LLM-based User Profile Management for Recommender System

Seunghwan Bang
UNIST
Ulsan, South Korea
shbang1422@unist.ac.kr

Hwanjun Song†
KAIST
Daejeon, South Korea
songhwanjun@kaist.ac.kr

Abstract

The rapid advancement of Large Language Models (LLMs) has opened new opportunities in recommender systems by enabling recommendations without conventional training. Despite their potential, many existing works rely solely on users' purchase histories, leaving significant room for improvement by incorporating user-generated textual data, such as reviews and product descriptions. Addressing this gap, we propose PURE, a novel LLM-based recommendation framework that builds and maintains evolving user profiles by systematically extracting and summarizing key information from user reviews. PURE consists of three core components: a Review Extractor for identifying user preferences and key product features, a Profile Updater for refining and updating user profiles, and a Recommender for generating personalized recommendations using the most current profile. To evaluate PURE, we introduce a continuous sequential recommendation task that reflects real-world scenarios by adding reviews over time and updating predictions incrementally. Our experimental results on Amazon datasets demonstrate that PURE outperforms existing LLM-based methods, effectively leveraging long-term user information while managing token limitations.

CCS Concepts

• **Information systems** → **Recommender systems**; *Personalization*.

Keywords

Large Language Models (LLMs), Recommender Systems (RS), Personalization

ACM Reference Format:
Seunghwan Bang and Hwanjun Song†. 2025. LLM-based User Profile Management for Recommender System. In *Proceedings of The 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (GENNEXT@SIGIR 2025).* ACM, New York, NY, USA, 5 pages.

1 Introduction

The rapid advancement of Large Language Models (LLMs) [1, 6, 23, 24] has significantly impacted various domains, such as text summarization [16] and search [13]. Recent studies leverage LLMs in recommender systems for their human-like reasoning and external knowledge integration through in-context learning [3] and retrieval-augmented generation [17]. As such, LLMs exhibit the potential to be used as *train-free* recommendation models without conventional training, which traditionally relies on explicit user-item interactions and training data [7, 8, 12].

Despite the advanced capability of LLMs, most recent works [9, 11, 21, 27, 28] rely solely on users' past purchase history (*i.e.,* list of purchased items). This leaves significant room for further improvement by incorporating additional user-generated textual information, such as user reviews and product descriptions, which have yet to be fully leveraged. In other words, they still fail to fully leverage various text data due to their inability to retain and process the increasing contextual information as users continue to make purchases, leading to longer recommendation sessions. This issue is primarily attributed to the *omission* of the context, either due to the information loss within the LLM's memory [19] or the memory capacity by the token limit [5, 18]. Thus, extracting key features from a user's textual sources is essential, as demonstrated in MemoryBank [30], a framework that enhances LLMs with *long-term* memory by summarizing key information from conversations and updating user profiles.

Building on this foundation, we take the first step in extending LLMs' long-term memory beyond conversations in MemoryBank, adapting it to the evolving dynamics of recommendation systems. We propose PURE, a novel LLM-based **P**rofile **U**pdate for **RE**commender that constructs a user profile by integrating users' purchase history and user-generated reviews, which naturally expand as the recommendation sessions progress. As illustrated in Fig. 1, PURE systematically extracts user likes, dislikes, and key features from reviews and integrates them into structured, dynamic user profiles. Specifically, PURE consists of three main components: *"Review Extractor"*, which analyzes user reviews to identify and extract user likes, dislikes, and preferred product features, referred to as "key features", offering a comprehensive view of user interests and purchase-driving attributes; *"Profile Updater"*, which refines newly extracted representations by eliminating redundancies and resolves conflicts with the existing user profile, ensuring a compact and coherent user profile; and *"Recommender"*, which utilizes the most up-to-date user profile for recommendation task.

Our main contributions are as follows: (1) We propose PURE, a novel framework that systematically extracts and stores key information from user reviews, optimizing LLM memory management for the recommendation. (2) We validate the effectiveness of PURE by introducing a more realistic sequential recommendation setting, where reviews are incrementally added over time, allowing the model to update user profiles and predict the next purchase continuously. This

† Corresponding Author.

GENNEXT@SIGIR 2025, Padova, Italy

ACM ISBN 978-1-4503-XXXX-X/2025/07

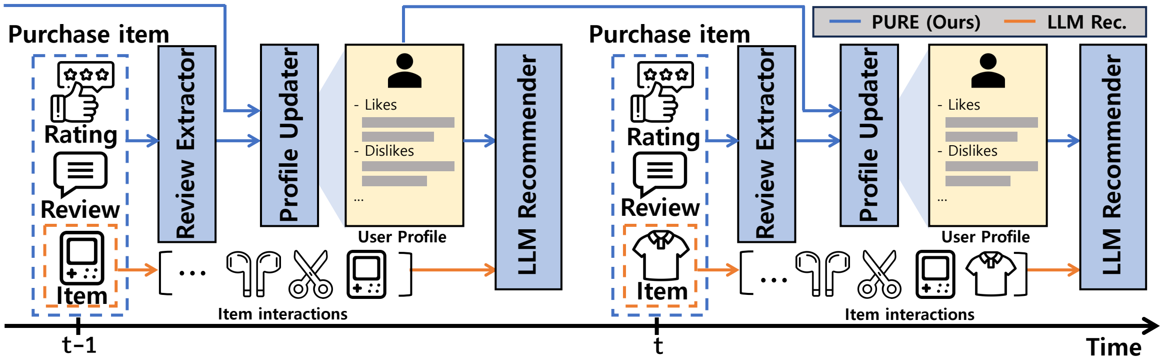


Figure 1: Overall system of PURE. PURE incorporates reviews, ratings, and item interactions, whereas LLM Recommender handles only item interactions. By using the "*Review Extractor*" to identify key information and the "*Profile Updater*" to refine the user profile, PURE addresses scalability issue (*i.e.,* growth of input token size).

setup more accurately reflects real-world recommendation scenarios compared to prior works, which assume all past purchases are provided at once, ignoring the evolving nature of user preferences. (3) We empirically show that PURE surpasses existing LLM-based recommendation methods on Amazon data, demonstrating its effectiveness in leveraging lengthy purchase history and user reviews.

2 Method

2.1 Problem Formulation

In our recommender system, we consider the user u dataset as follow: $\mathfrak{D}_u = \{\mathbf{R}_u, \mathbf{I}_u\}$, where $\mathbf{R}_u = \{r_u^1, \cdots, r_u^{k_u}\}$ represents the historical reviews, $\mathbf{I}_u = \{i_u^1, \cdots, i_u^{k_u}\}$ denotes the corresponding purchased items, and k_u is the total number of purchased items from user u. Leveraging the user's dataset $\mathfrak{D}_u$, we aim to predict the next purchased item $i_u^{k_u+1}$ from a candidate set $C_u^{k_u+1}$, which contains the ground-truth item.

One-shot Sequential Recommendation. It predicts a single next item based on a static history of user interactions up to timestep k_u-1. Given the dataset $\mathfrak{D}_u$, the model observes $\mathfrak{D}_u^{k_u-1} = \{\mathbf{R}_u^{k_u-1}, \mathbf{I}_u^{k_u-1}\}$ and predicts the last item $i_u^{k_u}$ from the candidate set $C_u^{k_u}$. This focuses on a one-time prediction without considering future timesteps.

Continuous Sequential Recommendation. This setup predicts the next item at every timestep ($4 \le t \le k_u - 1$), making it a multi-step prediction task. At each timestep t, the model observes the updated interaction history $\mathfrak{D}_u^t = \{\mathbf{R}_u^t, \mathbf{I}_u^t\}$ and predicts the next item i_u^{t+1} from the candidate set C_u^{t+1}. This multi-step prediction process effectively captures temporal dependencies and allows continuous updates of user preferences, making it more aligned with real-world scenarios.

2.2 PURE: Profile Update for REcommender

In this section, we introduce PURE, novel framework that manages the user profile $\mathbf{P}_u$ from user reviews $\mathbf{R_u}$ and predict the next item with user profile. Algorithm 1 can be divided into three steps.

STEP 1: Extract User Representation.
We begin by providing the LLM with raw inputs, including user reviews $\mathbf{R_u}$ and product names $\mathbf{I_u}$. The LLM extracts $\tilde{l}_u^t$(items the user likes), $\tilde{d}_u^t$(items the user dislikes), and $\tilde{f}_u^t$(key user features) from the incoming review as user representation. To do so, we utilized the following prompt template:

```
Algorithm 1: PURE
Input: Review extractor $\mathcal{E}(\cdot)$, User profile updater $\mathcal{U}(\cdot)$,
       Recommender $\mathcal{R}(\cdot)$, Dataset $\mathfrak{D}_u = \{\mathbf{R}_u, \mathbf{I}_u\}$ for user $u$,
       User profile $\mathbf{P}_u^t$, next purchase candidates $C_u^{t+1}$, timestep $t$
# Extract representations from reviews
$\tilde{l}_u^t, \tilde{d}_u^t, \tilde{f}_u^t = \mathcal{E}(r_u^t)$
$\hat{l}_u^t = l_u^{t-1} \cup \tilde{l}_u^t$      ▷ List of items user likes
$\hat{d}_u^t = d_u^{t-1} \cup \tilde{d}_u^t$      ▷ List of items user dislikes
$\hat{f}_u^t = f_u^{t-1} \cup \tilde{f}_u^t$      ▷ List of user's key features
# Update user profile after redundancy removal
$l_u^t, d_u^t, f_u^t = \mathcal{U}(\hat{l}_u^t, \hat{d}_u^t, \hat{f}_u^t)$
$\mathbf{P}_u^t = \{l_u^t, d_u^t, f_u^t\}$
# Recommend next purchase item
pred = $\mathcal{R}(\mathbf{P}_u^t, \mathbf{I}_u^t, C_u^{t+1})$
Output: pred
```

```
I purchased the following products and left reviews in chronological order: {Asins, product names, input reviews}. Analyze user's likes/dislikes/key features by referring to their reviews.
```

STEP 2: Update User Profile.
After the extraction in STEP 1, the extracted representation <$\tilde{l}_u^t$, $\tilde{d}_u^t$, $\tilde{f}_u^t$> concatenates with previous user profile $\mathbf{P}_u^{t-1} = \{l_u^{t-1}, d_u^{t-1}, f_u^{t-1}\}$. However, this faces a scalability issue as the number of reviews increases. Thus, leveraging the previous profile, we use an LLM to remove redundant and conflicting content from the extracted representation, yielding a more compact and up-to-date user profile $\mathbf{P}_u^t$ after concatenation. To achieve this, we utilized the following prompt template:

```
You are given a list: {list}. Update this list by removing redundant or overlapping information. Note that crucial information should be preserved.
```

STEP 3: Recommend Next Purchase Item.

		Games				Movies			
Data	Method	N@1	N@5	N@10	N@20	N@1	N@5	N@10	N@20
items	Sequential	10.75	18.25	23.13	28.97	9.99	15.92	20.17	26.94
	Recency	15.34	24.31	28.82	34.24	12.17	17.75	22.18	28.19
	ICL	14.28	26.57	30.51	35.72	12.03	19.56	23.36	29.91
items + reviews	Sequential†	11.14	19.95	24.97	32.00	8.05	13.11	17.72	25.57
	Recency†	12.19	23.64	28.37	35.35	8.54	15.78	21.31	29.21
	ICL†	15.11	26.34	31.25	37.39	12.24	22.10	27.31	34.52
	PURE (Sequential)	15.06	25.71	31.08	38.28	12.59	21.33	25.96	32.21
	PURE (Recency)	**18.18**	28.90	33.91	40.69	13.85	21.99	26.53	33.37
	PURE (ICL)	16.62	**29.81**	**35.60**	**42.00**	**15.80**	**26.32**	**32.03**	**38.93**

Table 1: Comparison PURE with Baselines. We evaluate performance under two data settings: using only item interactions and using item interactions augmented with reviews. † indicates customized baselines where review data is naively incorporated into the original prompt templates designed for item interactions only.

	Data		Components			Games					Movies				
Method	items	reviews	Rec.	Ext.	Upd.	N@1	N@5	N@10	N@20	$\lvert T\rvert$	N@1	N@5	N@10	N@20	$\lvert T\rvert$
Sequential	✓		✓			10.75	18.25	23.13	28.97	245.52	9.99	15.92	20.17	26.94	243.89
	✓	✓	✓			11.14	19.95	24.97	32.00	29165.17	8.05	13.11	17.72	25.57	60429.80
	✓	✓	✓	✓		16.09	26.94	32.35	40.08	486.49	13.05	21.38	26.11	32.62	459.69
	✓	✓	✓	✓	✓	15.06	25.71	31.08	38.28	415.01	12.59	21.33	25.96	32.21	384.87
Recency	✓		✓			15.34	24.31	28.82	34.24	253.31	12.17	17.75	22.18	28.19	249.64
	✓	✓	✓			12.19	23.64	28.37	35.35	29235.16	8.54	15.78	21.31	29.21	60509.43
	✓	✓	✓	✓		20.85	31.36	36.51	43.19	602.13	16.00	24.81	29.66	36.98	565.13
	✓	✓	✓	✓	✓	18.18	28.90	33.91	40.69	485.85	13.85	21.99	26.53	33.37	458.60
ICL	✓		✓			14.28	26.57	30.51	35.72	268.40	12.03	19.56	23.36	29.91	261.58
	✓	✓	✓			15.11	26.34	31.25	37.39	29388.72	12.24	22.10	27.31	34.52	60800.61
	✓	✓	✓	✓		19.60	32.96	38.21	44.97	803.60	16.05	27.25	33.11	40.15	867.36
	✓	✓	✓	✓	✓	16.62	29.81	35.60	42.00	592.48	15.80	26.32	32.03	38.93	634.02

Table 2: Component-wise study of PURE. Each configuration varies which data sources (items, reviews) and which PURE components are used (Rec. = Recommendation, Ext. = Extractor, Upd. = Updater), as indicated by ✓. We report N@k scores ($k \in \{1, 5, 10, 20\}$) and average of input token size (|T|) for Recommender.

Recommender $\mathcal{R}$ reranks the given candidate item list to predict the user's next purchase by leveraging the updated profile $\mathbf{P}_u^t$ and purchased items $\mathbf{I}_u$. As such, here is the prompt template that we utilized:

```
Positive aspects: {likes}
Negative aspects: {dislikes}
Key Features: {key features}
Based on these inputs, rank the {candidate list} from 1
to 20 by evaluating their likelihood of being purchased.
```

3 Experiment

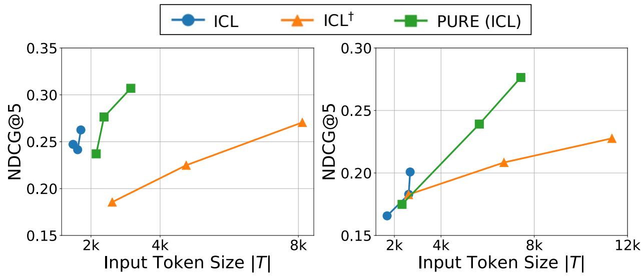
Datasets. For a thorough evaluation, we utilize two datasets from the Amazon collection [20]: Video Games and Movies & TV. To ensure a comprehensive analysis, we select datasets with diverse statistical properties, particularly in terms of the number of items. Each dataset includes ASINs, product names, and user reviews, which are chronologically sorted per user to reflect real-world behavior.

Baselines. LLMRank [11] is the recommendation method that utilizes pre-trained LLMs without additional training or fine-tuning, making it a suitable baseline. It describes three approaches for LLM-based recommendation: Sequential, Recency, and in-context learning (ICL). We compare our method with all three approaches and demonstrate the superiority of PURE when these techniques were applied to our framework, further highlighting its effectiveness. In the following, we describe each approach:

1) Sequential. We provide the LLM with instructions, supplying only the user-item interactions and the candidate list. The LLM is then tasked with ranking the items in the candidate list based on the likelihood of being purchased at time step t.

2) Recency-Focused. In the *sequential* prompt above, we add an instruction to emphasize the most recently purchased item, specifically the one bought at time step $(t-1)$. The additional prompt is: *"Note that my most recently purchased item is {recent item}."*

3) In-Context Learning. Unlike the previous *sequential* and *recency-focused* prompts, this approach utilizes user-item interactions only up to time step $(t-2)$ and the recently purchased item at $(t-1)$. The



(a) Video Games

(b) Movies and TV

Figure 2: Trade-off between NDCG and token size.

additional prompt is: *"I've purchased the following products: {user-item interactions}, then you should recommend {recent item} to me, and now that I've bought {recent item}."*

Evaluation Setting. To evaluate the performance of PURE, we adopt a continuous sequential recommendation. In this setup, the LLM is tasked with predicting the item a user is most likely to purchase at time step t. The model receives the user's interaction history up to time step $(t-1)$ in chronological order, along with a candidate set comprising one ground-truth item and 19 randomly sampled non-interacted items. Here, time step t spans from the user's 4th purchase to their final purchase k. To reflect the sequential nature of the task, NDCG scores are first aggregated across multiple recommendation sessions for each user and then averaged across all users.

Implementation Details. To perform this framework, we utilize Llama-3.2-3B-Instruct [24] as the backbone model for all experiments. Due to the inherent nature of generative language models, it is not always guaranteed that the output will follow the desired format in every response. This issue can be mitigated using structured output formats such as JSON or XML. These formats enable us to enforce consistency and completeness in the model's output by explicitly defining the expected response structure [2, 11]. In our implementation, we prompt the LLM to respond using JSON schemas, which improves reliability during post-processing and facilitates automatic evaluation of model outputs.

3.1 Experimental Results

Impact of Review Extractor. Tab. 1 compares PURE with (1) three baselines solely based on purchased items; (2) modified baselines, marked with †, that additionally utilize users' raw reviews. The results reveal that baselines that simply combine item interactions with raw reviews show inconsistent performance improvements. In contrast, PURE, which leverages the review extractor and profile updater, significantly outperforms all baselines. This demonstrates that processing reviews at three levels, i.e., like, dislike, and key features, is essential for enhancing performance.

Component-wise Study. Tab. 2 shows the ablation study of PURE, where we analyze the impact of reviews (using or not using) and the effect of components (enabling or disabling the review extractor and profile updater). The use of reviews bring high performance gains only when accompanied by Review Extractor (Ext.). This is due to the sharp increase in input tokens (see the $|T|$ column of the 2nd and 3rd rows of each method) as the user continues purchases.

Notably, the best recommendation performance is achieved when Profile Updater (Upd.) is disabled (see the 3rd and 4th rows for each method). This suggests that the well-structured context provided by the Review Extractor alone can lead to strong performance when directly concatenated, even without profile updating. However, it may face a challenge, as the number of purchases grows, leading to significant computational overhead. Thus, we use Profile Updater (Upd.) to maintain compact user profiles, reducing input token size by 15–20% with only a slight 1–3% performance drop. This trade-off underscores the need for Profile Updater for long-term recommendations.

Trade-off Analysis. We categorize users into three groups based on the total cumulative review token count per user, as the criterion: 0–500 (short), 500–1000 (middle), and 1000–2000 (long) tokens. Fig. 2 presents the trade-off between recommendation performance and input token length of the three models including PURE.

PURE achieves the best trade-off, showing the steepest NDCG increase compared to other methods as input token size grows. Therefore, this demonstrates that PURE accurately distills key information from long reviews, while achieving efficiency by minimizing input token growth without information loss, even for long-group users.

4 Related Works

Recommendation Setup. Conventional sequential recommendation methods [10, 12, 15, 22, 26] have followed a one-shot prediction setup, in which a user's interaction history is split such that the most recent item is held out as the test set, the second-most recent as the validation set, and the remaining history is used for training. While this setup simplifies the evaluation pipeline, it restricts the model to predicting a single target item, thereby failing to capture the nuanced and evolving nature of user preferences over time.

LLM-based Recommendation. A notable example is EXP3RT [14], which constructs static user profiles by fine-tuning LLMs directly on target recommendation datasets. These fine-tuned models are then used to compute preference scores over candidate items. Tallrec [2] proposed the parameter-efficient finetuning (PEFT) method in recommender system, and A-LLMRec [15] proposed to finetune the embedding model for LLM to leverage the collaborative knowledge. In contrast, train-free models [25] guided users through a conversational process to elicit responses and extract multiple features. These features are then used to make personalized recommendations in a conversation-based recommendation framework. Also, uncovering ChatGPT's capabilities of recommendation [4] shows ChatGPT is good at reranking the candidates and choosing user preference items while less good at rating. InstructRec [29] designed the instruction to recognize the users' intention and preference from context.

5 Conclusion

We propose PURE, a train-free LLM-based recommendation system that operates within a limited input token budget, while maintaining flexibility and eliminating the need for task-specific training. PURE reduces computational costs compared to train-based systems and adapts to various domains. Additionally, we introduce a sequential recommendation task to better model the evolving nature of user preferences over time, moving beyond conventional sequential setups. This work highlights the potential of train-free LLMs in real-world recommendation scenarios.

Limitations. A notable limitation of our approach is the tendency of the LLM to exhibit hallucination by occasionally recommending items beyond the predefined candidate set, even when explicitly instructed to select from it. This phenomenon underscores the difficulty in imposing strict constraints within LLM-based recommendation models while maintaining flexibility and accuracy. Also, our study was constrained by the inability to utilize datasets containing a larger number of user reviews, which may have provided richer context.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2024-00445087). Additionally, this research was supported by the National Research Foundation of Korea (NRF) funded by Ministry of Science and ICT (RS-2022-NR068758).

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *ArXiv:2303.08774* (2023).

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

[3] Tom B Brown, Benjamin Mann, Nick Ryder, et al. 2020. Language Models are Few-Shot Learners. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 33. 1877–1901.

[4] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 1126–1132.

[5] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. Longrope: Extending llm context window beyond 2 million tokens. *ArXiv:2402.13753* (2024).

[6] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *ArXiv:2407.21783* (2024).

[7] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of international ACM SIGIR conference on research and development in Information Retrieval*. 639–648.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of International Conference on World Wide Web (WWW)*. 173–182.

[9] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. 2023. Large language models as zero-shot conversational recommenders. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*. 720–730.

[10] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 843–852.

[11] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *Proceedings of European Conference on Information Retrieval*. Springer, 364–381.

[12] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *Proceedings of International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[13] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*. 6769–6781.

[14] Jieyong Kim, Hyunseo Kim, Hyunjin Cho, SeongKu Kang, Buru Chang, Jinyoung Yeo, and Dongha Lee. 2024. driven Personalized Preference Reasoning with Large Language Models for Recommendation. *arXiv preprint arXiv:2408.06276* (2024).

[15] 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*. 1395–1406.

[16] Mark Lewis, Yinhan Liu, et al. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

[17] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 33. 9459–9474.

[18] Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. 2024. A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth* 1, 1 (2024), 9.

[19] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics* 12 (2024), 157–173.

[20] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*. 188–197.

[21] Xubin Ren, Wei Wei, Lianghao Xia, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Representation learning with large language models for recommendation. In *Proceedings of the ACM on Web Conference*. 3464–3475.

[22] 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*. 1441–1450.

[23] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *ArXiv:2408.00118* (2024).

[24] Hugo Touvron, Thibaut Lavril, et al. 2023. LLaMA: Open and Efficient Foundation Language Models. *ArXiv:2302.13971* (2023).

[25] Lei Wang and Ee-Peng Lim. 2023. Zero-shot next-item recommendation using large pretrained language models. *ArXiv:2304.03153* (2023).

[26] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z Sheng, and Mehmet Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *Proceedings of International Joint Conference on Artificial Intelligence Organization (IJCAI)*. 6332–6338.

[27] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the ACM International Conference on Web Search and Data Mining*. 806–815.

[28] Jianyang Zhai, Xiawu Zheng, Chang-Dong Wang, Hui Li, and Yonghong Tian. 2023. Knowledge prompt-tuning for sequential recommendation. In *Proceedings of ACM International Conference on Multimedia*. 6451–6461.

[29] Junjie Zhang, Ruobing Xie, Yupeng Hou, Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *ACM Transactions on Information Systems* (2023).

[30] Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*, Vol. 38. 19724–19731.