REPREC: Representation Driven Parameter-Efficient Recommendation System

HARSHINI KAVURU, The Ohio State University, USA
DWIPAM KATARIYA, Capital One, AI Foundations, USA
GIRI IYENGAR, Capital One, AI Foundations, USA
PRANAB MOHANTY, Capital One, AI Foundations, USA
KALANAND MISHRA, Capital One, AI Foundations, USA
RAGHU MACHIRAJU, The Ohio State University, USA

Large language models (LLMs) have been applied to sequential recommendation by formulating it as a natural language task. Previous work has improved personalization by incorporating collaborative and sequential signals through input conditioning or LLM fine-tuning. However, existing approaches often rely on one or more of the following: LLM fine-tuning, additional architectural modules, representation distillation, or item-level conditioning over long interaction histories, increasing training complexity and deployment cost. We propose REPREC, a lightweight framework that reformulates LLM-based sequential recommendation through lightweight user representation alignment. REPREC maps a fixed-size user embedding from a frozen sequential encoder into a small set of learned soft tokens through a lightweight MLP injector that conditions a frozen LLM, leaving both pretrained backbones unchanged while training only the injector. We conducted exhaustive experiments on multiple benchmark datasets and demonstrate that REPREC consistently outperforms LoRA while remaining compatible with different pretrained sequential encoders and LLM backbones, enabling a modular and production-friendly recommendation pipeline without modifying either pretrained component. The gains are particularly pronounced for casual and core users across all datasets, highlighting REPREC's effectiveness in low-data regimes. Finally, when trained on short prompt histories and evaluated with longer contexts, REPREC maintains 85–100% of LoRA's performance while reducing per-epoch training time by an average of 1.51×, demonstrating an effective balance between recommendation quality and computational efficiency for production deployment. *The code is available at* https://github.com/phdbotcode/REPREC

CCS Concepts: • **Information systems** → **Recommender systems**; *Sequential recommendation.*

Additional Key Words and Phrases: Recommendation Systems, User Representation, Large Language Models

ACM Reference Format:
Harshini Kavuru, Dwipam Katariya, Giri Iyengar, Pranab Mohanty, Kalanand Mishra, and Raghu Machiraju. 2026. REPREC: Representation Driven Parameter-Efficient Recommendation System. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX).* ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

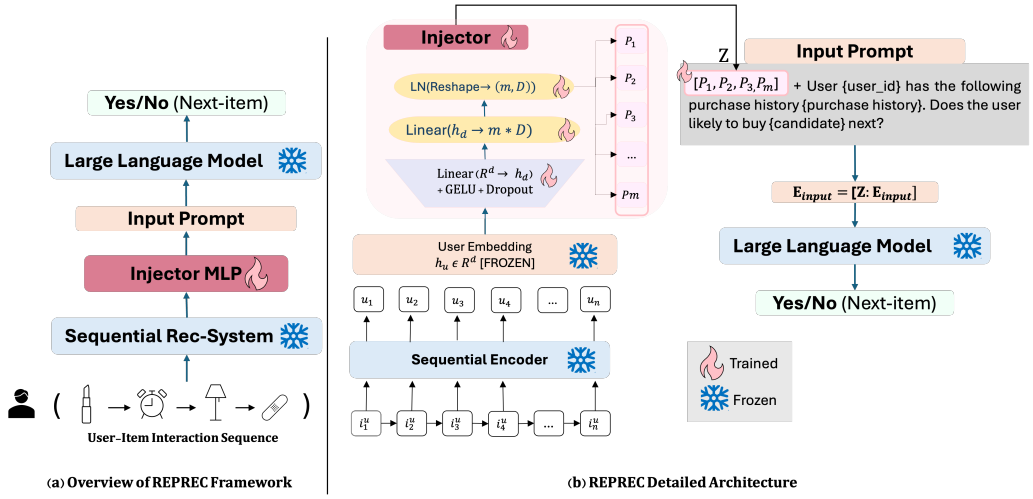

Fig. 1. Overview of the REPREC architecture. The sequential encoder produces a user embedding projected into m soft tokens that condition the frozen LLM for recommendation.

1 Introduction

Sequential recommendation focuses on modeling the order and context of user interactions to predict future behavior. Classical approaches such as collaborative filtering [5, 21, 25] and sequential models [10, 20, 22] have been highly effective in learning compact user representations that capture behavioral patterns and support personalized ranking. More recently, large language models (LLMs) have been adapted for recommendation by reformulating item prediction as a natural language task [2, 4, 23]. These approaches show that pretrained LLMs can leverage world knowledge and instruction-following capabilities to perform competitive recommendation without relying on specialized architectures.

However, textual interaction histories alone may not preserve the structured collaborative signals captured by conventional recommenders. Existing work therefore augments LLMs with representations learned by pretrained collaborative or sequential models. Item-level methods preserve fine-grained behavioral information by conditioning the LLM on representations of individual interactions. For example, LLaRA [16] maps each sequential item embedding to a behavioral token through a learned projection, while A-LLMRec [12] incorporates collaborative representations of users and items into the LLM input. Although this fine-grained conditioning can improve recommendation quality, injecting one representation per interaction causes the input sequence length to grow with the user's history, increasing attention

Authors' Contact Information: Harshini Kavuru, kavuru.7@osu.edu, The Ohio State University, Columbus, Ohio, USA; Dwipam Katariya, dwipam.katariya@capitalone.com, Capital One, AI Foundations, Virginia, USA; Giri Iyengar, Capital One, AI Foundations, McLean, VA, USA; Pranab Mohanty, Capital One, AI Foundations, McLean, VA, USA; Kalanand Mishra, Capital One, AI Foundations, San Jose, CA, USA; Raghu Machiraju, machiraju.1@osu.edu, The Ohio State University, Columbus, Ohio, USA.

Manuscript submitted to ACM

FLOPs and, consequently, training and inference latency. To avoid representing every historical item independently, subsequent methods move toward user-level integration, but typically require stronger coupling between the recommender and the LLM. CoLLM [24] combines user-level collaborative conditioning with a multi-stage training pipeline that includes LoRA-based LLM adaptation. User-LLM [18] enables richer interaction between recommendation and language representations through dedicated cross-attention modules and joint optimization of the recommendation encoder and LLM. LLM-SRec [13] keeps the LLM frozen but transfers sequential knowledge through a representation distillation objective and additional projection modules. Thus, while user-level methods reduce dependence on the raw history length, they often introduce backbone adaptation, joint optimization, architectural changes, or encoder-specific alignment procedures. Although these approaches achieve strong recommendation performance, they introduce trade-offs that can limit practical deployment. Most existing methods require modifying or fine-tuning the LLM backbone, jointly optimizing multiple pretrained components, introduce additional architectural modules, or condition on item-level representations whose computational cost grows with interaction history length. Consequently, these gains often come at the cost of increased training complexity, larger parameter footprints, reduced modularity, and more difficult integration and maintenance. In real-world recommendation systems, however, model quality is only one consideration: inference latency, training speed, scalability, serving cost, and ease of deployment are equally important, as highlighted by prior industry-focused systems [1, 3, 9, 11, 19].

These observations motivate a different design objective: rather than maximizing recommendation performance through increasingly sophisticated adaptation mechanisms, we investigate whether a frozen sequential encoder and a frozen LLM can be connected solely through lightweight user-level representation alignment, while preserving modularity, supporting encoder-agnostic integration, and minimizing deployment complexity. Inspired by multimodal approaches such as ClipCap [17] and BLIP-2 [15], which condition frozen language models through learned soft tokens, we study the viability of this fully frozen alignment paradigm for sequential recommendation. To the best of our knowledge, REPREC is the first work to systematically investigate whether a frozen sequential recommender and a frozen LLM can be connected using only a trainable user-level representation injector, without adapting either pretrained backbone. We introduce **REPREC** (**Rep**resentation-driven **P**arameter-**E**fficient **Rec**ommendation), a lightweight framework that maps a user representation produced by a frozen sequential encoder into a small set of soft conditioning tokens prepended to a frozen LLM. All adaptation is confined to a compact MLP injector, avoiding modifications to either backbone. We investigate the following research questions:

- **RQ1:** Can soft token injection align user representations with the LLM input space for effective recommendation without modifying the backbone?
- **RQ2:** How does performance vary across casual, core, and power user groups?
- **RQ3:** Can REPREC trained on short prompt histories generalize to longer histories at inference time?

2 Methodology

2.1 Problem Definition

Let $\mathcal{D}$ denote a historical interaction dataset consisting of triplets (u, i, y), where $u \in \mathcal{U}$ is a user, $i \in \mathcal{V}$ is an item, and $y \in \{0, 1\}$ indicates whether the interaction occurred. Each user u is associated with a chronological interaction sequence:

$$S_u = (i_1^{(u)}, i_2^{(u)}, \ldots, i_{n_u}^{(u)}), \tag{1}$$

where $i_t^{(u)} \in \mathcal{V}$ and n_u denotes the number of interactions of user u. We adopt the standard next-item recommendation setting. Given a user u and their interaction prefix the objective is to predict the next item $i_{t+1}^{(u)}$ among candidate items from $\mathcal{V}$. We adopt evaluation strategy as described by sasrec, specifically the model ranks the ground-truth next item against sampled negative items (see Appendix B for the sampling procedure). Performance is measured using HIT@K.

2.2 REPREC

REPREC consists of three components as illustrated in Figure 1: (i) a sequential encoder f_θ (the sequential encoder is modular and can be replaced with any recommendation backbones), (ii) a lightweight injector g_ϕ that maps structured user embeddings into the LLM embedding space, and (iii) a frozen LLM backbone $\mathcal{M}$.

2.2.1 Sequential Encoder. We first learn structured behavioral representations using a sequential recommender model. Given a user interaction prefix $S_u^{\leq t} = (i_1^{(u)}, \ldots, i_t^{(u)})$ we train a sequential encoder

$$f_\theta : \mathcal{V}^* \to \mathbb{R}^d$$

that maps the sequence of item IDs into a d-dimensional embedding. The encoder is trained using standard next-item prediction, where prediction is computed via a dot-product between the user representation and candidate item embeddings. After training converges, f_θ is frozen. In this work, we instantiate f_θ as SASRec [10] and BERT4Rec [20].

2.2.2 Injector: User-to-LLM Projection. We introduce a projection function $g_\phi : \mathbb{R}^d \to \mathbb{R}^{m \times D}$, parameterized by ϕ:

- d is the sequential embedding dimension,
- D is the LLM hidden size,
- m is the number of injected soft tokens.

The injector is implemented as a multi-layer perceptron(MLP):

$$g_\phi(\mathbf{u}) = \mathrm{LN}_D\big(\mathrm{reshape}_{m \times D}(\mathbf{W}_2\, \sigma(\mathbf{W}_1 \mathbf{u} + \mathbf{b}_1) + \mathbf{b}_2)\big),$$

where σ is a non-linear activation (GELU in our implementation). The projected output is reshaped into m token embeddings of dimension D, then normalized via a single shared LayerNorm applied independently over the D-dimensional embedding of each token before being prepended to the LLM input space. The output $\mathbf{Z} = g_\phi(\mathbf{u}) \in \mathbb{R}^{m \times D}$ constitutes m soft conditioning tokens in the LLM embedding space.

2.2.3 Prompt Construction and Training: Given a candidate item i, we construct a binary decision prompt(i) and prepend the injected user tokens $\mathbf{Z}$. $\mathbf{E}_{input} = [\mathbf{Z}; \mathbf{E}_{prompt}]$, where $[\cdot;\cdot]$ denotes sequence-dimension concatenation. The frozen LLM $\mathcal{M}$ produces vocabulary logits over $\mathbf{E}_{input}$, and we extract the probability of the target answer token (`Yes`/`No`):

$$p(y = 1 \mid u, i) = \mathrm{softmax}(\mathbf{W}_{llm}\mathbf{h}_T)_{\mathrm{Yes}},$$

where $\mathbf{h}_T \in \mathbb{R}^D$ is the hidden state of $\mathcal{M}$ at the final token position T. We compute cross-entropy loss only at the answer position, masking all prompt tokens:

$$\mathcal{L}(\phi) = \mathbb{E}_{(u,i,y)\sim\mathcal{D}}\left[-\log p_\theta(y \mid u, i)\right],$$

where gradients flow through LLM activations into the injector parameters ϕ while the backbone remains frozen.

3 Experiments

3.1 Data

We evaluate on five Amazon product review benchmarks from the 2018 release [6]. For each user, the last interaction is held out for testing, the second-to-last for validation, and the remainder for training. All datasets are highly sparse, with average sequence lengths between 7.95 and 8.88 interactions per user. Table 1 summarizes the statistics after preprocessing.

Table 1. Statistics of datasets after preprocessing. Avg. Len denotes the average sequence length per user. Sparsity is computed as $1 - \frac{\#\text{Int.}}{\#\text{Users} \times \#\text{Items}}$. Casual, Core and Power report the percentage of users in each activity group.

Dataset	#Users	#Items	#Int.	Avg. Len	Sparsity	Casual	Core	Power
Beauty	22,363	12,101	198,498	8.88	99.927%	62.9%	32.9%	4.2%
Sports & Outdoors	35,598	18,357	296,241	8.32	99.955%	63.6%	33.6%	2.8%
Toys & Games	19,412	11,924	167,247	8.62	99.928%	65.3%	30.9%	3.8%
Pet Supplies	19,856	8,510	157,836	7.95	99.907%	66.1%	31.8%	2.1%
Tools & Home Improvement	16,638	10,217	134,476	8.08	99.921%	66.0%	31.5%	2.5%

3.2 Baselines

We compare REPREC against two groups of baselines: conventional sequential recommendation models and LLM-based recommendation methods. For the sequential recommendation baselines, we evaluate SASRec [10] and BERT4Rec [20] as standalone models. These models also serve as the pretrained sequential encoders used by REPREC. For the LLM-based baselines, we consider Zero-shot LLM, Frozen Injector, LoRA-FT [7], and LLaRA [16]. The Zero-shot LLM provides a frozen LLM with the user's recent interactions represented as text, without parameter adaptation or injected collaborative information. Frozen Injector prepends soft tokens generated by a randomly initialized, untrained projection module while keeping the LLM frozen, thereby controlling for improvements caused solely by introducing continuous prompt tokens. LoRA-FT adapts the LLM through low-rank fine-tuning using the interaction history in textual form. LLaRA projects item-level representations from a pretrained sequential recommender into the LLM token space; following its original configuration, we use **S**ASRec(LLaRA-**S**) and **B**ERT4Rec(LLaRA-**B**) as its sequential encoders. We evaluate all four LLM-based baselines using LLaMA as the backbone. To examine generalization across LLM backbones, we additionally evaluate Qwen, using LoRA-FT as the adapted-LLM baseline for comparison with REPREC. REPREC is evaluated with both **S**ASRec(REPREC-**S**) and **B**ERT4Rec(REPREC-**B**) as sequential encoders under the LLaMA and Qwen backbones.

3.3 Implementation Details

We use a maximum interaction-history length of 50 for all models and evaluate recommendation performance using HIT@5 and HIT@10, respectively. For the LLM-based methods, recommendation is formulated as a binary candidate-relevance task. Candidate items are ranked using the model's relevance scores, and Hit Rate is computed from the resulting ranking. To ensure comparison under matched trainable-parameter budgets, we use LoRA rank r=8 for LoRA-FT and LLaRA and m=6 soft tokens for REPREC; additional parameter-budget details are provided in Appendix C. The results for REPREC and LoRA-FT are averaged over five random seeds and reported as mean and standard deviation to assess robustness to variations in model initialization.

Table 2. Performance comparison across five datasets with `max_history=50`. REPREC and LoRA-FT results are averaged over five seeds and reported as mean ± standard deviation. Green and light-blue cells indicate the best and second-best performance, respectively, in each metric column. Underlined values indicate the best mean performance among methods using Qwen as the LLM. Tied results receive the same formatting.

Setting	Backbone	Method	Beauty		Sports		Toys		Pet Supplies		Tools	
			HIT@5	HIT@10	HIT@5	HIT@10	HIT@5	HIT@10	HIT@5	HIT@10	HIT@5	HIT@10
Sequential	–	SASRec	0.172	0.247	0.130	0.202	0.177	0.246	0.126	0.204	0.097	0.153
		BERT4Rec	0.080	0.141	0.026	0.054	0.063	0.116	0.081	0.153	0.045	0.085
Baseline	LLaMA	Zero-shot	0.070	0.105	0.050	0.090	0.050	0.090	0.051	0.089	0.053	0.085
		Frozen Inj.	0.065	0.101	0.040	0.072	0.040	0.072	0.042	0.073	0.052	0.083
		LLaRA-S	0.1522	0.2882	0.1694	0.2509	0.310	0.395	0.230	0.327	0.190	0.267
		LLaRA-B	0.1190	0.1815	0.0454	0.0734	**0.346**	0.423	0.248	0.359	0.229	0.312
		LoRA-FT	0.337 ± 0.011	0.435 ± 0.013	0.248 ± 0.008	0.365 ± 0.007	**0.346 ± 0.006**	0.421 ± 0.003	0.264 ± 0.006	0.368 ± 0.002	0.233 ± 0.001	0.319 ± 0.001
	Qwen	LoRA-FT	0.317 ± 0.001	0.412 ± 0.002	0.251 ± 0.010	0.360 ± 0.007	0.334 ± 0.002	0.408 ± 0.006	0.259 ± 0.003	0.361 ± 0.001	0.221 ± 0.010	0.307 ± 0.002
REPREC	LLaMA	REPREC-S	**0.351 ± 0.010**	**0.450 ± 0.010**	**0.295 ± 0.012**	**0.407 ± 0.013**	**0.346 ± 0.001**	**0.425 ± 0.001**	0.268 ± 0.004	0.375 ± 0.003	**0.238 ± 0.005**	0.319 ± 0.003
		REPREC-B	0.346 ± 0.001	0.444 ± 0.001	0.275 ± 0.0015	0.3847 ± 0.000	0.339 ± 0.004	0.415 ± 0.005	**0.270 ± 0.010**	**0.378 ± 0.012**	0.237 ± 0.004	**0.323 ± 0.003**
	Qwen	REPREC-S	0.320 ± 0.002	0.414 ± 0.004	0.257 ± 0.006	0.364 ± 0.007	0.333 ± 0.007	0.410 ± 0.005	0.263 ± 0.004	0.367 ± 0.001	0.222 ± 0.006	0.303 ± 0.008
		REPREC-B	0.316 ± 0.002	0.414 ± 0.004	0.265 ± 0.007	0.370 ± 0.010	0.317 ± 0.001	0.394 ± 0.002	0.256 ± 0.004	0.361 ± 0.001	0.226 ± 0.003	0.304 ± 0.002

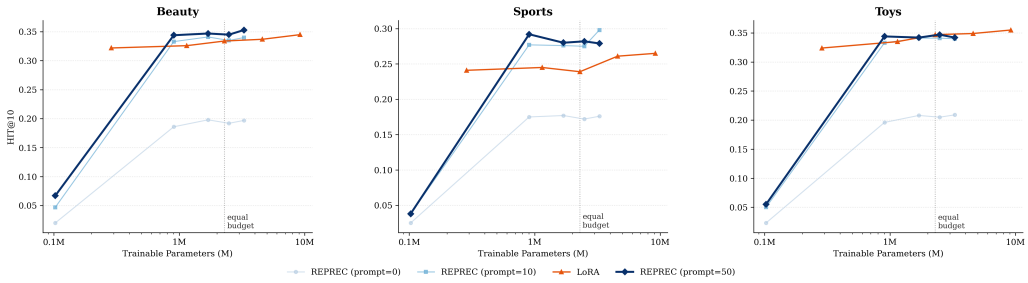

Fig. 2. HIT@5 versus the number of trainable parameters on Beauty, Sports, and Toys using LLaMA as LLM. For REPREC, we vary the number of soft tokens as $m \in \{2, 4, 6, 8\}$, while for LoRA we vary the rank as $r \in \{4, 8, 16, 32\}$. Each marker corresponds to one value of m for REPREC or one value of r for LoRA, with larger values appearing from left to right. The dashed vertical line indicates the matched trainable-parameter budget. For REPREC, prompt=0, prompt=10, and prompt=50 denote the number of recent interaction history items included in the textual prompt during inference, while the injected soft tokens remain unchanged.

3.4 Performance and Efficiency Comparison

3.4.1 ***Overall Performance(RQ1):*** Table 2 reports the main comparison of all baselines. The Frozen Injector consistently performs worse than the zero-shot baseline, confirming that simply prepending untrained embeddings is insufficient and that learning an alignment between collaborative representations and the LLM embedding space is essential. Consistent with this observation, Figure 2 shows that injecting as few as two learned soft tokens already provides substantial improvements over the zero-shot baseline, indicating that only a compact user representation is required to effectively condition the frozen LLM.

Compared with the standalone sequential recommenders, REPREC consistently outperforms both SASRec and BERT4Rec, demonstrating that aligning collaborative user representations with a pretrained LLM allows the model to leverage semantic knowledge beyond what the recommendation encoder alone can capture. Under matched parameter budgets (see Appendix A for details), REPREC further outperforms or matches LoRA and LLaRA across all datasets while leaving the LLM backbone unchanged. These improvements are consistently observed across two sequential encoders (REPREC-S and REPREC-B) and two LLM backbones (LLaMA and Qwen), demonstrating that the proposed representation alignment mechanism generalizes across different pretrained recommendation and language models.

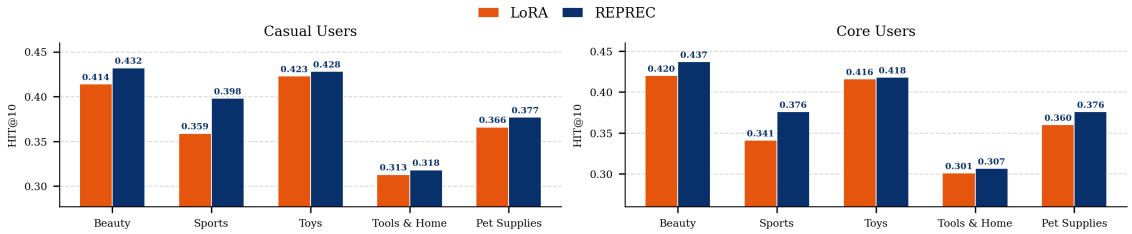

Fig. 3. Comparing REPREC-S and LoRA with LLaMA across Casual and Core users.

Figure 2 further shows that soft-token conditioning alone is insufficient. Using no textual interaction history (prompt=0) consistently underperforms configurations that include recent interaction histories (prompt=10 and prompt=50). As additional prompt history is incorporated, REPREC achieves progressively better recommendation performance, suggesting that the learned user representation and textual interaction history provide complementary information. Finally, both REPREC and LoRA exhibit diminishing returns as model capacity increases, with only marginal gains beyond m=2 soft tokens for REPREC and r=8 for LoRA, suggesting that additional trainable parameters provide limited benefit in this setting.

3.4.2 ***User Regimes(RQ2):*** To examine how performance varies with user activity, we divide users into three activity-based personas according to sequence length: Casual users with 0–5 interactions, Core users with 6–20 interactions, and Power users with more than 20 interactions. The proportion of users in each persona is reported in Table 1.Figure 3 compares REPREC-S and LoRA-FT using the LLaMA backbone for casual and core users. Among casual users, REPREC-S outperforms LoRA-FT across all five datasets, with relative improvements in HIT@10 ranging from 1.17% to 10.86%. REPREC-S also consistently outperforms LoRA-FT for core users, with gains of up to 10.26%. These results suggest that representations learned by the sequential encoder provide valuable collaborative signals when interaction histories are limited. In such sparse settings, the injected user representation complements the semantic knowledge of the LLM without requiring adaptation of the LLM backbone.

For power users, Figure 4 shows that the performance gap between REPREC-S and LoRA-FT becomes substantially smaller. REPREC-S performs comparably to or better than LoRA-FT on most datasets, but slightly underperforms on Beauty. Overall, REPREC provides its clearest benefits for casual and core users, while its advantage narrows as longer interaction histories become available.

3.4.3 ***Training and Inference Prompt Lengths (RQ3):*** We study the effect of prompt history length during training and inference. We first train REPREC(REPREC-S) with a shorter prompt history ($\ell = 10$) and evaluate it under the same setting. As shown in Figure 2, the overall performance remains comparable to training with $\ell = 50$, as the aggregate metrics are dominated by casual and core users. However, when segmented by user regime cohorts; this setting significantly degrades performance for power users. As shown in Figure 4, training and evaluating with $\ell = 10$ leads to a substantial drop in HIT@10 (up to 60.41%), indicating that longer prompt histories are important for modeling power users. To address this, we evaluate a hybrid setting where REPREC is trained with a short prompt history ($\ell = 10$) but evaluated with a longer history ($\ell = 50$) (REPREC$_{\text{cheap}}$). Table 3 shows that REPREC$_{\text{cheap}}$ recovers most of the lost performance, achieving 0.85×–1.00× of LoRA performance on power users across datasets. Importantly, training with shorter histories reduces per-epoch training time by 1.43×–1.81×, with average speedup of 1.51×.

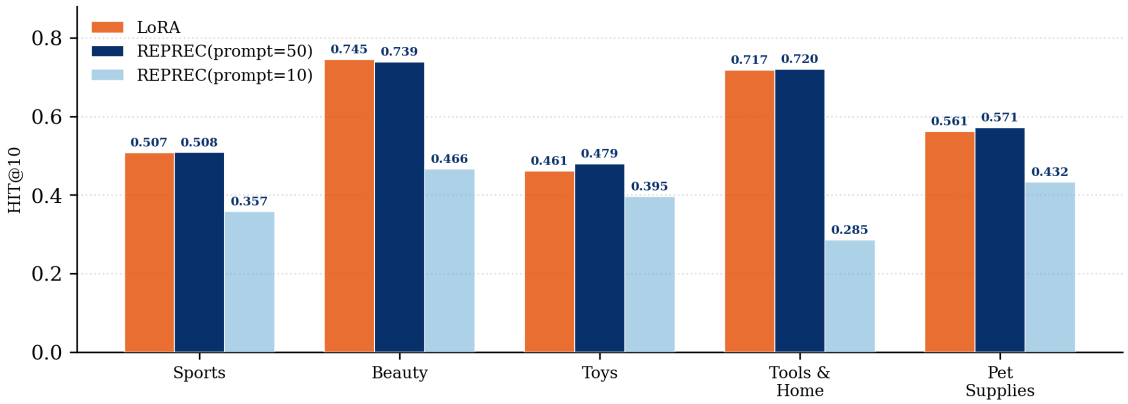

Fig. 4. Comparing REPREC-S, LoRA, and REPREC-S variant for Power users.

Table 3. Performance and efficiency evaluated for Power User persona. $\text{REPREC}_{\text{cheap}}$ (train ℓ=10, test ℓ=50) is compared to REPREC (ℓ=50) and LoRA. Ratios denote relative HIT@10; times are per epoch on an A100 (40GB).

Dataset	HIT@10			Train Time (min/epoch)		Performance Ratio		Efficiency	
	REPREC	LoRA	$\text{REPREC}_{\text{cheap}}$	LoRA	$\text{REPREC}_{\text{cheap}}$	vs REPREC	vs LoRA	Saved (min)	Speedup
Beauty	0.739	0.745	0.628	430	314	0.85×	0.84×	116	1.37×
Sports	0.508	0.507	0.485	425	283	0.95×	0.96×	142	1.50×
Toys	0.479	0.462	0.451	288	159	0.94×	0.98×	129	1.81×
Tools&Home	0.720	0.717	0.628	284	199	0.87×	0.88×	85	1.43×
Pet Supplies	0.571	0.561	0.563	249	174	0.99×	1.00×	75	1.43×

4 Conclusion

This work examined whether structured sequential representations can improve LLM-based recommendation without adapting either pretrained backbone. Our results show that a lightweight user-level alignment module can effectively transfer collaborative signals from a frozen sequential encoder to a frozen LLM, without requiring LLM fine-tuning, joint backbone optimization, or complex architectural changes. REPREC is especially effective for casual and core users and generalizes from shorter training prompts to longer histories at inference time, providing a favorable balance between efficiency and recommendation quality. By keeping both backbones independent and connecting them through a compact alignment layer, REPREC reduces trainable parameters and system coupling while remaining compatible with multiple sequential encoders and LLM backbones. These findings establish user-level representation alignment as a simple, modular, and practical approach for integrating collaborative signals into LLM-based recommendation.

For future work we want to extend REPREC to a broader range of sequential encoders, such as GRU4Rec and MAMBA. We also plan to evaluate REPREC on datasets with substantially longer interaction histories to further investigate its ability to train with shorter prompt histories while effectively leveraging longer contexts at inference time. Finally, exploring larger and more capable LLM backbones, such as LLaMA-3.1-8B and Mistral [8], will help assess the scalability of lightweight representation alignment across increasingly powerful foundation models.

References

[1] Marco De Nadai, Francesco Fabbri, Paul Gigioli, Alice Wang, Ang Li, Fabrizio Silvestri, Laura Kim, Shawn Lin, Vladan Radosavljevic, Sandeep Ghael, David Nyhan, Hugues Bouchard, Mounia Lalmas, and Andreas Damianou. 2024. Personalized Audiobook Recommendations at Spotify Through Graph Neural Networks. In *Companion Proceedings of the ACM Web Conference 2024* (Singapore, Singapore) *(WWW '24)*. Association for Computing Machinery, New York, NY, USA, 403–412. doi:10.1145/3589335.3648339

[2] Zhiang Dong, Liya Hu, Jingyuan Chen, Zhihua Wang, and Fei Wu. 2025. Comprehend Then Predict: Prompting Large Language Models for Recommendation with Semantic and Collaborative Data. *ACM Trans. Inf. Syst.* 43, 5, Article 115 (July 2025), 26 pages. doi:10.1145/3716499

[3] Ghazal Fazelnia, Sanket Gupta, Claire Keum, Mark Koh, Ian Anderson, and Mounia Lalmas. 2024. Generalized User Representations for Transfer Learning. arXiv:2403.00584 [cs.IR] https://arxiv.org/abs/2403.00584

[4] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). In *Proceedings of the 16th ACM Conference on Recommender Systems* (Seattle, WA, USA) *(RecSys '22)*. Association for Computing Machinery, New York, NY, USA, 299–315. doi:10.1145/3523227.3546767

[5] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web* (Perth, Australia) *(WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 173–182. doi:10.1145/3038912.3052569

[6] 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* (2024).

[7] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

[8] Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

[9] Wenqi Jiang, Zhenhao He, Shuai Zhang, Kai Zeng, Liang Feng, Jiansong Zhang, Tongxuan Liu, Yong Li, Jingren Zhou, Ce Zhang, and Gustavo Alonso. 2021. FleetRec: Large-Scale Recommendation Inference on Hybrid GPU-FPGA Clusters. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 3097–3105. doi:10.1145/3447548.3467139

[10] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. 197–206. doi:10.1109/ICDM.2018.00035

[11] Dwipam Katariya, Snehita Varma, Akshat Shreemali, Benjamin Wu, Kalanand Mishra, and Pranab Mohanty. 2025. FinTRec: Transformer Based Unified Contextual Ads Targeting and Personalization for Financial Applications. *arXiv preprint arXiv:2511.14865* (2025).

[12] Sein Kim, Hongseok Kang, Seungyoon Choi, Donghyun Kim, Minchul Yang, and Chanyoung Park. 2024. Large Language Models meet Collaborative Filtering: An Efficient All-round LLM-based Recommender System. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Barcelona, Spain) *(KDD '24)*. Association for Computing Machinery, New York, NY, USA, 1395–1406. doi:10.1145/3637528.3671931

[13] Sein Kim, Hongseok Kang, Kibum Kim, Jiwan Kim, Donghyun Kim, Minchul Yang, Kwangjin Oh, Julian McAuley, and Chanyoung Park. 2025. Lost in Sequence: Do Large Language Models Understand Sequential Recommendation?. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2* (Toronto ON, Canada) *(KDD '25)*. Association for Computing Machinery, New York, NY, USA, 1160–1171. doi:10.1145/3711896.3737035

[14] Walid Krichene and Steffen Rendle. 2022. On sampled metrics for item recommendation. *Commun. ACM* 65, 7 (June 2022), 75–83. doi:10.1145/3535335

[15] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of the 40th International Conference on Machine Learning* (Honolulu, Hawaii, USA) *(ICML'23)*. JMLR.org, Article 814, 13 pages.

[16] Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. LLaRA: Large Language-Recommendation Assistant. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington DC, USA) *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 1785–1795. doi:10.1145/3626772.3657690

[17] Ron Mokady, Amir Hertz, and Amit Bermano. 2021. ClipCap: CLIP Prefix for Image Captioning. doi:10.48550/arXiv.2111.09734

[18] Lin Ning, Luyang Liu, Jiaxing Wu, Neo Wu, Devora Berlowitz, Sushant Prakash, Bradley Green, Shawn O'Banion, and Jun Xie. 2025. User-LLM: Efficient LLM Contextualization with User Embeddings. In *Companion Proceedings of the ACM on Web Conference 2025* (Sydney NSW, Australia) *(WWW '25)*. Association for Computing Machinery, New York, NY, USA, 1219–1223. doi:10.1145/3701716.3715463

[19] Nikil Pancha, Andrew Zhai, Jure Leskovec, and Charles Rosenberg. 2022. PinnerFormer: Sequence Modeling for User Representation at Pinterest. doi:10.48550/arXiv.2205.04507

[20] 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* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 1441–1450. doi:10.1145/3357384.3357895

[21] Riya Widayanti, Mochamad Heru Chakim, Chandra Lukita, Untung Rahardja, and Ninda Lutfiani. 2023. Improving Recommender Systems using Hybrid Techniques of Collaborative Filtering and Content-Based Filtering. *Journal of Applied Data Sciences* 4, 3 (2023), 289–302. doi:10.47738/jads.v4i3.115

[22] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive Learning for Sequential Recommendation. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. 1259–1273. doi:10.1109/ICDE53745.2022.00099

[23] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2025. Recommendation as Instruction Following: A Large Language Model Empowered Recommendation Approach. *ACM Trans. Inf. Syst.* 43, 5, Article 114 (July 2025), 37 pages. doi:10.1145/3708882

[24] Yang Zhang, Fuli Feng, Jizhi Zhang, Keqin Bao, Qifan Wang, and Xiangnan He. 2025. CoLLM: Integrating Collaborative Embeddings Into Large Language Models for Recommendation. *IEEE Transactions on Knowledge and Data Engineering* 37, 5 (2025), 2329–2340. doi:10.1109/TKDE.2025.3540912

[25] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. 2013. Interactive collaborative filtering. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management* (San Francisco, California, USA) *(CIKM '13)*. Association for Computing Machinery, New York, NY, USA, 1411–1420. doi:10.1145/2505515.2505690

A Parameter Count Analysis

A.1 LoRA Trainable Parameters

We apply LoRA to the query (`q_proj`) and value (`v_proj`) projection matrices of every transformer layer in the frozen LLaMA-3.2-3B-Instruct backbone (L=28 layers, hidden size D=3072, 24 attention heads, 8 key-value heads via Grouped Query Attention). Each low-rank adapter introduces two factor matrices per target projection, giving a per-rank scaling of 286,720 parameters: $|\theta_{\text{LoRA}}| = 286{,}720 \times r$. Table 4 reports the resulting counts for $r \in \{1, 4, 8, 16, 32\}$.

A.2 REPREC Trainable Parameters

The sequential encoder (SASRec) is pre-trained and frozen prior to injector training; only the injector MLP parameters are updated end-to-end. The injector maps the frozen SASRec user embedding $\mathbf{u} \in \mathbb{R}^d$ to m soft conditioning tokens in the LLM embedding space: $g_\phi(\mathbf{u}) = \text{LN}_D\big(\text{reshape}_{m\times D}(\mathbf{W}_2\,\sigma(\mathbf{W}_1\mathbf{u} + \mathbf{b}_1) + \mathbf{b}_2)\big)$,, where $\mathbf{W}_1 \in \mathbb{R}^{h_d \times d}$, $\mathbf{W}_2 \in \mathbb{R}^{(m\cdot D)\times h_d}$, h_d=128 is the intermediate hidden size, D=3072 is the LLM hidden size, and LN is a LayerNorm applied over the D-dimensional soft token embeddings after reshape. The total parameter count is:

$$|\theta_{\text{inj}}| = \underbrace{(d \cdot h_d + h_d)}_{\text{Layer 1}} + \underbrace{(h_d \cdot mD + mD)}_{\text{Layer 2}} + \underbrace{2D}_{\text{LayerNorm}} \tag{2}$$

B Negative Sampling Strategy

To ensure reproducible and fair evaluation across all methods, we fix the negative item sets prior to training using a hybrid **Random + Popularity-based Negative Sampling** (PNS) strategy. For each test user, we sample K=200 negative items from the full item pool. Negatives are drawn from the *entire* item catalogue without excluding user-interacted items, following standard practice in sequential recommendation evaluation.

Hybrid sampling procedure. Let $\mathcal{V}$ denote the full item vocabulary with $|\mathcal{V}|$ items. For each item $i \in \mathcal{V}$, let c_i be its interaction count in the training split. We define a popularity weight with Laplace smoothing:

$$w_i = c_i + 1, \quad i \in \mathcal{V}, \tag{3}$$

so that every item (including unseen ones) has non-zero sampling probability. Given a popularity ratio $\rho \in [0, 1]$, for each test user u we draw:

- $K_{\text{pop}} = \lfloor \rho K \rfloor$: negatives sampled proportionally to item popularity weights $\{w_i\}$.
- $K_{\text{rand}} = K - K_{\text{pop}}$: negatives sampled uniformly at random from $\mathcal{V}$.

Table 4. Trainable-parameter counts for LoRA and the REPREC injector under different configurations. Rows marked † denote the configurations used in the main experiments.

(a) LoRA trainable parameters

Rank r	Params / layer	Total params
1	10,240	286,720
4	40,960	1,146,880
8†	81,920	2,293,760
16	163,840	4,587,520
32	327,680	9,175,040

`q_proj` and `v_proj` only, using LLaMA-3.2-3B-Instruct with D=3072, L=28, and GQA with 8 KV heads.

(b) REPREC injector parameters

Soft tokens m	Trainable params
2	807,040
4	1,599,616
6†	2,392,192
8	3,184,768

Injector dimensions are d=64, h_d=128, and D=3072. SASRec remains frozen. The m=6 configuration approximately matches LoRA r=8 (2.39M versus 2.29M parameters).

The two sets are concatenated and shuffled. We use ρ=0.5, yielding 100 popularity-weighted and 100 uniform-random negatives per user. Popularity-weighted negatives serve as *informative hard negatives*: frequently interacted items that the target user did not purchase are more discriminative than rare items, providing a more realistic evaluation signal than pure random sampling [14].