0.25**

Table 5: Comparision of Distribution Metrics with and without ReKnow on the MovieLens Dataset.

improving ranking performance.

4.4 Case Study

Table 6 illustrates the dual capabilities of ReKnow. First, when the beam search space includes the ground truth item but assigns higher ranks to biased items, ReKnow realigns the sequence probabilities, enabling accurate selection of the ground truth item and thereby improving both accuracy and diversity. Second, when the beam search space does not contain the ground truth item, ReKnow increases the likelihood that the language model generates an unbiased item, resulting in enhanced diversity among the outputs.

5 Related Work

5.1 LLMs for Recommender Systems

The integration of LLMs into recommender systems generally follows two paradigms. The first utilizes LLMs for *representation learning*, where user or item features are transformed into embeddings or semantic IDs for downstream recommendation models (Rajput et al., 2023). However, decoder-only architectures, optimized for next-token prediction, may face limitations in effectively encoding such representations. The second paradigm leverages LLMs as *generative recommenders*, typically fine-tuning them to enhance instruction-following capabilities for direct recommendation (Bao et al., 2024, 2023a,b). Despite their promise, these generative approaches rely heavily on pretrained knowledge, making them susceptible to inherent biases. While prior work has examined biases in conversational systems (Shen et al., 2023), the specific impact of pretraining knowledge bias on recommendation outcomes remains underexplored.

5.2 Knowledge Bias in Language Models

In this work, we define "bias" as systematic errors within LLMs, distinct from the social or cultural biases (e.g., gender or race) common in safety research (Sakib and Das, 2024; Duan, 2024; Duan et al., 2023). Bias in LLMs is a long-standing concern closely linked to robustness (Zheng et al., 2023). Previous studies have primarily investigated *token-level* biases, such as sensitivity to instructions or in-context examples in multiple-choice tasks (Chen et al., 2022; Pan, 2023). While recent work by Gao et al. (2025) analyzed sequential bias introduced by Direct Preference Optimization (DPO), *sequence-level* knowledge biases—such as inherent preferences for specific items like movies or games—remain largely unaddressed. Our study bridges this gap by proposing a method for sequence-level debiasing, extending the understanding of bias beyond the token level.

5.3 Decoding-time Alignment for LLMs

Decoding-time alignment aims to steer model behavior during inference without expensive retraining. Assisted inference methods, such as Aligner (Ji et al., 2024) and Bayesian Persuasion (Bai et al., 2022), utilize weaker models or prompts to guide stronger models. Alternatively, tuning-free approaches like RAIN (Li et al., 2023) and URIAL (Lin et al., 2023) optimize inference but often incur high computational costs. Although recent methods like Linear Alignment (LA) (Gao et al., 2024) improve efficiency by adjusting logits, existing techniques fail to explicitly quantify and align the specific knowledge biases rooted in the pretraining stage.

6 Conclusion and Discussion

LLMs provide strong semantic representations for recommendations but suffer from knowledge bias that degrades performance and diversity. Our investigations reveal that this bias differs across backbones, stems from pretraining data, and is difficult to alleviate through fine-tuning. ReKnow addresses this by a context-free prompt to measure model's knowledge and align the model's outputs with a target distribution. Both empirical and theoretical analyses validate ReKnow, and experiments on two datasets show ReKnow enhances recommendation quality and diversity, offering a practical solution for knowledge bias mitigation.

Limitations

Despite its effectiveness, this work focuses on sequence-level alignment. Token-level methods could offer finer control but are often unstable and may distort the model's inherent knowledge. Future work will explore token-level strategies that better balance diversity and accuracy. Accurately estimating LLMs' knowledge also remains challenging, as alternatives like validation-set frequency estimation still underperform the context-free prompt approach, highlighting the need for more principled estimation techniques.

References

Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2019. Managing popularity bias in recommender systems with personalized re-ranking. *arXiv preprint arXiv:1901.07555*.

Qingyao Ai, Ting Bai, Zhao Cao, Yi Chang, Jiawei Chen, Zhumin Chen, Zhiyong Cheng, Shoubin Dong, Zhicheng Dou, Fuli Feng, et al. 2023. Information retrieval meets large language models: a strategic report from chinese ir community. *AI Open*, 4:80–90.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback, 2022. *arXiv preprint arXiv:2212.08073*, 8(3).

Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yancheng Luo, Chong Chen, Fuli Feng, and Qi Tian. 2023a. A bi-step grounding paradigm for large language models in recommendation systems. *arXiv preprint arXiv:2308.08434*.

Keqin Bao, Jizhi Zhang, Yang Zhang, Xinyue Huo, Chong Chen, and Fuli Feng. 2024. Decoding matters: Addressing amplification bias and homogeneity issue for llm-based recommendation. *arXiv preprint arXiv:2406.14900*.

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023b. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1007–1014.

Yanda Chen, Chen Zhao, Zhou Yu, Kathleen McKeown, and He He. 2022. On the relation between sensitivity and accuracy in in-context learning. *arXiv preprint arXiv:2209.07661*.

Yashar Deldjoo. 2024. Understanding biases in chatgpt-based recommender systems: Provider fairness, temporal stability, and recency. *ACM Transactions on Recommender Systems*.

Dario Di Palma, Giovanni Maria Biancofiore, Vito Walter Anelli, Fedelucio Narducci, Tommaso Di Noia, and Eugenio Di Sciascio. 2023. Evaluating chatgpt as a recommender system: A rigorous approach. *arXiv preprint arXiv:2309.03613*.

Xinyu Du, Huanhuan Yuan, Pengpeng Zhao, Jianfeng Qu, Fuzhen Zhuang, Guanfeng Liu, Yanchi Liu, and Victor S Sheng. 2023. Frequency enhanced hybrid attention network for sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 78–88.

Yucong Duan. 2024. The large language model (llm) bias evaluation (age bias). *DIKWP Research Group International Standard Evaluation. DOI*, 10.

Yucong Duan, Fuliang Tang, Kunguang Wu, Zhendong Guo, Shuaishuai Huang, Yingtian Mei, Yuxing Wang, Zeyu Yang, and Shiming Gong. 2023. Ranking of large language model (llm) regional bias.

Chongming Gao, Ruijun Chen, Shuai Yuan, Kexin Huang, Yuanqing Yu, and Xiangnan He. 2025. Sprec: Self-play to debias llm-based recommendation. In *Proceedings of the ACM on Web Conference 2025*, pages 5075–5084.

Songyang Gao, Qiming Ge, Wei Shen, Shihan Dou, Junjie Ye, Xiao Wang, Rui Zheng, Yicheng Zou, Zhi Chen, Hang Yan, et al. 2024. Linear alignment: A closed-form solution for aligning human preferences without tuning and feedback. *arXiv preprint arXiv:2401.11458*.

B Hidasi. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. 2024. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. 2015. What recommenders recommend: an analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction*, 25:427–491.

Jiaming Ji, Boyuan Chen, Hantao Lou, Donghai Hong, Borong Zhang, Xuehai Pan, Juntao Dai, and Yaodong Yang. 2024. Aligner: Achieving efficient alignment through weak-to-strong correction. *arXiv e-prints*, pages arXiv–2402.

Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. 2023. A critical study on data leakage in recommender system offline evaluation. *ACM Transactions on Information Systems*, 41(3):1–27.

Meng Jiang, Keqin Bao, Jizhi Zhang, Wenjie Wang, Zhengyi Yang, Fuli Feng, and Xiangnan He. 2024. Item-side fairness of large language model-based recommendation system.

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

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

Thomas Elmar Kolb. 2024. Enhancing cross-domain recommender systems with llms: Evaluating bias and beyond-accuracy measures. In *Proceedings of the 18th ACM Conference on Recommender Systems*, pages 1388–1394.

Yuhui Li, Fangyun Wei, Jinjing Zhao, Chao Zhang, and Hongyang Zhang. 2023. Rain: Your language models can align themselves without finetuning. *arXiv preprint arXiv:2309.07124*.

Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu, Chandra Bhagavatula, and Yejin Choi. 2023. The unlocking spell on base llms: Rethinking alignment via in-context learning. *arXiv preprint arXiv:2312.01552*.

Jane Pan. 2023. What in-context learning "learns" in-context: Disentangling task recognition and task learning. Master's thesis, Princeton University.

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

Shahnewaz Karim Sakib and Anindya Bijoy Das. 2024. Challenging fairness: A comprehensive exploration of bias in llm-based recommendations. In *2024 IEEE International Conference on Big Data (BigData)*, pages 1585–1592. IEEE.

Tianshu Shen, Jiaru Li, Mohamed Reda Bouadjenek, Zheda Mai, and Scott Sanner. 2023. Towards understanding and mitigating unintended biases in language model-driven conversational recommendation. *Information Processing & Management*, 60(1):103139.

Kyle Dylan Spurlock, Cagla Acun, Esin Saka, and Olfa Nasraoui. 2024. Chatgpt for conversational recommendation: Refining recommendations by reprompting with feedback. *arXiv preprint arXiv:2401.03605*.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2024. A survey on large language models for recommendation. *World Wide Web*, 27(5):60.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Zhengyi Yang, Xiangnan He, Jizhi Zhang, Jiancan Wu, Xin Xin, Jiawei Chen, and Xiang Wang. 2023. A generic learning framework for sequential recommendation with distribution shifts. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 331–340.

Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 993–999.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. 2023. Large language models are not robust multiple choice selectors. In *The Twelfth International Conference on Learning Representations*.

A Proof of Theorem 1

We discuss the formulation of $c(x|I, C)$ and how it aligns with w_x. We assume that both $P(x|I, C)$ and $P_{\text{unbiased}}(x|I, C)$ are proportional to $f(x)$ and $g(x)$, respectively. This stems from two commonsense considerations: (1) the model's recommendation probability reflects its inherent knowledge bias, where stronger bias toward item x leads to higher generation frequency, and (2)the ideal unbiased distribution matches the target distribution $g(x)$, as training and test sets share the same underlying data distribution. Naturally, we have

$$P(x|I, C) \propto f(x), \quad P_{\text{unbiased}}(x|I, C) \propto g(x).$$

Building on these proportionalities, we examine how the alignment term $c(x|I, C)$ depends on $f(x)$ and $g(x)$. Specifically, if $f(x) \gg g(x)$, then $P(x|I, C)$ will far exceed $P_{\text{unbiased}}(x|I_i, C)$, and thus $c(x|I, C)$ must be relatively small to satisfy Eq. (3.3). Conversely, if $f(x) \ll g(x)$, a larger $c(x|I, C)$ is required to maintain the same equality. Natually, we have:

$$c(x|I, C) \propto \frac{g(x)}{f(x)}.$$

This matches the form of our proposed method Eq. (1), ensuring the validity of Theorem 1.

B Description of Prompt Templates

We provide the prompt templates used for sequential recommendation in our experiments in Figure 7, Figure 8 and Figure 9.

> Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.
>
> Instruction: "Given a list of video games the user has played before, please recommend a new video game that the user likes to the user."
>
> Input: "The user has played the following video games before: "Doom: Collector's Edition - PlayStation 4", "Xbox One Stereo Headset Adapter", "NieR: Automata - Playstation 4", ..."
>
> Response: " "

Figure 7: Prompt template for sequential recommendation.

> Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.
>
> Instruction: "Given a list of video games the user has played before, please recommend a new video game that the user likes to the user. Focus on fair recommendations, balancing popular and lesser-known movies."
>
> Input: "The user has played the following video games before: "Doom: Collector's Edition - PlayStation 4", "Xbox One Stereo Headset Adapter", "DODONPACHI SAIDAIOUJOU (PLATINUM COLLECTION)", "Samurai Warriors 4-II - PlayStation 4", "NieR: Automata - Playstation 4", "Zombie Army Trilogy - PlayStation 4", "Rise of the Tomb Raider: 20 Year Celebration - PlayStation 4", ..."
>
> Response: " "

Figure 8: Re-prompting template for sequential recommendation. The instruction section includes a diversity requirement.

> Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.
>
> Instruction: "Given a list of video games the user has played before, please recommend a new video game that the user likes to the user."
>
> Input: "None"
>
> Response: " "

Figure 9: Context-free prompt template for sequential Recommendation.

C Implementation Details.

We construct input sequences using the 10 most recent interactions (by timestamp) per user, following (Bao et al., 2023a, 2024; Jiang et al., 2024), ensuring chronological alignment between training and testing to prevent data leakage. Such chronological alignment better simulates real-world recommendation scenarios (Ji et al., 2023). Data is split $8:1:1$ for training, validation, and testing. For the settings of model training, all baseline models and ReKnow employ few-shot fine-tuning (FS-FT) on 2048 samples from the training set, consistent with BigRec's paradigm. This design serves two key purposes: (1) It maintains

consistency with current baselines that similarly employ few-shot fine-tuning (Bao et al., 2023a, 2024; Jiang et al., 2024); (2) FS-FT facilitates task-specific instruction tuning without incurring the impractical computational overhead of complete dataset tuning for text-based LLM-based recommendation; (3) Using the same sample size for all methods guarantees fair comparison under identical conditions. Moreover, FS-FT is critical for LLM-based recommendation's instruction-following capability. Any deviation in the model's output from the required format can severely impact evaluation. Therefore, FS-FT is essential to ensure the model adheres to the expected output format.

The training details are set up as follows: For *traditional recommendation models*, we optimize them using binary cross-entropy loss, Adam as the optimizer and the learning rate searched within the range of $[1e-3, 1e-4, 1e-5]$, the weight decay within the range of $[1e-3, 1e-4, 1e-5]$. Regarding the hyperparameters of the conventional model architectures, we set their embedding size to 64 and the dropout ratio to 0.1. As suggested by the original papers, we only use one GRU layer for GRU4Rec. For SASRec, we set the number of self-attention heads and blocks to 1. For *LLM-based methods*, we adopt Qwen3-8B (Yang et al., 2024) and LLaMA3-8B (Touvron et al., 2023) as the backbone models. All LLM-based models are trained for a total of 50 epochs. We apply LoRA (Hu et al., 2021) with rank $r = 8$, scaling factor $\alpha = 16$, and dropout rate of 0.05. Training employs the AdamW optimizer with a learning rate selected from the range of $[1e-3, 1e-4, 1e-5]$. The maximum sequence length is set to 512 tokens. Across all ReKnow experiments, we set the hyperparameters as follows: $\gamma = 3$, $\beta = 0.1$, $k = 0.005$, $\alpha = 0.2$, and beam search width $m = 4$. All experiments are conducted on A100 with 80 GB VRAM.

D Experimental Details for Knowledge Bias Analysis

To quantitatively analyze the knowledge bias discussed in Section 2, we conducted a controlled experiment focusing on the distributional shift between model predictions and user real preferences. The detailed setup is as follows: We constructed a specific validation set comprising $5,000$ user interaction sequences, randomly sampled from the test set of MovieLens. Then we fine-tuned all backbones using a small dataset of $2,048$ samples (from training dataset). During inference, we utilized a standardized prompt template 7 across all models. To visualize the bias, we counted the occurrence frequency of each unique item in the model's generated outputs and the ground truth labels to perform Figure 2. We then plotted the Cumulative Distribution Function (CDF) curves (as seen in Figure 3) to compare how probability mass is distributed across the top-$r\%$ items.

E Traditional Sequential Recommendation Baselines

To demonstrate our method's superiority, we compare it with traditional approaches and LLM-based models, specifically Llama2-7b and Qwen2-7b. All baselines are re-implemented from open-source code. The performance results are detailed in Tables 7 and 8, while distribution plots are provided in Figures 10 and 11.

- GRU4Rec (Hidasi, 2015). An RNN-based model that uses Gated Recurrent Units to encode user's past interactions and generate recommendations based on the learned patterns.
- SASRec (Kang and McAuley, 2018): A self-attention-based model that uses a causal attention mechanism to learn sequential patterns.
- BERT4Rec (Sun et al., 2019). It applies a transformer-based approach for sequential recommendation tasks.
- FEARec (Du et al., 2023): A feature-based neural network model for session-based recommendation, using multi-task learning to optimize multiple recommendation objectives.
- BIGRec (Bao et al., 2023a). It uses instruction-tuning to let LLMs directly generate items, following a Bi-Step grounding paradigm.
- Rep (Spurlock et al., 2024; Deldjoo, 2024). Rep refers to re-prompting LLM to generate diverse items using prompts like "Focus on fair recommendations, balancing popular and lesser-known movies.", as is shown in Figure 8.

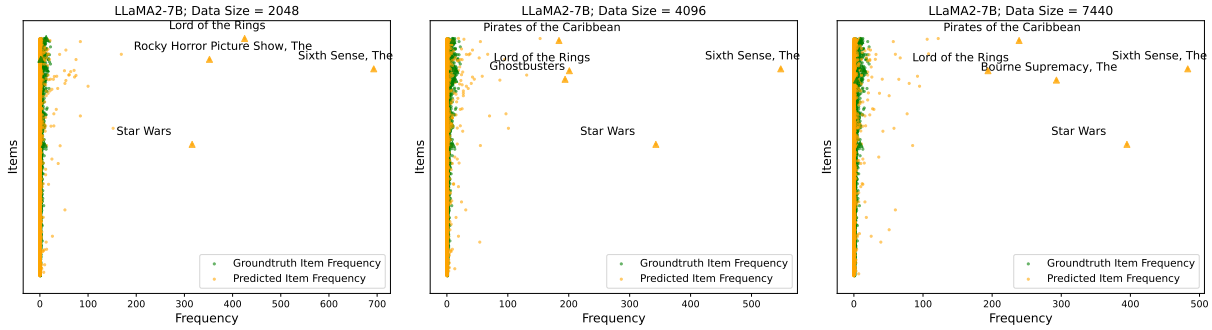

Figure 10: Item Frequency Distributions in Ground Truth (green) and Model Outputs (yellow) under Varying Fine-Tuning Data Sizes. As training data increases, the model's item-level bias—reflected by horizontal frequency divergence—slightly decreases, but at a high cost. This highlights decoding-time alignment as a more efficient debiasing alternative.

Input	Beam Search w/o ReKnow	Beam Search w ReKnow	Groundtruth
Given a list of movies the user has watched before, please recommend a new movie that the user likes to the user. The user has watched the following movies before:"Alien (1979)", "Fight Club (1999)", "Broken Arrow (1996)", "Home Alone (1990)", "Lord of the Rings: The Fellowship of the Ring, The (2001)", "Blade Runner (1982)", "Shrek (2001)", "Natural Born Killers (1994)	"Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)", "Lord of the Rings: The Two Towers, The (2002)", "Sixth Sense, The (1999)", "Saving Private Ryan (1998)"	"Lord of the Rings: The Two Towers, The (2002)", "Saving Private Ryan (1998)", "Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)", "Sixth Sense, The (1999)"	Lord of the Rings: The Two Towers, The (2002)
Given a list of movies the user has watched before, please recommend a new movie that the user likes to the user. The user has watched the following movies before:"Princess Bride, The (1987)", "Sleepless in Seattle (1993)", "Willy Wonka & the Chocolate Factory (1971)", "Star Wars: Episode I - The Phantom Menace (1999)", "Groundhog Day (1993)", "Waterworld (1995)", "Indiana Jones and the Last Crusade (1989)", "Alien (1979)", "Ghostbusters (a.k.a. Ghost Busters) (1984)", "Home Alone (1990)"	"Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)", "Home Alone 2: Lost in New York (1992)", "Star Wars: Episode VI - Return of the Jedi (1983)", "Aliens (1986)"	"Home Alone 2: Lost in New York (1992)", "Star Wars: Episode VI - Return of the Jedi (1983)", "Aliens (1986)". "Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977)"	Shrek (2001)

Table 6: **Case Study Demonstrating the Effectiveness of ReKnow.**

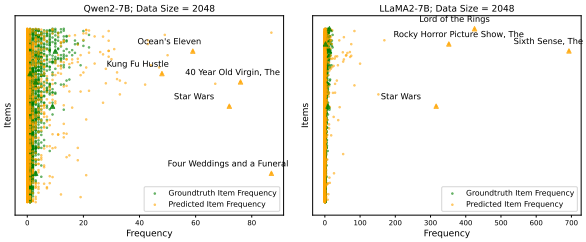

Figure 11: Comparison of item frequency distributions between the ground truth (green) and model predictions (yellow) across two backbones. The results reveal strong item-level biases in LLM predictions, consistent across backbones, with distinct bias patterns likely originating from pretraining (knowledge bias) rather than fine-tuning.

- Re-weighting (RW) (Jiang et al., 2024). RW improves fairness in BIGRec by balancing recommendations across categories through dataset-based training weights. Note that RW requires re-weighting of item categories, but the Game dataset lacks item category features, therefore we only performed the replication on the MovieLens dataset.

- Debiasing-Diversifying Decoding (D3) (Bao et al., 2024). D3 enhances diversity in BIGRec using a decoding strategy guided by SASRec.

Model	GRU4Rec	SASRec	BERT4Rec	FEARec	BIGRec (Backbone = LLaMA2-7B)	ReP (Backbone = LLaMA2-7B)	D3 (Backbone = LLaMA2-7B)	RW (Backbone = LLaMA2-7B)	ReKnow (Backbone = LLaMA2-7B)	BIGRec (Backbone = Qwen2-7B)	ReP (Backbone = Qwen2-7B)	D3 (Backbone = Qwen2-7B)	RW (Backbone = Qwen2-7B)	ReKnow (Backbone = Qwen2-7B)
NDCG@1 ↑	0.002	0.001	0.003	0.001	0.019	0.018	0.019	*0.019*	**0.020**	*0.012*	0.012	0.006	0.003	**0.012**
Hit@1 ↑	0.002	0.001	0.003	0.001	0.019	0.018	0.019	*0.019*	**0.020**	*0.012*	0.012	0.006	0.003	**0.012**
ARP@1 ↓	166,916	138,399	230,757	128,392	69,544	70,970	*69,543*	76,449	**64,630**	56,861	56,862	59,068	61,558	**55,011**
JS@1 ↓	0.68	0.65	0.69	0.68	0.47	0.48	0.47	*0.46*	**0.43**	0.35	0.36	0.45	0.65	**0.35**
NDCG@5 ↑	0.004	0.004	0.009	0.003	0.023	0.023	0.023	*0.024*	**0.025**	*0.015*	0.015	0.008	0.005	**0.015**
Hit@5 ↑	0.007	0.007	0.014	0.005	0.028	0.028	0.028	*0.029*	**0.029**	**0.018**	*0.018*	0.009	0.008	0.017
ARP@5 ↓	139,379	131,426	224,786	135,696	44,601	46,243	*44,604*	46,644	**43,290**	*37,196*	37,350	39,256	42,709	**37,093**
JS@5 ↓	0.67	0.63	0.68	0.66	0.41	0.42	*0.41*	0.41	**0.37**	*0.29*	0.29	0.37	0.61	**0.28**
NDCG@10 ↑	0.007	0.006	0.011	0.006	0.025	0.024	0.025	*0.026*	**0.027**	**0.017**	0.016	0.009	0.006	*0.016*
Hit@10 ↑	0.016	0.012	0.023	0.014	0.033	0.033	0.033	*0.034*	**0.036**	**0.023**	0.022	0.014	0.011	*0.022*
ARP@10 ↓	132,275	123,159	191,375	141,196	*37,246*	39,298	37,300	37,987	**37,018**	**31,761**	*31,921*	34,931	35,907	32,033
JS@10 ↓	0.67	0.61	0.68	0.65	*0.37*	0.38	0.37	0.37	**0.34**	*0.25*	0.26	0.33	0.59	**0.25**

Table 7: Comparision of ReKnow with SOTA sequential recommendation on Movielens. ↑ indicates higher is better, ↓ indicates lower is better. Best results for each backbone model are shown in **bold**, and the second best results are shown in *italics*.

Model	GRU4Rec	SASRec	BERT4Rec	FEARec	BIGRec (Backbone = LLaMA2-7B)	ReP (Backbone = LLaMA2-7B)	D3 (Backbone = LLaMA2-7B)	ReKnow (Backbone = LLaMA2-7B)	BIGRec (Backbone = Qwen2-7B)	ReP (Backbone = Qwen2-7B)	D3 (Backbone = Qwen2-7B)	ReKnow (Backbone = Qwen2-7B)
NDCG@1 ↑	0.0030	0.0008	0.0002	0.0008	*0.012*	0.008	0.012	**0.014**	*0.012*	0.012	0.002	**0.013**
Hit@1 ↑	0.003	0.001	0.000	0.001	*0.012*	0.008	0.012	**0.014**	*0.012*	0.012	0.002	**0.013**
ARP@1 ↓	299	3903	3367	3852	53	69	*50*	**47**	*50*	52	45	**44**
JS@1 ↓	0.69	0.52	0.69	0.53	0.70	**0.57**	0.73	*0.63*	0.74	0.73	*0.65*	**0.64**
NDCG@5 ↑	0.0031	0.0012	0.0004	0.0008	0.014	0.009	*0.015*	**0.017**	**0.015**	*0.015*	0.004	0.014
Hit@5 ↑	0.003	0.002	0.001	0.001	0.017	0.009	*0.017*	**0.019**	**0.018**	*0.017*	0.007	0.016
ARP@5 ↓	16,903	4287	1480	4150	59	**50**	59	*58*	55	55	*54*	**54**
JS@5 ↓	0.69	0.52	0.69	0.51	*0.65*	0.72	0.66	**0.56**	0.68	0.68	*0.59*	**0.57**
NDCG@10 ↑	0.0034	0.0013	0.0005	0.0008	0.017	0.009	*0.017*	**0.018**	*0.016*	0.016	0.005	**0.016**
Hit@10 ↑	0.004	0.002	0.001	0.001	**0.025**	0.012	0.024	*0.024*	**0.022**	0.020	0.009	*0.020*
ARP@10 ↓	16,062	4,519	1,169	4,184	61	**57**	64	*61*	61	*61*	**61**	62
JS@10 ↓	0.69	0.52	0.69	0.50	0.65	0.70	*0.65*	**0.53**	0.66	0.66	*0.55*	**0.53**

Table 8: Comparison of ReKnow with SOTA sequential recommendation on Amazon Games. ↑ indicates higher is better, ↓ indicates lower is better. Best results for each backbone model are shown in **bold**, and second best results are shown in *italics*.

Algorithm 1: ReKnow (Realignment at Decoding-time to Alleviate Knowledge Bias)

User interaction history I, items X, context-free prompt C, alignment scale β, the number of biased items k, the number of tokens per item n.

// Step 1: Identify Biased Items

Detect biased items set X_b using validation data ;

// Step 2: Quantify Model's Inherent Knowledge

for *each item* $x = (t_1, \ldots, t_n)$ *in* X_b **do**

  Compute $f(x|C) = \sum_{j=1}^{n} \log(P(t_j|t_{<j}, C)) \cdot \alpha^j$;

end

// Step 3: Compute Target Distribution

for *each item* x **do**

  Compute $g(x) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{ x_i = x \}$;

end

// Step 4: Realignment for Model Output

for *each item* $x = (t_1, \ldots, t_n)$ **do**

  Compute $w_x = \beta \cdot \{\log(g(x)) - \gamma * \log(f(x|C))\}$;

  Apply $P_{\text{Align}}(x|I, C) = P(x|I, C) \cdot \exp(w_x)$;

end

Aligned probability distribution $P_{\text{Align}}(x|I, C)$.