A Survey of LLM-Powered Agents for Recommender Systems: Taxonomy, Challenges, and Future Directions

Anonymized Author

Abstract

The rapid progress of large language models (LLM) has accelerated the use of LLM-powered agents in recommendation, enabling environment perception, autonomous task execution, and reasoning-based decision-making. However, a systematic overview of how agents are utilized within recommender systems is lacking. This paper surveys recent research on LLM-powered agents for recommender systems through four key dimensions: **WHERE** agents are deployed, **WHAT** agents are employed, **WHY** agents are utilized, and **HOW** agents are designed, offering practical guidance for LLM agent-based engineering applications. The **"WHERE"** perspective lists three types of strategies integrating LLM-powered agents with recommender systems, namely agents functioning as the recommender ("Agent as Rec"), agents assisting the recommender ("Agent for Rec"), and agents embedded within specific stages of recommendation pipeline ("Agent in Rec"). The **"WHAT"** perspective classifies existing works by the number of agents, mainly distinguishing LLM-powered single-agent systems from multi-agent systems. The **"WHY"** perspective groups existing works by their primary objectives into four categories, including improving general recommendation, addressing domain-specific recommendation, enhancing interactive recommendation, and evaluating recommender systems. The **"HOW"** perspective focuses on four core modules in agent, including Profile, Memory, Planning, and Action. Then, we identify six key challenges in integrating LLM-powered agents into recommender systems, including low efficiency, high computational cost, limited reliability, insufficient security, inadequate robustness, and the scalability and coordination issues of multi-agent integration. At last, we summarize the survey and discuss the future prospects. Furthermore, we maintain a curated

Preprint submitted to Engineering Applications of Artificial IntelligenceJanuary 22, 2026

list of relevant papers at Github Repository[1].

Keywords: Agent, recommender systems, Large Language Model, recommendation

1. INTRODUCTION

As the volume of information available on digital platforms continues to grow, recommender systems [1, 2, 3, 4] become increasingly important in alleviating information overload [5] and aligning content with user preferences [6, 7]. Their primary function is to mitigate information overload by predicting user preferences and recommending items or content likely to be of interest. Traditionally, these systems rely on collaborative filtering [8, 9], content-based filtering [10], or hybrid approaches [11], typically implemented using machine learning models trained on users' historical interactions [12]. Despite their success, these methods face several challenges, including cold-start issues for new users or items, difficulty in modeling subtle or evolving preferences, limited explainability [13], and the inability to support interactive, real-time conversations [14].

Meanwhile, natural language processing (NLP) has advanced substantially with the emergence of large language models (LLMs). In particular, models such as GPT-3, GPT-4 [15] and their successors have demonstrated strong capabilities in understanding, generating, and reasoning with human language. They exhibit abilities previously associated with human cognition, including following complex instructions, applying commonsense reasoning, acquiring world knowledge, and engaging in coherent dialogues [16, 17]. These advances have motivated exploration beyond conventional NLP tasks, including information retrieval [18, 19] and recommender systems [20, 21, 22]. For example, Li et al. [23] used LLMs as the core component of generative recommender systems to perform recommendations directly. Similarly, Sakurai et al. [24] proposed a novel approach that leverages LLMs as knowledge-graph (KGs) inferencers to enhance recommender systems.

However, inherent limitations of LLMs pose challenges in recommender applications, including constrained memory capacity [25], insufficient task decomposition and planning [26], and limited controllability [27]. To address these drawbacks, researchers integrate LLMs with agents—originally

[1] `https://github.com/dongjingWANG/Agent4Rec_Survey/`

studied in artificial intelligence and distributed systems, to create a new framework namely LLM-based (or LLM-powered) recommendation agents [28, 29]. LLM-based agents mitigate these limitations by leveraging advanced comprehension, reasoning, and decision-making capabilities. Their capacity to process and interpret large-scale information is expected to drive significant transformation in information retrieval [30]. Unlike static models, these agents act as autonomous entities that perceive their environment through natural language, reason about tasks, and plan actions. This enables dynamic adaptation to evolving inputs and user requirements [31]. By integrating the comprehension, reasoning, and interaction capabilities of LLMs, the agent paradigm shows substantial potential to address limitations of existing recommender systems. Specifically, agents are capable of interpreting complex natural language queries, leveraging broad external knowledge, and conducting multi-turn conversations to refine recommendations [32]. Compared with previous works that primarily target specific aspects of agents in recommender systems, this paper discusses a comprehensive framework for systematically studying the integration of LLM-based agents and recommender systems, as shown in Figure 1. The analysis specifically revolves around four fundamental aspects:

- In the **"WHERE"** perspective, we explore the integration strategies employed by LLM-powered agents within recommender systems, focusing on three distinct paradigms: Agent as Rec, Agent for Rec, and Agent in Rec. Firstly, in "Agent as Rec" paradigm, the agent serves as the primary decision-making entity, bearing the main responsibility for generating recommendations. Secondly, in "Agent for Rec" paradigm, the agent serves an auxiliary role, enhancing recommendation performance by augmenting the data utilization in the recommender system. Lastly, in "Agent in Rec" paradigm, the agent is embedded into specific stages of the pipeline to enhance performance by interacting with existing components.

- In the **"WHAT"** perspective, we classify systems by the number of agents: LLM-powered single-agent systems and LLM-powered multi-agent systems. Note that this classification is essential for understanding system complexity, scalability, and collaborative functionality. We further analyze the task allocation across agents to elucidate architectural design.

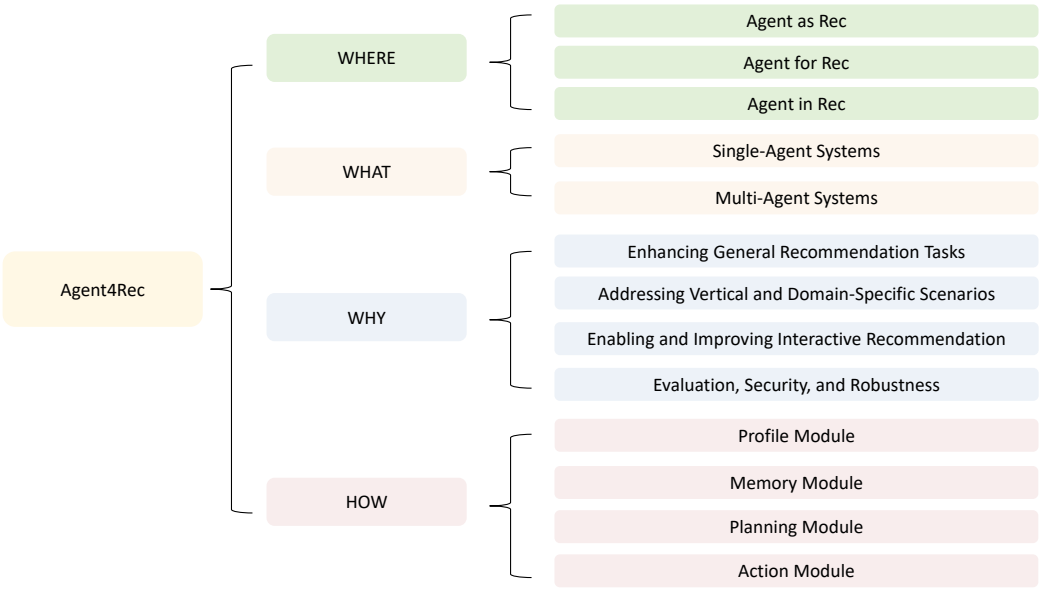

Figure 1: The decomposition of our core research question about adapting LLM-powered recommender systems. We analyze the question from four orthogonal perspectives: (1) where agents are deployed, (2) what agents are employed, (3) why agents are utilized, and (4) how agents are designed.

- In the **"WHY"** perspective, we examine motivations by linking the challenges agents address to their application domains. We categorize them into four areas: enhancing general recommendation, addressing vertical and domain-specific problems, facilitating and refining interactive recommendation, and evaluating recommender systems.

- In the **"HOW"** perspective, we examine optimization strategies grounded in modular agent design, focusing on the Profile, Memory, Planning, and Action modules. This perspective helps to reveal the internal mechanisms underpinning agent intelligence.

This survey investigates the utilization of LLM-powered agents within recommender systems, and it is different with recent surveys on LLM-powered agents for recommendation. Especially, we provide a thorough analysis of the role of intelligent agents in recommender systems, with an emphasis on integration strategies. By contrast, existing surveys overlap with ours in at most two dimensions and either do not adequately address integration strategies or application scenarios, or examine them from alternative perspectives.

Table 1: The comparison between this work and existing surveys.

Paper	Where	What	Why	How	Pipeline	Highlights
Zhu et al. [33]		✓		✓	(1) Agents for RS (2) RS for Agents (3) Trustworthy Agents and RS	Examines the two-way optimization of LLM agents and recommender systems.
Peng et al. [34]		✓		✓	(1) Recommender approaches (2) Interaction approaches (3) Simulation approaches	Identifies three key research paradigms and details their architectures and evaluations.
Zhang et al. [30]		✓		✓	(1) User Interaction (2) Item Representation (3) System Integration (4) Environment Simulation	Emphasizes the comprehensive improvement of recommender systems by LLM agents through enhanced semantic understanding, task decomposition, and user engagement.
Ours	✓	✓	✓	✓	(1) Where agents are deployed (2) What agents are employed (3) Why agents are utilized (4) How agents are architected	Proposes a "Where, What, Why, How" (4W) framework for comprehensive analysis of existing research.

Detailed comparisons between our survey and existing works are summarized in Table 1.

The contributions of this paper can be summarized as follows:

- We propose a systematic taxonomy that classifies the integration of LLM-powered agents into three paradigms: Agent as Rec (primary recommendation entity), Agent for Rec (auxiliary enhancer), and Agent in Rec (embedded component for optimization). This taxonomy provides architectural guidance in research and practice.

- We design a thorough four-dimensional analytical framework—WHERE (integration strategy), WHAT (agent configuration), WHY (problem domain), and HOW (component optimization)—and systematically relate representative system designs to these dimensions.

- In order to facilitate empirical research, we have assembled a collection of more than 50 closely related research papers published over the last three years, focusing on the integration of LLM-powered agents

within recommender systems. These papers are available on Github Repository[2].

The remainder of this paper is organized as follows. Section 2 overviews foundational concepts in recommender systems, large language models, LLM-powered agents, and their use in recommendation. Section 3 conducts a comprehensive review of existing works along four dimensions: **"WHERE", "WHAT", "WHY",** and **"HOW".** And it discusses relevant developmental pathways. In Section 4, examines challenges for LLM-powered agents in recommendation, including **high computational costs, inefficiency, limited reliability, inadequate security, limited robustness,** and **scalability and coordination issues in multi-agent integration**. Finally, Section 5 concludes the paper and outlines future directions for LLM-powered agents in recommender systems.

2. BACKGROUND

A structured overview is essential to contextualize the significance and potential of LLM-powered agents in recommender systems. This begins with the fundamental tasks and core concepts of recommender systems themselves. Next, it is crucial to recognize the inherent advantages of LLMs and the specific characteristics that make them suitable for recommendation tasks, including an understanding of their generative nature. Furthermore, a clear distinction must be drawn between using LLMs and LLM-powered agents within a system architecture. Finally, exploring the practical applications of these agents is key—specifically, how they enhance recommendation performance, how their state and action sets are defined, how reward functions are formulated for training, and what prevalent strategy learning approaches are employed.

2.1. Recommender Systems

Recommender systems [35, 36, 37, 38] are specialized information retrieval systems that analyze user preferences based on profiles and behavior. The principal aim of these systems is to facilitate the interaction between users and items that they are expected to find pertinent or captivating. Let $\boldsymbol{U} = \{u_1, u_2, \ldots, u_m\}$ represent the set of users and $\boldsymbol{I} = \{i_1, i_2, \ldots, i_n\}$ denote the

[2] `https://github.com/dongjingWANG/Agent4Rec_Survey/`

set of items. The interactions between the users in $\boldsymbol{U}$ and the items in $\boldsymbol{I}$ can be encapsulated in a matrix $\boldsymbol{D} \in \mathbb{R}^{m \times n}$, The primary objective is to develop a preference function f based on parameters θ that effectively predict user preferences. We can achieve this goal by minimizing the loss function $\mathcal{L}$.

$$\min_{\theta} \sum_{(u,i) \in \mathcal{D}} \mathcal{L}\left(f_{\theta}(u,i), g_{u,i}\right) \tag{1}$$

where $f_{\theta}(u,i)$ represents the predicted preference and $g_{u,i}$ denotes the ground truth interaction. In the early stages, these systems predominantly employed fundamental statistical methodologies or matrix factorization techniques to utilize historical interaction data for the purpose of forecasting user ratings or item purchases [39].

The advent of deep learning has led to the integration of sophisticated neural network architectures within recommender systems, enabling the modeling of user-item interactions, sequential patterns, and comprehensive auxiliary information, such as item characteristics and user demographics [40]. Approaches utilizing Recurrent Neural Networks (RNNs) and Transformer networks [41] have been employed to effectively capture sequential user behavior, resulting in significant progress in the field of sequential recommendation. For instance, BERT4Rec leverages the Bidirectional Encoder Representations from the Transformer architecture to model complex item relationships within user behavior sequences, thereby improving recommendation accuracy [42].

Despite the advancements brought by deep learning to recommender systems, they continue to encounter several challenges, including data sparsity [43, 44], cold-start issues related to recommending items to new users or introducing new items [45, 46], as well as concerns regarding explainability and robustness against malicious attacks. Furthermore, these systems struggle to effectively capture the complexities of evolving user preferences and the influence of external contexts.

2.2. Large Language Models

Language models [47, 48, 49, 50] are developed to engage in probabilistic modeling of natural language, facilitating the prediction of word tokens based on a defined textual context. The core mechanism of an LLM is to learn the probability distribution of a sequence of tokens $\boldsymbol{X} = \{x_1, x_2, \ldots, x_T\}$. This is achieved by factorizing the joint probability of the sequence into a product

of conditional probabilities, where the likelihood of each token is dependent on the tokens that precede it. This autoregressive process is captured by the following formula, where the model's parameters, denoted by θ, are optimized over a vast text corpus to maximize this probability.

$$P(X;\theta) = \prod_{t=1}^{T} P(x_t|x_1, x_2, ..., x_{t-1};\theta) \tag{2}$$

Currently, LLMs predominantly utilize the Transformer architecture and are trained on extensive datasets encompassing both text and code. These models are distinguished by their immense scale, often containing billions or even trillions of parameters. This massive capacity allows them to learn from vast training datasets, enabling them to identify intricate linguistic patterns, grasp deep semantic connections, and internalize a rich repository of world knowledge. Their proficiency in understanding and generating human-like text makes them ideal tools for enabling natural language interaction [51]. Capitalizing on their extensive knowledge and robust reasoning capabilities, modern LLMs demonstrate exceptional performance in zero-shot and few-shot scenarios, where they can comprehend and execute new tasks without task-specific training [52, 53]. This aptitude stems from the instruction-following behaviors acquired during their comprehensive pre-training phase. Additionally, the capabilities of LLMs are not confined to their internal knowledge. By integrating with external tools, they can access real-time information and interface with outside systems, thereby transcending the limitations of their static knowledge base [54]. This feature is particularly crucial for developing autonomous agents in recommender systems, as it empowers these agents to interact with real-world or domain-specific environments [55]. In summary, the combination of vast knowledge, powerful reasoning, and the ability to interact with external environments allows LLMs to accurately assess user preferences by analyzing behavioral sequences, leading to significant improvements in recommendation quality and effectiveness [56, 57, 58, 59].

2.3. LLM-Powered Agents

An agent is conventionally characterized as an autonomous entity that can perceive its surroundings, make decisions, and execute actions to fulfill specific objectives. LLM-powered agents utilize the LLM as their cognitive core for reasoning, planning, and both understanding and generating natural language. These agents are designed to demonstrate enhanced **autonomy**

and **goal-directed behavior** in comparison to basic LLM prompts. Unlike simpler LLM applications, these agents go beyond basic prompt-response interactions by actively interpreting their surroundings, making independent decisions, and executing actions to achieve specified objectives through iterative processes that involve planning, memory, and the use of external tools. Accordingly, an agent may be conceptually represented as:

$$A_{LLM} = \big(P, T, M, \Omega\big), \tag{3}$$

where:

- $\boldsymbol{P}$ denotes the agent's perception module, which enables it to autonomously sense the external environment for self-regulation. In recommender systems, the standard workflow typically encompasses data acquisition, preprocessing, and feature engineering. Within this framework, the Profile module and Memory module can be conceptualized as integral components of the sensor mechanism.

- $\boldsymbol{T}$ denotes a set of callable tools, such as functions and APIs. These tools allow an agent to augment its environmental perception, access specialized capabilities beyond its core model, including object detection and database queries, and perform actions to alter the state of the environment. In multi-agent systems, agents utilize shared tools and communication protocols to coordinate their efforts and accomplish complex collaborative tasks.

- $\boldsymbol{M}$ constitutes a large language model that has undergone instruction tuning or alignment. It is capable of text generation and reasoning through natural language. By comprehending user instructions, it can perform task decomposition and multi-step planning, subsequently generating corresponding action decisions, such as requests to invoke specific tools or instructions to query memory. An outer agent framework is responsible for parsing and executing these decisions, thereby orchestrating tools and memory systems to accomplish complex tasks.

- $\boldsymbol{\Omega}$ refers to the agent's memory and storage system, which maintains dialogue history, internal agent states such as current goals and subtask progress, and behavioral policies learned from experience or defined externally. In practice, this system typically employs a combination of vector databases, traditional relational or key-value databases, and

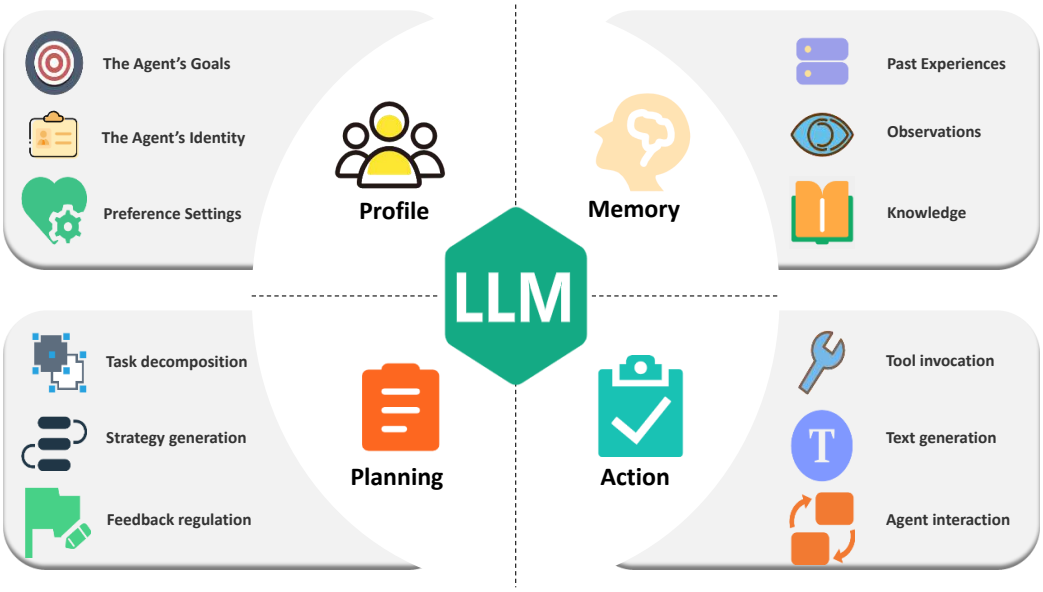

Figure 2: Illustration of Agent Components and Corresponding Functions.

configuration files like JSON to store perceived environmental data, user demographics, and preferences. This stored information provides the necessary context for the model's planning and decision-making processes, thereby facilitating a complete perceptual-decisive-active-mnemonic loop.

A widely accepted conceptual framework for LLM-powered agents, shown in Figure 2, outlines several key components. It is crucial to note, however, that this framework is not definitive. The definition of an agent is based on its inherent autonomous capabilities, not a fixed architecture. Modular systems represent just one implementation within the broader agent concept. Therefore, some studies rightly claim the use of agent technology in their frameworks, despite not enumerating these specific modules.

- **Profile Module**: The identity of the agent includes its characteristics, goals, and potentially its knowledge or preferences. In the context of recommender systems, this may relate to the specific role adopted by either the user or the agent [60, 61, 62, 63].

- **Memory Module**: Memory refers to the capacity to retain past experiences, observations, interactions, and knowledge acquired over time.

It can be categorized into short-term memory, often described as a context window, and long-term memory, which may involve an external knowledge base or vector database. Additionally, memory may incorporate processes for reflection and consolidation [64].

- **Planning Module**: The methodology employed by the agent to ascertain a series of actions aimed at accomplishing a specific objective encompasses the decomposition of intricate tasks into manageable subtasks, the evaluation of possible outcomes, and the modification of strategies in response to feedback [65, 66, 67].

- **Action Module**: The implementation of procedures may encompass the generation of textual content, the utilization of external resources (such as search engines or conventional recommender system APIs), engagement with other agents, or the simulation of specific behaviors [68, 69].

These components enable agents to exhibit more sophisticated behaviors, including self-reflection, experiential learning, and complex problem-solving. The application of this agentic paradigm to recommender systems promises to create more intelligent, adaptive, and user-centric recommendation experiences. The four modules should not be viewed as independent entities; instead, they constitute a closely integrated and interdependent system that is crucial for the effective functioning of an LLM-powered agent.

2.4. LLM-Powered agents for Recommendation

The primary rationale for integrating LLM-powered agents into recommender systems lies in their ability to transcend conventional static recommendation frameworks, thus enabling the development of genuinely dynamic, adaptive, and interactive systems [70, 71, 72]. These agents possess the capacity to engage in sophisticated cognitive processes, analyze intricate tasks, and iteratively refine strategies in response to real-time user feedback and environmental changes, which markedly enhances personalization and responsiveness. Furthermore, LLM-powered agents are adept at assessing complex, multifaceted user inquiries and decomposing them into manageable subtasks, thereby improving execution efficiency. For example, in travel recommendations, an agent can break down a complex request into multiple elements, including destination selection, itinerary planning, flight booking, and budget assessment, ultimately generating a comprehensive report. This

capability is essential for effectively tackling the challenges associated with complex information retrieval and recommendation.

To operationalize an agent's dynamic decision-making process in a recommendation scenario, it is typically modeled through the framework of a Partially Observable Markov Decision Process (POMDP). This approach requires defining three core components: a state space $\boldsymbol{S}$, an action space $\boldsymbol{A}$, and a reward function $\boldsymbol{R}$. The state space $\boldsymbol{S}$ serves to represent the environmental information of an agent within a recommender system, which primarily includes user profiles $\boldsymbol{profile}_u$, historical interactions $\boldsymbol{H}_t$, the current context $\boldsymbol{c}_t$, and mechanisms for handling uncertainty $\boldsymbol{b}_t$. The state $\boldsymbol{s}_t$ at time step $\boldsymbol{t}$ can be represented in an abstract form as

$$s_t = f(\text{profile}_u, H_t, c_t, b_t). \tag{4}$$

In a POMDP, the agent has access to only partial information. As a result, the state is characterized by $\boldsymbol{b}_t \in \Delta(S)$, where $\boldsymbol{\Delta}(S)$ denotes the probability distribution over the state space.

In contrast to the state space, the action space is defined with greater diversity, influenced by the granularity of actions and the distinct roles of each agent. From a broader perspective, this behavior is linked to the agent's Action module and can be categorized into discrete and hierarchical actions. The discrete action space is denoted as $\boldsymbol{A}_d = \{a_1, a_2, \ldots, a_N\}$, where $\boldsymbol{a}_i$ corresponds to either a recommended item $\boldsymbol{i}$ or discrete actions (such as "click"). In contrast, the hierarchical action space is defined as $\boldsymbol{A}_h = A_{macro} \times A_{micro}$, where macro-level actions $\boldsymbol{a}_{macro}$ (e.g., "explore") are chosen first, followed by micro-level actions $\boldsymbol{a}_{micro}$ (e.g., "recommend item A"). The specific action set generally encompasses item recommendations, explanation generation, user response simulation (e.g., like/dislike), and system actions (e.g., search, exit).

These concepts are closely tied to reinforcement learning (RL) in agent-based systems. In recommender systems, when an agent improves its strategy through learning, it often employs RL methods, which are particularly suited for sequential decision-making tasks that involve balancing immediate user feedback with long-term engagement objectives. In contrast, contextual learning—often inherent in LLMs—excels at leveraging semantic patterns and real-time user signals to deliver situation-aware recommendations, though it typically operates in a more static, single-turn manner. Some approaches also leverage the inherent contextual learning ability of LLMs, utilizing mechanisms such as feedback loops and multi-agent collaboration. This

section focuses on key RL components for agents—specifically the POMDP framework and reward functions—which are essential for understanding the dimensional categorization introduced in Section 2. In recommender systems, reward function design is crucial for optimizing recommendation strategies. Rewards can be classified by timing and source. Timing-based categories include immediate rewards (e.g., clicks), short-term rewards (e.g., session engagement), and long-term rewards (e.g., user retention). Source-based categories comprise explicit user feedback (e.g., ratings), implicit behavioral signals (e.g., dwell time), business-aligned metrics (e.g., conversion rates), and strategic rewards (e.g., diversity control). This multidimensional framework supports the design of balanced reward systems that incorporate both immediate feedback and long-term value. The corresponding mathematical formulation is provided below.

$$\mathcal{R}_{\text{rec}}(s_t, a_t) = \begin{cases} \lambda_1 \cdot r_{\text{im}}(s_t, a_t) + \lambda_2 \cdot r_{\text{pro}}(s_t, a_t) \\ r_{\text{short}} \\ r_{\text{long}} \end{cases} \tag{5}$$

In this context, $\boldsymbol{r}_{im}$ represents the click reward (immediate), $\boldsymbol{r}_{pro}$ denotes the diversity reward (process), $\boldsymbol{r}_{short}$ indicates the purchase reward (short-term), $\boldsymbol{r}_{long}$ corresponds to the retention reward (long-term), and $\boldsymbol{\lambda}_1, \lambda_2$ serves as the weight parameter.

3. RECENT ADVANCES OF LLM-POWERED AGENTS FOR RECOMMENDATION

In Table 2, we present the 58 papers collected for this review, listing their publication years, journals, and main contributions. We can systematically analyze the current landscape by examining where agents are applied, what types of agent systems are being developed, why they are designed, and how they are optimized.

Table 2: A summary of the collected papers.

Category	Methods	Year	Venue	Key Contribution
	AutoConcierge [73]	2024	PADL	LLM-ASP Integrated Dialog Comprehension

A summary of the collected papers

Table 2 – Continued from previous page

Category	Methods	Year	Venue	Key Contribution
Agent as Rec	Rec4Agentverse [74]	2024	arXiv	Agent Collaborative Evolution Framework
	PMS [75]	2024	arXiv	Multi-agent multimodal synergy
	MACRS [66]	2024	arXiv	Multi-agent planning with feedback-aware reflection
	MACRec [76]	2024	SIGIR	Modular multi-agent interoperability
	RecAI [77]	2024	WWW	Multifaceted LLM-integration framework
	Shuang et al. [78]	2024	KDD	Generative Trajectory-Driven Data Efficiency
	InteRecAgent [67]	2024	ACM TRANS	Tool-Augmented LLM-Brain Recommendation Agent
	RecMind [79]	2024	NAACL	Self-Inspiring Path Integration
	MAS4POI [80]	2024	PAKDD	Cold-start resilient agent architecture
	TAIRA [81]	2025	arXiv	Thought Pattern Distillation
	ChatCRS [82]	2025	NAACL	Knowledge-goal dual-agent orchestration
	MADREC [83]	2025	arXiv	Aspect-Driven Re-Ranking
	PUMA [84]	2025	WWW	Memory-Augmented Personalized Alignment
	MMAgentRec [85]	2025	Sci. Rep	Cross-modal self-reflective agent framework
	CDA4Rec [86]	2025	arXiv	Cloud-Device Dual-Agent Planning
	STARec [87]	2025	CIKM	Anchored Reinforcement Training
	REMI [88]	2025	KDD	Causal Schema Memory
	RouteLLM [89]	2025	arXiv	Hierarchical Agent-Router Fusion
	AgentDR [90]	2025	arXiv	Dual S&C Reranking
	MARC [91]	2025	CIKM	Agentic Graph RAG
	MATCHA [92]	2025	arXiv	Role-specialized multi-agent orchestration
	iEvaLM [93]	2023	EMNLP	Interactive LLM-simulated evaluation framework
	CORE [94]	2023	NeurIPS	Uncertainty-minimized plug-and-play integration

A summary of the collected papers

Table 2 – Continued from previous page

Category	Methods	Year	Venue	Key Contribution
Agent for Rec	Agent4Rec [63]	2024	SIGIR	Generative-Agent Recommendation Simulator
	RAH [95]	2024	TCSS	Human-centered LLM-agent alignment via learn-act-critic loop
	PEARL [96]	2024	EMNLP	Exemplar-augmented multi-agent preference navigation
	USimAgent [68]	2024	SIGIR	Context-integrated reasoning-action simulation
	Zhang et al. [97]	2024	AAAI	Explicit Preference Modeling with Ensemble Reliability
	CheatAgent [98]	2024	KDD	LLM-agent-driven perturbation optimization
	PEPPER [99]	2024	arXiv	Preference-guided interaction design
	ToolRec [69]	2024	SIGIR	Surrogate-user-guided recommendation exploration
	Yoon et al. [100]	2024	ACL	Task-structured evaluation protocol for simulator-human alignment
	SimpleUserSim [101]	2024	WWW	Cognitive-fidelity simulation enhancement
	SUBER [102]	2024	arXiv	LLM-integrated synthetic environment simulation
	CSHI [103]	2024	WWW	Plugin-orchestrated behavior modulation
	TextSimu [104]	2024	SIGIR	Popularity-simulated text poisoning attack mechanism
	Lusifer [105]	2025	IEEE	Preference-transparent dynamic simulation
	iALP [106]	2025	WSDM	LLM-preference-distilled adaptive policy exploration
	iAgent [107]	2025	ACL	Agent-mediated preference shielding
	RecAgent [108]	2025	ACM TRANS	Programmable human behavior simulation
	CARTS [109]	2025	arXiv	Refinement Circle Arbitration
	RecUserSim [60]	2025	WWW	Bounded-rationality-based action-response emulation
	DiscomfortFilter [110]	2025	WWW	Conversational preference masking framework

A summary of the collected papers

Table 2 – Continued from previous page

Category	Methods	Year	Venue	Key Contribution
	CreAgent [111]	2025	arXiv	Information-asymmetry-aware creator simulation
	Chirag Shah et al. [61]	2025	arXiv	Multi-session dynamic adaptability framework
	DrunkAgent [112]	2025	arXiv	Transferable-imperceptible memory perturbation
	RuleAgent [62]	2025	arXiv	Reflection-enhanced Rule Discovery Framework
Agent in Rec	CSA [113]	2023	SIGIR	State-augmented generalization boost
	AgentCF [114]	2024	WWW	User-item agent co-learning
	Jie Wang et al. [115]	2024	SIGIR	LLM-simulated Environment with Triple-function Feedback
	BiLLP [116]	2024	SIGIR	Hierarchical macro-micro planning mechanism
	AFL [117]	2024	SIGIR	Bidirectional Agentic Synergy Framework
	KGLA [118]	2024	arXiv	KG-path-integrated rationale enhancement
	VRAgent-R1 [119]	2025	arXiv	Dual-Agent Reinforcement Framework
	ARAG [120]	2025	SIGIR	Semantic Alignment and Feedback-Driven Ranking
	AgentCF++ [121]	2025	SIGIR	Interest-group-constrained popularity propagation
	RPP [122]	2025	ACM TRANS	Sentence-level RL Search

3.1. Where Agents Are Applied: Integration Strategies

In this section, we examine the strategies employed for the integration of contemporary LLM-powered agents within recommender systems. As illustrated in Figure 3, we can categorize these integration strategies into three primary paradigms: Agent as Recommender, Agent for Recommendation, and Agent in Recommendation. Each paradigm, in turn, comprises several specific subcategories along with representative studies, as depicted in Figure 4. This classification facilitates a deeper understanding of the multi-faceted roles that agents assume and the architectural consequences of their implementation. It is essential to clarify the distinction between the "Agent

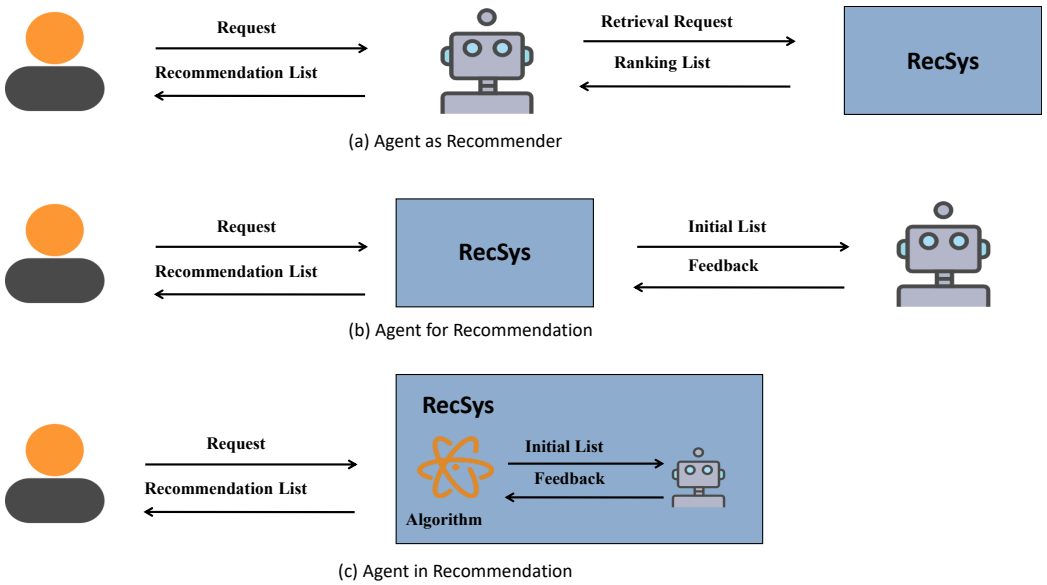

Figure 3: (a) **Agent as Rec**: The user requests the agent to perform the core recommendation task. The agent uses tools like retrieval and ranking models to obtain and rank candidate items based on user profiles and context, returning a recommendation list. (b) **Agent for Rec**: The user requests the RecSys, which generates an initial recommendation list based on historical behavior. This list is sent to the agent for optimization, and the RecSys updates it based on feedback. (c) **Agent in Rec**: The user requests recommendations from the RecSys, which generates an initial list. The agent analyzes this list and provides feedback, collaborating with the recommender system to improve the accuracy of the final recommendations.

as Rec" and "Agent in Rec" paradigms. The primary criterion centers on whether the agent occupies a central role within the overall recommender system. If an agent interacts with multiple components—seemingly embedded within the system—yet fundamentally acts as a core coordinator, it falls under the "Agent as Rec" paradigm. The defining characteristic is not independence but rather whether the agent serves as the principal recommender or managerial entity.

3.1.1. Agent as Recommender

In this paradigm, the LLM-powered agent acts as the primary engine responsible for generating recommendations or guiding the user through the recommendation process. The agent leverages its inherent capabilities, such as natural language understanding, reasoning, planning, and potentially ex-

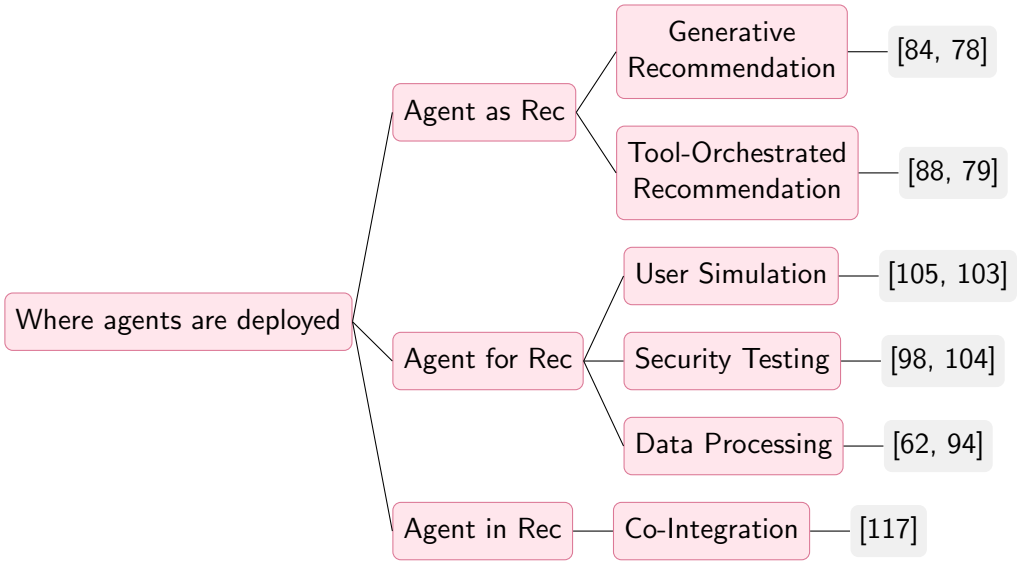

Figure 4: A Detailed Taxonomy of Agent-Recommender System Integration.

ternal knowledge or tools, to directly interact with the user and provide personalized suggestions. As shown in Figure 4, the "Agent as Recommender" paradigm comprises two distinct categories: Generative Recommendation and Tool-Orchestrated Recommendation. The conceptual framework for this taxonomy is further illustrated in Figure 5. Two distinct paradigms must be clearly differentiated: one employs the internal components of an intelligent agent or lightweight API interfaces to generate recommendations, while the other relies on independent systems or models that function as task orchestrators to fulfill recommendation objectives. Both paradigms serve as the primary recommendation mechanism within a system; however, the former operates through intrinsic capabilities, potentially augmented by simple APIs, whereas the latter acts more as a supervisory entity—akin to a manager in a multi-agent system—by coordinating traditional recommendation models or standalone systems to execute tasks.

Within the "Generative Recommendation" sub-paradigm, agents utilize their internally integrated capabilities to produce recommendations without requiring external model invocations. Multiple frameworks exemplify this approach through their autonomous generative reasoning mechanisms. For instance, PUMA [84] employs a dynamic user memory repository and task-aware retrieval to generate personalized recommendations in a fully au-

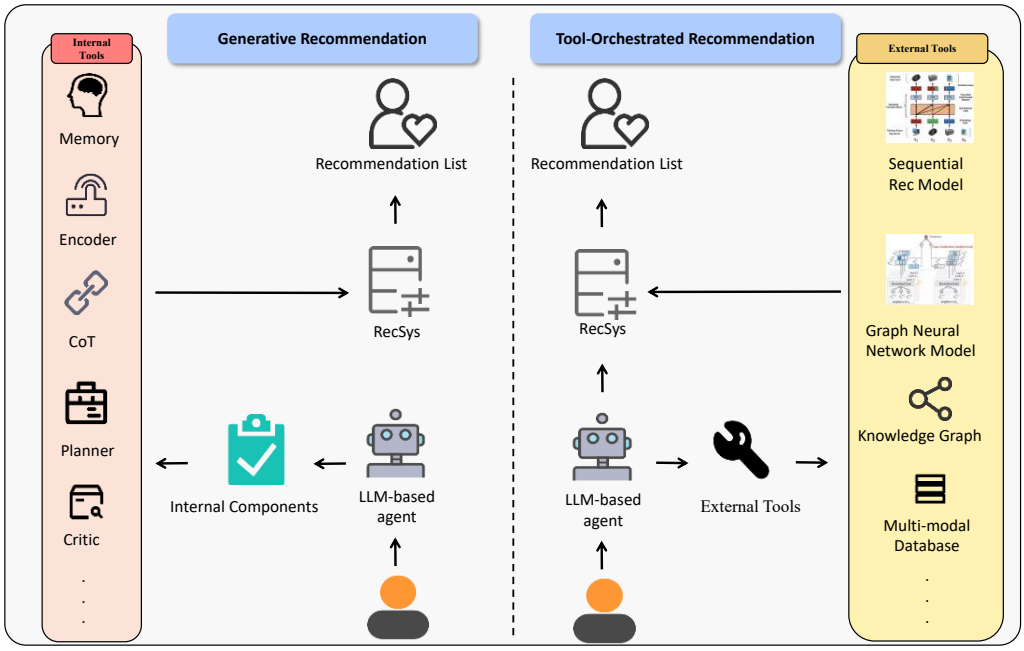

Figure 5: (a) **Generative Recommendation**: The agent invokes its internally integrated core capabilities, which can be directly utilized without the need for external service calls. (b) **Tool-Orchestrated Recommendation**: The agent invokes independent systems or models (such as SASRec [123], LightGCN [124], etc.) that it does not inherently possess but can access through APIs or interfaces.

tonomous manner. Shuang et al. [78] simulate user preferences via LLM-generated trajectories and refine recommendation strategies through Direct Preference Optimization [125], achieving notable data efficiency through endogenous learning. ChatCRS [82] implements specialized agents for knowledge retrieval, goal planning, and response generation to form an end-to-end generative workflow. A particularly comprehensive embodiment of this paradigm is MACRec [76], which establishes a multi-agent system where specialized components collaborate directly to address complex recommendation tasks. The framework integrates a Manager agent for coordination, User and Item Analysts for profiling, a Reflector for iterative refinement, a Searcher for knowledge retrieval, and an Interpreter for parsing conversational input. Through coordinated reasoning and iterative cycles guided by the Manager agent, the system autonomously handles various recommendation tasks, such as rating prediction, sequential recommendation, and conversational recom-

mendation. This integrated approach, which combines multi-role collaboration with systematic task decomposition, allows MACRec to significantly improve recommendation accuracy, enhance explainability, and increase contextual adaptability, all while maintaining full operational autonomy through its internal capabilities.

Additional frameworks further demonstrate the versatility of this generative paradigm. MACRS [66] deploys three agent types for inquiry, chit-chat, and recommendation tasks, with a dedicated planner agent selecting responses through internal evaluation. PMS [75] assigns specialized agents to recommendation tasks and optimizes outputs through built-in iterative feedback mechanisms. MATCHA [92] integrates multiple generative agents covering intent analysis, candidate generation, ranking, re-ranking, explanation, and safety functions, enabling complex query processing within a unified architecture. MADREC [83] leverages dynamically constructed multi-aspect user and item profiles, which undergo aspect-driven re-ranking to prioritize optimal candidates, subsequently processed through LLM-powered chain-of-thought reasoning for final selection. STARec [87] employs a dual-process agent architecture that integrates fast thinking for immediate ranking and slow thinking for chain-of-thought reasoning. Through iterative self-reflection and anchored reinforcement training, it achieves fully internal recommendation generation without external tool invocation.

Beyond **direct recommendation generation**, agents can also act as recommenders by **orchestrating RS tools**. The "Tool-Orchestrated Recommendation" sub-paradigm represents a distinct approach within agent-based recommender systems, where agents coordinate external models and systems through structured interfaces to enhance their capabilities. Unlike systems relying solely on internal reasoning, these frameworks strategically integrate specialized tools such as SASRec [123], LightGCN [124], or SQL-based retrievers through API calls to accomplish complex recommendation tasks.

AgentDR [90] harnesses traditional recommendation models as tools by delegating full-ranking tasks to them, while dynamically orchestrating their outputs through LLM-driven personalized tool suitability assessment and intent-aware reranking based on implicit substitute-complement relationships. RecMind [79] utilizes self-inspiration planning to decompose tasks and employs multi-path reasoning, accessing personalized memories through SQL tools and prioritizing items using external models like SASRec. This orchestration enables the generation of balanced recommendation lists that

optimize both relevance and diversity. InteRecAgent [67] demonstrates tool orchestration through its direct utilization of specialized tools in response to explicit user requests. The system processes criteria such as genre or price through SQL queries to filter candidates, then applies ranking mechanisms incorporating user profiles to generate personalized suggestions. REMI [88] extends this approach through a causal schema memory architecture where an LLM orchestrator integrates multiple external components, including a personal causal knowledge graph, a reasoning engine, and a schema-based planner to produce contextually grounded recommendations. Further illustrating this paradigm, CDA4Rec [86] operates as a dual-agent framework that orchestrates external sequential models including SASRec and SURGE. This enables structured user modeling and efficient inference through cloud-device collaboration. Complementing these systems, RecAI [77] provides a comprehensive toolkit where a recommender AI agent coordinates with specialized language models and knowledge plugins to enhance recommendation processes. These frameworks collectively demonstrate how agents can effectively leverage external systems and models through structured orchestration, forming a robust sub-paradigm that significantly expands the capabilities of agent-based recommender systems.

The "Agent as Rec" paradigm offers distinct advantages, including superior interactivity, the capability to address complex and novel problems (e.g., in zero-shot scenarios), integration of external knowledge, the ability to provide explanations, and enhanced user engagement through human-like interaction. This approach positions the agent as the central component of the recommender system, enabling end-to-end control, coherent reasoning processes, and significant flexibility—particularly suited to complex interactive settings. However, a key limitation lies in its potentially high computational demands and its reliance on the agent's internal capabilities, such as content reasoning or tool orchestration, which may restrict specialization in certain domains.

3.1.2. Agent for Recommendation

In this paradigm, the LLM-powered agent does not directly provide recommendations to the end-user but instead serves a function that supports, evaluates, or improves the recommender system itself. This includes roles such as simulating user behavior for training or evaluation, attacking the system to test its robustness, or assisting in data processing and analysis for RS development. The "Agent for Recommendation" paradigm includes

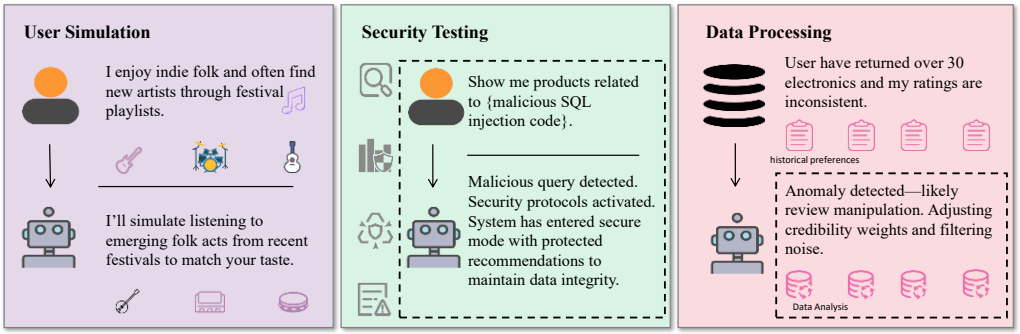

Figure 6: Illustration of Different Method Objectives. We classify existing methods into the following three categories: (1) User Simulation; (2) Security Testing; (3) Data Processing.

three main approaches: a) user simulation, b) security and robustness testing, and c) data processing and analysis tasks. Figure 6 summarizes representative cases corresponding to the three primary methodologies. The primary methodology discussed here, while also operating from a "WHY" dimension, diverges in its perspective. This study investigates the "WHY" from a pragmatic standpoint, emphasizing application domains and practical implementation. In contrast, the main approach referenced explores the issue on a more abstract, conceptual level, an angle that remains aligned with the perspective adopted by existing research in this dimension.

A significant application of agents within recommender systems is **user simulation**. The development of realistic user simulators is essential for the training of reinforcement learning (RL)-based recommender systems, the offline evaluation of new algorithms prior to their potentially expensive online implementation, and the generation of synthetic data to mitigate issues related to data sparsity or cold-start scenarios. Conventional user simulators frequently exhibit deficiencies in realism, diversity, and the ability to accurately represent complex human decision-making processes [126]. In contrast, LLM-powered agents possess the capability to replicate human behavior and reasoning, thereby effectively mitigating these deficiencies [127].

Lusifer [105] creates dynamic user profiles by analyzing historical interactions and textual metadata (such as movie descriptions and tags) with LLMs to generate context-aware user states. This approach further simulates user feedback on new items, effectively addressing cold-start challenges. In contrast, CSHI [103] uses a plugin manager to regulate agent behavior,

improving response generation to align with user logic during the recommendation process. Rather than offering direct suggestions, its agents simulate feedback to help conversational recommender systems better understand user needs. Agent4Rec [63] utilizes LLM-powered agents to simulate user interactions with recommender systems. These agents autonomously take taste- and emotion-driven actions based on personalized preferences and real-time satisfaction, thereby enabling the evaluation of filter bubbles and causal relationships. Meanwhile, PEPPER [99] deploys target-free user simulators that are developed from authentic interaction histories and reviews, facilitating dynamic exploration of user preferences through enhanced dialogues. USimAgent [68] receives a recommendation scenario and uses LLM to generate user queries (such as search keywords), simulating a user-initiated request. ToolRec [69] employs LLM-powered surrogate users that iteratively refine recommendations by leveraging attribute-oriented tools (e.g., rank/retrieval tools) to explore item pools, dynamically adjusting suggestions based on multi-round feedback loops.

iALP [106] leverages LLM-generated user preferences to pretrain RL policies, which not only improves initial recommendation quality but also enables dynamic adaptation to online environments. This framework adopts an actor-critic architecture and continuously updates its policy based on simulated user feedback, thereby ensuring long-term performance stability. RecUserSim [60] enhances simulation realism and diversity by incorporating explicit rating evaluations. Its architecture includes a profile module for different roles, a memory module for tracking preferences, and a core action module built on bounded rationality theory to support fine-grained decision-making. Yoon et al. [100] design an LLM-based user simulator that interacts with external recommender systems through five key tasks: item selection, binary preference judgment, open-ended preference expression, recommendation request generation, and feedback processing. RecAgent [108] initializes agent profiles to capture user characteristics and conducts iterative interactions in a sandbox environment, using memory modules to store historical data and action modules to generate responses. CreAgent [111] serves as an LLM-based generative simulation agent that leverages partial user feedback, such as exposure and click data, to generate and upload item content. Through iterative interactions, it continually optimizes content creation. The framework also introduces a belief mechanism inspired by game theory and a dual-system thinking approach (fast and slow thinking) to enhance recommendation relevance and long-term system performance.

Another critical role for agents for RS is in **security and robustness testing**. As recommender systems grow more complex with LLMs, understanding their vulnerability to malicious attacks is crucial. LLM-powered agents can mimic advanced human strategies and may serve as potential attack vectors.

For instance, CheatAgent [98] identifies optimal insertion points for adversarial perturbations and employs prompt tuning guided by system feedback to refine its attack strategies. TextSimu [104] uncovers a vulnerability in identity-free recommender systems, where multiple LLM-based agents can collaboratively generate deceptive item descriptions that imitate popular content. In a related vein, DrunkAgent [112] integrates a generative module for crafting adversarial triggers and a strategic module that disrupts memory updates; optimized via a surrogate model, it improves attack transferability and stealth, highlighting architectural weaknesses in agent memory mechanisms. Beyond attacks, evaluation frameworks also leverage agent-based simulation. Chirag Shah et al. [61] introduce a dynamic assessment approach in which agents collect user preferences and deliver recommendations through interactions with simulated personas, followed by LLM-powered evaluation to ensure recommendation reliability and adaptability. Collectively, these studies underscore the dual role of agents—both as tools for probing system vulnerabilities and as instruments for enhancing robustness. They emphasize the importance of rigorously evaluating safety implications when deploying LLM-powered agents in recommender systems, and provide insights for designing more resilient architectures.

Agents can also assist in **data processing and analysis tasks** relevant to RS. RuleAgent [62] functions as an autonomous real-time recommendation engine that utilizes LLM-powered agents to identify denoising rules dynamically, thereby improving the quality of recommendations. It is seamlessly incorporated into recommender systems and is capable of continuously adapting to changing data patterns through its self-evolving rule discovery mechanism. This capability allows RuleAgent to sustain strong performance autonomously, without dependence on predetermined or static denoising rules. CORE [94] enhances recommendation performance by dynamically checking estimated relevance scores through conversational interactions, minimizing uncertainty via attribute or item queries. PEARL [96] uses collaborative LLM-powered agents for preference extraction in multi-turn conversational settings, generating and retrieving in-context learning exemplars. This shows agents assisting in the crucial task of understanding

user preferences from natural language interactions, which is fundamental for personalized recommendations.

The "Agent for Rec" paradigm enhances recommender systems in a flexible, non-intrusive way. It employs auxiliary tools like user simulation and security testing to improve systems without altering their core algorithms. This approach ensures better stability and easier integration than the all-encompassing "Agent as Rec" model or the deeply embedded "Agent in Rec" alternative. Nevertheless, its benefits are indirect, as it operates upstream of the main recommendation process. It also cannot achieve the fine-grained, stage-specific control that the tight component collaboration in "Agent in Rec" enables.

3.1.3. Agent in Recommendation

This paradigm describes scenarios where agents are integrated into a specific component of the recommender system, closely interacting with the system. In contrast to "Agent as Rec," where the agent acts as the primary recommender, or "Agent for Rec," where the agent works externally to enhance the system, "Agent in Rec" emphasizes **a more interactive and synergistic role**, where the agent collaborates seamlessly with the system's algorithms and components to deliver personalized, dynamic recommendations. The agent becomes an intrinsic part of the system's decision-making process, enhancing its adaptability and responsiveness.

In the "Agent in Rec" paradigm, agents are deeply integrated into the internal components of a recommender system, working in close synergy with system algorithms to mutually enhance performance and enable personalized and dynamic recommendation generation. For instance, AFL [117] employs a bidirectional learning mechanism between user agent feedback and item explanations, establishing an iterative collaborative process that improves recommendation accuracy and the authenticity of user behavior simulation, reflecting a tight coupling between the system and the agents. AgentCF [114] and its enhanced version, AgentCF++ [121], further articulate the core principles of "Agent in Rec." This framework models both users and items as autonomous language agents, simulating user-item relationships through their interactions while embedding collaborative filtering mechanisms. AgentCF++ introduces a dual-level memory structure alongside a collective memory module, effectively strengthening cross-domain recommendation capabilities and improving the handling of item popularity bias. Essentially, the recommender system emerges from the interactions

and collaborations between user and item agents, highlighting the role of agents as intrinsic, synergistic components rather than external tools.

Other representative methods also demonstrate the versatility of this paradigm: CSA [113] integrates state enhancement and contrastive learning into reinforcement learning-based recommender systems to refine user state representations; KGLA [118] combines knowledge graphs with language agents, leveraging structured relationships to enhance user preference modeling and recommendation interpretability; ARAG [120] uses task specialization to strengthen semantic understanding and optimize personalized outcomes; RPP [122] functions as an internal coordinating module by employing a multi-agent reinforcement learning framework to iteratively refine sentence-level actions across four prompt patterns, thereby dynamically optimizing LLM-input interactions for enhanced recommendation performance. Together, these approaches illustrate the broad potential of "Agent in Rec" in advancing system adaptability and responsiveness.

The "Agent in Rec" paradigm allows for more nuanced and collaborative improvements than the all-encompassing "Agent as Rec" model, as it works within existing system components to enhance rather than replace them. It also provides more direct, real-time impact on recommendations compared to the peripheral, upstream role of "Agent for Rec". However, this integrated approach requires more complex implementation than the standalone "Agent as Rec" architecture. It also lacks the modular flexibility of "Agent for Rec", making it less suitable for external tasks like system-wide evaluation or data simulation.

In conclusion, the "WHERE" perspective indicates that LLM agents are being incorporated into recommender systems through various approaches. These approaches include functioning directly as the recommender (Agent as Rec), serving as essential instruments for the development and assessment of the system (Agent for Rec), and being integrated as collaborative elements within the system's architecture (Agent in Rec). It is important to note that these paradigms are not mutually exclusive; thus, certain systems may demonstrate features characteristic of multiple approaches.

3.2. What Agents Are Used: System Architectures

In addition to their function or position within the recommender system framework, LLM-powered agent systems can be further classified based on their internal architecture, particularly in terms of whether they consist of a single agent or multiple agents that collaborate. Table 3 presents

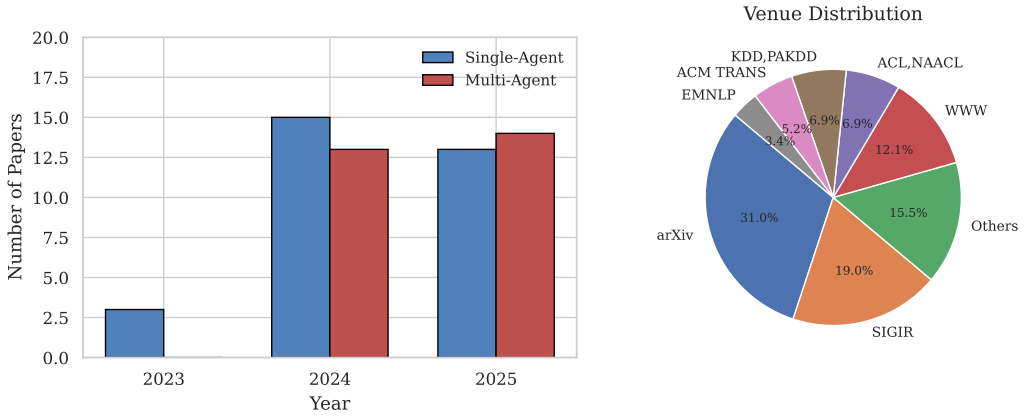

Figure 7: The statistics of publications related to the "WHAT" research question with year and venue.

a categorization of the reviewed paper according to the distinction between single-agent and multi-agent systems. Figure 7 shows the number of collected papers according to publication time and journal. The bar chart presented on the left illustrates the continued advancement of multi-agent systems within the domain of recommender systems. Despite the relatively small number of collected publications, this observation suggests a prevailing trend in development.

3.2.1. Single-Agent Systems

Single-agent recommender systems rely on a single LLM-powered agent to perform designated functions, interact with users or traditional recommenders, and make independent decisions. Compared with multi-agent coordination, this approach emphasizes design simplicity and lower computational overhead.

Frameworks such as iEvaLM [93] consolidate preference reasoning and response generation within one agent, enabling end-to-end recommendations through natural language understanding. Zhang et al. [97] enhance this model with planning and memory modules, allowing the agent to simulate user decision-making and manage the recommendation pipeline autonomously. The REMI framework [88] extends this paradigm by positioning an LLM as the central orchestrator coordinating components such as a causal knowledge graph, reasoning engine, and schema planner to deliver personalized, explainable lifestyle recommendations. Similarly, RecAI [77],

Table 3: A summary of the "WHAT" research question.

Category	Methods
Single-Agent Systems	CSA [113], iEvaLM [93], CORE [94], Jie Wang et al. [115], USimAgent [68], Zhang et al. [97], AutoConcierge [73], CheatAgent [98], CSHI [103], PEPPER [99], ToolRec [69], Yoon et al. [100], SimpleUserSim [101], SUBER [102], Lusifer [105], RecAI [77], Shuang et al. [78], InteRecAgent [67], RecMind [79], PUMA [84], iAgent [107], iALP [106], RuleAgent [62], RecAgent [108], DrunkAgent [112], CreAgent [111], STARec [87], REMI [88], MADREC [83], AgentDR [90], RecUserSim [60]
Multi-Agent Systems	AgentCF [114], Agent4Rec [63], RAH [95], PEARL [96], Rec4Agentverse [74], PMS [75], MACRS [66], ChatCRS [82], BiLLP [116], MACRec [76], MAS4POI [80], TextSimu [104], AFL [117], KGLA [118], MMAgentRec [85], AgentCF++ [121], DiscomfortFilter [110], Chirag Shah et al. [61], CARTS [109], VRAgent-R1 [119], ARAG [120], CDA4Rec [86], TAIRA [81], RouteLLM [89], RPP [122], MARC [91], MATCHA [92]

InteRecAgent [67], and AutoConcierge [73] adopt a unified LLM that integrates retrieval, ranking, and intent parsing, handling user interaction and task execution in a single framework. This architecture is particularly suited for sequential or self-contained tasks. SimpleUserSim [101] refines recommendations through iterative feedback, while Yoon et al. [100] employ unified prompting to generate recommendations directly from dialogue history. CSHI [103] further shows how one agent can simulate multi-stage user interactions—from profile setup to response generation—while maintaining data security.

The performance of single-agent systems, however, depends heavily on the LLM's reasoning strength and the robustness of key modules such as memory, planning, and tool integration. These systems can struggle when facing tasks that require diverse expertise or parallel reasoning. To address such limitations, several studies have introduced self-improvement mechanisms. Jie Wang et al. [115] unify environment simulation and decision-making within one agent that models user states and generates reward signals for iterative optimization. SUBER [102] and iAgent [107] extend this direction by dynamically generating rewards or self-reflecting to adapt to changing preferences. RecAgent [108] demonstrates how a single agent can emulate realistic behaviors such as searching and clicking to improve responsiveness.

The single-agent approach offers the advantage of design simplicity and potentially reduced overhead in comparison to the coordination of multiple agents [128]. Although this approach is conceptually more straightforward, it is heavily dependent on the capabilities of the underlying LLM and the

efficacy of the agent's fundamental components, such as memory, planning, and tools, to manage task complexity [129]. Single-agent systems are typically well-suited for tasks that can be executed sequentially or where the necessary expertise is centralized within a single entity [130]. Nevertheless, a single agent may encounter difficulties when faced with complex tasks that necessitate a range of expertise or require parallel processing capabilities [131].

3.2.2. Multi-Agent Systems

Multi-agent systems consist of several LLM-powered agents that work collaboratively or engage with one another to accomplish a shared objective or perform distinct functions within the system. This framework facilitates the breakdown of intricate tasks into smaller, manageable sub-tasks, which are addressed by specialized agents. This approach capitalizes on the principle of division of labor, thereby fostering more nuanced interactions among the agents. A multi-agent system can be formally represented as

$$MAS = \big(A, S, T, \Pi\big), \tag{6}$$

where $\boldsymbol{A} = \{a_1, a_2, \ldots, a_N\}$ represents a finite set of agents, where each agent, akin to individuals in a human society, assumes distinct roles and is entrusted with specific tasks. $\boldsymbol{S}$ defines the topological structure among agents within a multi-agent system (e.g., hierarchical, centralized). This structure governs the pathways for message propagation. $\boldsymbol{T} = \{t_1, t_2, \ldots, t_N\}$ corresponds to the collection of tasks pursued by individual agents, where a complex objective is decomposed in a divide-and-conquer manner and tackled through a collective effort involving collaborative or competitive interactions. $\boldsymbol{\Pi}$ defines the interaction protocol among agents, establishing a framework for message passing that can be either predefined or adaptively regulated.

Recent studies demonstrate diverse implementations of this paradigm. DiscomfortFilter [110] employs cooperative agents to generate personalized recommendations while minimizing user discomfort. Perceive Agent extracts item features from user interactions, and Reflect Agent integrates these insights to refine preference profiles through guided conversations that iteratively filter sensitive content. Agent4Rec [63] uses generative LLM agents initialized with real-world user profiles and equipped with coordinated functional modules to emulate authentic behaviors. RAH [95] defines five specialized roles—Perceive, Learn, Act, Critic, and Reflect—forming a continuous

feedback loop that adjusts user preferences and resolves conflicts, thereby improving recommendation control and privacy. BiLLP [116] follows a hierarchical structure of four agents (Planner, Reflector, Actor, and Critic), combining macro-level planning with micro-level personalization.

Other frameworks further refine agent collaboration. Rec4Agentverse [74] structures the system around two main agents—Agent Items and Agent Recommender—that coordinate information flow to enhance personalization. MACRec [76] integrates multiple agents—Manager, Analyst, Reflector, Searcher, and Interpreter—to analyze tasks from different perspectives and refine decisions through reflective learning. AFL [117] adopts an iterative dialogue loop between a Recommendation Agent and a User Agent, enabling mutual learning and preference refinement. AgentCF [114] and AgentCF++ [121] extend this concept by modeling users and items as autonomous agents that interact bilaterally, capturing evolving user–item relationships to improve recommendation precision.

The multi-agent paradigm presents numerous advantages, including (1) **Specialization**, wherein agents can be tailored to possess specific expertise or fulfill designated roles, thereby improving performance on intricate sub-tasks; (2) **Modularity**, which allows for the construction of the system using reusable components; (3) **Robustness**, as the failure of an individual agent does not necessarily compromise the functionality of the entire system; and (4) **Parallelism**, enabling agents to operate concurrently. Nonetheless, multi-agent systems also encounter challenges, such as coordination and communication overhead, the need to maintain consistent behavior, and the management of system complexity. The decision to utilize a single-agent versus a multi-agent system is contingent upon the specific requirements of the task, the desired level of complexity management, and the computational resources at hand [132].

3.3. Why Agents Are Used: Problems and Applications

Integrating LLM-powered agents into recommender systems is driven by their potential to overcome existing limitations and enable new functions. These agents are designed to tackle a variety of challenges, which can be categorized into general recommendations, specific scenarios, interactive contexts, and assessments pertaining to security or robustness. Table 4 provides a breakdown of the related paper based on this classification. Here, general-domain and vertical-domain recommendations are distinguished along the

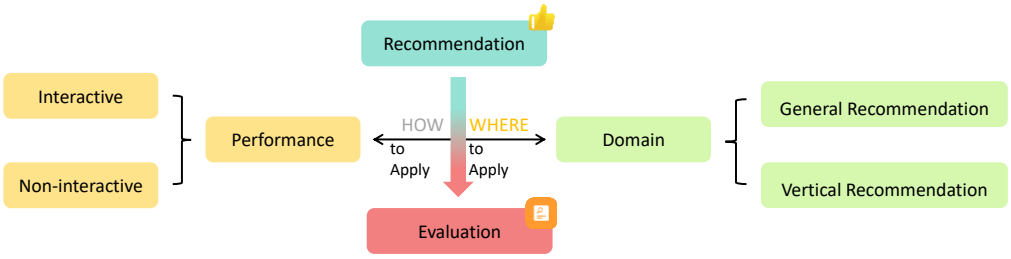

Figure 8: A structural breakdown of the research question "why". The framework delineates two high-level facets: Recommendation and Evaluation. The Recommendation facet comprises Domain (General or Vertical) and Performance (Interactive or Non-interactive), derived from the model's primary motivation.

spectrum of application scope, whereas interactive recommendation is characterized as a distinct functional capability. A single system can integrate interactive recommendation with either general or vertical-domain functionality. It is important to note that "interaction" here refers specifically to dialogue-based communication, excluding implicit feedback such as user clicks or browsing behaviors. The evaluation scenario, in contrast, operates orthogonally to these three functional scenarios, serving to quantitatively assess the effectiveness of systems within any of the above contexts. The interrelationships among these four scenarios are illustrated in Figure 8. It is important to note that these four categories are not mutually exclusive. A single model's design motivation may span multiple scenarios. In this study, differing from prior research, we classify models based on their primary motivational scenario from the perspective of practical application, rather than their functional positioning.

3.3.1. Enhancing General Recommendation Tasks

Recommendation agents are evolving from domain-specific tools towards a general paradigm. This next-generation paradigm aims to create agile systems that transcend vertical boundaries (e.g., e-commerce, music, news), interpret multifaceted user intents, and execute a diverse set of recommendation tasks leveraging a consolidated architectural and knowledge framework. Let $\boldsymbol{q} \in Q$ denote the problem to be solved by the agent, $\boldsymbol{u} \in U$ the user profile vector (typically sourced from a memory module), and $\boldsymbol{\omega} \in \Omega$ a general corpus or external knowledge base. The generic recommendation task

Table 4: A summary of the "WHY" research question.

Category	Methods
General Recommendation	RecMind [79], MACRec [76], RecAI [77], AgentCF++ [121], RAH [95], Zhang et al. [97], Jie Wang et al. [115], CSA [113], KGLA [118], STARec [87], MADREC [83], AgentDR [90], RPP [122], ARAG [120]
Domain-Specific	ToolRec [69], Shuang et al. [78], AutoConcierge [73], MMAgentRec [85], MAS4POI [80], DiscomfortFilter [110], Lusifer [105], RuleAgent [62], PEARL [96], CARTS [109], VRAgent-R1 [119], PUMA [84], CDA4Rec [86], REMI [88], RouteLLM [89], MARC [91], AFL [117]
Interactive Improvement	iAgent [107], InteRecAgent [67], MACRS [66], Rec4Agentverse [74], TAIRA [81], MATCHA [92], CSHI [103], iALP [106], CORE [94], ChatCRS [82], AgentCF [114], PMS [75]
Evaluation & Security	iEvaLM [93], BiLLP [116], RecAgent [108], DrunkAgent [112], SimpleUserSim [101], CreAgent [111], Yoon et al. [100], USimAgent [68], Chirag Shah et al. [61], TextSimu [104], AFL [117], RecUserSim [60], PEPPER [99], Agent4Rec [63], SUBER [102], CheatAgent [98]

can be concisely abstracted as

$$R_\theta : (q, u) \rightarrow \big(\omega_1, \omega_2, ..., \omega_L\big), \tag{7}$$

where θ is the agent's mapping policy that, by combining the user's behavior history and current preference vector, retrieves the L most probable items or solutions from the general corpus.

Recent research has increasingly focused on developing general-purpose recommendation frameworks that adapt across diverse scenarios. AgentDR [90] achieves domain-agnostic recommendation through a modular architecture that orchestrates heterogeneous models via LLM-mediated tool suitability learning and transferable intent reasoning, enabling consistent adaptation across diverse data distributions. RecMind [79] utilizes LLMs to deliver personalized recommendations in a zero-shot manner, eliminating the need for task-specific fine-tuning. It improves general recommendation capabilities through an innovative Self-Inspiring planning approach, which incorporates various reasoning pathways and external tools for knowledge acquisition. Similarly, CSA [113] addresses various recommendation scenarios via unified state augmentation strategies, combining offline reinforcement learning with contrastive learning to mitigate distribution shifts in static datasets. MADRec [83] orchestrates recommendation through an integrated workflow where unsupervised aspect-based profiling, multi-factor re-ranking, and self-refining feedback loops collectively enable adaptive and explainable item se-

lection. Beyond zero-shot approaches, several frameworks employ specialized architectures to achieve broader applicability. RAH [95] tackles significant challenges in recommender systems by implementing a human-centered methodology aimed at enhancing personalization, alleviating user burden, mitigating biases, and augmenting user control over recommendations and privacy. MACRec [76] advances this direction through multi-agent collaboration, where distinct agents collectively improve recommendation precision, interpretability, and adaptability across rating prediction, sequential recommendation, and conversational tasks. Building on agent-based designs, STARec [87] models users as autonomous reasoning agents with a dual-process architecture that dynamically adapts to preferences through integrated knowledge distillation and reinforcement learning, achieving robust performance in sparse data environments with minimal training requirements.

The integration of external knowledge represents another pathway to generalization. KGLA [118] employs knowledge graph pathways as natural language justifications to augment agent memory, thereby facilitating a more precise representation of user preferences and interactions, which ultimately leads to enhanced recommendation outcomes. Similarly, ARAG [120] enhances the RAG framework through multi-agent collaboration, where specialized agents handle user profiling, semantic filtering, chain-of-thought reasoning, and evidence summarization to support comprehensive recommendation decisions across varying contexts. Meanwhile, a separate line of research concentrates on refining model architectures and training paradigms to address challenges such as data sparsity and domain adaptation. AgentCF++ [121] employs a dual-layer memory architecture to filter irrelevant information and integrate cross-domain preferences while accounting for popularity effects among similar users. Jie Wang et al. [115] utilize LLMs as simulated environments to overcome data sparsity, generating user states and reward signals from limited data for robust cross-domain RL training.

The applications of these general recommender systems indicate that agents extend beyond specialized domains and possess the potential to enhance performance and mitigate the limitations associated with conventional recommendation tasks. This is achieved through the incorporation of advanced capabilities such as zero-shot learning [84, 77], the utilization of tools for precise control [95, 84, 76], and the automation of data quality enhancement [113, 77, 118, 121, 115].

3.3.2. Addressing Vertical and Domain-Specific Scenarios

LLM-powered agents are especially adept in vertical or domain-specific recommendation contexts, where the application of specialized knowledge and comprehension of particular item characteristics are essential. Their capacity to analyze and interpret textual data renders them highly effective in fields characterized by an abundance of textual information. Owing to its domain-specific corpora Ω and mapping strategies θ, this approach surpasses general recommendation in both recommendation quality and problem-solving efficacy.

E-commerce is a prime vertical scenario. Shuang et al. [78] concentrate on enhancing product recommender systems within e-commerce contexts, such as the WebShopp platform, by utilizing structured item attributes. Their objective is to achieve a precise correspondence between user specifications and the distinctive features of niche products. Hybrid-MACRS [133] is purposefully developed for conversational recommendation within the e-commerce sector. It utilizes a search agent to navigate product catalogs and employs a central LLM-powered agent to comprehend shopping intentions and provide product recommendations. PEARL [96] is designed to extract customer preferences within e-commerce dialogues, effectively tackling challenges such as the management of extensive product catalogs, the comprehension of intricate attributes, and assisting users in locating specific items. CARTS [109] introduces a collaborative multi-agent framework specifically designed for e-commerce recommendation. Its three-stage methodology—Generation Augmented Generation, Refinement Circle, and Arbitration—succinctly addresses key constraints like character limits, semantic richness, and alignment with business metrics.

In the field of POI recommendation, MAS4POI [80] employs a multi-agent framework where specialized agents collaborate in a closed-loop process. By integrating diverse data sources and dynamically optimizing tasks, the system effectively addresses cold-start challenges and enhances POI recommendation performance. Similarly, RouteLLM [89] operates as a grounded multi-agent system where a manager agent orchestrates three functional agents—Constraint, POI, and Path Refinement—to transform natural language queries into executable routes. The system translates abstract preferences into quantifiable weights, invokes a traditional routing engine for spatial optimization, and finally delivers verified routes through a dedicated Verifier Agent, ensuring both interpretability and constraint compliance. Cai and

Niu [134] propose a methodology within the educational domain that employs autonomous agents to customize assessment task recommendations for learners, considering their capabilities, academic emotions, and the expectations of both learners and educators through dynamic strategy adjustments and agent negotiation. In the domain of transportation, Li et al. [135] propose a multi-agent reinforcement learning recommender system for electric vehicle charging stations that considers real-time traffic conditions and uses centralized learning with decentralized execution for optimal station recommendations. In the field of video recommendation, VRAgent-R1 [119] uses agents based on Multimodal Large Language Models (MLLM) refined through reinforcement learning to dynamically optimize video recommendations. It integrates multimodal video semantics and employs a dual-agent approach to adapt to changing user preferences. The MARC framework [91] utilizes task-aware routing to identify user intent and drives targeted graph retrieval algorithms. A reflection module then iteratively refines the results to achieve precise cocktail recommendations.

Some system frameworks can be adapted for different domains through training, but they cannot operate across multiple domains simultaneously, limiting their ability to offer universal recommendations. Lusifer [105] shows notable adaptability in vertical domains by using dynamic user profiles from LLMs, allowing for the transparent evolution of user preferences. It excels in movie recommendations, e-commerce suggestions, and music streaming, where contextual metadata like genres and tags is crucial. ToolRec [69] utilizes attribute-oriented tools, such as ranking and retrieval mechanisms, whereby LLM-powered agents assess particular attributes, including movie genres or book categories, to effectively retrieve and rank items. This approach guarantees that the recommendations are aligned with the relevant domain. PUMA [84] addresses the challenge of aligning user intentions with web-based recommendations by integrating user directives, past behaviors, and contextual task specifications. Its framework supports adaptive alignment across diverse web contexts, ensuring strong performance in both single-turn and multi-turn interactions. RuleAgent [62] employs an autonomous rule-discovery framework designed for the denoising of implicit feedback. The primary innovation of this system is its ability to emulate domain experts in order to dynamically formulate denoising rules that are tailored to specific datasets, such as those found in e-commerce or media platforms.

The deployment of agents within specialized contexts leverages their comprehension of domain-specific language and knowledge, frequently in con-

junction with pertinent external tools or databases. This approach yields recommendations that are more precise and contextually relevant compared to those generated by general-purpose models.

3.3.3. Enabling and Improving Interactive Recommendation

A significant impact of LLM-powered agents is their capacity to enhance and optimize interactive recommendation experiences, particularly within the framework of conversational recommendations. These agents are especially adept at engaging in dialogue-driven interactions due to their sophisticated language processing abilities. Let $\boldsymbol{Q}$ denote the query space of users and $\boldsymbol{A}$ the action space of the LLM-powered agents. The session history at time t can then be represented as

$$C_{1:t} = \big(q_1, a_1, ..., q_{t-1}, a_{t-1}, q_t\big), \tag{8}$$

where $\boldsymbol{q}_i$ denotes a user's query utterance and $\boldsymbol{a}_i$ represents the agent's response. The complete interactive recommendation task can thus be defined as

$$\Theta : (C_{1:t}, E) \rightarrow \big(i_1, i_2, ..., i_L\big), \tag{9}$$

that selects the top-L items with the highest output probabilities for user recommendation. $\boldsymbol{E}$ represents the environment in which the LLM agent operates and conducts external interactions.

Conversational Recommender Systems represent a crucial evolution in recommendation research, emphasizing dynamic, dialogue-driven user engagement rather than static one-shot prediction. Recent advances—such as ChatCRS [82], MATCHA [92], MACRS [66], and CORE [94]—have demonstrated how interactive recommendation can transform passive consumption into an adaptive and explanatory process, where systems continuously refine understanding of user preferences through conversation. In contrast to traditional recommenders that rely solely on historical user–item interactions, conversational recommenders center on interactive preference elicitation: the agent interprets natural language, asks clarifying questions, and adjusts its outputs based on real-time feedback. For instance, ChatCRS decomposes the conversational recommendation task into multiple coordinated agents—one for goal planning and another for knowledge retrieval—thus enabling the system to guide dialogues toward recommendation goals while grounding responses in domain-specific external knowledge. This design enriches the interactivity of the system, as each user turn dynamically influences both

reasoning paths and recommendation results. Similarly, CORE introduces a unified uncertainty minimization mechanism that bridges conversational agents and recommender systems in a plug-and-play fashion. The conversational agent acts as an online checker, strategically querying item attributes or options to minimize uncertainty about user preferences. Through this adaptive interaction, CORE transforms recommendation into a process of information exchange—each conversational turn reduces ambiguity and progressively narrows the search space, ensuring efficient and explainable recommendation outcomes.

Extending this paradigm, MATCHA proposes a multi-agent collaboration framework to enhance personalization, safety, and trustworthiness in interactive recommendations. Within MATCHA, distinct agents cooperate on subtasks such as intent analysis, candidate generation, ranking and reflection, and explanation generation. These agents communicate iteratively, balancing recommendation accuracy and diversity while maintaining interpretability and robustness. Particularly, MATCHA's reflection agent revisits prior dialogue states and recommendation rationales to refine subsequent responses, creating a continuous feedback loop that deepens system–user alignment. The addition of a risk control agent further ensures secure and ethical interactions, addressing potential jailbreak or harmful query issues inherent in open-ended dialogue systems. Beyond these frameworks, systems such as MACRS and TAIRA [81] exemplify the broader movement toward interactive and hierarchical reasoning in CRS. MACRS regulates dialogue flow based on user feedback to sustain coherent, goal-oriented interaction, while TAIRA employs a multi-agent hierarchical architecture leveraging Thought Pattern Distillation to extract reasoning schemas from prior interactions. Through a manager–executor collaboration, it transforms ambiguous conversational intents into specific, actionable recommendations, closing the loop between understanding and generation. Meanwhile, CSHI [103] extends interactivity to user simulation and adaptive profiling, enabling real-time modification of user models and plugin-based dialogue control. This design allows CRS to adapt dynamically to evolving conversational contexts, maintaining consistency across user sessions while preserving personalization depth. InteRecAgent [67] operates as a conversational recommender system by employing an LLM-powered agent to coordinate traditional recommender system tools and engage with users through natural language interactions.

Interactive scenarios leverage the core functionalities of LLMs in natural language processing and generation to improve the engagement, adaptabil-

ity, and user-centric nature of recommendation experiences. Such scenarios support dynamic preference elicitation, enhance explainability, and allow for adaptability, thus moving beyond conventional static recommendation lists and encouraging collaborative discovery processes.

3.3.4. Evaluation, Security, and Robustness

In LLM-powered recommender systems, evaluation plays a central role in ensuring accuracy, reliability, and real-world adaptability. It serves not only to measure the quality of recommendations but also to assess the system's security and robustness under diverse conditions. Evaluation metrics commonly examine the precision and relevance of recommendations, often complemented by user satisfaction indicators. Meanwhile, security evaluation safeguards user data and prevents malicious manipulation through encryption and access control, while robustness evaluation ensures system stability when facing noisy data or adversarial perturbations. In these contexts, agents typically assume the role of users, performing a series of system evaluations by simulating user behaviors or environmental conditions. In the context of recommendation performance evaluation, these agents can be abstracted as

$$R_\theta : E \times I^L \rightarrow A_{user}, A_{user} \in \{Accept, Reject\}, \tag{10}$$

where θ denotes the internal preference vector and $\boldsymbol{I}$ represents the set of candidate items for recommendation.

Recent research has advanced evaluation from simple accuracy measurement to comprehensive agent-based testing frameworks. For instance, RecAgent [108] incorporates adversarial perturbation analysis to examine how systems respond to malicious or noisy inputs, thereby exposing potential vulnerabilities. Agent4Rec [63] introduces generative LLM-powered agents that emulate human behaviors, enabling evaluations of how well systems reproduce user interactions, preferences, and decision-making patterns. Similarly, PEPPER [99] proposes target-free user simulators built from authentic interaction data and introduces metrics such as Preference Coverage (PC) and Preference Coverage Increase Rate (PCIR) to comprehensively assess both preference elicitation and recommendation accuracy. Yoon et al. [100] evaluate LLM-powered user simulators through five user behavior tasks—such as item selection, feedback generation, and preference articulation—revealing gaps between simulated and real users while demonstrating that model selection and prompting techniques can improve simulator fidelity. SUBER

[102] further highlights fundamental challenges in evaluating reinforcement learning-based recommenders, particularly the scarcity of real interaction data and the difficulty of establishing reliable evaluation frameworks.

Several studies use simulations to assess performance. USimAgent [68] reproduces user search behaviors—including query formulation and click actions—to automate evaluation under controlled settings, while also using adversarial simulations to probe algorithmic weaknesses. TextSimu [104] evaluates system robustness against adversarial attacks by simulating manipulations in item descriptions, quantified via Hit Ratio and NDCG metrics. Building on this line, DrunkAgent [112] employs cross-model transferability tests (AgentCF, AgentSEQ, and AgentRAG) and perplexity-based stealth evaluation to quantify attack efficacy, incorporating dual-layer defenses such as paraphrasing and confidence thresholding to enhance adversarial robustness. RecUserSim [60] extends this paradigm by modeling user profiles, memory, and actions to produce quantitative ratings, thus evaluating recommendation quality while also testing resilience against diverse user behaviors and adversarial scenarios. In parallel, CreAgent [111] leverages LLMs to simulate content creators under information asymmetry, enabling fairness- and diversity-oriented evaluation across user preference, activity, and content distributions.

Overall, these studies establish evaluation as a multifaceted process encompassing performance, security, and robustness testing. By simulating both user interactions and adversarial attacks, agent-based evaluation frameworks not only quantify system effectiveness but also expose weaknesses that inform model improvement. Through this integrated approach, recommender systems can better adapt to dynamic real-world environments and maintain reliable, secure, and robust performance.

3.4. How Agents Are Designed: Component Optimization

The performance of LLM-powered agents in recommender systems largely depends on their internal architecture and the components integrated into their design. To better suit recommendation tasks, researchers often enhance core modules such as Profile, Memory, Planning, and Action by introducing additional mechanisms tailored to task-specific requirements. This section examines the "HOW" dimension—how these agents operate and are optimized for recommendation. Table 5 summarizes the optimization components adopted in existing studies, along with available open-source implementations. It also lists the underlying language models, frameworks, and

datasets used for experimental evaluation.

Table 5: A summary of the "HOW" research question.

Methods	**Profile** Module	**Memory** Module	**Planning** Module	**Action** Module	**Code** (Y/N)	**Model** (Framework)	**Dataset**
CSA	×	✓	×	✓	Y[1]	Tensorflow	RC15, RetailRocket and Meituan
iEvaLM	×	×	×	✓	Y[2]	gpt-3.5-turbo	ReDial, OpenDialKG
CORE	×	×	✓	×	N	gpt-3.5-turbo	Amazon, Last.fm, Yelp, Taobao
AgentCF	×	✓	×	✓	N	RecBole	CDs and Vinyl, Office Products
Agent4Rec	✓	✓	×	✓	Y[3]	LangChain	MovieLens-1M, Steam, Book
Jie Wang et al.	×	✓	✓	✓	N	Mistral 7B	LFM, Industry
RAH	✓	✓	✓	✓	N	GPT-4-0613	Movies, Books, Video Games
PEARL	×	✓	×	✓	N	Claude-instant-v1	Internal Dataset, MultiWOZ-H
USimAgent	×	×	✓	✓	N	GPT-4	UserStudy
Zhang et al.	✓	×	✓	×	Y[4]	ChatGLM-6B	Yelp, Music, Games
AutoConcierge	✓	✓	✓	✓	N	GPT-3	No available datasets
CheatAgent	×	×	✓	×	N	T5	MovieLens-1M, LastFM, Taobao
PEPPER	×	✓	×	✓	N	GPT-3.5-turbo	IMDb, ReDial, OpenDialKG
ToolRec	×	×	✓	✓	Y[5]	gpt-3.5-turbo-16k	ML-1M, Book, Yelp2018
Yoon et al.	×	✓	×	✓	Y[6]	PyABSA	ReDial, Reddit, MovieLens
SimpleUserSim	✓	×	×	✓	N	iEvaLM	ReDial, OpenDialKG
SUBER	×	✓	×	×	Y[7]	Sentence-T5	ml-latest-small, Book
Lusifer	×	✓	×	✓	Y[8]	GPT-4o-mini	MovieLens
CSHI	✓	✓	×	✓	Y[9]	GPT-3.5-turbo	ReDial, OpenDialKG, MovieLens
Rec4Agentverse	✓	✓	✓	✓	N	GPT-4	Generate data
PMS	✓	✓	✓	✓	N	LangChain	Created dataset
MACRS	✓	✓	✓	✓	N	GPT-3.5-turbo-0613, Llama-2-70b-chat-hf	MovieLens

Table 5, continued

Methods	**Profile** Module	**Memory** Module	**Planning** Module	**Action** Module	**Code** (Y/N)	**Model** (Framework)	**Dataset**
ChatCRS	✓	×	✓	×	Y[10]	gpt-3.5-turbo-1106, LLaMA-7b	DuRecDial, TG-Redial
BiLLP	×	×	✓	×	Y[11]	GPT-3.5-turbo-16k	Steam, Book
MACRec	✓	✓	✓	✓	Y[12]	Not Specified	Generate data
RecAI	✓	✓	✓	×	Y[13]	GPT-4	Not Specified
Shuang et al.	×	×	×	✓	N	BERT	WebShop
InteRecAgent	✓	✓	✓	×	Y[14]	LangChain	Steam, MovieLens, Beauty
RecMind	×	✓	✓	✓	N	GPT-3.5-turbo-16k	Reviews, Yelp
MAS4POI	✓	✓	✓	✓	Y[15]	Six Distinct LLMs	NYC, TKY
TextSimu	✓	✓	✓	✓	N	GPT-4o mini	Beauty, Instrument, Office
RecAgent	✓	✓	×	✓	N	ChatGPT	MovieLens-1M, Beauty, Book-Crossing
AFL	×	✓	✓	✓	Y[16]	GPT-4o-mini	Lastfm, Steam, MovieLens
KGLA	×	✓	×	×	N	Claude3-Haiku-20240307	CDs, Clothing, Beauty
iALP	×	×	✓	✓	N	Mistral 7B	LFM, Industry, Coat
REMI	×	✓	✓	✓	N	Gemini-2.0-Flash	Generate data
iAgent	✓	✓	×	×	Y[17]	GPT-4o-mini	Book, Movie, Goodreads
PUMA	×	✓	×	✓	Y[18]	LLaMA-2-7B	Review
MMAgentRec	✓	✓	×	✓	N	BERT	Guangdong Tourism Dataset, QK-Video
CDA4Rec	✓	×	✓	✓	N	LLaMA-3.1-8B	ReDial, Music4All, Sports
MATCHA	×	×	✓	✓	N	Multiple LLMs	OMuleT, WildJailbreak
CARTS	×	✓	×	✓	N	GPT-4o	Beauty, Electronics, Fashion
VRAgent-R1	×	✓	✓	✓	N	Qwen2.5-7B	MicroLens-100K, MovieLens-1M
MADREC	✓	✓	×	×	N	GPT-4.1-nano	Beauty, Sports, Toys
ARAG	×	✓	✓	✓	N	GPT-3.5-turbo	Review

Table 5, continued

Methods	**Profile** Module	**Memory** Module	**Planning** Module	**Action** Module	**Code** (Y/N)	**Model** (Framework)	**Dataset**
TAIRA	×	×	✓	×	Y[19]	GPT-4o	Clothing & Shoes, Beauty, Music
RecUserSim	✓	✓	×	✓	N	Multiple LLMs	Generate data
DiscomfortFilter	✓	×	×	×	N	Multiple LLMs	MIND
STARec	×	×	✓	✓	N	Multiple LLMs	MovieLens 1M, CDs and Vinyl
CreAgent	✓	✓	×	✓	Y[20]	Llama3-8B	YouTube
Chirag Shah et al.	✓	✓	✓	✓	N	Not Specified	Not Specified
DrunkAgent	×	✓	×	×	N	LLaMA 3-8B-Instruct	CDs & Vinyl, Office Products, Musical
RuleAgent	✓	✓	✓	✓	N	GPT-4o mini	Beauty, Yelp2018, Gowalla
AgentCF++	×	✓	×	×	Y[21]	Not Specified	Books, CDs, Movies, Games
AgentDR	×	✓	×	✓	N	Phi-4	Instacart, Electronics, Sports
RouteLLM	×	×	✓	✓	Y[22]	GPT-4o	Generate data
MARC	×	×	✓	✓	Y[23]	GPT-4o	cocktails, ingredients and instructions
RPP	×	×	×	✓	Y[24]	LLaMa2-7B-chat	ML-1M, Games, Lastfm

3.4.1. Profile Module

The profile component of an LLM-powered agent in a recommender system is tasked with encapsulating the attributes, preferences, and possible objectives of the entity that the agent is either simulating or representing, typically a user or an item. This profile functions as the agent's identity and significantly impacts its behavior and decision-making processes.

Rec4Agentverse [74] integrates dynamic social identity, intrinsic motivation, and characteristics of creative activities, thereby facilitating a more sophisticated simulation of agent behavior. Zhang et al. [97] explicitly model user preferences by employing sentiment analysis generated by LLMs alongside behavioral logic. This approach incorporates both the attributes of items and historical interactions to develop dynamic and nuanced user profiles. CSHI [103] uses a dual initialization approach, blending manual configuration of personality traits with LLM-powered generation for scalable profiling. DiscomfortFilter [110] develops a modifiable preference profile by employing LLM analysis of user behavior, specifically focusing on clicks and non-clicks.

[1]https://github.com/HN-RS/CSA
[2]https://github.com/RUCAIBox/iEvaLM-CRS
[3]https://github.com/LehengTHU/Agent4Rec
[4]https://github.com/Applied-Machine-Learning-Lab/LLM_User_Simulator
[5]https://github.com/Go0day/ToolRec-Code
[6]https://github.com/granelle/naacl24-user-sim
[7]https://github.com/SUBER-Team/SUBER
[8]https://github.com/danialebrat/Lusifer
[9]https://github.com/zlxxlz1026/CSHI
[10]https://github.com/lichuangnus/ChatCRS
[11]https://github.com/jizhi-zhang/BiLLP
[12]https://github.com/wzf2000/MACRec
[13]https://github.com/microsoft/RecAI
[14]https://aka.ms/recagent
[15]https://github.com/yuqian2003/MAS4POI
[16]https://github.com/Lanyu0303/AFL
[17]https://github.com/wujiangxu/iAgent
[18]https://github.com/HongruCai/PersonalWAB
[19]https://github.com/Alcein/TAIRA
[20]https://github.com/shawnye2000/CreAgent
[21]https://github.com/jhliu0807/AgentCF-plus
[22]https://anonymous.4open.science/r/RouteLLM_ACL-DE52
[23]https://github.com/diddbwls/cocktail_rec_agentrag
[24]https://github.com/maowenyu-11/RPP

This approach transforms raw interaction data into comprehensible feature-based preferences, which are subsequently ranked using the PageRank algorithm. iAgent [107] improves the Profile Module by implementing conflict resolution strategies that are designed to more effectively align with user preferences. Furthermore, it integrates real-time feedback mechanisms and personalized trait modeling to achieve a more precise representation of users. MMAgentRec [85] integrates cross-attention mechanisms and self-reflection mechanisms to enhance the understanding of user preferences and social characteristics. RecAgent [108] generates detailed user personas with nuanced social attributes and unique preferences, utilizing real-world datasets for authenticity. It also includes a conflict resolution mechanism to resolve inconsistencies in user profiles, improving realism and diversity.

The design of the profile component is essential for achieving personalization and realistic simulation. It establishes the initial conditions and intrinsic biases of the agent, which in turn influence its perception of the environment and its subsequent behaviors. Contemporary profile modules, while capable of interacting with external environments through conflict resolution mechanisms and enabling dynamic adjustments via feedback loops to construct scalable user profiles, exhibit significant susceptibility to external influences. In offline settings, their effectiveness is constrained by the quality of available datasets, whereas in real-world deployments, the increased complexity of external environments presents substantial challenges in selectively updating user profiles. Looking forward, the optimization of profile modules is expected to become increasingly implicit—for instance, through integration with memory components. As previously noted, these four fundamental modules are not isolated constructs but rather function as interconnected and coordinated elements within the system.

3.4.2. Memory Module

Memory constitutes an essential element that facilitates the capacity of LLM-powered agents to preserve information from previous interactions and experiences. This capability enables these agents to learn, adapt, and sustain contextual awareness over time. In the context of recommender systems, memory serves to archive interaction histories, acquired preferences, external knowledge, and even internal states or reflections.

CSA [113] introduces contrastive state augmentations for dynamically generating synthetic state representations, improving memory retrieval robustness during distributional shifts. It also uses hierarchical memory com-

pression to store original and augmented interaction sequences, enhancing knowledge transfer across different recommendation contexts. RecAI [77] improves the memory module with dynamic updates and hierarchical architectures for better storage and retrieval of user interaction data. It also uses contrastive learning to enhance memory representations, allowing the model to better identify subtle user preferences over time. AFL [117] enhances the memory module with a dual-agent synchronization mechanism, where the recommendation and user agents collaboratively update shared interaction histories, including rejected items and feedback. This enables real-time adjustments to user behavior and reduces popularity bias through balanced memory weighting. SUBER [102] utilizes a dual-structure framework that separates user and item datasets while updating interaction histories. It features preprocessing elements that convert raw data into prompts for LLMs and postprocessing components that transform LLM outputs into actionable rewards, facilitating real-time adaptation to user behavior.

Yoon et al. [100] enhance the memory module with a novel preference excitation mechanism that dynamically identifies latent user interests, allowing for adaptive updates to preferences when recommendations align with unarticulated desires. Jie Wang et al. [115] incorporate a contrastive learning framework that enhances state representations by evaluating the distinctions between positive and negative actions. Additionally, they employ autoregressive tokenization to dynamically update user-item interaction histories, thereby facilitating efficient semantic storage. InteRecAgent [67] features a "shared candidate bus" for efficient tool communication and a dual-profile system for monitoring user preferences. It also includes advanced memory mechanisms, such as dynamic task planning and reflection strategies, to enhance robustness and error correction. CreAgent [111] employs a dual-layer architecture that integrates feedback memory and creation memory. This design purposefully retains user feedback and historical creation patterns, thereby enhancing the simulation of creator behavior in contexts characterized by information asymmetry. AgentCF++ [121] prioritizes target domain information through a two-step fusion process, reducing cross-domain noise. It also includes interest groups that leverage shared memory to account for popularity among users with similar interests.

The design of the memory component involves key considerations, such as selecting information to store, representation methods (e.g., raw text, structured data, embeddings), efficient retrieval, and updating processes, including reflective practices. Effective memory management is crucial for agents to

demonstrate consistent, adaptive, and contextually aware behavior in recommendation tasks. Future advancements in memory components will exhibit increasing diversification and socialization. Diversification refers to the expanding variety of memory sources, forms, and operations—ranging from interaction histories to external knowledge, from internal weight parameters to external textual representations, and from memory writing to memory management. This evolution will transform memory from a simple modular component into a rich, multi-faceted memory entity. Socialization, on the other hand, describes a shift in research focus from treating memory as a set of static parameters to viewing it as an integral part of an agent and eventually, as a distributed system analogous to the human brain. In the pursuit of AGI, studies will progressively incorporate principles of human cognition and social behavior into the design of memory systems. Although current research remains focused on designing dynamic memory mechanisms—such as collaborative and hierarchical optimization—these mechanisms are already inspired by real-world processes. Thus, the diversification and socialization of memory components represent an inevitable and meaningful direction for future development.

3.4.3. Planning Module

Planning refers to the methodology employed by an LLM-powered agent to establish a series of actions aimed at accomplishing a designated objective. Within the framework of recommender systems, this may encompass the orchestration of a multi-turn dialogue to ascertain user preferences, the arrangement of a sequence of tool interactions to produce recommendations, or the delineation of steps within a simulation or attack scenario.

RecMind [79] presents a Self-Inspiring (SI) planning algorithm that retains historical states from previous reasoning pathways, unlike traditional Chain-of-Thought [136] and Tree-of-Thought [137] methods that discard this data. This approach enhances planning and improves recommendation accuracy by leveraging accumulated knowledge. AutoConcierge [73] incorporates goal-oriented Answer Set Programming (ASP) reasoning through s(CASP) [138], which actively detects absent user preferences and formulates specific inquiries to address informational deficiencies throughout interactions. MAS4POI [80] enhances its Planning Module with multi-agent collaborative planning. Specialized agents, such as the Analyst and Reflector, iteratively refine Points of Interest (POI) recommendations using hierarchical task decomposition and reflective mechanisms. RAH [95] improves its Planning

Module with a multi-path strategy that integrates confidence scores, denoising rules, and historical actions. Using reflection mechanisms, the Planner refines rules and adjusts confidence levels based on interactions, enabling adaptive decision-making that responds to user preferences.

BiLLP [116] employs a two-phase plan refinement approach, wherein the Reflector agent conducts a post-processing analysis of finalized trajectories to derive overarching principles. This methodology facilitates the iterative enhancement of planning strategies without necessitating gradient updates. CheatAgent [98] introduces a mechanism to localize insertion positions, identifying tokens that significantly affect recommendation outcomes. It utilizes LLM-powered agents for semantic perturbations, replacing traditional reinforcement learning agents. Chirag Shah et al. [61] propose a multi-path planning approach that leverages confidence scores, denoising protocols, and historical actions to enhance agent decision-making and adapt to changing user interactions. The framework features real-time reflection mechanisms for evaluating interaction trajectories and improving long-term recommendation strategies.

Effective planning enables agents to demonstrate goal-oriented behavior and manage intricate tasks that necessitate multiple steps or interactions. In contrast to profile and memory components, the planning module primarily relies on the inherent reasoning capabilities of LLMs. However, when confronted with complex tasks, these intrinsic reasoning abilities often fall short of adequately addressing user needs, necessitating the integration of advanced planning algorithms and reasoning techniques to enhance LLM performance. Current training paradigms—such as supervised fine-tuning and reinforcement learning—appear effective in enabling agents to better acquire strategic competencies and thereby improve reasoning. Looking ahead, more deep learning methodologies may emerge that more closely emulate human learning processes, ultimately leading to reasoning mechanisms better suited for autonomous agents.

3.4.4. Action Module

The Action component delineates the array of operations that an LLM-powered agent can execute to engage with its environment or affect the recommendation process. These actions represent the integration of the agent's perceptual capabilities, profiling, memory, and planning mechanisms.

iEvaLM [93] presents a multi-dimensional rating framework that assesses Conversational recommender systems responses based on various criteria, in-

cluding the quality of actions and the relevance of recommendations. This approach facilitates more sophisticated decision-making processes. MACRec [76] improves the Action Module by implementing role-specific specialization among agents. In this framework, various agents, such as the Searcher for information retrieval and the Task Interpreter for intent parsing, work collaboratively to perform recommendation actions through a process of dynamic task decomposition. Shuang et al. [78] propose a methodology using contrastive trajectory pairs to compare human-generated actions with model-generated ones in action selection. This allows the DPO policy to identify optimal actions through relative reward comparisons. Agent4Rec [63] improves its Action Module with a CoT mechanism for emotion-driven decision-making. This allows agents to assess their satisfaction and fatigue levels based on emotions, enabling them to dynamically select actions, like exiting the system, to enhance behavioral realism.

AgentCF [114] enables autonomous interactions between user and item agents through decision-making and collaborative reflection. This ongoing adjustment process aligns actions with user behaviors, improving their relevance and effectiveness over time. SimpleUserSim [101] presents a three-tier mechanism termed "Rating-Action-Response," which is informed by the principles of Bounded Rationality theory. This framework facilitates more sophisticated decision-making processes and allows for tailored responses. MATCHA [92] improves its Action Module by incorporating multi-agent collaborative decision-making. In this framework, specialized agents, such as those responsible for ranking and re-ranking, engage in a dynamic process of refining recommendations through iterative reflection mechanisms. iALP [106] incorporates item preferences generated by LLMs to dynamically determine the most effective actions in the context of recommendations. Its adaptive counterpart, A-iALP, enhances this process by utilizing real-time user feedback and a combination of policies to address the constraints associated with exploration.

The action space of an agent delineates the range of actions it can undertake. The development of a suitable and adequately comprehensive action space is essential for the agent to fulfill its designated function effectively. Unlike casual observers who might assume that an agent's action is limited to recommending items, the actual definition of actions varies significantly based on the agent's state and role, as noted in prior discussions of action sets. For instance, in a multi-agent route recommender system comprising POI agents and their corresponding Manager agent, their action spaces dif-

fer fundamentally: the Manager agent typically handles task planning and operational decomposition, while a POI agent may only be responsible for POI extraction and database queries. Current optimizations of action components are predominantly conducted jointly with planning modules or tailored to specific application scenarios—an approach that mirrors how human behavior in real-world contexts is shaped by both cognitive processes and environmental factors.

4. CHALLENGES FROM REAL-WORLD APPLICATIONS

This section delineates the principal challenges associated with the adaptation of agents to recommender systems. We examine the initial endeavors undertaken by current research, alongside alternative potential solutions. The subsequent challenges are delineated across five dimensions: (1) **Efficiency and Computational Cost**, (2) **Trustworthiness**, (3) **Safety**, (4) **Robustness**, and (5) **Multi-agent Integration**.

4.1. Efficiency and Computational Cost

The issues faced by recommender agents in terms of efficiency and computational costs constitute a vital focus of current research. In recent years, as recommender systems have advanced, the importance of addressing concerns related to efficiency and computational expenditures has grown markedly [139]. Recommender agents are tasked with processing extensive amounts of user data and item information, which substantially heightens the demand for computational resources [140]. For instance, when addressing complex multimodal data, such as the integration of visual, linguistic, and action data, these agents necessitate considerable computational power to facilitate effective model training and inference [141].

Moreover, to enhance accuracy and personalization, agents frequently implement more advanced algorithms and model architectures, which consequently escalate computational expenses [142]. Current methodologies prioritize balancing efficiency and cost through parameter-efficient fine-tuning and data-minimization strategies [143]. Wenyu Zhang et al. [144] introduce a parameter-efficient tuning approach that significantly diminishes the computational costs associated with full-model fine-tuning through staged training and knowledge-guided strategies. Similarly, Ardalan Arabzadeh et al. [145] strategically reduce the volume of training data, achieving approximately a 50% decrease in training computational resources and energy consumption

while maintaining a majority of performance metrics. Furthermore, recommender agents must navigate trade-offs in real-time performance to fulfill users' expectations for prompt recommendations [146]. This necessity compels recommender systems to produce high-quality recommendations swiftly, even within the constraints of limited computational resources, thereby elevating the standards for the efficiency of recommender agents.

In conclusion, the difficulties encountered by recommender agents regarding efficiency and computational costs largely arise from the handling of extensive datasets, the implementation of sophisticated algorithms, and the necessity for real-time processing capabilities. Future research should prioritize the development of lightweight and efficient agent-based recommender systems, aiming to enhance recommendation efficiency while minimizing computational overhead to better align with practical application requirements. The current gap between academic research and industrial application is significantly influenced by real-world constraints, where balancing timeliness with recommendation quality can be framed as a trade-off between computational resources and operational efficiency. In the future, techniques such as knowledge distillation are poised to demonstrate substantially enhanced performance in addressing these challenges.

4.2. Trustworthiness

Trustworthiness is particularly vital, as users are more inclined to depend on recommendations from agents they consider reliable. However, recommending agents that use LLMs as their "brains" means that inherent issues such as biased recommendations and a lack of transparency in the underlying algorithms [147, 148], which can severely undermine user trust, also become factors that the recommending agents need to address. A substantial body of research has indicated that LLMs can severely undermine the reliability of systems functioning in high-risk domains, such as finance and healthcare, particularly when they generate content that, while seemingly plausible, is factually inaccurate. For instance, incorrect stock recommendations or flawed evaluations of market trends can lead to significant financial losses. Wang et al. [149] underscore the occurrence of "hallucinations" in LLMs, particularly within the financial sector, where these models may produce information that appears reasonable but is fundamentally erroneous. This phenomenon poses a considerable threat to the integrity of recommender systems. In the context of financial investments, reliance on such models for providing flawed analyses of market trends or misguided investment advice could result in

substantial financial harm to users. Moreover, Junyi Chen [150] observes that while the incorporation of LLMs into recommender systems has significantly improved the personalization and explainability of recommendations through enhanced textual data processing capabilities, it has also introduced new challenges, including those related to cold start, fairness, and bias. For example, in the healthcare sector, LLMs may generate medically implausible or inaccurate information, which can adversely affect patient care. Current approaches primarily employ multi-agent ensemble learning to address the inherent limitations of LLMs, integrate agents with Retrieval-Augmented Generation (RAG) to mitigate hallucination issues through external knowledge bases, or design sophisticated planning modules for agents using methods such as task decomposition, thereby enhancing the reliability of recommendation outcomes. Ruixin Yang et al. [151] conduct a simulation of the human deliberation process by employing multi-agent group interactions. They utilized tool-enhanced LLM-powered agents to collaboratively refine confidence assessments, which resulted in enhanced accuracy and calibration of decision-making outcomes. It is noteworthy that as LLMs become more powerful, they are paradoxically prone to generating more subtle and harder-to-detect hallucinations or fabrications. When serving as the central processing unit of an agent, the trustworthiness issues, while potentially less acute than in raw LLM applications, remain a critical concern. This is particularly true in domains sensitive to recommendation outcomes. Therefore, ensuring the trustworthiness of these systems will persist as a significant and ongoing challenge that demands continuous attention and improvement.

4.3. Safety

Security remains a major concern, as recommender agents often handle sensitive user data. Data and privacy breaches carry significant consequences for all parties involved [152]. Recent research has underscored the significant susceptibility of recommender systems to poisoning attacks. Malicious actors can introduce meticulously designed "malicious samples" into the training datasets, thereby influencing recommendation outcomes, such as enhancing the visibility of long-tail items or diminishing the prominence of popular content through modifications to labels or metadata. Moreover, comprehensive examinations of the recommendation ecosystem have indicated that such attacks can affect various stages, including recall, ranking, and feedback mechanisms.

Nevertheless, current defensive strategies predominantly concentrate on the identification or mitigation of isolated attack vectors, lacking a cohesive, end-to-end protective framework. Wan et al. [153] identify vulnerabilities in neural code search systems, where the injection of malicious samples into training data can lead to erroneous outputs from the recommender systems. This highlights the security vulnerabilities inherent in recommender agents, which can be exploited through data poisoning, ultimately undermining the reliability of the recommendations provided. Liangxuan Wu et al. [154] further emphasize that LLM-powered agents present eleven distinct attack surfaces, with these security weaknesses potentially resulting in behavioral biases, privacy violations, and even execution hijacking. This situation accentuates the pressing necessity for the establishment of standardized security protocols. Existing research has primarily concentrated on traditional threats, such as adversarial sample attacks [155] and prompt injection attacks, while offering insufficient in-depth analysis of emerging attack vectors, including multimodal attack chains and distributed collaborative attacks. More critically, current security evaluations often depend on static testing methodologies and have yet to implement a continuous assessment framework suitable for dynamic environments, thereby complicating the detection of adaptive attacks that may evolve over time.

In terms of defensive strategies, recent studies have made strides in addressing collaborative federated poisoning attacks (e.g., FedAttack [156]) and metadata manipulation (e.g., Poison-RAG [157]). However, these investigations frequently operate under the assumption of static environments and singular attack pathways. In practice, recommender systems have evolved into complex, multimodal, multistage, and federated processes. There is an urgent requirement to formulate a unified defense framework that encompasses various modalities and stages, alongside dynamic, continuous evaluation and early warning systems. This is particularly vital for mitigating potential cascading risks within LLM-driven chains, as there currently exists no established quantitative evaluation model or early warning standards.

4.4. Robustness

Robustness is essential for recommender agents to effectively manage a variety of data types and user behaviors without experiencing significant declines in performance [158]. In particular, in dynamic environments where user preferences and data patterns frequently fluctuate, the ability of recommender agents to adapt and maintain high-quality recommendations presents

a substantial challenge. Zhang et al. [159] highlight the absence of consistent evaluation methodologies and standardized benchmark datasets within the realm of robust recommender system research. Predominantly, existing approaches evaluate the robustness of recommender systems indirectly by analyzing shifts in performance metrics; however, this method may not reliably reflect true robustness. Furthermore, the random nature of many dataset splits can introduce variability and uncertainty into the evaluation process, resulting in inconsistent performance of recommender agents across different datasets, thereby complicating efforts to ensure their robust functionality in diverse scenarios.

Moreover, recommender systems encounter a multitude of intricate challenges. Variations in data distribution and shifts in user preferences can adversely affect the system's adaptability, leading to diminished accuracy in recommendations [160]. Additionally, noise disturbances and adversarial attacks can distort recommendation outcomes, thereby compromising the system's stability and reliability [158]. With the advent of multimodal and multi-behavior data, there is an increasing demand for enhanced generalization capabilities in recommender systems [161]; however, current systems often struggle to effectively manage such complex data. The absence of dynamic defense mechanisms and early warning systems further hampers the ability of these systems to respond to rapidly evolving threats, ultimately impacting their overall stability.

Contemporary research predominantly emphasizes adversarial modeling as a means to improve the robustness of systems. The study conducted by MG-SPA [162] tackles the challenge of robustness within the context of multi-agent reinforcement learning by employing adversarial control strategies, thereby offering theoretical foundations for addressing state uncertainty. Additionally, MvDN [163] integrates attention mechanisms to refine the feature fusion process, thereby bolstering the model's robustness against noise and malicious attacks through the implementation of an attention correction mechanism.

4.5. Multi-agent Integration

Recent studies have advanced the multi-agent framework considerably, providing novel insights and methodologies for optimization in areas such as recommender systems. This framework improves decision-making and adaptability in intricate environments by facilitating both collaboration and competition among agents. Nevertheless, several challenges persist, including

issues related to scalability, constraints on cross-domain collaborative efforts, and obstacles in achieving effective communication and semantic translation among multiple intelligent agents.

4.5.1. Scalability

In practical multi-agent systems, agents often exhibit diverse functions and strategies. Nevertheless, current scalable multi-agent frameworks encounter challenges in accommodating heterogeneity [164]. To realize zero-shot scalability, where a trained multi-agent model can be effectively utilized for novel tasks involving varying agent quantities, it is essential to incorporate heterogeneity through parameter sharing. R. M. Aratchige et al. [165] conduct an analysis focusing on four fundamental domains: the architecture of LLM multi-agent systems, memory, planning, and technical frameworks. Their findings indicate that while advancements such as hybrid agent architectures and the ReAct planning model have enhanced role allocation and decision-making processes, issues related to scalability and real-time responsiveness continue to limit overall system performance. They propose that improving the system's adaptability and collective resilience can be achieved by refining collaboration mechanisms. Bingyu Yan et al. [166] assert that multi-agent systems leveraging LLMs can promote collective intelligence and facilitate flexible collaboration through communication mechanisms that utilize natural language interaction. However, they note that challenges pertaining to scalability, security, and multimodal integration must be addressed to further the progress of this domain. The current research primarily provides a theoretical framework and technical implementation strategies designed to tackle scalability issues through improvements in algorithmic architecture and task allocation techniques. SS-MARL [167] significantly improves the scalability of extensive multi-agent environments by incorporating safety constraints and utilizing hierarchical policy learning, thereby ensuring system stability. Additionally, Khanh-Tung Tran et al. [168] emphasize that decomposing complex tasks into subtasks and delegating them to specialized agents can diminish overall interaction complexity while preserving the system's flexibility and adaptability.

4.5.2. Cross-Domain Collaborative

Research into the interoperability of cross-domain collaborative weak heterogeneous agent systems remains nascent, primarily characterized by three critical dimensions. From a **technological** perspective, the mechanisms

for semantic understanding and knowledge transfer among agents with disparate architectures are inadequately developed, resulting in a low success rate for existing protocol conversion methods in complex task scenarios [169]. For instance, within the realm of vision-language-action models, numerous challenges persist regarding semantic comprehension and knowledge transfer among various models and agents [140]. In terms of **temporal** considerations, there is a notable deficiency in modeling the dynamic effects associated with long-term collaboration, as the majority of existing studies concentrate solely on short-term interactions, thereby neglecting the time-varying characteristics of system performance. This trend is similarly observed in the domain of vision-language-action models, where research predominantly emphasizes short-term interactions and lacks adequate modeling for sustained collaboration. With respect to **domain-specific** challenges, significant obstacles hinder the integration of cross-industry knowledge, particularly due to the complexities associated with specialized terminologies and business logic prevalent in vertical sectors such as healthcare and finance. For example, in the healthcare sector, the integration of cross-industry knowledge encounters numerous difficulties stemming from the intricate nature of specialized terminologies and business logic. Furthermore, within multi-agent systems, conflicts arising from divergent assumptions among agents can result in inconsistent decision-making when addressing complex tasks.

4.5.3. Information Transfer and Semantic Understanding

A fundamental challenge in multi-agent systems pertains to the efficacy of information transfer, semantic compression, and collaborative decision-making between the primary agent and subordinate agents, as well as among the subordinate agents themselves [170]. Current agents exhibit limitations in task allocation and real-time coordination, with the problem of "cross-agent context transfer" remaining unresolved. For example, in the context of multi-agent reinforcement learning, agents operate within a shared environment and update their strategies in a synchronous manner, resulting in each agent lacking awareness of the complete dynamics of the environment. The subsequent state is influenced not only by the actions and current state of the individual agent but is also altered by the actions of other agents, thereby contravening the Markov assumption [171]. Existing research indicates that the effectiveness of multi-agent collaboration can be systematically enhanced through efficient semantic encoding and dynamic optimization of communication protocols. For example, semantic compression technology can be used to

reduce the dimensions of information transmission, and pre-trained semantic models can achieve efficient encoding and decoding of information, further lowering communication costs [168]. Yuheng Cheng et al. [172] propose that by designing standardized communication protocols, the efficient transmission of information in distributed environments can alleviate communication conflicts among agents.

5. CONCLUSION AND FUTURE PROSPECTS

The integration of LLM-powered agents is ushering in a transformative era for recommender systems [34, 173]. By utilizing the advanced capabilities of LLMs in natural language understanding, reasoning, and generation, these agents are being developed to interpret their environments, make autonomous decisions, plan actions, and engage in complex interactions. This advancement presents innovative solutions to persistent challenges within the recommendation field. This survey has conducted a comprehensive review of the current literature by analyzing the application of LLM-powered agents from four critical perspectives: where they are applied, what their structure is, why they are used, and how they are designed.

- For the **"WHERE"** question, we analyze three integration strategies for LLM agents and recommender systems: Agent as Rec, Agent for Rec, and Agent in Rec.
- For the **"WHAT"** question, we examine the system architecture, categorizing it into single-agent systems and multi-agent systems based on the number of agents involved.
- For the **"WHY"** question, we investigate the underlying rationale for the design of each agent. This analysis reveals four distinct classification criteria: the enhancement of general recommendation tasks, the consideration of vertical and domain-specific scenarios, the facilitation and improvement of interactive recommendations, and the assessment of the performance of recommender systems.
- For the **"HOW"** question, we analyzed the optimization modules of each agent, including the Profile Module, Memory Module, Planning Module, and Action Module.

Regarding future prospects, in addition to the five dimensions discussed in Section 4—specifically, efficiency and computational cost, trustworthiness, safety and robustness, as well as the challenges associated with multi-agent integration—we wish to express our optimistic outlook on the future advancement of the integration of LLM-powered agents with recommender systems:

- **Enhanced Multimodal Integration & Understanding**: Recent studies, including the BoNMF model [174], have started to incorporate multimodal data, such as text and images, but cross-modal semantic alignment remains challenging. User preferences may be expressed in text, while project attributes are derived from visuals, necessitating cohesive alignment. Future research should prioritize cross-modal attention mechanisms and adaptive fusion technologies to reduce information asymmetry between modalities, such as analyzing audio sentiment alongside visual themes in video recommendations. Furthermore, real-time integration of sensor data from wearable devices should be investigated to develop dynamic, multimodal user profiles.

- **Optimization of Dynamic Interactions & User Behavior Simulation**: Current interaction simulations like AgentCF [114] generate basic user behaviors but lack adaptability in complex scenarios, such as sudden changes in user interest. To improve robustness, it is crucial to incorporate causal reasoning methodologies to differentiate between randomness and stability in user behaviors [175] and to develop adversarial training frameworks. Furthermore, incremental learning mechanisms are needed for agents to adjust their conversational strategies and recommendations based on real-time feedback [117], creating an "interaction-optimization" feedback loop.

References

[1] C. Gao, Y. Zheng, N. Li, Y. Li, Y. Qin, J. Piao, Y. Quan, J. Chang, D. Jin, X. He, et al., A survey of graph neural networks for recommender systems: Challenges, methods, and directions, ACM Transactions on Recommender Systems 1 (1) (2023) 1–51.

[2] D. Wang, H. Yao, D. Yu, S. Song, H. Weng, G. Xu, S. Deng, Graph intention embedding neural network for tag-aware recommendation, Neural Networks 184 (2025) 107062.

[3] I. Akdim, L. Mekouar, Y. Iraqi, Trust in recommender systems: A survey, Expert Systems with Applications (2025) 129653.

[4] X. Zhang, H. Weng, Y. Wei, D. Wang, J. Chen, T. Liang, Y. Yin, Multivariate hawkes spatio-temporal point process with attention for point of interest recommendation, Neurocomputing 619 (2025) 129161.

[5] J. Lin, Y. Qu, W. Guo, X. Dai, R. Tang, Y. Yu, W. Zhang, Map: A model-agnostic pretraining framework for click-through rate prediction, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023, pp. 1384–1395.

[6] L. Fu, J. Lin, W. Liu, R. Tang, W. Zhang, R. Zhang, Y. Yu, An f-shape click model for information retrieval on multi-block mobile pages, in: Proceedings of the sixteenth ACM international conference on web search and data mining, 2023, pp. 1057–1065.

[7] M. Verma, V. Parganiha, Privacy-preserving context-aware recommendation system with federated neural collaborative filtering, Knowledge-Based Systems (2025) 114997.

[8] H. Li, X. Zhang, H. Weng, Y. Shen, K. Cai, D. Wang, Z. Qin, S. Deng, Disentangled progressive negative sampling for graph collaborative filtering recommendation, Knowledge-Based Systems (2025) 114133.

[9] J. Qian, S. Song, X. Zhang, D. Wang, H. Weng, H. Zhang, D. Yu, Tcf-mamba: Trajectory collaborative filtering mamba for debiased point-of-interest recommendation, in: Proceedings of the 34th ACM International Conference on Information and Knowledge Management, 2025, pp. 2409–2419.

[10] Q. Yang, S. Wang, D. Guo, D. Yu, Q. Xiao, D. Wang, C. Luo, Cascading multimodal feature enhanced contrast learning for music recommendation, in: 2024 IEEE International Conference on Data Mining (ICDM), IEEE, 2024, pp. 905–910.

[11] S. Bansal, K. Gowda, M. Z. U. Rehman, C. S. Raghaw, N. Kumar, A hybrid filtering for micro-video hashtag recommendation using graph-based deep neural network, Engineering Applications of Artificial Intelligence 138 (2024) 109417.

[12] M. M. Alam, M. Ahmed, Deep learning based collaborative filtering recommendation system, Procedia Computer Science 258 (2025) 2362–2371.

[13] Z. Sun, Q. Guo, J. Yang, H. Fang, G. Guo, J. Zhang, R. Burke, Research commentary on recommendations with side information: A survey and research directions, Electronic Commerce Research and Applications 37 (2019) 100879.

[14] X. Zhu, Y. Wang, H. Gao, W. Xu, C. Wang, Z. Liu, K. Wang, M. Jin, L. Pang, Q. Weng, et al., Recommender systems meet large language model agents: A survey, Foundations and Trends® in Privacy and Security 7 (4) (2025) 247–396.

[15] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al., Gpt-4 technical report, arXiv preprint arXiv:2303.08774 (2023).

[16] J. Huang, K. C.-C. Chang, Towards reasoning in large language models: A survey, in: Findings of the association for computational linguistics: ACL 2023, 2023, pp. 1049–1065.

[17] M. Arslan, S. Munawar, Large language models in building energy applications: a survey, Energy and Buildings (2025) 116800.

[18] A. Zhang, Y. Deng, Y. Lin, X. Chen, J.-R. Wen, T.-S. Chua, Large language model powered agents for information retrieval, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 2989–2992.

[19] S. Cheng, X. Xiao, L. Wang, Deep hashing image retrieval based on cnn and visual transformer network, Applied Soft Computing 177 (2025) 113244.

[20] Y. Zhu, L. Wu, Q. Guo, L. Hong, J. Li, Collaborative large language model for recommender systems, in: Proceedings of the ACM Web Conference 2024, 2024, pp. 3162–3172.

[21] K. Sivamayilvelan, E. Rajasekar, S. Vairavasundaram, S. Balachandran, V. Suresh, Building explainable artificial intelligence for reinforcement learning based debt collection recommender system using

large language models, Engineering Applications of Artificial Intelligence 159 (2025) 111622.

[22] X. Zheng, G. Wang, G. Xu, J. Yang, B. Han, J. Yu, A llm-driven and motif-informed linearizing graph transformer for web api recommendation, Applied Soft Computing 169 (2025) 112547.

[23] L. Li, Y. Zhang, D. Liu, L. Chen, Large language models for generative recommendation: A survey and visionary discussions, in: Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), 2024, pp. 10146–10159.

[24] K. Sakurai, R. Togo, T. Ogawa, M. Haseyama, Llm is knowledge graph reasoner: Llm's intuition-aware knowledge graph reasoning for cold-start sequential recommendation, in: European Conference on Information Retrieval, Springer, 2025, pp. 263–278.

[25] C. Zhang, Y. Jian, Z. Ouyang, S. Vosoughi, Working memory identifies reasoning limits in language models, in: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, 2024, pp. 16896–16922.

[26] X. Huang, W. Liu, X. Chen, X. Wang, H. Wang, D. Lian, Y. Wang, R. Tang, E. Chen, Understanding the planning of llm agents: A survey, arXiv preprint arXiv:2402.02716 (2024).

[27] D. Zhang, W. Li, K. Song, J. Lu, G. Li, L. Yang, S. Li, Memory in large language models: Mechanisms, evaluation and evolution, arXiv preprint arXiv:2509.18868 (2025).

[28] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, et al., A survey on large language model based autonomous agents, Frontiers of Computer Science 18 (6) (2024) 186345.

[29] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou, et al., The rise and potential of large language model based agents: A survey, Science China Information Sciences 68 (2) (2025) 121101.

[30] Y. Zhang, S. Qiao, J. Zhang, T.-H. Lin, C. Gao, Y. Li, A survey of large language model empowered agents for recommendation and search: Towards next-generation information retrieval, arXiv preprint arXiv:2503.05659 (2025).

[31] L. Zhang, X. Fu, Y. Li, J. Chen, Large language model-based agent schema and library for automated building energy analysis and modeling, Automation in Construction 176 (2025) 106244.

[32] A. Zhao, D. Huang, Q. Xu, M. Lin, Y.-J. Liu, G. Huang, Expel: Llm agents are experiential learners, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 38, 2024, pp. 19632–19642.

[33] X. Zhu, Y. Wang, H. Gao, W. Xu, C. Wang, Z. Liu, K. Wang, M. Jin, L. Pang, Q. Weng, et al., Recommender systems meet large language model agents: A survey, Foundations and Trends® in Privacy and Security 7 (4) (2025) 247–396.

[34] Q. Peng, H. Liu, H. Huang, Q. Yang, M. Shao, A survey on llm-powered agents for recommender systems, arXiv preprint arXiv:2502.10050 (2025).

[35] S. Deng, D. Wang, X. Li, G. Xu, Exploring user emotion in microblogs for music recommendation, Expert Systems with Applications 42 (23) (2015) 9284–9293.

[36] D. Wang, X. Zhang, Y. Yin, D. Yu, G. Xu, S. Deng, Multi-view enhanced graph attention network for session-based music recommendation, ACM Transactions on Information Systems 42 (1) (2023) 1–30.

[37] Y. Yang, S. Zhou, H. Weng, D. Wang, X. Zhang, D. Yu, S. Deng, Siamese learning based on graph differential equation for next-poi recommendation, Applied Soft Computing 150 (2024) 111086.

[38] M. Acharya, K. K. Mohbey, Time-aware cross-domain point-of-interest recommendation in social networks, Engineering Applications of Artificial Intelligence 139 (2025) 109630.

[39] S. Raza, M. Rahman, S. Kamawal, A. Toroghi, A. Raval, F. Navah, A. Kazemeini, A comprehensive review of recommender systems: Transitioning from theory to practice, Computer Science Review 59 (2026) 100849.

[40] S. Zhang, L. Yao, A. Sun, Y. Tay, Deep learning based recommender system: A survey and new perspectives, ACM computing surveys (CSUR) 52 (1) (2019) 1–38.

[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[42] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, P. Jiang, Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer, in: Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 1441–1450.

[43] L. Ferreira, D. C. Silva, M. U. Itzazelaia, Recommender systems in cybersecurity, Knowledge and information systems 65 (12) (2023) 5523–5559.

[44] G. Guo, Resolving data sparsity and cold start in recommender systems, in: International conference on user modeling, adaptation, and personalization, Springer, 2012, pp. 361–364.

[45] S. N. Hasan, R. Khatwal, Challenges and issues analysis in cold start recommendation system, in: AIP Conference Proceedings, Vol. 2930, AIP Publishing LLC, 2023, p. 020017.

[46] H. Wang, Dotmat: Solving cold-start problem and alleviating sparsity problem for recommender systems, in: 2022 IEEE 5th International Conference on Electronics Technology (ICET), IEEE, 2022, pp. 1323–1326.

[47] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, A. Mian, A comprehensive overview of large language models, ACM Transactions on Intelligent Systems and Technology 16 (5) (2025) 1–72.

[48] J. Lyu, Z. Ying, N. Shi, J. Xue, Z. Pan, W. Ding, W. Zhou, Harnessing large language models for question answering over complex tables, Engineering Applications of Artificial Intelligence 160 (2025) 111910.

[49] Y. Jiang, Z. Li, B. Song, Fine-tuning large language models in federated learning with fairness-aware prompt selection, Neural Networks (2025) 108160.

[50] H. Zou, Y. Wang, A. Huang, Large language model augmented joint learning framework for entity-relation extraction, Applied Soft Computing (2025) 114094.

[51] B. Sindhu, R. Prathamesh, M. Sameera, S. KumaraSwamy, The evolution of large language model: Models, applications and challenges, in: 2024 international conference on current trends in advanced computing (ICCTAC), IEEE, 2024, pp. 1–8.

[52] D. Sileo, W. Vossen, R. Raymaekers, Zero-shot recommendation as language modeling, in: European conference on information retrieval, Springer, 2022, pp. 223–230.

[53] L. Couto Seller, Í. Sanz Torres, A. Vogel-Fernández, C. González Carballo, P. M. Sánchez Sánchez, A. Carruana Martín, E. de Miguel Ambite, Evaluating compact llms for zero-shot iberian language tasks on end-user devices., Procesamiento del lenguaje natural 75 (2025).

[54] C. Qu, S. Dai, X. Wei, H. Cai, S. Wang, D. Yin, J. Xu, J.-R. Wen, Tool learning with large language models: A survey, Frontiers of Computer Science 19 (8) (2025) 198343.

[55] Y. Zhang, Y. Yang, J. Shu, X. Wen, J. Sang, Agent models: Internalizing chain-of-action generation into reasoning models, arXiv preprint arXiv:2503.06580 (2025).

[56] Y. Xi, W. Liu, J. Lin, X. Cai, H. Zhu, J. Zhu, B. Chen, R. Tang, W. Zhang, Y. Yu, Towards open-world recommendation with knowledge augmentation from large language models, in: Proceedings of the 18th ACM Conference on Recommender Systems, 2024, pp. 12–22.

[57] J. Xiao, L. Wu, F. Zhong, J. Zhang, Knowledge-aware neighbor collaborative multi-relationship multi-interest comparative recommender system for diversified book recommendations, Expert Systems with Applications 297 (2026) 129235.

[58] X. Li, Z. Huang, Z. Wu, C. Wang, Y. Chen, Cross-domain recommendation via knowledge distillation, Knowledge-Based Systems 311 (2025) 113112.

[59] Z. Meng, F. Meng, R. Lin, B. Wu, Lgmcrec: Large language models-augmented light graph model for multi-criteria recommendation, Knowledge-Based Systems (2025) 114219.

[60] L. Chen, Q. Dai, Z. Zhang, X. Feng, M. Zhang, P. Tang, X. Chen, Y. Zhu, Z. Dong, Recusersim: A realistic and diverse user simulator for evaluating conversational recommender systems, in: Companion Proceedings of the ACM on Web Conference 2025, 2025, pp. 133–142.

[61] C. Shah, H. Joho, K. Kaur, P. P. S. Dammu, Dynamic evaluation framework for personalized and trustworthy agents: A multi-session approach to preference adaptability, arXiv preprint arXiv:2504.06277 (2025).

[62] Z. Wang, M. Gao, J. Yu, Y. Hou, S. Sadiq, H. Yin, Ruleagent: Discovering rules for recommendation denoising with autonomous language agents, arXiv preprint arXiv:2503.23374 (2025).

[63] A. Zhang, Y. Chen, L. Sheng, X. Wang, T.-S. Chua, On generative agents in recommendation, in: Proceedings of the 47th international ACM SIGIR conference on research and development in Information Retrieval, 2024, pp. 1807–1817.

[64] X. Wang, X. Wang, Y. Luo, Y. Yu, M. Fan, J. Zhang, L. Ren, Scene-aware vectorized memory multi-agent framework with cross-modal differentiated quantization vlms for visually impaired assistance, Expert Systems with Applications (2025) 130662.

[65] Y. Ding, X. Zhang, S. Amiri, N. Cao, H. Yang, A. Kaminski, C. Esselink, S. Zhang, Integrating action knowledge and llms for task planning and situation handling in open worlds, Autonomous Robots 47 (8) (2023) 981–997.

[66] J. Fang, S. Gao, P. Ren, X. Chen, S. Verberne, Z. Ren, A multi-agent conversational recommender system, arXiv preprint arXiv:2402.01135 (2024).

[67] X. Huang, J. Lian, Y. Lei, J. Yao, D. Lian, X. Xie, Recommender ai agent: Integrating large language models for interactive recommendations, ACM Transactions on Information Systems 43 (4) (2025) 1–33.

[68] E. Zhang, X. Wang, P. Gong, Y. Lin, J. Mao, Usimagent: Large language models for simulating search users, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 2687–2692.

[69] Y. Zhao, J. Wu, X. Wang, W. Tang, D. Wang, M. De Rijke, 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, 2024, pp. 1796–1806.

[70] Y. Cheng, J. Zheng, B. Wu, Q. Ma, Sequential recommendation via agent-based irrelevancy skipping, Neural Networks 185 (2025) 107134.

[71] B. Farhadi, P. Asghari, A. Zamanifar, H. H. S. Javadi, An innovative edge-driven social iot service recommender framework utilizing multi-agent deep reinforcement learning, Knowledge-Based Systems 317 (2025) 113465.

[72] H. Liu, J. Wei, K. Zhu, P. Li, P. Zhao, X. Wu, A multi-agent reinforcement learning framework for cross-domain sequential recommendation, Neural Networks 185 (2025) 107192.

[73] Y. Zeng, A. Rajasekharan, P. Padalkar, K. Basu, J. Arias, G. Gupta, Automated interactive domain-specific conversational agents that understand human dialogs, in: International Symposium on Practical Aspects of Declarative Languages, Springer, 2024, pp. 204–222.

[74] J. Zhang, K. Bao, W. Wang, Y. Zhang, W. Shi, W. Xu, F. Feng, T.-S. Chua, Prospect personalized recommendation on large language model-based agent platform, arXiv preprint arXiv:2402.18240 (2024).

[75] P. Thakkar, A. Yadav, Personalized recommendation systems using multimodal, autonomous, multi agent systems, arXiv preprint arXiv:2410.19855 (2024).

[76] Z. Wang, Y. Yu, W. Zheng, W. Ma, M. Zhang, Macrec: A multi-agent collaboration framework for recommendation, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 2760–2764.

[77] J. Lian, Y. Lei, X. Huang, J. Yao, W. Xu, X. Xie, Recai: Leveraging large language models for next-generation recommender systems, in: Companion Proceedings of the ACM Web Conference 2024, 2024, pp. 1031–1034.

[78] S. Feng, G. Feng, An extremely data-efficient and generative llm-based reinforcement learning agent for recommenders, arXiv preprint arXiv:2408.16032 (2024).

[79] Y. Wang, Z. Jiang, Z. Chen, F. Yang, Y. Zhou, E. Cho, X. Fan, Y. Lu, X. Huang, Y. Yang, Recmind: Large language model powered agent for recommendation, in: Findings of the Association for Computational Linguistics: NAACL 2024, 2024, pp. 4351–4364.

[80] Y. Wu, Y. Peng, J. Yu, R. Lee, Mas4poi: a multi-agents collaboration system for next poi recommendation, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2025, pp. 356–367.

[81] H. Yu, Y. Wu, H. Wang, W. Guo, Y. Liu, Y. Li, Y. Ye, J. Du, E. Chen, Thought-augmented planning for llm-powered interactive recommender agent, arXiv preprint arXiv:2506.23485 (2025).

[82] C. Li, Y. Deng, H. Hu, M.-Y. Kan, H. Li, Chatcrs: Incorporating external knowledge and goal guidance for llm-based conversational recommender systems, in: Findings of the Association for Computational Linguistics: NAACL 2025, 2025, pp. 295–312.

[83] J. Park, M. Kim, Madrec: A multi-aspect driven llm agent for explainable and adaptive recommendation, arXiv preprint arXiv:2510.13371 (2025).

[84] H. Cai, Y. Li, W. Wang, F. Zhu, X. Shen, W. Li, T.-S. Chua, Large language models empowered personalized web agents, in: Proceedings of the ACM on Web Conference 2025, 2025, pp. 198–215.

[85] X. Xiao, Mmagentrec, a personalized multi-modal recommendation agent with large language model, Scientific Reports 15 (1) (2025) 12062.

[86] J. Long, S. Huang, H. Huo, T. Chen, H. Yin, G. Xu, Cloud-device collaborative agents for sequential recommendation, arXiv preprint arXiv:2509.01551 (2025).

[87] C. Wu, R. Ren, J. Zhang, R. Wang, Z. Ma, Q. Ye, W. X. Zhao, Starec: An efficient agent framework for recommender systems via autonomous deliberate reasoning, in: Proceedings of the 34th ACM International Conference on Information and Knowledge Management, 2025, pp. 3355–3365.

[88] V. Raman, A. Ragav, et al., Remi: A novel causal schema memory architecture for personalized lifestyle recommendation agents, arXiv preprint arXiv:2509.06269 (2025).

[89] T. Zhe, R. Liu, F. Memar, X. Luo, W. Fan, X. Ye, Z. Peng, D. Wang, Constraint-aware route recommendation from natural language via hierarchical llm agents, arXiv preprint arXiv:2510.06078 (2025).

[90] M. Yang, N. Choudhary, J. Du, E. W. Huang, P. S. Yu, K. Subbian, D. Kourta, Agentdr dynamic recommendation with implicit item-item relations via llm-based agents, arXiv preprint arXiv:2510.05598 (2025).

[91] S. H. Cho, Y. Yang, D. Baeck, M. Kim, Y.-M. Kim, H. Lee, S. Park, Marc: Multimodal and multi-task agentic retrieval-augmented generation for cold-start recommender system, arXiv preprint arXiv:2511.08181 (2025).

[92] Z. Hui, X. Wei, Y. Jiang, K. Gao, C. Wang, F. Ong, S.-e. Yoon, R. Pareek, M. Gong, Matcha: Can multi-agent collaboration build a trustworthy conversational recommender?, arXiv preprint arXiv:2504.20094 (2025).

[93] X. Wang, X. Tang, W. X. Zhao, J. Wang, J.-R. Wen, Rethinking the evaluation for conversational recommendation in the era of large language models, arXiv preprint arXiv:2305.13112 (2023).

[94] J. Jin, X. Chen, F. Ye, M. Yang, Y. Feng, W. Zhang, Y. Yu, J. Wang, Lending interaction wings to recommender systems with conversational

agents, Advances in Neural Information Processing Systems 36 (2023) 27951–27979.

[95] Y. Shu, H. Zhang, H. Gu, P. Zhang, T. Lu, D. Li, N. Gu, Rah! recsys–assistant–human: A human-centered recommendation framework with llm agents, IEEE Transactions on Computational Social Systems 11 (5) (2024) 6759–6770.

[96] V. Malik, A. Jagatap, V. S. Puranik, A. Majumder, Pearl: Preference extraction with exemplar augmentation and retrieval with llm agents, in: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, 2024, pp. 1536–1547.

[97] Z. Zhang, S. Liu, Z. Liu, R. Zhong, Q. Cai, X. Zhao, C. Zhang, Q. Liu, P. Jiang, Llm-powered user simulator for recommender system, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 39, 2025, pp. 13339–13347.

[98] L.-b. Ning, S. Wang, W. Fan, Q. Li, X. Xu, H. Chen, F. Huang, Cheatagent: Attacking llm-empowered recommender systems via llm agent, in: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2024, pp. 2284–2295.

[99] S. Kim, T. Kim, K. Seo, J. Yeo, D. Lee, Stop playing the guessing game! target-free user simulation for evaluating conversational recommender systems, arXiv preprint arXiv:2411.16160 (2024).

[100] S.-e. Yoon, Z. He, J. Echterhoff, J. McAuley, Evaluating large language models as generative user simulators for conversational recommendation, in: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), 2024, pp. 1490–1504.

[101] L. Zhu, X. Huang, J. Sang, How reliable is your simulator? analysis on the limitations of current llm-based user simulators for conversational recommendation, in: Companion Proceedings of the ACM Web Conference 2024, 2024, pp. 1726–1732.

[102] N. Corecco, G. Piatti, L. A. Lanzendörfer, F. X. Fan, R. Wattenhofer, Suber: An rl environment with simulated human behavior for recommender systems, arXiv preprint arXiv:2406.01631 (2024).

[103] L. Zhu, X. Huang, J. Sang, A llm-based controllable, scalable, human-involved user simulator framework for conversational recommender systems, in: Proceedings of the ACM on Web Conference 2025, 2025, pp. 4653–4661.

[104] Z. Wang, M. Gao, J. Yu, X. Gao, Q. V. H. Nguyen, S. Sadiq, H. Yin, Id-free not risk-free: Llm-powered agents unveil risks in id-free recommender systems, in: Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2025, pp. 1902–1911.

[105] D. Ebrat, E. Paradalis, L. Rueda, Lusifer: Llm-based user simulated feedback environment for online recommender systems, in: 2025 IEEE 4th International Conference on Computing and Machine Intelligence (ICMI), IEEE, 2025, pp. 1–6.

[106] J. Wang, A. Karatzoglou, I. Arapakis, J. M. Jose, Large language model driven policy exploration for recommender systems, in: Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining, 2025, pp. 107–116.

[107] W. Xu, Y. Shi, Z. Liang, X. Ning, K. Mei, K. Wang, X. Zhu, M. Xu, Y. Zhang, iagent: Llm agent as a shield between user and recommender systems, arXiv preprint arXiv:2502.14662 (2025).

[108] L. Wang, J. Zhang, H. Yang, Z.-Y. Chen, J. Tang, Z. Zhang, X. Chen, Y. Lin, H. Sun, R. Song, et al., User behavior simulation with large language model-based agents, ACM Transactions on Information Systems 43 (2) (2025) 1–37.

[109] J. Chen, K. Yao, R. Y. Maragheh, K. Zhao, J. Xu, J. Cho, E. Korpeoglu, S. Kumar, K. Achan, Carts: Collaborative agents for recommendation textual summarization, arXiv preprint arXiv:2506.17765 (2025).

[110] J. Liu, Y. Shao, P. Zhang, D. Li, H. Gu, C. Chen, L. Du, T. Lu, N. Gu, Filtering discomforting recommendations with large language models, in: Proceedings of the ACM on Web Conference 2025, 2025, pp. 3639–3650.

[111] X. Ye, C. Xu, Z. Sun, J. Xu, G. Wang, Z. Dong, J.-R. Wen, Creagent: Towards long-term evaluation of recommender system under platform-creator information asymmetry, arXiv preprint arXiv:2502.07307 (2025).

[112] S. Yang, Z. Hu, X. Li, C. Wang, T. Yu, X. Xu, L. Zhu, L. Yao, Drunkagent: Stealthy memory corruption in llm-powered recommender agents, arXiv preprint arXiv:2503.23804 (2025).

[113] Z. Ren, N. Huang, Y. Wang, P. Ren, J. Ma, J. Lei, X. Shi, H. Luo, J. Jose, X. Xin, Contrastive state augmentations for reinforcement learning-based recommender systems, in: Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval, 2023, pp. 922–931.

[114] J. Zhang, Y. Hou, R. Xie, W. Sun, J. McAuley, W. X. Zhao, L. Lin, J.-R. Wen, Agentcf: Collaborative learning with autonomous language agents for recommender systems, in: Proceedings of the ACM Web Conference 2024, 2024, pp. 3679–3689.

[115] J. Wang, A. Karatzoglou, I. Arapakis, J. M. Jose, Reinforcement learning-based recommender systems with large language models for state reward and action modeling, in: Proceedings of the 47th International ACM SIGIR conference on research and development in information retrieval, 2024, pp. 375–385.

[116] W. Shi, X. He, Y. Zhang, C. Gao, X. Li, J. Zhang, Q. Wang, F. Feng, 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, 2024, pp. 1893–1903.

[117] S. Cai, J. Zhang, K. Bao, C. Gao, Q. Wang, F. Feng, X. He, Agentic feedback loop modeling improves recommendation and user simulation, in: Proceedings of the 48th International ACM SIGIR conference on Research and Development in Information Retrieval, 2025, pp. 2235–2244.

[118] T. Guo, C. Liu, H. Wang, V. Mannam, F. Wang, X. Chen, X. Zhang, C. K. Reddy, Knowledge graph enhanced language agents for recommendation, arXiv preprint arXiv:2410.19627 (2024).

[119] S. Chen, B. Chen, C. Yu, Y. Luo, O. Yi, L. Cheng, C. Zhuo, Z. Li, Y. Wang, Vragent-r1: Boosting video recommendation with mllm-based agents via reinforcement learning, arXiv preprint arXiv:2507.02626 (2025).

[120] R. Y. Maragheh, P. Vadla, P. Gupta, K. Zhao, A. Inan, K. Yao, J. Xu, P. Kanumala, J. Cho, S. Kumar, Arag: Agentic retrieval augmented generation for personalized recommendation, arXiv preprint arXiv:2506.21931 (2025).

[121] J. Liu, S. Gu, D. Li, G. Zhang, M. Han, H. Gu, P. Zhang, T. Lu, L. Shang, N. Gu, 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, 2025, pp. 2566–2571.

[122] W. Mao, J. Wu, W. Chen, C. Gao, X. Wang, X. He, Reinforced prompt personalization for recommendation with large language models, ACM Transactions on Information Systems 43 (3) (2025) 1–27.

[123] W.-C. Kang, J. McAuley, Self-attentive sequential recommendation, in: 2018 IEEE international conference on data mining (ICDM), IEEE, 2018, pp. 197–206.

[124] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, Lightgcn: Simplifying and powering graph convolution network for recommendation, in: Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, 2020, pp. 639–648.

[125] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, C. Finn, Direct preference optimization: Your language model is secretly a reward model, Advances in neural information processing systems 36 (2023) 53728–53741.

[126] D. Amin, J. Salminen, B. J. Jansen, J. Shin, D. H. Kim, Generative ai personas considered harmful? putting forth twenty challenges of algorithmic user representation in human-computer interaction, International Journal of Human-Computer Studies (2025) 103657.

[127] M. A. Ferrag, N. Tihanyi, M. Debbah, Reasoning beyond limits: Advances and open problems for llms, ICT express (2025).

[128] I. O'Flynn, H. Šiljak, Emergence of roles in robotic teams with model sharing and limited communication, arXiv preprint arXiv:2505.00540 (2025).

[129] H. P. Zou, W.-C. Huang, Y. Wu, Y. Chen, C. Miao, H. Nguyen, Y. Zhou, W. Zhang, L. Fang, L. He, et al., Llm-based human-agent collaboration and interaction systems: A survey.

[130] E. Feng, W. Zhou, Z. Liu, L. Chen, Y. Dong, C. Zhang, Y. Zhao, D. Du, Z. Hua, Y. Xia, et al., Get experience from practice: Llm agents with record & replay, arXiv preprint arXiv:2505.17716 (2025).

[131] H. Jin, L. Huang, H. Cai, J. Yan, B. Li, H. Chen, From llms to llm-based agents for software engineering: A survey of current, challenges and future, arXiv preprint arXiv:2408.02479 (2024).

[132] S. Chen, Y. Liu, W. Han, W. Zhang, T. Liu, A survey on llm-based multi-agent system: Recent advances and new frontiers in application, arXiv preprint arXiv:2412.17481 (2024).

[133] G. Nie, R. Zhi, X. Yan, Y. Du, X. Zhang, J. Chen, M. Zhou, H. Chen, T. Li, Z. Cheng, et al., A hybrid multi-agent conversational recommender system with llm and search engine in e-commerce, in: Proceedings of the 18th ACM Conference on Recommender Systems, 2024, pp. 745–747.

[134] Q. Cai, L. Niu, Agent-based personalized assessment tasks recommendation considering objective and subjective factors, Ieee Access 11 (2023) 44377–44390.

[135] X. Li, X. Lei, X. Liu, H. Xiao, Collision-free parking recommendation based on multi-agent reinforcement learning in vehicular crowdsensing, Digital Communications and Networks 10 (3) (2024) 609–619.

[136] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al., Chain-of-thought prompting elicits reasoning in large language models, Advances in neural information processing systems 35 (2022) 24824–24837.

[137] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, K. Narasimhan, Tree of thoughts: Deliberate problem solving with large language

models, Advances in neural information processing systems 36 (2023) 11809–11822.

[138] J. Arias, M. Carro, E. Salazar, K. Marple, G. Gupta, Constraint answer set programming without grounding, Theory and Practice of Logic Programming 18 (3-4) (2018) 337–354.

[139] S. Raza, M. Rahman, S. Kamawal, A. Toroghi, A. Raval, F. Navah, A. Kazemeini, A comprehensive review of recommender systems: Transitioning from theory to practice, Computer Science Review 59 (2026) 100849.

[140] R. Sapkota, Y. Cao, K. I. Roumeliotis, M. Karkee, Vision-language-action models: Concepts, progress, applications and challenges, arXiv preprint arXiv:2505.04769 (2025).

[141] Q. Liu, J. Hu, Y. Xiao, X. Zhao, J. Gao, W. Wang, Q. Li, J. Tang, Multimodal recommender systems: A survey, ACM Computing Surveys 57 (2) (2024) 1–17.

[142] Z. Zhao, W. Fan, J. Li, Y. Liu, X. Mei, Y. Wang, Z. Wen, F. Wang, X. Zhao, J. Tang, et al., Recommender systems in the era of large language models (llms), IEEE Transactions on Knowledge and Data Engineering 36 (11) (2024) 6889–6907.

[143] A. Arabzadeh, Optimal dataset size for recommender systems: Evaluating algorithms' performance via downsampling, arXiv preprint arXiv:2502.08845 (2025).

[144] W. Zhang, J. Luo, X. Zhang, Y. Fang, Bridging domain gaps between pretrained multimodal models and recommendations, arXiv preprint arXiv:2502.15542 (2025).

[145] A. Arabzadeh, T. Vente, J. Beel, Green recommender systems: Optimizing dataset size for energy-efficient algorithm performance, in: International Workshop on Recommender Systems for Sustainability and Social Good, Springer, 2024, pp. 73–82.

[146] Y. Ye, Z. Zheng, Y. Shen, T. Wang, H. Zhang, P. Zhu, R. Yu, K. Zhang, H. Xiong, Harnessing multimodal large language models for multimodal

sequential recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 39, 2025, pp. 13069–13077.

[147] S. Wang, X. Zhang, Y. Wang, F. Ricci, Trustworthy recommender systems, ACM Transactions on Intelligent Systems and Technology 15 (4) (2024) 1–20.

[148] Y. Ge, S. Liu, Z. Fu, J. Tan, Z. Li, S. Xu, Y. Li, Y. Xian, Y. Zhang, A survey on trustworthy recommender systems, ACM Transactions on Recommender Systems 3 (2) (2024) 1–68.

[149] K. Wang, G. Zhang, Z. Zhou, J. Wu, M. Yu, S. Zhao, C. Yin, J. Fu, Y. Yan, H. Luo, et al., A comprehensive survey in llm (-agent) full stack safety: Data, training and deployment, arXiv preprint arXiv:2504.15585 (2025).

[150] J. Chen, A survey on large language models for personalized and explainable recommendations, arXiv preprint arXiv:2311.12338 (2023).

[151] R. Yang, D. Rajagopal, S. A. Hayati, B. Hu, D. Kang, Confidence calibration and rationalization for llms via multi-agent deliberation, arXiv preprint arXiv:2404.09127 (2024).

[152] Z. Wang, J. Yu, T. Chen, H. Yin, S. Sadiq, M. Gao, Towards secure and robust recommender systems: A data-centric perspective, in: Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining, 2025, pp. 1020–1023.

[153] Z. Yang, Z. Sun, T. Z. Yue, P. Devanbu, D. Lo, Robustness, security, privacy, explainability, efficiency, and usability of large language models for code, arXiv preprint arXiv:2403.07506 (2024).

[154] L. Wu, C. Wang, T. Liu, Y. Zhao, H. Wang, From assistants to adversaries: Exploring the security risks of mobile llm agents, arXiv preprint arXiv:2505.12981 (2025).

[155] K. Zhang, Q. Cao, Y. Wu, F. Sun, H. Shen, X. Cheng, Improving the shortest plank: Vulnerability-aware adversarial training for robust recommender system, in: Proceedings of the 18th ACM Conference on Recommender Systems, 2024, pp. 680–689.

[156] T. T. Nguyen, N. Quoc Viet Hung, T. T. Nguyen, T. T. Huynh, T. T. Nguyen, M. Weidlich, H. Yin, Manipulating recommender systems: A survey of poisoning attacks and countermeasures, ACM Computing Surveys 57 (1) (2024) 1–39.

[157] F. Nazary, Y. Deldjoo, T. d. Noia, Poison-rag: Adversarial data poisoning attacks on retrieval-augmented generation in recommender systems, in: European Conference on Information Retrieval, Springer, 2025, pp. 239–251.

[158] L. Cheng, X. Huang, J. Sang, J. Yu, Towards robust recommendation: A review and an adversarial robustness evaluation library, IEEE Transactions on Knowledge and Data Engineering (2025).

[159] K. Zhang, Q. Cao, F. Sun, Y. Wu, S. Tao, H. Shen, X. Cheng, Robust recommender system: a survey and future directions, ACM Computing Surveys 58 (1) (2025) 1–38.

[160] X. Zhao, B. Hu, Y. Zhong, S. Huang, Z. Zheng, M. Wang, H. Wang, M. Zhang, Raserec: Retrieval-augmented sequential recommendation, arXiv preprint arXiv:2412.18378 (2024).

[161] W. Wei, L. Xia, C. Huang, Multi-relational contrastive learning for recommendation, in: Proceedings of the 17th ACM conference on recommender systems, 2023, pp. 338–349.

[162] S. He, S. Han, S. Su, S. Han, S. Zou, F. Miao, Robust multi-agent reinforcement learning with state uncertainty, arXiv preprint arXiv:2307.16212 (2023).

[163] L. Yuan, F. Chen, Z. Zhang, Y. Yu, Communication-robust multi-agent learning by adaptable auxiliary multi-agent adversary generation, Frontiers of Computer Science 18 (6) (2024) 186331.

[164] X. Guo, D. Shi, J. Yu, W. Fan, Heterogeneous multi-agent reinforcement learning for zero-shot scalable collaboration, Neurocomputing (2025) 130716.

[165] R. Aratchige, W. Ilmini, Llms working in harmony: A survey on the technological aspects of building effective llm-based multi agent systems, arXiv preprint arXiv:2504.01963 (2025).

[166] B. Yan, Z. Zhou, L. Zhang, L. Zhang, Z. Zhou, D. Miao, Z. Li, C. Li, X. Zhang, Beyond self-talk: A communication-centric survey of llm-based multi-agent systems, arXiv preprint arXiv:2502.14321 (2025).

[167] H. Du, F. Gou, Y. Cai, Scalable safe multi-agent reinforcement learning for multi-agent system, arXiv preprint arXiv:2501.13727 (2025).

[168] K.-T. Tran, D. Dao, M.-D. Nguyen, Q.-V. Pham, B. O'Sullivan, H. D. Nguyen, Multi-agent collaboration mechanisms: A survey of llms, arXiv preprint arXiv:2501.06322 (2025).

[169] M. Zhan, M. Xu, W. Lin, H. He, C. He, Graphene oxide research: Current developments and future directions, Nanomaterials 15 (7) (2025) 507.

[170] R. Zhang, J. Hou, F. Walter, S. Gu, J. Guan, F. Röhrbein, Y. Du, P. Cai, G. Chen, A. Knoll, Multi-agent reinforcement learning for autonomous driving: A survey, arXiv preprint arXiv:2408.09675 (2024).

[171] H. Su, R. Chen, S. Tang, Z. Yin, X. Zheng, J. Li, B. Qi, Q. Wu, H. Li, W. Ouyang, et al., Many heads are better than one: Improved scientific idea generation by a llm-based multi-agent system, in: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2025, pp. 28201–28240.

[172] Y. Cheng, C. Zhang, Z. Zhang, X. Meng, S. Hong, W. Li, Z. Wang, Z. Wang, F. Yin, J. Zhao, et al., Exploring large language model based intelligent agents: Definitions, methods, and prospects, arXiv preprint arXiv:2401.03428 (2024).

[173] C. Huang, H. Huang, T. Yu, K. Xie, J. Wu, S. Zhang, J. Mcauley, D. Jannach, L. Yao, A survey of foundation model-powered recommender systems: From feature-based, generative to agentic paradigms, arXiv preprint arXiv:2504.16420 (2025).

[174] A. Xiang, B. Huang, X. Guo, H. Yang, T. Zheng, A neural matrix decomposition recommender system model based on the multimodal large language model, in: Proceedings of the 2024 7th International Conference on Machine Learning and Machine Intelligence (MLMI), 2024, pp. 146–150.

[175] N. Sukiennik, H. Wang, Z. Zeng, C. Gao, Y. Li, Simulating filter bubble on short-video recommender system with large language model agents, arXiv preprint arXiv:2504.08742 (2025).