CogRec: A Cognitive Recommender Agent Fusing Large Language Models and Soar for Explainable Recommendation

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have demonstrated a remarkable capacity in understanding user preferences for recommendation systems. However, they are constrained by several critical challenges, including their inherent "Black-Box" characteristics, susceptibility to knowledge hallucination, and limited online learning capacity. These factors compromise their trustworthiness and adaptability. Conversely, cognitive architectures such as Soar offer structured and interpretable reasoning processes, yet their knowledge acquisition is notoriously laborious. To address these complementary challenges, we propose a novel cognitive recommender agent called CogRec which synergizes the strengths of LLMs with the Soar cognitive architecture. CogRec leverages Soar as its core symbolic reasoning engine and leverages an LLM for knowledge initialization to populate its working memory with production rules. The agent operates on a Perception-Cognition-Action(PCA) cycle. Upon encountering an impasse, it dynamically queries the LLM to obtain a reasoned solution. This solution is subsequently transformed into a new symbolic production rule via Soar's chunking mechanism, thereby enabling robust online learning. This learning paradigm allows the agent to continuously evolve its knowledge base and furnish highly interpretable rationales for its recommendations. Extensive evaluations conducted on three public datasets demonstrate that CogRec demonstrates significant advantages in recommendation accuracy, explainability, and its efficacy in addressing the long-tail problem.

1 Introduction

Recommender systems have become a cornerstone of the modern digital information ecosystem, profoundly influencing users' decision-making processes. The recent ascendancy of LLMs (Zhao et al., 2023; Wu et al., 2024) has reinvigorated this field. Leveraging their vast world knowledge acquired from massive text corpora and their powerful contextual understanding, LLMs have demonstrated unprecedented potential in capturing nuanced and dynamic user preferences (Geng et al., 2022a; Cui et al., 2022). Techniques such as Chain of Thought (CoT) (Wei et al., 2022; Lyu et al., 2023; Xia et al., 2025) have enabled LLMs to generate seemingly plausible recommendation explanations, marking a significant step towards more transparent systems.

Despite these significant advancements, the current LLM-based recommendation paradigm confronts fundamental challenges: (1)**"Black-Box" Reasoning**. Although methods like CoT can produce fluent explanatory text, their reasoning process is inherently generative rather than structured. It lacks rigorous logical constraints and state tracking, making it prone to "knowledge hallucination"—generating outputs that are factually incorrect or logically inconsistent, especially in complex, multi-step decision-making scenarios (Zhang et al., 2025; Ji et al., 2023). (2)**Limited online learning**. the knowledge within LLMs is embedded in their vast parameters, making real-time, efficient updates from sparse new interaction data difficult (Zheng et al., 2025; Pope et al., 2023; Meng et al., 2022). This renders them ill-suited for adapting to rapidly changing user preferences or addressing the cold-start problem.

In stark contrast, cognitive architectures, exemplified by Soar (Laird, 2019), offer a distinct alternative. Soar is inherently characterized by several advantages stemming from its production rule-based reasoning engine, explicit goal hierarchy, and its ”chunking” mechanism for learning new rules from experience. It naturally possesses advantages such as high interpretability, strong online learning ability, and efficient symbolic computing. However, Soar's knowledge acquisition bottleneck is pronounced: initial rules must be manually encoded, a process that is both laborious and difficult to scale to complex domains. These complementary challenges motivate the exploration of neuro-symbolic integration (Garcez and Lamb, 2023; Pan et al., 2024). We observe that the semantic understanding and knowledge generation capabilities of LLMs can address Soar's knowledge initialization problem, while Soar's symbolic reasoning and online learning can mitigate the black-box and hallucination issues of LLMs.

To this end, this paper proposes CogRec, a cognitive recommender agent that deeply integrates LLMs with Soar. CogRec designates Soar as its core "cognitive brain," responsible for symbolic-level reasoning and learning. It utilizes LLMs as an external knowledge source for initial rule generation and for dynamic querying during an impasse. An innovative Bridge Module facilitates bidirectional neuro-symbolic translation: extracting symbolic rules from the natural language outputs of LLMs, and generating precise prompts from Soar's state to guide the LLMs. This framework operates on a PCA cycle, supports online evolution, and outputs traceable recommendation explanations.The major contributions of this paper are summarized as follows:

- We propose CogRec, the first framework to synergize LLMs and the Soar cognitive architecture, delivering an end-to-end solution for explainable recommendation.
- We design a neuro-symbolic bridge module for efficient, bidirectional knowledge translation, which enhances system robustness and enables real-time online learning.
- Extensive experiments on public datasets validate CogRec's superior performance in accuracy and explainability, along with its effectiveness in mitigating the long-tail issue.

2 Related Work

2.1 LLM for Recommendation

The advent of LLMs has ushered in a paradigm shift in the field of recommender systems (Wu et al., 2024; Zhao et al., 2023). Initial research primarily centered on leveraging the powerful semantic representation capabilities of LLMs. Researchers employed LLMs to encode textual descriptions of items or user reviews, thereby augmenting the feature representations of conventional recommender models, such as matrix factorization or graph neural networks (Li et al., 2023a,c; Xu et al., 2024). This approach proved effective in mitigating data sparsity and enhancing performance (Wang and Lim, 2023).

As LLMs have grown in scale and capability, the research focus has shifted towards harnessing their generative and reasoning abilities. A prominent line of work has recast the recommendation task as a sequence-to-sequence language generation problem. For instance, P5 (Geng et al., 2022a) and TALLRec (Bao et al., 2023) introduced a unified pre-training framework that consolidated diverse recommendation sub-tasks—such as sequential recommendation, rating prediction, and explanation generation—under a singular text-to-text paradigm. Such approaches capitalize on the contextual understanding of LLMs, demonstrating significant promise in zero-shot and few-shot recommendation scenarios (Li et al., 2024b; Hollmann et al., 2022). More recently, researchers have begun to explore the use of LLMs as decision-making agents. These agents are designed to handle complex recommendation tasks by engaging in multi-step reasoning and interacting with external tools, such as search engines and databases (Wang et al., 2023; Zhao et al., 2024; Yao et al., 2022).

2.2 Cognitive-enhanced Recommendation

Cognitive-enhanced recommender systems seek to construct models that better align with human decision-making processes by drawing upon theories from cognitive science, thereby enhancing both recommendation accuracy and explainability (Beheshti et al., 2020; Von Rueden et al., 2021). Early works primarily concentrated on modeling specific cognitive concepts, such as dual-process theory (Chen et al., 2023), introducing attention mechanisms to simulate user attention allocation (Zhou et al., 2018) or employing memory networks to emulate users' short- and long-term memory (Cho et al., 2023; Shin et al., 2025). Another line of research has focused on modeling user intent, attempting to disentangle multiple latent intents from user behavior sequences to better comprehend the motivations behind user actions (Ma et al., 2019; Li et al., 2023c; Cen et al., 2020).

The integration of symbolic reasoning into recommendation represents another significant research direction. Several studies have incorporated Knowledge Graphs (KGs) into recommender models, leveraging path-based reasoning over the graph to furnish interpretable rationales for recommendations (Wang et al., 2019; Li et al., 2025). For instance, PGPR (Xian et al., 2019) and CKAN (Wei et al., 2023) employ reinforcement learning and attention networks to discover inferential paths from users to items within the KG. In contrast to these methods, the impasse-driven online learning paradigm designed for CogRec enables the agent to learn and evolve with greater autonomy and continuity.

3 Preliminary

The problem can be formally defined as follows: Given a set of users, denoted by $\mathcal{U}$, and a set of items, denoted by $\mathcal{I}$, the historical interaction data is represented by a matrix $\mathbf{R} \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{I}|}$, where r_{ui} denotes the preference of user $u \in \mathcal{U}$ for item $i \in \mathcal{I}$ (e.g., a rating or a click). For a target user u, the system's task is to generate a ranked list of recommendations $\mathcal{L}_u = [i_1, i_2, \ldots, i_k]$, where each item $i_j \in \mathcal{I}$ is an item not previously interacted with by user u.

The cognitive recommendation task proposed in this paper transcends simple item prediction. We aim to construct a cognitive agent whose task is twofold:

- **Next-Item Prediction**. To generate a high-quality list of candidate recommendations $\mathcal{L}_u$.

- **Generation of Explainable Reasoning Traces**. To produce the symbolic and traceable decision-making process, Eu, that leads to the recommendations.

Therefore, the ultimate objective of the CogRec agent is to learn a function f such that:

$$f : (\mathcal{U}, \mathcal{I}, \mathbf{R}) \rightarrow (\mathcal{L}_u, \mathcal{E}_u) \tag{1}$$

where $\mathcal{E}_u = [e_1, e_2, \ldots, e_k]$ is the set of explanations corresponding to the recommendation list $\mathcal{L}_u$. This function, given a user u, outputs a tuple $(\mathcal{L}_u, \mathcal{E}_u)$ comprising the recommended items and their corresponding reasoning traces, thereby simultaneously optimizing for both recommendation Accuracy and Transparency.

4 Methodology

4.1 Framework Overview

The core tenet of CogRec is to build a recommender agent that emulates human cognitive patterns. Rather than employing an LLM as a monolithic, end-to-end recommender, CogRec designates Soar as its core ”cognitive brain,” tasked with reasoning, decision-making, and learning. The LLM, in turn, serves as a powerful, on-demand external source of knowledge and guidance. As illustrated in Figure 1, the operational workflow of CogRec follows a PCA feedback loop:

- **Perception**. The user interface receives the user's historical interaction sequence S_u, and the current query q_u. This information is first fed into the LLM module for initial semantic encoding and parsing, which translates the raw input into a symbolic state comprehensible to Soar.

- **Cognition**. The Soar core receives this symbolic state and updates its Working Memory. It then leverages the production rules within its procedural memory to reason about the recommendation task, such as decomposing the problem, filtering candidates, and making decisions. If its existing knowledge is sufficient, the process proceeds directly to the action phase. However, if a knowledge gap or conflict is encountered, an impasse is triggered. This is the critical juncture where CogRec's online learning occurs.

- **Action & Learning**. The Soar core ultimately outputs a specific operator, leading to a recommendation. When an impasse occurs, Soar initiates a structured query to the LLM via the bridge module. The LLM's response, after being parsed by the bridge module, is transformed into a new production rule. This new rule is then permanently internalized into Soar's procedural memory through the chunking mechanism. This entire process actualizes the agent's capacity for online learning and evolution.

4.2 Knowledge Generation and Encoding

We employ the Deepseek-V3 model as the foundational component of our LLM module, where it serves a dual role.

4.2.1 Initial Semantic Encoder.

For a given user sequence S_u and query q_u, the LLM first transforms them into high-dimensional semantic vectors. Concurrently, it extracts key entities and intents, furnishing a rich contextual foun-

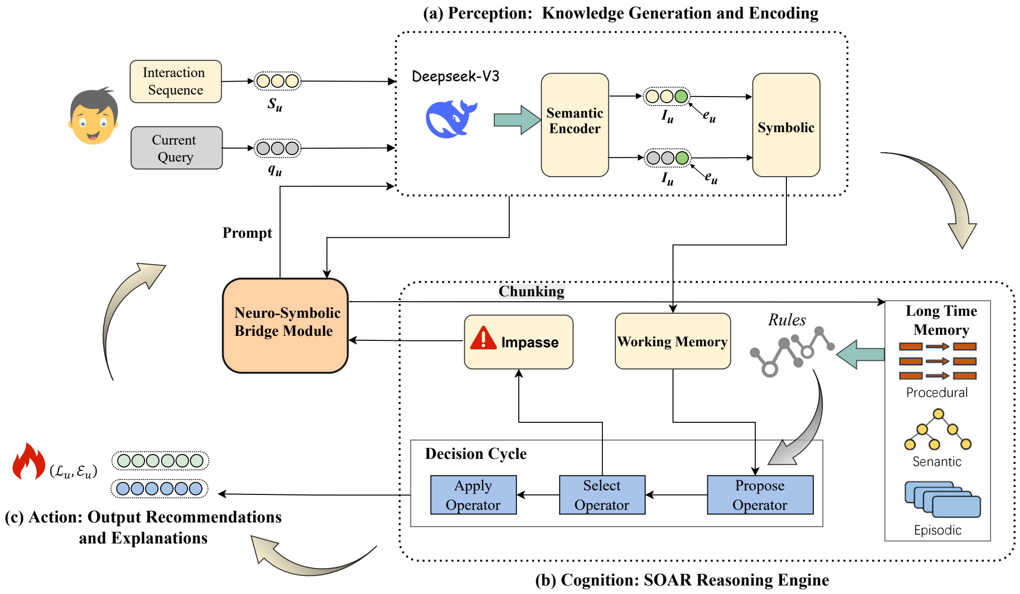


Figure 1: Overview of the CogRec Framework. The figure illustrates the Soar-centric cognitive cycle, with the LLM serving as an external knowledge module. Neuro-symbolic interaction and learning are facilitated by the Bridge Module. The information flow begins with user input, proceeds through LLM-based encoding into the Soar core for reasoning, and culminates in the output of recommendations and their corresponding explanations.

dation for Soar's subsequent symbolic processing. This can be formalized as:

$$\mathbf{e}_u, I_u = \text{LLM}_{\text{encode}}(S_u, q_u) \tag{2}$$

where e_u represents the semantic embedding and I_u denotes the extracted preliminary user intent (e.g., "looking for action movies").

4.2.2 Knowledge Generator.

Upon encountering an impasse, Soar queries the LLM, which then generates a potential solution based on the provided context. Leveraging meticulously crafted prompts, we can guide the LLM to generate domain-specific knowledge or inferential steps (Yadav et al., 2020). For instance, during the bootstrapping phase, we inject initial domain knowledge into Soar using the following prompt.

> *Prompt: "For a movie recommender system, generate common-sense rules for the 'sci-fi' genre. The rules must be in IF-THEN format. For example: IF a user likes 'Star Wars', THEN recommend 'Blade Runner'."*

4.3 Cognitive Reasoning Engine

The cognitive cycle of Soar is precisely mapped onto the recommendation task as follows:

The agent perceives and parses the symbolic information received from the LLM module, which is then used to update its WM. The WM maintains dynamic information pertinent to the current task, including the user model, recommendation goals, and the set of candidate items. For example:

$$\begin{aligned} WM = {} & (\text{goal} < g1 >^{\text{type}} \text{recommend}), \\ & (\text{user} < u1 >^{\text{history}} < h1 >), \\ & (\text{candidates} < c1 >^{\text{items}} v1, v2, \ldots) \end{aligned} \tag{3}$$

The Soar decision cycle commences. Production rules (P_i:C_i $\rightarrow$ A_i) stored in procedural memory are matched against the current state of the WM. This involves two substages: (1)**Proposal**. All rules whose conditions C_i are satisfied by the WM state are activated. Each activated rule proposes an operator O_i to be applied. For example, a rule might propose the operator: `Filter-by-Genre(genre='sci-fi')`. (2)**Decision**. Based on preferences or learned heuristics, Soar selects the optimal operator, O^*, from the set of proposed operators.

The selected operator O^* is applied, and its resulting actions modify the state of the WM. For instance, after applying the filter operator, the candidate set `<c1>` in the WM would be updated to remove non-sci-fi movies. This cycle continues iteratively until the goal is achieved.

4.4 Neuro-Symbolic Bridge Module

This module represents the core innovation of CogRec. It bridges the chasm between the sub-symbolic representations of the LLM and the symbolic representations of Soar, enabling bidirectional information flow and synergy.

4.4.1 From LLM to Soar.

When Soar enters an impasse due to insufficient knowledge, the natural language response generated by the LLM must be transformed into symbolic knowledge comprehensible to Soar. This process is performed by the bridge module's Text-to-Chunk Converter, denoted as $\phi_L \rightarrow S$ (ϕ from Language to Symbol).

Consider a scenario where Soar enters an impasse, unable to decide among the candidate set $\{v_A, v_B, v_C\}$, and subsequently queries the LLM. The LLM returns a COT-style response R_{LLM}:

> *"Recommend 'Blade Runner 2049' (v_A). Reason (1): The user's history indicates a preference for the cyberpunk genre. Reason (2): 'Blade Runner 2049' is a quintessential cyberpunk film."*

The converter $\phi_L \rightarrow S$ parses this text into a structured causal pair: Chunkraw = Parse(RLLM) = ⟨C1, C2, A1⟩, where the components are:

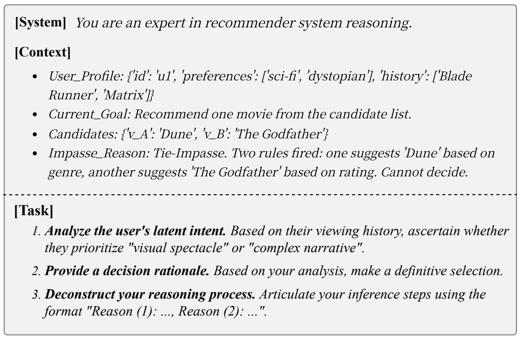
$$\text{Chunk}_{\text{raw}} = \text{Parse}(R_{\text{LLM}}) = \langle \{C_1, C_2\}, A_1 \rangle \tag{4}$$

where the components are:

$$Condition1(C_1) : (user < u1 >^{preference} \, 'cyberpunk'),$$
$$Condition2(C_2) : (item < v_A >^{genre} \, 'cyberpunk'),$$
$$Action(A_1) : (propose - operator < op1 >^{name}$$
$$select - item^{item} < v_A >)$$

Soar's chunking mechanism then ingests this raw chunk and automatically compiles a new and generalizable production rule P_{new}, which is then stored in its procedural memory. A simplified representation is:

```
P_new: sp {
   IF(< s > ^state-is-valid true)
     (< s > ^user < u >)
     (< u > ^preference ?g)
     (< s > ^candidate-item ?i)
     (?i ^genre ?g)
   -->
     (< s > ^operator < o > +)
     (< o > ^name select-item)
     (< o > ^item ?i)
}
```



[System] *You are an expert in recommender system reasoning.*

[Context]

- *User_Profile: {'id': 'u1', 'preferences': ['sci-fi', 'dystopian'], 'history': ['Blade Runner', 'Matrix']}*
- *Current_Goal: Recommend one movie from the candidate list.*
- *Candidates: {'v_A': 'Dune', 'v_B': 'The Godfather'}*
- *Impasse_Reason: Tie-Impasse. Two rules fired: one suggests 'Dune' based on genre, another suggests 'The Godfather' based on rating. Cannot decide.*

[Task]

1. ***Analyze the user's latent intent.*** *Based on their viewing history, ascertain whether they prioritize "visual spectacle" or "complex narrative".*
2. ***Provide a decision rationale.*** *Based on your analysis, make a definitive selection.*
3. ***Deconstruct your reasoning process.*** *Articulate your inference steps using the format "Reason (1): ..., Reason (2): ...".*

Figure 2: An example of a structured query automatically generated by CogRec's Symbol-to-Text Converter upon encountering a decision impasse.

This new rule enables the agent to handle similar situations in the future without re-querying the LLM, thus achieving autonomous knowledge internalization and generalization, which significantly enhances its efficiency and robustness.

4.4.2 From Soar to LLM.

To prevent the LLM from generating hallucinations due to vague queries, Soar generates a highly structured prompt when initiating a query. This is accomplished via the bridge module's Symbol-to-Text Converter, $\phi_S \rightarrow L$. When an impasse occurs, the $WM_{impasse}$ contains a wealth of contextual information. $\phi_S \rightarrow L$ transforms this context into a precise prompt Q_{Soar}, An example of such a prompt is shown in Figure 2.

$$Q_{\text{Soar}} = \Phi_{\text{S}\rightarrow\text{L}}(WM_{\text{impasse}}) \tag{5}$$

By translating the symbolic state from Soar's working memory (e.g., user profile, goals, candidate set, and the reason for the impasse) into a precise textual prompt, this mechanism effectively guides the LLM to focus on the specific cognitive bottleneck. This, in turn, minimizes knowledge hallucination and ensures the relevance and accuracy of the generated content.

5 Experiment

5.1 Experimental Setup

5.1.1 Datasets.

Our evaluation is performed on three public datasets from distinct domains. These datasets are widely used and are rich in metadata, making them suitable for our study: (1)**MovieLens-1M (ML-1M)**(Harper and Konstan, 2015): We

Dataset	#Users	#Items	#Inters	#Sparsity
ML-1M	6,040	3,416	999,611	95.16%
Movies	123,960	50,052	1,697,533	99.97%
Yelp	43,702	51,068	1,689,188	99.93%

Table 1: Statistics of the experimented datasets.

leverage its rich genre information as symbolic attributes for Soar's reasoning processes. (2)**Amazon Review (Movies)**(McAuley et al., 2015): We select the "Movies" subset from the Amazon Review dataset. (3)**Yelp** (Asghar, 2016): A dataset comprising user reviews for local businesses. This dataset is selected to evaluate CogRec's generalizability in recommendation scenarios beyond media consumption, such as local services. We preprocess all datasets by filtering out users and items with fewer than 10 interactions. Detailed statistics for the datasets are presented in Table 1.

5.1.2 Evaluation Metrics.

We employ two standard Top-K ranking metrics to assess recommendation performance: Hit Rate (HR@K) and Normalized Discounted Cumulative Gain (N@K), where K is set to 10 and 20.

5.1.3 Baselines.

To comprehensively evaluate CogRec, we compare it against a diverse set of representative baselines. First, for Collaborative and Sequential Recommendation, we include foundational and state-of-the-art(SOTA) models that rely on interaction patterns: the classic BPR-MF (Rendle et al., 2012), the efficient GNN-based LightGCN (He et al., 2020), the seminal sequential model SASRec (Kang and McAuley, 2018) and its contrastive learning successor CL4SRec (Xie et al., 2022). Second, to assess performance against models leveraging richer information, our Feature-enhanced and Causal Models category includes the classic multimodal VBPR (He and McAuley, 2016), the GNN-based MMGCN (Wei et al., 2019), the foundation model-powered FREEDOM (Li et al., 2024a), and the SOTA causal model MACR (Gao et al., 2023) for disentangled representation learning. Finally, to provide a direct comparison with our neuro-symbolic paradigm, we select several LLM-based and Symbolic Systems: P5 (Geng et al., 2022b), which fine-tunes an LLM for recommendation tasks; GPT4Rec (Li et al., 2023b), a generative agent approach; LLM-Direct (Liu et al., 2024), a zero-shot baseline using direct prompting; and a pure Soar (Laird, 2019) agent with manually-coded rules to serve as a symbolic-only benchmark.

5.1.4 Implementation Details.

The CogRec framework is implemented in Python. The Soar core is built upon the pySoar library, while the LLM module interacts with the Deepseek-V3 model via its API. During the knowledge bootstrapping phase, we design approximately 10 high-level prompt templates for each dataset, which are used to generate an initial set of around 500 production rules to populate Soar's procedural memory. For all models, the embedding dimension is set to 128, and they are trained using the Adam optimizer (Loshchilov and Hutter, 2017; Kingma and Ba, 2014) with a learning rate of 1e-4 and a weight decay of 0.01. Training proceeds for a maximum of 20 epochs. All experiments are conducted on a server equipped with an NVIDIA 4090 GPU.

5.2 Performance Comparison

We evaluate CogRec against all baseline models across three datasets, with the results presented in Table 2. Our analysis leads to the following conclusions: (1) CogRec consistently and significantly outperforms all baseline models across all datasets and metrics. This result unequivocally validates the potent synergy of integrating the rich, implicit knowledge from LLMs with the structured, explicit reasoning of the Soar cognitive architecture. (2) The performance gains over the strongest LLM-native baselines, such as P5 and FREEDOM, are particularly noteworthy. This indicates that our neuro-symbolic framework leverages knowledge more effectively than purely generative or prompt-based paradigms. By transforming transient LLM insights into permanent, reusable symbolic rules via Soar's chunking mechanism, CogRec achieves a more robust and adaptive decision-making process. (3) CogRec's significant superiority over the LLM-Direct baseline provides compelling evidence that the Soar cognitive architecture is indispensable, not redundant. It offers crucial capabilities such as goal decomposition, constraint satisfaction, and structured reasoning, which effectively mitigate the logical inconsistencies and knowledge hallucinations that can arise from direct LLM-based inference, ultimately leading to more accurate and reliable recommendations.

Models	ML-1M				Movies				Yelp			
	HR@10	HR@20	N@10	N@20	HR@10	HR@20	N@10	N@20	HR@10	HR@20	N@10	N@20
BPR-MF	0.1025	0.1833	0.0512	0.0764	0.0814	0.1521	0.0401	0.0623	0.0622	0.1219	0.0315	0.0501
LightGCN	0.2851	0.4189	0.1765	0.2243	0.3140	0.4601	0.2032	0.2566	0.2015	0.3044	0.1347	0.1789
SASRec	0.2815	0.4152	0.1743	0.2218	0.3122	0.4589	0.2015	0.2543	0.1988	0.3015	0.1321	0.1764
CL4SRec	0.2942	0.4295	0.1881	0.2384	0.3241	0.4703	0.2120	0.2675	0.2099	0.3172	0.1448	0.1885
VBPR	0.1533	0.2510	0.0802	0.1124	0.1259	0.2188	0.0677	0.0953	0.1011	0.1832	0.0543	0.0805
MMGCN	0.2879	0.4223	0.1798	0.2281	0.3185	0.4640	0.2071	0.2605	0.2046	0.3088	0.1384	0.1820
FREEDOM	0.2968	0.4325	0.1912	0.2415	<u>0.3289</u>	<u>0.4751</u>	<u>0.2163</u>	<u>0.2724</u>	0.2127	0.3214	0.1479	0.1921
MACR	0.2921	0.4265	0.1856	0.2352	0.3207	0.4673	0.2095	0.2644	0.2113	0.3195	0.1465	0.1904
P5	<u>0.2985</u>	<u>0.4344</u>	<u>0.1930</u>	<u>0.2439</u>	0.3271	0.4730	0.2148	0.2705	<u>0.2141</u>	<u>0.3235</u>	<u>0.1493</u>	<u>0.1938</u>
GPT4Rec	0.2956	0.4301	0.1899	0.2398	0.3258	0.4712	0.2134	0.2688	0.2105	0.3188	0.1456	0.1892
LLM-Direct	0.2901	0.4215	0.1832	0.2311	0.3197	0.4655	0.2088	0.2621	0.2053	0.3102	0.1402	0.1823
Soar	0.0511	0.0982	0.0234	0.0356	0.0423	0.0811	0.0198	0.0299	0.0317	0.0624	0.0145	0.0221
CogRec	**0.3124**	**0.4522**	**0.2057**	**0.2581**	**0.3401**	**0.4905**	**0.2269**	**0.2853**	**0.2247**	**0.3351**	**0.1588**	**0.2045**
Improve	4.7%	4.1%	6.6%	5.8%	3.4%	3.2%	4.9%	4.7%	5.0%	3.6%	6.4%	5.5%

Table 2: Overall performance comparison on the three datasets. Models are grouped by category. The best results are in **bold**, and the second-best are <u>underlined</u>.

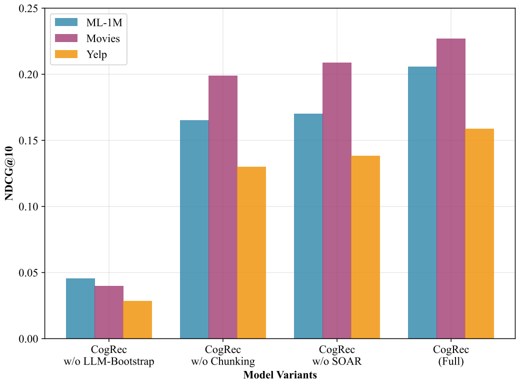


Figure 3: The performance of CogRec and its variants.

5.3 Ablation Study

To validate the necessity of each key component in CogRec, we designed and evaluated the following three ablation variants:

- **CogRec w/o LLM-Bootstrap**: This variant removes the LLM-based knowledge bootstrapping phase.
- **CogRec w/o Chunking**: This variant retains the full framework but disables Soar's chunking mechanism after a solution is obtained from the LLM.
- **CogRec w/o Soar**: This variant is equivalent to the LLM-Direct baseline, removing the entire symbolic reasoning engine.

As Figure 3 shows, removing LLM bootstrapping leads to a drastic performance drop, indicating that learning symbolic rules from scratch is exceedingly inefficient. The LLM provides the agent with a high-quality 'innate' knowledge base to start from. After disabling the chunking mechanism, performance degrades significantly, approaching that of LLM-Direct. This demonstrates that online learning is a cornerstone of CogRec's success, enabling it to transform transient solutions into permanent, reusable skills.

5.4 Further Analysis

5.4.1 Learning Curve Analysis.

As shown in Figure 4, as the number of interactions with users increases, CogRec's LLM-Call Frequency (LCF) declines significantly, eventually converging to a low, stable plateau. This indicates that the agent has learned a sufficient number of rules via chunking to handle most common situations autonomously. In contrast, the LCF of CogRec w/o Chunking remains consistently high, as it is incapable of learning from experience and must re-query the LLM for every similar problem. This provides strong evidence that CogRec's online learning capability substantially enhances its long-term operational efficiency and autonomy.

5.4.2 Long-Tail Item Recommendation Performance.

We partition the items in the test set into head (top 20%) and long-tail (bottom 80%) groups based on interaction frequency and evaluate model performance on each. As illustrated in Figure 5, all models experience a performance drop on long-tail items. However, the performance degradation for

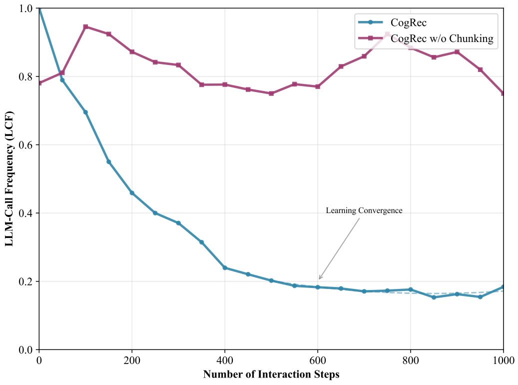


Figure 4: Variation of LCF with the number of interaction steps.

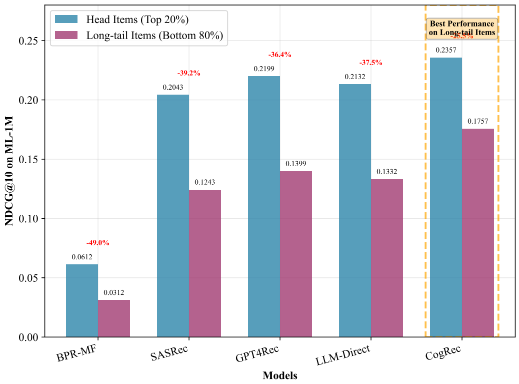
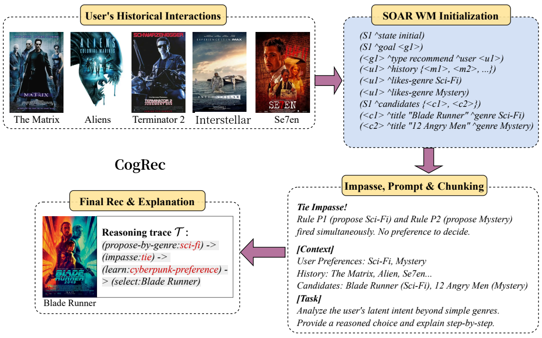


Figure 5: Performance comparison on head and long-tail items (N@10 on ML-1M)

CogRec is markedly smaller than that of SASRec and LLM-Direct. We attribute this to the fact that CogRec's reasoning does not rely solely on collaborative filtering signals (which are sparse for long-tail items), but rather leverages the common-sense knowledge about item attributes and types provided by the LLM. For instance, even if a niche film has few views, Soar can still make a high-quality recommendation if it knows the film's director or theme matches the user's preferences. This highlights CogRec's robustness in addressing data sparsity issues.

5.5 Case Study

To intuitively illustrate the explainability of CogRec, we trace the recommendation process for a user with a preference for "sci-fi" and "mystery" films, as depicted in Figure 6. The process unfolds as follows: (a) the user's partial viewing history is observed, leading to (b) the initialization of Soar's working memory. (c) A 'tie' impasse is then triggered due to multiple viable sci-fi candidates. In response, (d) Soar initiates a structured query to



```
(S1 ^state initial)
(S1 ^goal <g1>)
(<g1> ^type recommend ^user <u1>)
(<u1> ^history {<m1>, <m2>, ...})
(<u1> ^likes-genre Sci-Fi)
(<u1> ^likes-genre Mystery)
(S1 ^candidates {<c1>, <c2>})
(<c1> ^title "Blade Runner" ^genre Sci-Fi)
(<c2> ^title "12 Angry Men" ^genre Mystery)
```

Tie Impasse!
Rule P1 (propose Sci-Fi) and Rule P2 (propose Mystery) fired simultaneously. No preference to decide.

[Context]
User Preferences: Sci-Fi, Mystery
History: The Matrix, Alien, Se7en...
Candidates: Blade Runner (Sci-Fi), 12 Angry Men (Mystery)
[Task]
Analyze the user's latent intent beyond simple genres.
Provide a reasoned choice and explain step-by-step.

Reasoning trace $\mathcal{T}$:
(propose-by-genre:sci-fi) -> (impasse:tie) -> (learn:cyberpunk-preference) -> (select:Blade Runner)

Figure 6: The process of conducting an instance recommendation based on CogRec.

the LLM to probe for deeper-level preferences. (e) The LLM returns a Chain-of-Thought response, identifying a potential user preference for the 'cyberpunk' sub-genre. (f) Subsequently, Soar learns a new production rule via its chunking mechanism, which (g) culminates in the final recommendation of 'Blade Runner 2049,' accompanied by its fully traceable reasoning path.

This case study vividly demonstrates how CogRec reasons through symbolic operations and, when faced with a challenge, resolves it through deep interaction with the LLM to achieve self-evolution. The resulting reasoning trace $\mathcal{T}$, is machine-native and high-fidelity, making it far more trustworthy than any post-hoc generated natural language explanation.

6 Conclusion

In this paper, we introduced CogRec, a cognitive recommender agent designed to tackle the critical challenges of explainability, adaptability, and robustness in LLM-based recommendation. The framework employs Soar for transparent symbolic reasoning, while leveraging an LLM for knowledge bootstrapping and dynamic impasse resolution. This neuro-symbolic interaction, facilitated by a dedicated bridge module and Soar's chunking mechanism, enables continuous online learning. Extensive evaluations validate CogRec's efficacy, demonstrating its superior performance in recommendation accuracy and effectiveness in mitigating the long-tail problem. For future work, we plan to enhance the neuro-symbolic bridge with more advanced translation models and explore extending CogRec to support more complex, conversational recommendation scenarios.

Limitations

Despite the promising results demonstrated by our method, it is important to acknowledge its limitations. First, our CogRec is not immune to knowledge hallucination, and the chunking mechanism could internalize faulty rules. Additionally, the computational overhead from LLM API calls introduces latency, posing scalability challenges for real-time industrial systems. The neuro-symbolic translation bridge, while critical, represents a complex NLP task and a potential point of failure. Finally, the expressiveness of production rules may be limited in capturing abstract or subjective user preferences, a dimension not fully explored by our current evaluation datasets.

Ethics Statement

Our research utilizes public, anonymized datasets, minimizing privacy risks. The primary ethical concern is the potential for CogRec to inherit and amplify societal biases from the pre-trained LLM, which could be encoded into its symbolic production rules, leading to unfair recommendations. We acknowledge that explicit bias detection and mitigation are critical areas for future work.

Acknowledgements

We appreciate anonymous reviewers for their thorough comments and suggestions. This research was supported by the National Basic Research Program of China (Grant No. 2022-JCJQ-JJ-0683).

References

Nabiha Asghar. 2016. Yelp dataset challenge: Review rating prediction. *arXiv preprint arXiv:1605.05362*.

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

Amin Beheshti, Shahpar Yakhchi, Salman Mousaeirad, Seyed Mohssen Ghafari, Srinivasa Reddy Goluguri, and Mohammad Amin Edrisi. 2020. Towards cognitive recommender systems. *Algorithms*, 13(8):176.

Yukuo Cen, Jianwei Zhang, Xu Zou, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Controllable multi-interest framework for recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2942–2951.

Hongbo Chen, Yexin Wang, Zhen Liu, Weinan Cheng, and Jun Chen. 2023. Dppar: A dual-process persona-aware recommender system. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 343–352.

Junsu Cho, Dongmin Hyun, Dong won Lim, Hyeon jae Cheon, Hyoung-iel Park, and Hwanjo Yu. 2023. Dynamic multi-behavior sequence modeling for next item recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 4199–4207.

Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-rec: Generative pretrained language models are open-ended recommender systems. *arXiv preprint arXiv:2205.08084*.

Chongming Gao, Jiawei Chen, Shiqi Wang, Fuli Feng, Xiangnan He, and Biao Li. 2023. Multi-agent causal representation learning for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 954–964.

Artur d'Avila Garcez and Luis C Lamb. 2023. Neurosymbolic ai: The 3 rd wave. *Artificial Intelligence Review*, 56(11):12387–12406.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022a. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM conference on recommender systems*, pages 299–315.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022b. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM conference on recommender systems*, pages 299–315.

F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19.

Ruining He and Julian McAuley. 2016. Vbpr: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 639–648.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. 2022. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38.

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

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

John E Laird. 2019. *The Soar cognitive architecture*. MIT press.

Cheng Li, Yong Xu, Xin He, Yujun Zhu, Jinde Cao, and Qun Fang. 2025. Knowledge enhanced global graph contrastive denoising for recommendation. In *International Conference on Intelligent Computing*, pages 138–149. Springer.

Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023a. Text is all you need: Learning language representations for sequential recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1258–1267.

Jiatong Li, Zhengbang Yang, Zhaowei Liu, Shiyu Du, Qian-Yi, Chen Gao, and Aozora Project. 2024a. FREEDOM: A benchmark for foundation models on multimodal recommendation. *arXiv preprint arXiv:2402.13254*.

Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. 2023b. Gpt4rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879*.

Xinyi Li, Yongfeng Zhang, and Edward C Malthouse. 2023c. A preliminary study of chatgpt on news recommendation: Personalization, provider fairness, fake news. *arXiv preprint arXiv:2306.10702*.

Yangning Li, Shirong Ma, Xiaobin Wang, Shen Huang, Chengyue Jiang, Hai-Tao Zheng, Pengjun Xie, Fei Huang, and Yong Jiang. 2024b. Ecomgpt: Instruction-tuning large language models with chain-of-task tasks for e-commerce. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38(17), pages 18582–18590.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. In *The 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (IJCNLP-AACL 2023)*.

Jianxin Ma, Chang Zhou, Peng Cui, Hongxia Yang, and Wenwu Zhu. 2019. Learning disentangled representations for recommendation. *Advances in neural information processing systems*, 32.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. 2023. Efficiently scaling transformer inference. *Proceedings of machine learning and systems*, 5:606–624.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*.

Myung Eun Shin, Paula Parra-Bueno, and Ryohei Yasuda. 2025. Formation of long-term memory without short-term memory revealed by camkii inhibition. *Nature Neuroscience*, 28(1):35–39.

Laura Von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, and 1 others. 2021. Informed machine learning–a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):614–633.

Lei Wang and Ee-Peng Lim. 2023. Zero-shot next-item recommendation using large pretrained language models. *arXiv preprint arXiv:2304.03153*.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 950–958.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yanfei Wei, Jing Miao, Xiaodong Cheng, and Yanan Wang. 2023. A multi-head attention network integrating knowledge graph and collaborative filtering. In *2023 4th International Symposium on Computer Engineering and Intelligent Communications (ISCEIC)*, pages 648–654. IEEE.

Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of the 27th ACM international conference on multimedia*, pages 1437–1445.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, and 1 others. 2024. A survey on large language models for recommendation. *World Wide Web*, 27(5):60.

Yu Xia, Rui Wang, Xu Liu, Mingyan Li, Tong Yu, Xiang Chen, Julian McAuley, and Shuai Li. 2025. Beyond chain-of-thought: A survey of chain-of-x paradigms for llms. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10795–10809.

Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. 2019. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 285–294.

Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*, pages 1259–1273. IEEE.

Lanling Xu, Zhen Tian, Bingqian Li, Junjie Zhang, Daoyuan Wang, Hongyu Wang, Jinpeng Wang, Sheng Chen, and Wayne Xin Zhao. 2024. Sequence-level semantic representation fusion for recommender systems. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 5015–5022.

Satya Prakash Yadav, Dharmendra Prasad Mahato, and Nguyen Thi Dieu Linh. 2020. *Distributed artificial intelligence: A modern approach*. CRC Press.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Ekin Durmus, Maxwell Laskin, Clay Lin, Xuezhi Li, Kanishka Hausman, and Sharan Gu. 2022. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, and 1 others. 2025. Siren's song in the ai ocean: A survey on hallucination in large language models. *Computational Linguistics*, pages 1–45.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

Zihuai Zhao, Wenqi Fan, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Zhen Wen, Fei Wang, Xiangyu Zhao, Jiliang Tang, and 1 others. 2024. Recommender systems in the era of large language models (llms). *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6889–6907.

Junhao Zheng, Shengjie Qiu, Chengming Shi, and Qianli Ma. 2025. Towards lifelong learning of large language models: A survey. *ACM Computing Surveys*, 57(8):1–35.

Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1059–1068.

A Detailed Experimental Setup

A.1 Dataset Preprocessing

To ensure the fairness and validity of our experiments, we adopted a unified preprocessing pipeline for all datasets.

A.1.1 Data Filtering

We adhered to a 10-core filtering standard, meaning that users and items with fewer than 10 interactions were removed. This mitigates noise from data sparsity and ensures that each user and item has sufficient contextual information for modeling. The statistics presented in Table 1 of the main paper were compiled after this filtering step.

A.1.2 Sequence Construction and Splitting

For each user, we sorted their historical interactions chronologically by timestamp to construct an interaction sequence $S_u = [v_1, v_2, \ldots, v_{|S_u|}]$. We employed a leave-one-out evaluation strategy, implemented as follows:

- *Training Set:* For each user sequence S_u, the first $|S_u| - 2$ items, i.e., $[v_1, v_2, \ldots, v_{|S_u|-2}]$, were used to train the model.
- *Validation Set:* The second-to-last item in the sequence, $v_{|S_u|-1}$, was used for model selection and hyperparameter tuning.
- *Test Set:* The final item in the sequence $v_{|S_u|}$, was reserved for reporting the final model performance.

A.1.3 Negative Sampling

During both training and evaluation, we provided a set of negative instances for each positive instance (the item the user actually interacted with next). For each positive instance of a user, we randomly sampled 100 items from the set of all items with which the user had never interacted. During evaluation, the model is tasked with ranking the single positive instance as highly as possible among this set of 101 candidates (1 positive + 100 negative).

A.2 Mathematical Definitions of Evaluation Metrics

We used Hit Rate (HR@K) and Normalized Discounted Cumulative Gain (NDCG@K) in the main paper. Their precise definitions are as follows:

- **HR@K:** This metric measures whether the positive instance is present within the top-K recommended items. For a given user, the value is 1 if the ground-truth item is ranked within the top K, and 0 otherwise. The final HR@K is the average over all test users.

$$\text{HR@K} = \frac{1}{|U_{\text{test}}|} \sum_{u \in U_{\text{test}}} \mathbb{I}(\text{rank}_u \leq K) \quad (6)$$

where U_{test} is the set of test users, rank_u is the rank of the positive item for user u, and $\mathbb{I}(\cdot)$ is the indicator function.

- **NDCG@K:** This metric accounts for the position of the hit, assigning higher scores to hits at higher ranks.

$$\text{NDCG@K} = \frac{1}{|U_{\text{test}}|} \sum_{u \in U_{\text{test}}} \frac{\mathbb{I}(\text{rank}_u \leq K)}{\log_2(\text{rank}_u + 1)} \quad (7)$$

A.3 Implementation Details of Baselines

To ensure a fair comparison, we utilized publicly available and widely accepted implementations for the baselines wherever possible and tuned them in the same environment.

- **BPR-MF, LightGCN, SGL, SASRec & CL4SRec:** We implemented these models using the open-source recommendation library RecBole. We adopted their default hyperparameter settings and subsequently fine-tuned them on the validation set to achieve optimal performance.
- **VBPR, MMGCN & MACR:** For these feature-enhanced and causal models, we utilized the official open-source implementations provided by their respective authors. All key hyperparameters were tuned based on their reported ranges on our validation data.
- **FREEDOM:** We used the pre-trained vision and language encoders as specified by the authors to extract item features, ensuring a fair comparison of its multimodal understanding capabilities.
- **P5:** We re-implemented the P5 framework, which casts recommendation as a text-to-text language processing task. We fine-tuned the T5-base model on a task-specific mixture of our training data, converting user-item interactions into natural language prompts as described in the original paper. The best-performing model checkpoint was selected based on validation set performance.

- **GPT4Rec:** We referred to the implementation details in its original paper and used an open-source LLM (Deepseek-V3) as its core language model to maintain consistency with the LLM component of our own method. We designed multiple prompt templates for GPT4Rec and selected the best-performing one based on its validation set performance.

- **LLM-Direct:** This baseline is designed to simulate the recommendation capabilities of a pure LLM. Our prompt template included the user's complete historical interaction sequence (represented by item titles) and a list of 101 candidate items (also by title). The model was tasked with selecting the 10 most suitable items and ranking them. An example of the prompt template is as follows:

```
# Context
You are an expert movie
    ↪ recommender system. A
    ↪ user has watched the
    ↪ following movies in order
    ↪ : [Movie A, Movie B,
    ↪ Movie C, ...].
# Task
From the following list of
    ↪ candidate movies, please
    ↪ select the 10 movies that
    ↪  the user is most likely
    ↪ to watch next and rank
    ↪ them from most likely to
    ↪ least likely. Provide
    ↪ only the ranked list of
    ↪ movie titles.
# Candidate Movies
[Candidate 1, Candidate 2, ...,
    ↪ Candidate 101]
# Your Ranked Recommendation:
```

- **Soar:** This baseline involves no interaction with any LLM. For each dataset, we manually authored approximately 20 high-quality, generic, attribute-based recommendation rules. For example, on the MovieLens dataset, a rule might be: IF *a user's recently watched movies predominantly belong to the 'sci-fi' genre* AND *a candidate movie is also of the 'sci-fi' genre* THEN *propose recommending this candidate movie.*

A.4 Experimental Environment

All experiments were conducted on a server with the following specifications:

Component	Specification
CPU	Intel(R) Xeon(R) Platinum 8358 CPU @ 2.60GHz
GPU	NVIDIA RTX 4090 (24GB VRAM)
RAM	256 GB
OS	Ubuntu 20.04 LTS
Core Software	Python 3.9, PyTorch 2.0, pysoar 3.1.0, CUDA 11.8
LLM API	Deepseek-V3 (via official API)

Table 3: Hardware and Software Specifications.

B Algorithm Pseudocode

Algorithm 1 provides a detailed description of the complete process by which the CogRec agent generates a recommendation for a single user. It clearly illustrates the Perception-Cognition-Action cycle, along with the impasse-driven LLM interaction and the chunking-based learning mechanism.

Algorithm 1 The CogRec Cognitive Recommender

```
Require: User $u$, user's historical interaction sequence $S_u$,
    current query $q_u$
Ensure: Recommended item $v_{\text{rec}}$, reasoning trace $\mathcal{T}$

// Phase 1: Perception
$WM \leftarrow \text{INITIALIZESTATE}(S_u, q_u, \text{LLM}_{\text{encode}})$
$\mathcal{T} \leftarrow \emptyset$
while not $\text{ISTERMINAL}(WM)$ do
    // Phase 2: Cognition
    $\mathcal{O}_{\text{proposed}} \leftarrow \text{PROPOSEOPERATORS}(WM, \Pi_{\text{Soar}})$
    if $\text{ISIMPASSE}(\mathcal{O}_{\text{proposed}})$ then
        $Q_{\text{Soar}} \leftarrow \Phi_{S \to L}(WM)$
        $R_{\text{LLM}} \leftarrow \text{LLM}_{\text{generate}}(Q_{\text{Soar}})$
        $P_{\text{new}} \leftarrow \Phi_{L \to S}(R_{\text{LLM}})$
        $\Pi_{\text{Soar}} \leftarrow \Pi_{\text{Soar}} \cup \{P_{\text{new}}\}$
        continue
    // Phase 3: Decision-making & Action
    $O^* \leftarrow \text{SELECTOPERATOR}(\mathcal{O}_{\text{proposed}})$
    $WM \leftarrow \text{APPLYOPERATOR}(O^*, WM)$
    $\mathcal{T} \leftarrow \mathcal{T} \oplus \text{TRACE}(O^*)$
$v_{\text{rec}} \leftarrow \text{EXTRACTRECOMMENDATION}(WM)$
return $v_{\text{rec}}, \mathcal{T}$
```

C Hyperparameter Sensitivity Analysis

The performance of the CogRec framework is influenced by several key hyperparameters. We conduct a sensitivity analysis on the MovieLens-1M dataset to investigate how these parameters affect model performance and to validate our configuration choices.

C.1 Impact of the Number of Initial Rules

During the knowledge bootstrapping phase, the number of initial production rules generated by the LLM is crucial for CogRec's "cold-start" performance. Too few rules lead to frequent impasses and inefficiency, whereas too many may introduce noise and increase initialization overhead. We tested varying numbers of initial rules, ranging from 50 to 1,000.

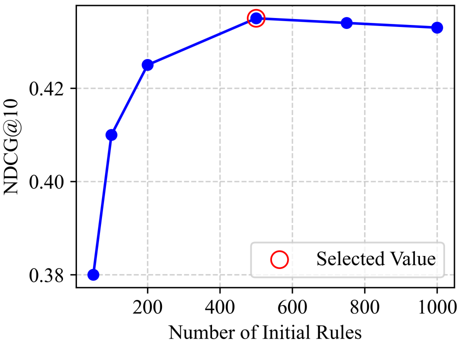


Figure 7: Impact of the number of initial rules on performance (NDCG@10 on ML-1M).

As depicted in Figure 7, performance steadily improves as the number of initial rules increases, since a richer initial knowledge base reduces the need for costly LLM calls during early interactions. Performance tends to plateau around 500 rules, after which the marginal benefits diminish, and may even slightly decrease due to potential rule conflicts. Therefore, we chose to generate approximately 500 initial rules in our experiments to strike a balance between performance and efficiency.

C.2 Impact of Embedding Dimension

Although CogRec's core is symbolic reasoning, its initial semantic encoding phase still relies on embedding representations. We analyze the impact of the embedding dimension from the LLM encoder on the final recommendation performance.

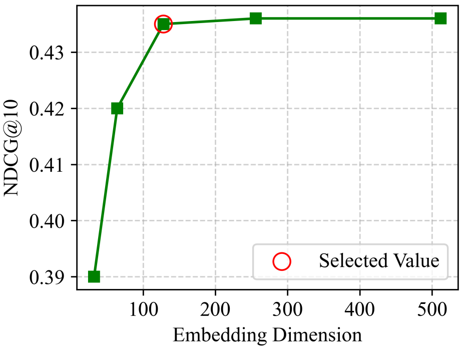


Figure 8: Impact of Embedding dimension on performance (NDCG@10 on ML-1M).

As shown in Figure 8, there is a significant performance gain as the embedding dimension increases from 32 to 128, indicating that a richer representational space can capture more nuanced semantic information. Beyond 128, performance growth slows and stabilizes. Considering the computational cost, we ultimately selected 128 as the embedding dimension for all experiments.

D Detailed Case Study

To more concretely illustrate the workflow of CogRec, we provide here the complete details of the case study presented in the main paper.

Scenario: Recommending a movie for User_123.

D.1 User Profile and Candidates

D.1.1 User History (Partial)

Movie Title	**Genre**
The Matrix (1999)	Action, Sci-Fi
Blade Runner (1982)	Sci-Fi, Thriller
Ghost in the Shell (1995)	Animation, Action, Sci-Fi

Table 4: User's Historical Viewing Information.

D.1.2 Candidate Set (Partial)

- v1: Blade Runner 2049 (Sci-Fi, Thriller)
- v2: Star Wars: Episode IV (Action, Adventure, Sci-Fi)
- v3: The Terminator (Action, Sci-Fi)
- v4: The Shawshank Redemption (Drama)

D.2 Soar State Before Impasse

Soar's initial rules (from LLM bootstrapping) identify the user's strong preference for the "Sci-Fi" genre. After filtering out non-sci-fi movies (e.g., The Shawshank Redemption), Soar's Working Memory (WM) appears as follows (simplified representation):

```
(goal <g1> ^type recommend ^user User_123)
(<g1> ^state <s>)
(<s> ^candidates {v1, v2, v3})
(<s> ^user_preference ^genre Sci-Fi)

# Existing rules propose multiple operators
(operator <op1> ^name recommend ^item v1)
(operator <op2> ^name recommend ^item v2)
(operator <op3> ^name recommend ^item v3)
```

At this point, since v1, v2, and v3 are all sci-fi films, the existing generic rules cannot make a unique choice, leading to a tie impasse.

D.3 Structured Prompt Generation

CogRec's bridge module $\Phi_{S \to L}$ converts the current WM state and impasse information into a precise LLM prompt. A JSON representation of this prompt is shown as follows:

```
{
  "task": "Resolve Recommendation Tie",
  "user_profile": {
    "user_id": "User_123",
    "known_preferences": ["Sci-Fi"],
    "interaction_history_titles": [
      "The Matrix",
      "Blade Runner",
      "Ghost in the Shell"
    ]
  },
  "decision_context": {
    "impasse_type": "tie",
    "reason": "Multiple candidates match
       ↪  the 'Sci-Fi' preference.",
    "tied_candidates": {
      "v1": {"title": "Blade Runner
         ↪ 2049", "genre": "Sci-Fi,
         ↪ Thriller"},
      "v2": {"title": "Star Wars:
         ↪ Episode IV", "genre": "
         ↪ Action, ..."},
      "v3": {"title": "The Terminator",
         ↪ "genre": "Action, Sci-Fi"}
    }
  },
  "request": "Analyze the user's history
     ↪  to find a more specific
  sub-preference or theme. Provide a
     ↪ step-by-step reasoning
  (Chain of Thought) to justify which of
     ↪  the tied candidates is the
  best fit, and formulate a new IF-THEN
     ↪ rule based on your reasoning."
}
```

D.4 LLM's Chain-of-Thought Response

Upon receiving the structured prompt, the LLM generates the following response:
(Thought): The user's history includes 'The Matrix', 'Blade Runner', and 'Ghost in the Shell'. These films are all not just 'Sci-Fi', but specifically belong to the 'Cyberpunk' subgenre. They explore themes of artificial intelligence, dystopian futures, and the nature of humanity.
(Analysis of Candidates):

- 'Blade Runner 2049' is a direct sequel to 'Blade Runner' and a quintessential Cyberpunk film.
- 'Star Wars' is Space Opera, a different Sci-Fi subgenre.
- 'The Terminator' has Cyberpunk elements but is more focused on action and time travel.

(Conclusion): The user shows a strong affinity for the Cyberpunk subgenre. 'Blade Runner 2049' is the most direct match for this nuanced preference.
(Generated Rule):

> `IF` a user has watched multiple 'Cyberpunk' movies `AND` a candidate movie's subgenre is 'Cyberpunk', `THEN` propose to recommend that candidate movie.

D.5 Chunking and New Rule Generation

The bridge module $\Phi_{L \to S}$ parses the LLM's response to extract the causal relationship. Soar's chunking mechanism then compiles it into a new, permanent production rule and adds it to the procedural memory. The Soar syntax form of this rule is as follows:

```
sp {propose*recommend*if-user-likes-
    ↪ cyberpunk-and-item-is-cyberpunk
   (state <s> ^operator <o> +)
   (<s> ^user <user>)
   (<user> ^preference <p>)
   (<p> ^subgenre cyberpunk)
   (<s> ^candidate <item>)
   (<item> ^subgenre cyberpunk)
-->
   (<s> ^operator <o> ^name recommend-
       ↪ item ^item <item>)
}
```

D.6 Impasse Resolution and Final Recommendation

With this new rule added, the Soar cognitive cycle restarts. Now, it possesses more refined knowledge to differentiate among the candidates.

1. The new rule is triggered because it matches the user's inferred preference for "Cyberpunk" and candidate v1's "Cyberpunk" attribute.
2. This new rule provides stronger support for recommending v1 than the other generic rules, thus breaking the tie.
3. Soar selects and applies the operator to recommend v1.

The final output result is as follows:

- **Recommendation:** Blade Runner 2049
- **Reasoning Trace** ($\mathcal{T}$): *rule:filter-by-genre-sci-fi* → *rule:identify-user-subgenre-cyberpunk* → *rule:propose*recommend*...*cyberpunk* → *rule:select-best-supported-recommendation.*

This detailed case study clearly demonstrates how CogRec seamlessly integrates the deep semantic understanding of an LLM (identifying "Cyberpunk") with the symbolic reasoning and learning capabilities of Soar to achieve a recommendation process that is both accurate and highly interpretable.