LLMEvalRec: An Agentic Framework for Simulating Users to Evaluate News Recommendation Systems

Yao Ma
Amazon AGI
Boston, United States
yaoom@amazon.com

Abhishek Tripathi
Amazon AGI
Cambridge, United Kingdom
dtriabhi@amazon.co.uk

Samuel Louvan
Amazon AGI
Cambridge, United Kingdom
slouvan@amazon.com

Wei Liu
Amazon AGI
London, United Kingdom
weliuz@amazon.co.uk

Murat Sensoy
Amazon AGI
London, United Kingdom
msensoy@amazon.co.uk

ABSTRACT

Evaluating news recommendation systems (NRS) presents unique challenges due to their dynamic and interactive nature coupled with evolving user interests. In the early stages of development, when user bases and historical data are scarce, it is difficult to conduct meaningful offline and online evaluations. This cold-start evaluation challenge hinders data-driven decision-making for product development and deployment. To address this, we propose LLMEvalRec, a framework that leverages Large Language Model (LLM) agents to simulate user behavior for NRS evaluation. Our approach features generative agents that automatically generate user profiles from a small number of user reading histories and perform realistic actions, while introducing the Guided Episodic Search (GUES) algorithm, which guides the automated prompt optimization process by exploring human prompt engineering practices. Experiments demonstrate that LLMEvalRec-generated data achieves 0.97 Spearman correlation with real evaluation rankings, significantly outperforming baseline simulators (0.4 and -0.05), and successfully predicts relative performance trends across both MIND benchmark and real customer datasets. Production environment validation shows consistent alignment between simulated metrics and real click-through rate (CTR) improvements.

KEYWORDS

Multi-Agent Systems; User Simulation; News Recommendation

ACM Reference Format:
Yao Ma, Abhishek Tripathi, Samuel Louvan, Wei Liu, and Murat Sensoy. 2026. LLMEvalRec: An Agentic Framework for Simulating Users to Evaluate News Recommendation Systems. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 8 pages. https://doi.org/10.65109/EFZY6916

1 INTRODUCTION

News Recommendation Systems (NRS) play a crucial role in alleviating information overload for users by providing relevant content [16]. These systems face unique challenges due to the dynamic nature of news content, evolving customer interests and varying engagement behaviour. A particularly critical challenge arises during early production stages, when evaluation options are limited. Insufficient user traffic often renders online A/B testing impractical, despite being the ideal evaluation method. Meanwhile, traditional offline evaluation using historical data may not be available or representative due to the limited customer base. This creates a fundamental "cold-start" evaluation problem that can hinder product development and deployment decisions.

Simulation-based evaluation offers a promising solution, combining benefits of both online and offline approaches [1, 4, 12]. It provides reproducibility and control while enabling interactive feedback through user simulators, thus capturing the dynamic nature of NRS more accurately. However, training realistic human behavior models for news recommendation remains challenging.

Prior work in recommender system simulation has explored various approaches. RecoGym [12] uses fixed user preferences with static behavior assumptions, while Sim2Rec [3] introduces dynamic modeling through historical interaction distributions. These methods, however, often require complex implementation, substantial expert knowledge, and extensive manual tuning, limiting their practicality in industrial settings.

Recent approaches have begun leveraging Large Language Models (LLMs). SUBER [4] generates synthetic user interactions for reinforcement learning, while Wang et al. [14] explores ChatGPT for conversational recommendation evaluation. Agent4Rec [18] pioneered LLM agents with predefined user profiles on the MovieLens dataset, and SimUSER [2] introduced persona matching for improved persona creation. While these LLM-based works show promise in offline datasets for domains like movies and books, they face two critical limitations: lack of comparison with online evaluation, and unclear effects and efforts required for prompt engineering, particularly in dynamic domains like news recommendation.

We introduce LLMEvalRec, a novel agent-based framework designed to address these limitations for practical implementation in industry settings. LLMEvalRec adapts to changing environments by using a small dataset to continuously evolve synthetic users, combining production-friendly implementation with robust evaluation capabilities. It provides an end-to-end solution from user profile construction to evaluation metric calculation, requiring minimal real user data and engineering effort. Our main contributions include:

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/EFZY6916

- LLMEvalRec: A production-ready framework that simplifies NRS evaluation through LLM agents, significantly reducing implementation complexity
- Guided Episodic Search (GUES): An algorithm for automated prompt optimization that improves upon sequential prompt optimization [9] through parallel exploration
- Experimental validation on SOTA recommender systems, demonstrating effective relative performance assessment in industrial settings, including real-world production environments

By leveraging LLM prompt optimization as its core mechanism, LLMEvalRec significantly reduces implementation complexity and effort for prompt engineering. To the best of our knowledge, this work is the first to conduct real-world experiments in production environments using LLM-based NRS evaluations.

2 LLMEVALREC FRAMEWORK

This section presents the complete LLMEvalRec framework architecture, detailing its key components and operational workflow. We begin with an overview of the system design, followed by detailed descriptions of user profile generation, LLM agent implementation, and the evaluation methodology that enables comprehensive recommender system assessment.

2.1 Overview

LLMEvalRec is a framework for evaluating news recommendation systems through LLM-based user simulation (Figure 1). It employs multiple generative agents, each initialized with a profile capturing user interests, reading history, and habits. These profiles can be derived from real user data or synthetically generated.

To ensure realistic behavior, LLMEvalRec incorporates a prompt optimization pipeline leveraging a small dataset

$$D = \{(u, h, c)\},$$

where u represents the user identifier, h denotes historical interactions, and c indicates observed clicks and non-clicks. This dataset drives the user profiling stage where realistic user profiles are extracted from D to ensure accurate simulation. This data also drives the optimization process through an observer module that tracks agent behavior and a prompt optimizer that continuously refines prompts based on observed patterns. The optimization process is guided by our GUES algorithm, which adapts successful human prompt engineering practices into an automated framework.

The framework operates sequentially: generating user profiles from D, initializing agents with optimized prompts, simulating system interactions, and computing evaluation metrics. This pipeline maintains minimal data requirements, making it a valuable evaluation framework for early-stage development when real user data is scarce. The following subsections detail each framework component, from user profile generation through LLM agent architecture to evaluation methodology.

2.2 User Profiles

We employed an LLM-based profile generation agent (profiler) to create diverse and realistic user profiles from reading histories. The profiler utilize the dataset D and generate a set of user profiles U. Each user $u \in U$ is represented as a 3-tuple (I_u, H_u, R_u), where $I_u = \{E_1, \ldots, E_{n_u}\}$ is the set of interest entities, representing subjects engaging the user; $H_u = \{N_1, \ldots, N_{m_u}\}$ is the reading history, a collection of news articles the user has engaged with; and $R_u \in \{Active, Inactive, Neutral\}$ indicates the user's engagement level. The profiler operates in steps: *analyzing* the reading history H_u of the user u to identify news categories following the IPTC[1] hierarchy; *Extracting* interest entities I_u that align with observed reading patterns; *Determining* the reading habit category R_u based on engagement frequency. The profiling process is done through an LLM agent that is prompted to act as a user profile analyzer. A system prompt example is presented in Example 2.1. Together with the query and user information, the user profier generates I_u and R_u to be consumed later in the simulation. A key advantage of our profiler is its ability to extrapolate dynamic behavior patterns from static histories, generating profiles that evolve with changing news content and user interests—crucial for the dynamic nature of news recommendation.

Example 2.1 (User Profier System Prompt Example). *You are a user profile analyzer for a new recommendation system. Your task is to analyze the user interests on news stories. Based on user's click history, understand what topics the user is interested in. A topic could be either in the for of IPTC category or an entitle name.*

Profile Evolution: Unlike static user models, our agent profiles adapt over time based on simulated interactions. As agents engage with recommendations during the simulation, their reading histories H_u are updated with clicked articles, and their interest entities I_u shift to reflect evolving preferences. This dynamic adaptation is essential for capturing the temporal aspects of user behavior in news consumption, where interests can change based on current events, personal circumstances, or content exposure.

2.3 LLM Agents for Behavior Simulation

LLMEvalRec employs LLM agents to simulate user interactions with news recommendation systems through a two-phase process: aligning and simulation. The framework comprises four key components: memory, actor, observer, and prompt optimizer.

Aligning Phase Components: An Observer and a prompt optimizer are vital components that enable the agents i.e., the LLM that represents a user, to learn realistic behavior patterns from the small dataset D of real user interactions. This ensures that the simulator is grounded to the real use case.

- **Observer**: Tracks agent actions, compares them with ground truth from D, and provides feedback for refining the agent. It generates insights on discrepancies in click behaviors, helping align agent behavior more closely with real user tendencies. An example prompt used in observer is in Example 2.2
- **Prompt optimizer**: Adjusts prompts based on the Observer's insight, ensuring agents remain aligned with real user behavior without significant divergence. The prompt optimizer will be discussed in detail in Section 3.

[1] IPTC categories are standardized metadata tags used in news and media to classify and organize content

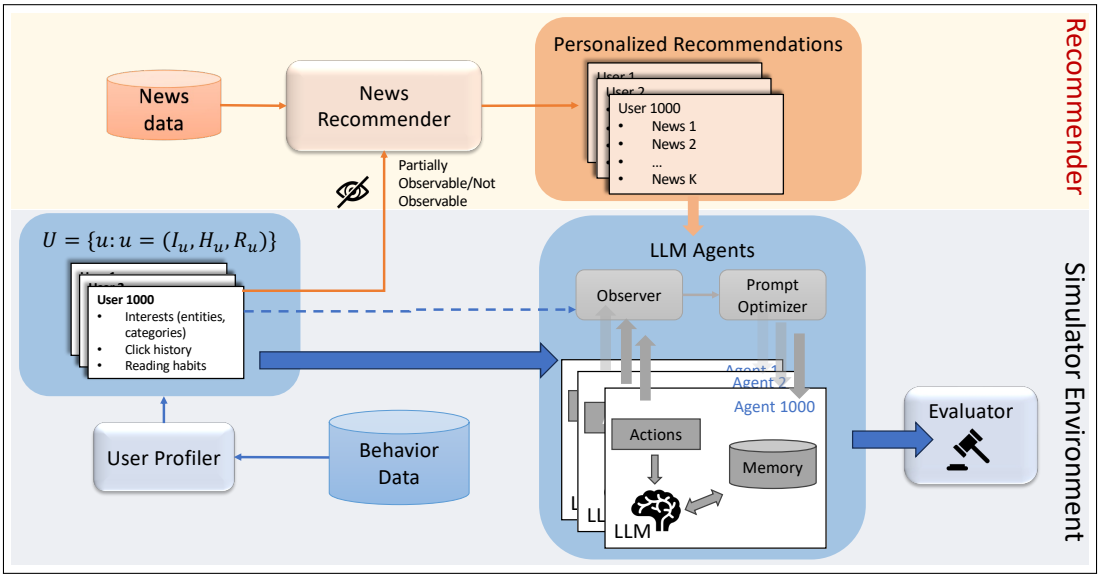


Figure 1: LLMEvalRec Framework

Example 2.2 (Observer Prompt Example). *There are some memories: {PLACEHOLDER}.Please infer the high-level insights for this person's character from these memories. The insight needs to be significantly different from the content and structure of the original memories. Response in one sentence.*

Core Simulation Components: Once the aligning phase is complete and the optimal prompt is learned, we deploy the agents for evaluation. During this phase, each agent operates using its memory, action modules and the optimized prompt. When presented with recommendations, the agent reads its memory to understand its current interests and recent behaviors, then uses the optimized prompt to make click decisions that reliably mimic real user behavior patterns.

Memory: Stores the agent's current user profile (I_u, H_u, R_u), tracks recent interactions, and updates history H_u dynamically as the agent engages with recommendations.

Actor: Defines the agent's possible actions (e.g., *click*, *no-click*) while interacting with the recommender systems and can be customized to capture diverse human behaviors. Constraining actions prevents hallucinations and ensures realistic behavior.

Agent Decision Process: Each agent follows a structured decision-making process when presented with news recommendations. The optimized prompt guides the agent through five key steps: (1) *Analyze click history*: identify main topics, themes, and patterns from previously clicked news titles; (2) *Evaluate recommendations*: extract keywords and themes from each recommendation and compare with identified interests; (3) *Rank news items*: order all recommendations from most to least relevant based on topic similarity and user preferences; (4) *Select articles to click*: choose 1-2 most relevant items that strongly align with user interests; (5) *Output decisions*: return a dictionary with ranked news list and clicked stories. This structured approach ensures consistent decision-making while allowing for natural behavioral variation across agents.

This two-phase approach, emphasizing thorough aligning followed by streamlined simulation, enables LLMEvalRec to generate highly realistic user interactions even with limited initial data.

2.4 Evaluation Module

The evaluation module utilizes the generated actions from the LLM agents as labels and calculates the evaluation metrics for assessing the performance of the recommender system. We employed multiple standard evaluation metrics for NRS: Area Under the ROC Curve (AUC), Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain (NDCG). These metrics were selected to comprehensively assess both the binary classification performance and the ranking quality of the recommendations at various positions.

3 PROMPT OPTIMIZATION

The effectiveness of LLMEvalRec heavily depends on the quality of the agents' input prompts. To optimize these prompts, we introduce an improved version of existing prompt optimization techniques, which we call *guided episodic search* (GUES). This approach builds upon methods like RecPrompt [9], incorporating successful human prompt engineering practices into an automated optimization process. The algorithm operates in two main steps:

Instruction Sampling: We maintain a set of instruction templates based on effective human prompt engineering practices. These include guiding the agent to think step by step, present examples, break tasks into subtasks, analyze before deciding, and consider alternatives. In each iteration, we sample an instruction from this set to guide the prompt modification. This approach helps introduce diversity and human-like reasoning into the prompts.

Episodic Optimization: Unlike sequential optimization as used in [9], GUES explores multiple optimization paths simultaneously. For each sampled instruction, we initiate multiple optimization

episodes, where each episode represents a different trajectory of prompt refinement. Within each episode, we generate simulated behaviors using the current prompt, evaluate their alignment with real user behaviors sampled from the small dataset D used in *Observer*. The LLM prompt optimizer composes the meta prompt with *previous prompt*, *sampled instruction*, and *feedback* from the observer (similar to the meta prompt used in RecPrompt[9]), and then propose the prompt modification. Some example meta-prompts with different instructions are presented in Figure 2. These parallel episodes allow diverse exploration of the prompt space, reducing the risk of converging to suboptimal solutions. At the end of each iteration, we select the best-performing prompt across all episodes to serve as the starting point for the next iteration.

You are a prompt optimizer for a human click simulator which simulates users' clicks on news recommendations. The simulator's goal is to generate clicks as close to the users' as possible. Your job is to optimize its prompt to reach its goal. Do not add your explanation and analysis in the output. Just output the improved prompt message.
You should generate an improved human message prompt template message based on the provided information, which consists of two parts:

- The current prompt template between "# Current Prompt Template Begin" and "# Current Prompt Template End"
- Some samples for observations including the generated clicks using the current prompt and the real human clicks.

Compare and analyze why the generated clicks are different from the real human clicks. And you should generate the improved prompt template message after learning from the difference. Do not include input variables. Focus on the text message. You should provide an enhanced prompt template After "# New Prompt Template". In the template, you must include placeholders for the users's history clicks and the recommended news only with history and recommendations within bracket. The input variables are only 'history'and 'recommendations'. Do not add any additional variables to the prompt template message.

#Current Prompt Template Begin
{current prompt}
#Current Prompt Template End
One example of the generated clicks with the current prompt is as follows. The simulator generated clicks for a user who has clicked on the following news in history: {history}. The simulator was shown {recommendation}. Among these, the simulator rank the news according to their relevancy as: {ranked_news}. The real user click on: {ground_truth}. Focus on the given example and analyze the content of the prompt based on real human's clicks. You can break the task step by step.

Define the output format, you need to output a ranked list of news and generate clicks. The response from both Sub-Tasks in a dictionary object. The first key of the dictionary must be 'ranked_news' whose value is a list of string of news id from Sub-Task 1. Make sure that news id in the list covers all ids the Recommendation List. The length of the list must be the same with the same number if the number of news in the Recommendation List. The second key of the dictionary must be 'clicked_stories' whose value is a list of string corresponds to news id that you decide to click from Sub-Task 2.

Generate an improved prompt template message after #New Prompt Template

Figure 2: Meta-prompt template used by GUES optimizer to iteratively improve agent prompts.

The complete GUES algorithm is presented in Algorithm 1, with the episodic search procedure detailed in Algorithm 2. In our implementation, we use a maximum of 2 iterations and 2 episodes per iteration. The instruction sampling draws from 3 instruction templates based on established prompt engineering practices, and each episodic search performs a maximum of 5 optimization steps.

To illustrate the effectiveness of GUES, we show examples of the initial handcrafted prompt and the optimized version. Our initial prompt follows a structured approach as shown in Example 3.1.

Example 3.1 (Initial Prompt). *You are a user of a news recommendation system with your own profile and interests. You have clicked on the following news titles: {history} ####Recommendation List#### {recommendations} Based on your click history, understand what topic you are interested in and perform the following sub-tasks: 1) Rank ALL the news from the recommendation list according to the relevance to your interests. 2) Select the news you want to click on from Sub-Task 1. Return your response in a dictionary object with keys 'ranked_news' (list of all news IDs ordered by relevance) and 'clicked_stories' (list of news IDs you decide to click).*

Algorithm 1 Guided Episodic Search

Require: Instruction Set P_I, training dataset D_{tr}, initial LLM prompt p^0
  for $i = 0, 1, ...,$ *iteration* **do**
    for $e = 0, 1, ..., E$ **do**
      Sample instruction $i_t \sim P_I$
      $p_i^e \leftarrow$ *Episodic prompt search*(i_t, p_i^0)
      Calculate the loss $L(y_u(p_i^e), \hat{y}_u(p_i^e))$
    end for
    $p_i^0 \leftarrow \arg\min_p L(y_u(p_i^e), \hat{y}_u(p_i^e))$
  end for

Algorithm 2 Episodic prompt search

Require: Instruction i, initial LLM agent prompt p_0
  for $t = 0, 1, ..., T-1$ **do**
    Generate actions $\hat{y}_u(p_t), \forall u$
    Calculate $L_t = \sum_u L(y_u(p_t), \hat{y}_u(p_t))$
    Sample m users U_t from the training dataset
    Construct meta-prompt for prompt optimizer with p, i, $\hat{y}_u(p_t)$, $y_u(p_t)$, for all $u \in U_t$
    Call the prompt optimizer and generate the new prompt p_{t+1}
  end for
  return Best prompt $p^* = \arg\min_{p_t} L_t, \forall t = 0, 1, ..., T$.

After GUES optimization, the prompt evolved into a more sophisticated chain-of-thought approach as presented in Example 3.2.

Example 3.2 (Optimized Prompt). *You are simulating a user with specific news interests. Your task is to analyze the given click history and recommended news, then make decisions on what to click based on the user's preferences. 1) Analyze the user's click history: Identify main topics, themes, and key interests from the previously clicked news titles. 2) Evaluate the recommended news: Carefully read each recommended news title, extract keywords and themes, compare with the user's identified interests. 3) Rank the recommended news: Order all recommended news from most to least relevant based on the user's interests, considering topic similarity and current trends. 4) Select news to click: Choose the most relevant items from your ranked list. Limit selection to 1-2 items that strongly align with the user's interests.*

Remember, simulate realistic user behavior that reflects demonstrated interests while allowing for some variation. Return response in dictionary format with 'ranked_news' and 'clicked_stories' keys.

The optimization process transforms the simple instruction-based prompt into a structured reasoning framework that better captures human decision-making patterns.

4 EXPERIMENTS

The purpose of our experiments is to answer the question: *Is LLMEvalRec's simulated evaluation data predictive of NRS performance?* Specifically, if NRS A outperforms NRS B in real-world evaluation, LLMEvalRec should predict the same relative performance.

Model	MIND-SIME2E				REAL MIND			
	AUC	MRR	NDCG@5	NDCG@10	AUC	MRR	NDCG@5	NDCG@10
DKN	0.500	0.256	0.199	0.288	0.500	0.263	0.246	0.315
NRMS	0.499	0.266	0.204	0.293	0.541	0.272	0.253	0.319
CAUM	0.509	0.275	0.219	0.305	0.595	0.331	0.312	0.377
NAML	0.501	0.232	0.186	0.275	0.502	0.334	0.318	0.381
NRMS-PLM	0.500	0.253	0.208	0.294	0.500	0.219	0.195	0.260
CAUM-PLM	0.510	0.252	0.198	0.282	0.597	0.328	0.310	0.372
NAML-PLM	0.500	0.282	0.225	0.312	0.528	0.300	0.282	0.347
GLIMPSE	0.589	**0.422**	**0.366**	**0.440**	**0.691**	0.339	**0.376**	**0.439**
MANNER	**0.609**	0.365	0.315	0.396	0.662	**0.367**	0.351	0.411

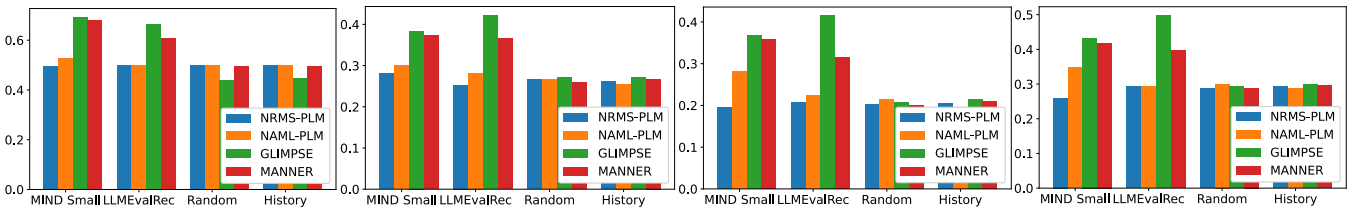
Table 1: Comparison of model performance using evaluation data generated by LLMEvalRec. Bold indicates the best performing model and underlined indicates the second best model.

Model	REAL-SIME2E			REAL DATASET		
	MRR	NDCG@5	NDCG@10	MRR	NDCG@5	NDCG@10
Bandit	0.230	0.198	0.400	0.218	0.176	0.255
Seq2Seq	0.459	0.473	0.584	0.228	0.193	0.276
MANNER	**0.507**	**0.555**	**0.620**	**0.256**	**0.217**	**0.298**

Table 2: Comparison of model performance on real world dataset

Agent Model	GUES Model	Acc	Prec.	Recall	F1
Mistral large	-	0.704	0.136	0.419	0.205
Sonnet 3	-	0.716	0.146	0.439	0.220
Sonnet 3	Sonnet 3.5	0.698	0.144	0.469	0.221
Sonnet 3	Sonnet 3	0.827	0.202	0.303	0.242

Table 3: Click behavior alignment comparison with different Agent model and GUES algorithm.



(a) AUC **(b) MRR** **(c) NDCG@5** **(d) NDCG@10**

Figure 3: The comparison of the recommenders' performance on different simulators

We validate this through experiments with multiple recommender models using both public benchmarks and online evaluation.

4.1 Experimental Settings

Datasets. To evaluate LLMEvalRec we generate two evaluation datasets using different sources: the public MIND benchmark[17] and a real-world NRS. Both datasets are created through end-to-end simulation using LLMEvalRec but with different news sources and impression generation strategies.

- **MIND-SIME2E**: We initialize synthetic profiles by sampling 1000 users from the MIND dataset, using their reading histories to ensure alignment with real user behaviors. To simulate temporal dynamics, we split MIND news articles into seven daily sets. We ran LLMEvalRecfor 7 days continuously and generated 7000 impressions for evaluating the NRSs. For the first six days, we generate random impressions for each user, simulate clicks using LLMEvalRec, and update user profiles with new interactions to capture evolving interests. On the final day, we create the test set by generating random impressions and simulating ground truth clicks using the accumulated six-day interaction history.
- **REAL-SIME2E**: For the real world evaluation dataset, we follow a similar process but use actual news articles and production impressions. We initialize user profiles using the

same approach with 1K sampled users from MIND. To construct news articles set, we use real news articles collected from 2024-09-01 to 2024-09-05. The impressions are generated by the production ranker used in the commercial NR system. LLMEvalRec is used to simulate user interactions over 5 days. Same as MIND-SIME2E, the final test set is constructed using the last day's data.

In both datasets, the simulation maintains user profiles that evolve based on interaction history, allowing us to evaluate recommender under realistic usage pattern. The key difference lies in the source of news content and impression generation methods, enabling us to validate our proposed framework under both controlled and real-world conditions.

Recommender Models. We evaluate six established recommender models[2]: GLIMPSE[7], NRMS[15], CAUM[11], NAML[15], DKN[13], and MANNER[6]. For each neural model (NAML, CAUM, and NRMS), we test versions using both static Glove[10] embeddings and Roberta pre-trained language model (PLM) embeddings. For real-world evaluation, we include three production rankers from a commercial NR system: a Rule-based ranker using expert-defined ranking criteria, a Seq2Seq model using attention-enhanced transformer for sequential prediction, and a Bandit model implementing generalized linear bandit with state-of-the-art sentence embeddings. To compare the performance of the models, we use standard ranking evaluation metrics: *AUC*, *MRR*, *NDCG@5*, and *NDCG@10*.

Baseline Behavior Models. We compare different behavior simulators using the mentioned recommender models in this experiment. We compare LLMEvalRec to: 1) Random click model, which generates clicks by randomly choosing news articles from the candidates; 2) History similarity-based click model, which generates clicks by measuring the average similarity between the candidate news and the ones in click history, using a state-of-the-art open-source text embedding model [8].

4.2 Results

Table 1 and Table 2 show the performance evaluations of recommender models using both MIND-simE2E and REAL-SIME2E as evaluation datasets. We assess the effectiveness of LLMEvalRec for evaluation by inspecting the *relative comparison* of different models and see whether the results are aligned with the ones with real data (original MIND and customer data).

Models Comparison. We observe that both MIND-SIME2E and REAL-SIME2E exhibit similar trends as in the original datasets. For MIND-simE2E, GLIMPSE and MANNER are consistently the two best performing models on all evaluation metrics, achieving AUC scores of 0.589 and 0.609 respectively, significantly outperforming traditional models like DKN (0.500) and NRMS (0.499). This performance gap demonstrates our framework's ability to distinguish between model capabilities effectively.

Neural-based models generally perform better with pre-trained language model embeddings, showing consistent improvements across CAUM-PLM and NAML-PLM variants. However, NRMS-PLM exhibits degraded performance (MRR: 0.253 vs 0.266, NDCG@10: 0.294 vs 0.293), consistent with known issues documented in [5] where NRMS suffers performance drops with PLM-based embeddings due to architectural incompatibilities.

For REAL-SIME2E evaluation, the ranking metrics obtained from simulated data strongly align with results from the real customer dataset. MANNER achieves the highest performance (MRR: 0.507, NDCG@10: 0.620) on simulated data, matching its superior performance on real data (MRR: 0.256, NDCG@10: 0.298). This consistency validates our framework's predictive capability in real-world scenarios [3].

LLMEvalRec vs other behavior models. We also compare the performance of LLMEvalRec with basic behavior simulators using evaluation results for NRMS-PLM, NAML-PLM, MANNER, and GLIMPSE, as shown in Figure 3. These models represent a spectrum from worst-performing (NRMS-PLM, NAML-PLM) to best-performing (MANNER, GLIMPSE) on the real MIND dataset, making them ideal for testing discriminative capability.

The results reveal critical differences in simulator effectiveness. LLMEvalRec successfully preserves the performance hierarchy observed in real data: GLIMPSE and MANNER consistently outperform NRMS-PLM and NAML-PLM across all metrics. In contrast, the random simulator fails to capture any meaningful performance differences, producing nearly identical scores across all models (AUC variance < 0.02). The history similarity-based simulator shows marginal improvement but still fails to distinguish between model capabilities effectively, demonstrating only weak correlation with true performance rankings.

The average Spearman correlation between real and simulated evaluation results provides quantitative validation: LLMEvalRec achieves 0.97, indicating near-perfect rank preservation, while *random* and *history similarity-based* simulators only reach -0.05 and 0.4, respectively (computed across AUC, MRR, NDCG@5, and NDCG@10 metrics for the four models). This substantial correlation difference demonstrates that LLMEvalRec captures nuanced user behavior patterns that are essential for accurate model evaluation, while simpler baselines fail to provide meaningful discriminative power.

Ablation Study: GUES Impact. To assess the contribution of prompt optimization and the observer component in LLMEvalRec we conducted a comprehensive ablation study using 1,000 datapoints from the MIND dataset. Our experimental design compared click prediction alignment across different LLM configurations with and without the GUES algorithm.

The study employed 5 distinct instruction templates based on established prompt engineering practices (step-by-step reasoning, example provision, task decomposition, alternative consideration, and analytical thinking). For each instruction, we initialized 3 independent optimization episodes with 5 refinement steps each, totaling 75 optimization trajectories per model configuration.

Results in Table 3 demonstrate substantial improvements from GUES optimization. The most significant gains appear in the Sonnet 3 configuration, where GUES optimization improves F1 score from 0.220 to 0.242 (10% relative improvement) and accuracy from 0.716

[2]We use the implementation of NRMS, CAUM, and MANNER from the Newsreclib library

[3]We exclude AUC from this analysis as it becomes inefficient when dealing with large impression sizes typical in real customer data.

Model	CTR	AUC	MRR	NDCG5	NDCG10
Rule	2.92%	0.350	0.196	0.141	0.394
Bandit	**3.84%**	**0.405**	**0.226**	**0.191**	**0.425**

Table 4: Online evaluation vs. simulation based evaluation.

to 0.827 (15.5% improvement). However, this comes with a trade-off: optimized recall decreases from 0.439 to 0.303, while precision increases from 0.146 to 0.202, indicating the system becomes more selective in identifying true positive clicks with higher confidence.

Cross-model analysis reveals that GUES effectiveness varies by base model capability. Mistral Large shows modest improvements (F1: 0.205), while Sonnet variants demonstrate stronger optimization potential. The consistency of improvements across different agent-optimizer combinations (Sonnet 3 + Sonnet 3.5: F1 0.221) suggests that GUES captures generalizable patterns rather than exploiting specific model quirks. This behavioral alignment improvement is crucial for maintaining evaluation accuracy as user preferences and news content evolve over time.

4.3 Online Evaluation

We demonstrated the effectiveness of LLMEvalRec in a critical real-world scenario: evaluating an under-development news recommendation system with limited user base (fewer than 1,000 internal users). This exemplifies the "cold-start" evaluation challenge where traditional A/B testing lacks statistical power, and offline evaluation suffers from data scarcity. The challenge is amplified by the news domain's dynamic nature, with over 30,000 new articles from diverse sources updated daily, making historical data quickly obsolete.

Experimental Design: Our online evaluation compared two production-ready news ranking architectures. The Rule-based ranker employs expert-defined similarity matching with pre-collected user interest profiles, representing traditional content-based filtering approaches. The Bandit ranker implements a contextual multi-armed bandit with generalized linear rewards, learning user preferences through online interactions while balancing exploration-exploitation trade-offs.

Both systems shared identical candidate generation pipelines (filtering, diversity promotion, and freshness constraints) to ensure fair comparison, differing only in the final ranking stage. This design isolates ranking effectiveness while controlling for upstream effects.

Validation Protocol: We conducted parallel evaluation over a one-week period (2024-04-27 to 2024-05-03), with daily CTR calculation to account for temporal variations in user engagement patterns. Our simulation protocol mirrored real system dynamics: the first six days accumulated user interaction histories through LLMEvalRec agents, while the final day generated test impressions for performance measurement.

Predictive Accuracy: The alignment between simulated and real metrics validates our framework's predictive power. The Bandit system achieved 31.5% higher CTR than the Rule-based system (3.84% vs 2.92%), closely matching the simulated performance ratios across multiple metrics (AUC ratio: 1.16, MRR ratio: 1.15, NDCG@10 ratio: 1.08). This consistency across different evaluation perspectives demonstrates that LLMEvalRec captures the fundamental user behavior patterns driving real system performance, enabling confident deployment decisions in data-scarce environments.

5 RELATED WORK

We position LLMEvalRec within three key research areas that inform our approach: agent-based simulation methodologies from multi-agent systems research, recommender system simulation techniques, and autonomous agent applications for system evaluation. This interdisciplinary foundation enables our novel contribution of LLM-powered agents for news recommendation evaluation.

5.1 Agent-Based Simulations

Our work builds upon the rich tradition of agent-based modeling in complex systems evaluation. Multi-agent systems have long been used to simulate emergent behaviors in economic markets [**?**], social networks, and information systems. In the recommender systems domain, agent-based approaches offer unique advantages by modeling individual user autonomy, heterogeneous preferences, and interactive behaviors that aggregate into system-level performance patterns.

Recent advances in LLM-powered autonomous agents have opened new possibilities for realistic user simulation. Unlike traditional rule-based agents that require extensive domain knowledge engineering, LLM agents can exhibit complex reasoning, contextual understanding, and adaptive behavior patterns through natural language interfaces. This paradigm shift enables more accessible and flexible agent design while maintaining behavioral authenticity.

5.2 Recommender System Simulation

Simulations of recommendation systems have evolved from simple statistical models to sophisticated behavioral simulators. Early approaches like RecoGym[12] employed fixed user preference models with predetermined interaction patterns, limiting their ability to capture the dynamic nature of real user behavior. Sim2Rec[3] advanced the field by modeling historical interaction distributions, enabling more realistic temporal dynamics but requiring substantial historical data.

Recent work has begun leveraging Large Language Models for user simulation. SUBER[4] generates synthetic user interactions for reinforcement learning environments, while Agent4Rec[18] pioneered LLM agents with explicit user profiles and memory systems. SimUSER[2] introduced persona matching techniques for improved profile creation from historical data.

5.3 Autonomous Agents for System Evaluation

The application of autonomous agents for system evaluation represents a growing research area within multi-agent systems. Traditional evaluation approaches often fail to capture the interactive and emergent properties that arise from user-system interactions. Autonomous agents offer a solution by providing controllable yet realistic interaction patterns that can be systematically varied to explore system behavior under different conditions.

Our work extends this paradigm to news recommendation, where the challenges of dynamic content and evolving user interests make traditional evaluation methods particularly inadequate. Unlike existing approaches, LLMEvalRec employs a multi-agent architecture

where individual agents maintain persistent profiles, adapt through experience, and collectively generate system-level evaluation metrics. This approach enables thorough evaluation of recommender system performance while requiring minimal real user data, addressing a critical need in early-stage system development.

5.4 Coordination and Prompt Optimization

A critical challenge in multi-agent simulation systems is ensuring consistent and coordinated behavior across agents while maintaining individual autonomy. Traditional multi-agent systems achieve coordination through explicit communication protocols, shared knowledge bases, or centralized coordination mechanisms. In LLM-based agent systems, coordination emerges through consistent prompt engineering and behavioral alignment techniques.

Our GUES algorithm addresses this challenge by establishing coordinated behavioral patterns across multiple agents through systematic prompt optimization. Rather than requiring explicit inter-agent communication, GUES ensures behavioral consistency by optimizing the prompt templates that guide individual agent decisions. This approach maintains the autonomy of individual agents while ensuring their collective behavior aligns with observed user patterns, enabling scalable deployment of large agent populations for comprehensive system evaluation.

The episodic nature of GUES optimization enables parallel exploration of behavioral strategies, similar to distributed optimization approaches in multi-agent learning. By running multiple optimization episodes simultaneously and selecting the best-performing prompts, GUES achieves robust behavioral alignment while avoiding local optima that could compromise evaluation accuracy. This coordination mechanism represents a novel approach to managing large-scale LLM agent deployments for system evaluation tasks.

6 CONCLUSIONS

We introduced LLMEvalRec, a novel framework for evaluating news recommendation systems using LLM-powered generative agents. Our approach addresses the challenges of NRS evaluation by providing a flexible, data-efficient simulation environment that bridges the gap between offline and online methods. The experiments conducted demonstrate the effectiveness of LLMEvalRec in several key aspects. First, the framework enables robust offline evaluation using both public benchmark data and real customer data. Second, it facilitates online evaluation in cold-start scenarios, where limited real user data is available, allowing for informed decisions on development and deployment. We showed that the relative performance of recommendation systems measured through LLMEvalRec's simulated evaluation aligns well with the real-world online click-through rates. This alignment underscores the predictive power of our approach, validating its usefulness as a tool for optimizing and comparing NRS models before deployment.

As a future work, we plan to extend our validation to larger-scale deployments with diverse user populations and explore the framework's applicability to other recommendation domains beyond news, such as e-commerce and entertainment.

REFERENCES

[1] Krisztian Balog and ChengXiang Zhai. 2023. User Simulation for Evaluating Information Access Systems. *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region* (2023). https://api.semanticscholar.org/CorpusID:259165129

[2] Nicolas Bougie and Narimasa Watanabe. 2025. SimUSER: Simulating User Behavior with Large Language Models for Recommender System Evaluation. arXiv:2504.12722 [cs.IR] https://arxiv.org/abs/2504.12722

[3] Xiong-Hui Chen, Bowei He, Yang Yu, Qingyang Li, Zhiwei Qin, Wenjie Shang, Jieping Ye, and Chen Ma. 2023. Sim2rec: A simulator-based decision-making approach to optimize real-world long-term user engagement in sequential recommender systems. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 3389–3402.

[4] Nathan Corecco, Giorgio Piatti, Luca A Lanzendörfer, Flint Xiaofeng Fan, and Roger Wattenhofer. [n.d.]. SUBER: An RL Environment with Simulated Human Behavior for Recommender Systems. ([n. d.]).

[5] Andreea Iana, Goran Glavaš, and Heiko Paulheim. 2023. NewsRecLib: A PyTorch-Lightning Library for Neural News Recommendation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Yansong Feng and Els Lefever (Eds.). Association for Computational Linguistics, Singapore, 296–310. https://doi.org/10.18653/v1/2023.emnlp-demo.26

[6] Andreea Iana, Goran Glavas, and Heiko Paulheim. 2023. Train Once, Use Flexibly: A Modular Framework for Multi-Aspect Neural News Recommendation. *ArXiv* abs/2307.16089 (2023). https://api.semanticscholar.org/CorpusID:260334338

[7] Nithish Kannen, Yao Ma, Gerrit J.J. van den Burg, and Jean Baptiste Faddoul. 2024. Efficient Pointwise-Pairwise Learning-to-Rank for News Recommendation. In *Conference on Empirical Methods in Natural Language Processing*. https://api.semanticscholar.org/CorpusID:272910764

[8] Sean Lee, Aamir Shakir, Darius Koenig, and Julius Lipp. 2024. *Open Source Strikes Bread - New Fluffy Embeddings Model*. https://www.mixedbread.ai/blog/mxbai-embed-large-v1

[9] Dairui Liu, Boming Yang, Honghui Du, Derek Greene, Neil Hurley, Aonghus Lawlor, Ruihai Dong, and Irene Li. 2024. RecPrompt: A Self-tuning Prompting Framework for News Recommendation Using Large Language Models. https://doi.org/10.1145/3627673.3679987 arXiv:2312.10463 [cs.IR]

[10] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Conference on Empirical Methods in Natural Language Processing*. https://api.semanticscholar.org/CorpusID:1957433

[11] Tao Qi, Fangzhao Wu, Chuhan Wu, and Yongfeng Huang. 2022. News Recommendation with Candidate-aware User Modeling. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2022). https://api.semanticscholar.org/CorpusID:248085044

[12] David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. 2018. Recogym: A reinforcement learning environment for the problem of product recommendation in online advertising. *arXiv preprint arXiv:1808.00720* (2018).

[13] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. *Proceedings of the 2018 World Wide Web Conference* (2018). https://api.semanticscholar.org/CorpusID:4889971

[14] Xiaolei Wang, Xinyu Tang, Xin Zhao, Jingyuan Wang, and Ji-Rong Wen. 2023. Rethinking the Evaluation for Conversational Recommendation in the Era of Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.621

[15] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Attentive Multi-View Learning. In *International Joint Conference on Artificial Intelligence*. https://api.semanticscholar.org/CorpusID:196471084

[16] Chuhan Wu, Fangzhao Wu, Yongfeng Huang, and Xing Xie. 2021. Personalized News Recommendation: Methods and Challenges. *ACM Transactions on Information Systems* 41 (2021), 1 – 50. https://api.semanticscholar.org/CorpusID:247084474

[17] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A Large-scale Dataset for News Recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 3597–3606. https://doi.org/10.18653/v1/2020.acl-main.331

[18] An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. 2024. On Generative Agents in Recommendation. arXiv:2310.10108 [cs.IR] https://arxiv.org/abs/2310.10108