AgentCF++: Memory-enhanced LLM-based Agents for Popularity-aware Cross-domain Recommendations

Jiahao Liu*
Shengkang Gu*
Fudan University
Shanghai, China
jiahaoliu21@m.fudan.edu.cn
gusk24@m.fudan.edu.cn

Dongsheng Li
Microsoft Research Asia
Shanghai, China
dongshengli@fudan.edu.cn

Guangping Zhang
Fudan University
Shanghai, China
gpzhang20@fudan.edu.cn

Mingzhe Han
Fudan University
Shanghai, China
mzhan22@m.fudan.edu.cn

Hansu Gu
Independent
Seattle, United States
hansug@acm.org

Peng Zhang†
Fudan University
Shanghai, China
zhangpeng_@fudan.edu.cn

Tun Lu†
Fudan University
Shanghai, China
lutun@fudan.edu.cn

Li Shang
Fudan University
Shanghai, China
lishang@fudan.edu.cn

Ning Gu
Fudan University
Shanghai, China
ninggu@fudan.edu.cn

Abstract

LLM-based user agents, which simulate user interaction behavior, are emerging as a promising approach to enhancing recommender systems. In real-world scenarios, users' interactions often exhibit cross-domain characteristics and are influenced by others. However, the memory design in current methods causes user agents to introduce significant irrelevant information during decision-making in cross-domain scenarios and makes them unable to recognize the influence of other users' interactions, such as popularity factors. To tackle this issue, we propose a dual-layer memory architecture combined with a two-step fusion mechanism. This design avoids irrelevant information during decision-making while ensuring effective integration of cross-domain preferences. We also introduce the concepts of interest groups and group-shared memory to better capture the influence of popularity factors on users with similar interests. Comprehensive experiments validate the effectiveness of AgentCF++. Our code is available at https://github.com/jhliu0807/AgentCF-plus.

CCS Concepts

• **Information systems** → **Recommender systems**.

*Equal contribution.
†Corresponding author.

SIGIR '25, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3730161

Keywords

LLM-based agents, user behavior simulation, recommendations

ACM Reference Format:
Jiahao Liu, Shengkang Gu, Dongsheng Li, Guangping Zhang, Mingzhe Han, Hansu Gu, Peng Zhang, Tun Lu, Li Shang, and Ning Gu. 2025. AgentCF++: Memory-enhanced LLM-based Agents for Popularity-aware Cross-domain Recommendations. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3726302.3730161

1 Introduction

Recommender systems play a pivotal role in the dissemination of information today [18, 20–25, 32, 45]. However, their development is hindered by challenges in effectively understanding user behavior [33]. One promising approach to overcoming these challenges is the reliable simulation of user interaction behavior in a controlled, privacy-preserving manner, thereby improving recommender systems by offering insights into user preferences and system performance [49]. Recent advancements in large language models (LLMs), renowned for their capabilities in understanding, reasoning, and generating content [56], have inspired significant efforts to develop LLM-based agents [40]. These agents often incorporate memory modules [54], utilize external tools [46], and engage in advanced reasoning [14], enabling them to exhibit emergent human-like behaviors [29]. In this context, researchers have begun investigating the potential of LLM-based agents to simulate user interaction behavior in the field of recommender systems [3, 37, 41, 48, 51, 55].

Accurately representing user preferences is crucial for a user agent to realistically simulate user behavior. While various terms are used across studies, this paper uniformly refers to these preferences as being stored in *memory*. In real-world scenarios, users' interactions with recommender systems often exhibit cross-domain characteristics [47]. Additionally, individual behaviors are frequently

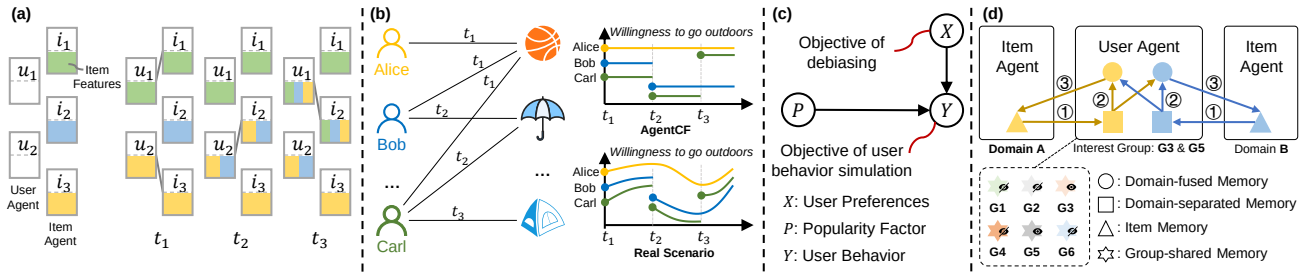

Figure 1: (a) The memory propagation process in AgentCF. (b) An example illustrating AgentCF's limitations in modeling user behavior influenced by popularity factors. (c) Illustration of why modeling popularity factors is necessary for accurately simulating user behavior. (d) Overview of the proposed AgentCF++ model, highlighting its improvements over AgentCF.

influenced by those of others [12]. For instance, popularity-related factors suggest that even in the absence of explicit social networks, such influences can be inferred from interaction graphs and propagated through interaction paths [53]. However, the current memory design causes the user agent to exhibit two significant limitations: **First**, user preferences from multiple domains are mixed into a single memory. However, only a portion of the preferences is relevant to decision-making in the target domain, leading the user agent to process a considerable amount of irrelevant information. **Second**, memory construction relies exclusively on individual user interactions, neglecting how external factors, such as popularity influences, shape user preferences.

In this paper, we present **AgentCF++**, an enhanced version of AgentCF [51]. Our approach introduces a *dual-layer memory architecture* comprising *domain-separated memory* and *domain-fused memory*, designed to prioritize target-domain-relevant information in decision-making for cross-domain scenarios. To refine this architecture, we propose an attention-inspired *two-step fusion mechanism*. This mechanism first identifies valuable cross-domain knowledge pertinent to the target domain and then integrates these critical preferences. Furthermore, we introduce the concept of *interest groups* and propose a *group-shared memory mechanism* to facilitate the transfer of popularity effects within the same interest group. By utilizing interest groups, we ensure that a user's behavior impacts only those with similar interests, effectively preventing the spread of popularity effects to unrelated users. Our experimental results on five cross-domain datasets demonstrate the effectiveness of the proposed modules.

2 Preliminaries

2.1 AgentCF

Unlike previous studies [3, 37, 41, 48, 55] that consider only users as agents, AgentCF [51] treats both users and items as agents. Each user agent is equipped with a memory to store individual preferences, while each item agent maintains a memory to track the interest levels of users with varying preferences towards it. At each step, leveraging LLMs for decision-making and reflection, these agents autonomously interact, compare their actions with real-world data, and collaboratively adjust their memories to better align with observed behaviors. As illustrated in Figure 1(a), user and item memories gradually propagate over time through interactions, embodying the principles of collaborative filtering.

2.2 Limitations of AgentCF

AgentCF employs a single memory for each user agent and item agent. In cross-domain scenarios, the propagation process integrates information from multiple domains into the memory of each user and item. On one hand, the mixing of cross-domain preferences in the user agent may introduce noise, thereby complicating decision-making on target domain. On the other hand, such intermingling may cause the item agent to lose its original domain characteristics.

Additionally, while AgentCF employs collaborative filtering to capture the influence of others' interactions, memory updates for user and item agents occur only during direct interactions, limiting its capacity to comprehensively model how popularity factors shape user behavior. To illustrate this limitation, consider the following example. Figure 1(b) visualizes user-item interaction dynamics using a timestamped bipartite graph. At t_1, Alice, Bob, and Carl purchase outdoor activity items; at t_2, Bob and Carl buy rain gear; at t_3, Carl purchases camping equipment. Under the AgentCF framework, Alice, who ceases interactions after t_1, is still assumed to engage in outdoor activities, despite worsening weather conditions at t_2. Similarly, Bob shows no interest at t_3, failing to account for the improved weather conditions. This indicates that, within the AgentCF framework, users' memories remain static in the absence of additional interactions. In practice, however, even without further direct participation, Alice might infer the weather changes at t_2 and t_3 by observing Bob's and Carl's actions. This underscores the need for user agents to update their memories not only through their own interactions with item agents but also in response to interactions by other user agents. In essence, a user's memory should evolve dynamically, even without direct participation—a critical capability currently missing in the AgentCF framework.

Clarification. As shown in Figure 1(c), user behavior (Y) is influenced by both user preferences (X) and popularity factors (P). Unlike debiasing approaches, which seek to model user preferences by removing the effects of popularity factors [4, 8], user behavior simulation aims to model user behavior. Therefore, in user behavior simulation, popularity factors are not a nuisance to be mitigated but a critical factor to be explicitly modeled.

3 AgentCF++

Similar to AgentCF, AgentCF++ simulates interactions between user agents and item agents from multiple domains and performs reflection during this process to update the memories of both sides, thereby aligning with user behavior.

3.1 Memory Architecture

AgentCF++ employs a similar memory architecture to AgentCF for the item agent, using a single memory to record the interest levels of users with various preferences towards it. Initially, the item's memory is seeded with its side information. However, AgentCF++ has meticulously designed the memory architecture for the user agent to enhance its functionality.

Firstly, each user agent in AgentCF++ is equipped with a **dual-layer memory architecture**. Specifically, each user agent maintains two types of memory for **each domain**: (1) **Domain-separated memory** retains the user's preferences specific to a single domain. (2) **Domain-fused memory** also stores preferences within a particular domain but integrates domain-separated memories from other domains. Initially, both memories are empty.

Additionally, each user agent is assigned to several **interest groups** through the following process: (1) *Building user-tag relationships*: The user agent's domain-fused memory is processed by an LLM to derive a set of interest tags representing the user's preferences. (2) *Merging synonymous tags*: The LLM transforms all tags into embedding vectors, which are then grouped into clusters based on semantic similarity using a K-means clustering algorithm. Each cluster corresponds to a specific area of interest, encompassing tags with high semantic similarity. (3) *Refining interest groups*: The LLM synthesizes the tags in each cluster to generate a consolidated interest group name. Ultimately, only the largest few interest groups are retained, collectively covering the majority of the user's interests. AgentCF++ periodically re-segments the interest groups to ensure they reflect any updates in user preferences.

Each interest group is equipped with a **group-shared memory**, enabling all user agents within the group to collaboratively access shared information. The shared memory is of fixed size, designed to store the recent interaction history of its associated users.

3.2 Inference Phase

We assume that the current interaction is (u, i, d), where u represents a user agent, i represents an item agent, and d denotes the domain of i. First, a negative sample j is selected from the domain d. Then, u receives the memories of i and j and is tasked with identifying the positive sample while explaining its reasoning. To mitigate potential position bias, in which LLMs tend to favor earlier options, j is deliberately placed before i. Note that u's decisions depend simultaneously on both domain-separated memory and domain-fused memory within domain d, as well as on the group-shared memories it can access.

Discussion. The dual-layer memory architecture includes a domain-separated memory and a domain-fused memory corresponding to each domain. With this memory enhancement, only information relevant to the target domain is utilized during decision-making, effectively reducing noise in cross-domain scenarios. On the other hand, the sharing mechanism allows user behavior to influence related users without directly updating their individual memories, incorporating the influence of popularity factors into preferences modeling. For instance, in the scenario depicted in Figure 1(b), Bob and Carl insert the behavior of purchasing rain gear into the memory shared with Alice at t_2. At t_3, Carl adds the behavior of purchasing camping gear into the memory shared with both Alice and Carl. This results in Alice's willingness to go outdoors decreasing at t_2 and increasing at time t_3, reflecting real-world patterns where user behavior is influenced by trends and popularity factors. Note that we segment users based on their interests rather than the similarity of their interaction history, to more precisely identify populations influenced by popularity factors.

3.3 Update Phase

The memories are updated using a reflection mechanism [26, 28, 36]. Specifically: (1) According to the results of the inference phase, u updates its domain-separated memory in domain d using the memories of i and j. This step enables u to learn what it like and dislike from the latest interaction. (2) We propose a **two-step fusion mechanism** to effectively integrate information from multiple domains. Firstly, u extracts preferences related to the target domain d from domain-separated memories of other domains. Then, u updates its domain-fused memory in domain d based on the extracted preferences. (3) i and j update their item memories using u's domain-fused memory in domain d. In this step, i learns which user preferences it appeals to, while j learns which user preferences it does not appeal to.

Discussion. Figure 1(d) illustrates the process of update phase. The two-step fusion mechanism implicitly incorporates the idea of the attention mechanism, ensuring that the domain-fused memory effectively integrates preferences from other domains while retaining only preferences relevant to the corresponding domain. Specifically, in the first step, only preferences related to the target domain are extracted, akin to the computation of attention scores. In the second step, the extracted preferences from different domains are integrated, akin to the weighted aggregation process in the attention mechanism. Additionally, with the aid of the reflection mechanism, the cyclic updates of item memory, domain-separated memory, and domain-fused memory enable all memories to achieve self-improvement.

4 Experiments

4.1 Settings

4.1.1 Datasets. We experimented with the Amazon review dataset [10]. We constructed the cross-domain datasets *Cross-1*—*Cross-5* by combining data from the Books, CDs, Movies, and Games domains, selecting 3 or 4 domains for each dataset. Then, we retained interaction records with ratings ≥ 4 and timestamps spanning six months, from October 2021 to March 2022. We further filtered the data to include only records of users who interacted across multiple domains and had ≥ 10 total interactions. Following AgentCF, we randomly sampled 100 users to minimize API call expenses. Next, we sorted these interaction records chronologically and split them into training, validation, and test sets with an 8:1:1 ratio.

4.1.2 Evaluation. For each ground truth item, we randomly sample 9 items from the same domain that the user has not interacted with to construct the candidate set. The user agent is then tasked with ranking these items, and the ranking performance is measured using NDCG and MRR. We report the average results over 5 runs.

4.1.3 Baselines. We used two traditional recommendation models, BPR-MF [31] and SASRec [16], as well as four training-free methods, Pop, LLMSeqSim [9], LLMRank [11], and AgentCF [51], as baseline methods for comparison. Specifically, Pop ranks candidates based on item popularity, LLMSeqSim evaluates candidates by measuring their similarity to the user's interaction history, and LLMRank employs an LLM as a zero-shot ranker to prioritize candidate items.

We compared solely with AgentCF, omitting other LLM-based user agent methods. Our goal is to demonstrate that the proposed module enhances AgentCF. Performance differences with other methods may stem from differing agent construction paradigms, making it challenging to attribute improvements directly to the proposed module. On the other hand, they all construct memory directly through the user's interaction history, which can, to some extent, be considered equivalent to LLMRank.

We also designed three ablation variants for AgentCF++: (1) *AgentCF + dual*: Extends AgentCF with only the dual-layer memory architecture. (2) *AgentCF + shared*: Extends AgentCF with only interest groups and group-shared memory. (3) *AgentCF++ w/o group*: Users are grouped based on their full interaction history rather than their interests. The other components, including treating users as agents, treating items as agents, the automatic interaction process, and the reflection mechanism, have already been validated as effective by AgentCF. Therefore, these components were not included in the ablation study.

4.2 Overall Performance

As shown in Table 1, on cross-domain datasets, AgentCF achieves comparable performance to training-free methods such as LLMRank but fails to surpass traditional recommendation models like SASRec. This suggests that traditional models inherently capture popularity factors and cross-domain collaborative information through their training mechanisms, giving them a clear advantage over LLM-based user agents in predicting user behavior. Encouragingly, the proposed AgentCF++ consistently outperforms both its ablation variants and all baselines. Moreover, the ablation variants consistently outperform AgentCF, further validating the effectiveness of the proposed modules. Notably, *AgentCF++ w/o group* performs not only worse than AgentCF++ but also worse than *AgentCF + dual*, further underscoring the importance of dividing users into interest groups. This indicates that assigning users to groups based on their full interaction history is too coarse, allowing the popularity factor to influence unrelated users, introducing noise, and ultimately reducing accuracy.

Table 1: Overall performance. Due to space constraints, we only report MRR results. NDCG results, available in the repository, lead to consistent conclusions.

Method	Cross-1	Cross-2	Cross-3	Cross-4	Cross-5
BPR-MF	0.2949	0.2959	0.3114	0.3012	0.3127
SASRec	0.3463	0.3154	0.3828	0.3118	0.3687
Pop	0.2589	0.2817	0.3094	0.2954	0.3089
LLMSeqSim	0.2646	0.2549	0.3101	0.2959	0.3124
LLMRank	0.3268	0.2730	0.3106	0.2970	0.3308
AgentCF	0.3284	0.2681	0.3114	0.3032	0.3480
AgentCF++	**0.3537**	**0.3176**	**0.3989**	**0.3321**	**0.3837**
AgentCF + dual	0.3495	0.2962	0.3581	0.3139	0.3581
AgentCF + shared	0.3488	0.2777	0.3190	0.3147	0.3689
AgentCF++ w/o group	0.3415	0.2724	0.3181	0.3126	0.3549

5 Related Work

LLM-based agents in recommender systems can be broadly divided into two categories. The first category focuses on **recommendation agents** that leverage LLMs to generate or improve recommendations [13, 35, 43, 44, 50, 52, 57]. The second category explores **user agents** that leverage LLMs to simulate user behavior. While some studies focus on simulating user dialogues in conversational recommendation [6, 17, 42, 58, 59], our emphasis is on simulating user interaction behavior. RecAgent [41] and Agent4Rec [48] employ LLM-based agents, incorporating user profiles, memory, and action modules, to simulate interactions with recommender systems. RAH [37] places LLM-based multi-agents between users and recommender systems, serving both as recommendation agents and as proxies for user interactions. FLOW [3] facilitates collaboration between recommendation agents and user agents by establishing a feedback loop. Zhang et al. [55] integrate explicit user preferences, LLM-driven sentiment analysis, a human engagement model, and a statistical framework to robustly simulate user interactions. AgentCF [51] proposes a novel approach, conceptualizing users and items as agents and employing a collaborative learning strategy to optimize them simultaneously.

Several studies have highlighted LLMs' generalization capabilities in **cross-domain recommendation** [1, 2, 30, 34, 38, 39] and explored **popularity bias** in LLM-based recommenders [5, 7, 15, 19, 27]. These works mainly use LLMs as recommenders, not for simulating user behavior. Importantly, we emphasize the need to explicitly model popularity factors when simulating user behavior, rather than merely reducing their influence.

6 Conclusions

We propose AgentCF++, which consists of: (1) a dual-layer memory architecture and a two-step fusion mechanism that allow the user agent to avoid introducing irrelevant in cross-domain scenarios; (2) the concept of interest groups and a shared memory mechanism that captures the influence of popularity among users with similar interests. Comprehensive experiments demonstrate the effectiveness of AgentCF++. In the future, we aim to adapt these designs to other LLM-based user agent frameworks.

Acknowledgments

This work is supported by National Natural Science Foundation of China (NSFC) under the Grant No. 62172106. Peng Zhang is a faculty of School of Computer Science, Fudan University. Tun Lu is a faculty of School of Computer Science, Shanghai Key Laboratory of Data Science, Fudan Institute on Aging, MOE Laboratory for National Development and Intelligent Governance, and Shanghai Institute of Intelligent Electronics & Systems, Fudan University.

References

[1] Zhuoxi Bai, Ning Wu, Fengyu Cai, Xinyi Zhu, and Yun Xiong. 2024. Aligning Large Language Model with Direct Multi-Preference Optimization for Recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 76–86.

[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] Shihao Cai, Jizhi Zhang, Keqin Bao, Chongming Gao, and Fuli Feng. 2024. FLOW: A Feedback LOop FrameWork for Simultaneously Enhancing Recommendation and User Agents. *arXiv preprint arXiv:2410.20027* (2024).

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

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

[6] Luke Friedman, Sameer Ahuja, David Allen, Zhenning Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, et al. 2023. Leveraging large language models in conversational recommender systems. *arXiv preprint arXiv:2305.07961* (2023).

[7] Chongming Gao, Ruijun Chen, Shuai Yuan, Kexin Huang, Yuanqing Yu, and Xiangnan He. 2024. SPRec: Leveraging Self-Play to Debias Preference Alignment for Large Language Model-based Recommendations. *arXiv preprint arXiv:2412.09243* (2024).

[8] Yingqiang Ge, Shuchang Liu, Zuohui Fu, Juntao Tan, Zelong Li, Shuyuan Xu, Yunqi Li, Yikun Xian, and Yongfeng Zhang. 2024. A survey on trustworthy recommender systems. *ACM Transactions on Recommender Systems* 3, 2 (2024), 1–68.

[9] Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging large language models for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 1096–1102.

[10] 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).

[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 *European Conference on Information Retrieval*. Springer, 364–381.

[12] Jen-Hung Huang and Yi-Fen Chen. 2006. Herding in online product choice. *Psychology & Marketing* 23, 5 (2006), 413–428.

[13] Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. 2023. Recommender ai agent: Integrating large language models for interactive recommendations. *arXiv preprint arXiv:2308.16505* (2023).

[14] Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. 2024. Understanding the planning of LLM agents: A survey. *arXiv preprint arXiv:2402.02716* (2024).

[15] 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. In *Proceedings of the ACM on Web Conference 2024*. 4717–4726.

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

[17] Sunghwan Kim, Tongyoung Kim, Kwangwook Seo, Jinyoung Yeo, and Dongha Lee. 2024. Stop Playing the Guessing Game! Target-free User Simulation for Evaluating Conversational Recommender Systems. *arXiv preprint arXiv:2411.16160* (2024).

[18] Dongsheng Li, Jianxun Lian, Le Zhang, Kan Ren, Tun Lu, Tao Wu, and Xing Xie. 2024. *Recommender Systems: Frontiers and Practices*. Springer Nature.

[19] Jan Malte Lichtenberg, Alexander Buchholz, and Pola Schwöbel. 2024. Large language models as recommender systems: A study of popularity bias. *arXiv preprint arXiv:2406.01285* (2024).

[20] Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Jiongran Wu, Peng Zhang, Li Shang, and Ning Gu. 2023. Recommendation unlearning via matrix correction. *arXiv preprint arXiv:2307.15960* (2023).

[21] Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, and Ning Gu. 2022. Parameter-free dynamic graph embedding for link prediction. *Advances in Neural Information Processing Systems* 35 (2022), 27623–27635.

[22] Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, Li Shang, and Ning Gu. 2023. Personalized graph signal processing for collaborative filtering. In *Proceedings of the ACM Web Conference 2023*. 1264–1272.

[23] Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, Li Shang, and Ning Gu. 2023. Triple structural information modelling for accurate, explainable and interactive recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1086–1095.

[24] Jiahao Liu, Yiyang Shao, Peng Zhang, Dongsheng Li, Hansu Gu, Chao Chen, Longzhi Du, Tun Lu, and Ning Gu. 2024. Filtering Discomforting Recommendations with Large Language Models. *arXiv preprint arXiv:2410.05411* (2024).

[25] Sijia Liu, Jiahao Liu, Hansu Gu, Dongsheng Li, Tun Lu, Peng Zhang, and Ning Gu. 2023. Autoseqrec: Autoencoder for efficient sequential recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 1493–1502.

[26] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2024. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems* 36 (2024).

[27] Gustavo Mendonça Ortega, Rodrigo Ferrari de Souza, and Marcelo Garcia Manzato. 2024. Evaluating zero-shot large language models recommenders on popularity bias and unfairness: a comparative approach to traditional algorithms. *Anais Estendidos* (2024).

[28] Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2023. Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies. *arXiv preprint arXiv:2308.03188* (2023).

[29] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*. 1–22.

[30] Alessandro Petruzzelli, Cataldo Musto, Lucrezia Laraspata, Ivan Rinaldi, Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. 2024. Instructing and prompting large language models for explainable cross-domain recommendations. In *Proceedings of the 18th ACM Conference on Recommender Systems*. 298–308.

[31] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[32] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2010. Introduction to recommender systems handbook. In *Recommender systems handbook*. Springer, 1–35.

[33] Guy Shani and Asela Gunawardana. 2011. Evaluating recommendation systems. *Recommender systems handbook* (2011), 257–297.

[34] Tingjia Shen, Hao Wang, Jiaqing Zhang, Sirui Zhao, Liangyue Li, Zulong Chen, Defu Lian, and Enhong Chen. 2024. Exploring User Retrieval Integration towards Large Language Models for Cross-Domain Sequential Recommendation. *arXiv preprint arXiv:2406.03085* (2024).

[35] Wentao Shi, Xiangnan He, Yang Zhang, Chongming Gao, Xinyue Li, Jizhi Zhang, Qifan Wang, and Fuli Feng. 2024. Large language models are learnable planners for long-term recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1893–1903.

[36] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2024).

[37] Yubo Shu, Haonan Zhang, Hansu Gu, Peng Zhang, Tun Lu, Dongsheng Li, and Ning Gu. 2024. RAH! RecSys–Assistant–Human: A Human-Centered Recommendation Framework With LLM Agents. *IEEE Transactions on Computational Social Systems* (2024).

[38] Zuoli Tang, Zhaoxin Huan, Zihao Li, Xiaolu Zhang, Jun Hu, Chilin Fu, Jun Zhou, and Chenliang Li. 2023. One model for all: Large language models are domain-agnostic recommendation systems. *arXiv preprint arXiv:2310.14304* (2023).

[39] Ajay Krishna Vajjala, Dipak Meher, Ziwei Zhu, and David S Rosenblum. 2024. Cross-Domain Recommendation Meets Large Language Models. *arXiv preprint arXiv:2411.19862* (2024).

[40] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science* 18, 6 (2024), 186345.

[41] Lei Wang, Jingsen Zhang, Hao Yang, Zhi-Yuan Chen, Jiakai Tang, Zeyu Zhang, Xu Chen, Yankai Lin, Hao Sun, Ruihua Song, et al. 2024. User Behavior Simulation with Large Language Model-based Agents for Recommender Systems. *ACM Transactions on Information Systems* (2024).

[42] Xiaolei Wang, Xinyu Tang, Wayne Xin Zhao, Jingyuan Wang, and Ji-Rong Wen. 2023. Rethinking the evaluation for conversational recommendation in the era of large language models. *arXiv preprint arXiv:2305.13112* (2023).

[43] Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Yingzhen Yang. 2023. Recmind: Large language model powered agent for recommendation. *arXiv preprint arXiv:2308.14296* (2023).

[44] Zhefan Wang, Yuanqing Yu, Wendi Zheng, Weizhi Ma, and Min Zhang. 2024. Multi-Agent Collaboration Framework for Recommender Systems. *arXiv preprint arXiv:2402.15235* (2024).

[45] Jiafeng Xia, Dongsheng Li, Hansu Gu, Jiahao Liu, Tun Lu, and Ning Gu. 2022. FIRE: Fast incremental recommendation with graph signal processing. In *Proceedings of the ACM Web Conference 2022*. 2360–2369.

[46] Siyu Yuan, Kaitao Song, Jiangjie Chen, Xu Tan, Yongliang Shen, Ren Kan, Dongsheng Li, and Deqing Yang. 2024. Easytool: Enhancing llm-based agents with concise tool instruction. *arXiv preprint arXiv:2401.06201* (2024).

[47] Tianzi Zang, Yanmin Zhu, Haobing Liu, Ruohan Zhang, and Jiadi Yu. 2022. A survey on cross-domain recommendation: taxonomies, methods, and future directions. *ACM Transactions on Information Systems* 41, 2 (2022), 1–39.

[48] An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. 2024. On generative agents in recommendation. In *Proceedings of the 47th international ACM SIGIR conference on research and development in Information Retrieval*. 1807–1817.

[49] Guangping Zhang, Dongsheng Li, Hansu Gu, Tun Lu, Li Shang, and Ning Gu. 2024. Simulating News Recommendation Ecosystems for Insights and Implications. *IEEE Transactions on Computational Social Systems* (2024).

[50] Jizhi Zhang, Keqin Bao, Wenjie Wang, Yang Zhang, Wentao Shi, Wanhong Xu, Fuli Feng, and Tat-Seng Chua. 2024. Prospect Personalized Recommendation on Large Language Model-based Agent Platform. *arXiv preprint arXiv:2402.18240* (2024).

[51] Junjie Zhang, Yupeng Hou, Ruobing Xie, Wenqi Sun, Julian McAuley, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2024. Agentcf: Collaborative learning with autonomous language agents for recommender systems. In *Proceedings of the ACM on Web Conference 2024*. 3679–3689.

[52] 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).

[53] Yao Zhang, Yun Xiong, Dongsheng Li, Caihua Shan, Kan Ren, and Yangyong Zhu. 2021. Cope: Modeling continuous propagation and evolution on interaction graph. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2627–2636.

[54] Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2024. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501* (2024).

[55] Zijian Zhang, Shuchang Liu, Ziru Liu, Rui Zhong, Qingpeng Cai, Xiangyu Zhao, Chunxu Zhang, Qidong Liu, and Peng Jiang. 2024. LLM-Powered User Simulator for Recommender System. *arXiv preprint arXiv:2412.16984* (2024).

[56] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

[57] Yuyue Zhao, Jiancan Wu, Xiang Wang, Wei Tang, Dingxian Wang, and Maarten de Rijke. 2024. Let me do it for you: Towards llm empowered recommendation via tool learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1796–1806.

[58] Lixi Zhu, Xiaowen Huang, and Jitao Sang. 2024. How Reliable is Your Simulator? Analysis on the Limitations of Current LLM-based User Simulators for Conversational Recommendation. In *Companion Proceedings of the ACM on Web Conference 2024*. 1726–1732.

[59] Lixi Zhu, Xiaowen Huang, and Jitao Sang. 2024. A LLM-based Controllable, Scalable, Human-Involved User Simulator Framework for Conversational Recommender Systems. *arXiv preprint arXiv:2405.08035* (2024).