From Enhancement to Exploitation: The Dual Role of LLMs in Recommender Systems

Yuyue Zhao

From Enhancement to Exploitation: The Dual Role of LLMs in Recommender Systems

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in
de Agnietenkapel
op donderdag 23 oktober 2025 te 10:00 uur

door

Yuyue Zhao

geboren te Hunan

Promotiecommissie

Promotor:	prof. dr. M. de Rijke	Universiteit van Amsterdam
Co-promotor:	dr. J. Huang	Universiteit van Amsterdam
Overige leden:	dr. M. Alian Nejadi	Universiteit van Amsterdam
	prof. dr. P.T. Groth	Universiteit van Amsterdam
	prof. dr. E. Kanoulas	Universiteit van Amsterdam
	dr. M. Lalmas	Spotify
	prof. dr. M. Zhang	Tsinghua University

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

The research was carried out at the Information Retrieval Lab at the University of Amsterdam.

Printed by Ridderprint, The Netherlands

ISBN: 978-94-6522-765-8

Acknowledgements

The years of pursuing my Ph.D. have passed more quickly than I could have imagined. Looking back, this journey was filled with setbacks, failed experiments, and moments of doubt. Yet it was equally filled with the joy of exploration, the sense of achievement, and the happiness of being recognized. My name, Yuyue, literally means "happy" in Chinese, which might explain why, despite the challenges, I look back on my doctoral years with mostly fond memories. Now, standing at the end of this chapter, I feel overwhelmed with gratitude. I have so many people to thank—those who supported me through this time, helping me learn to think deeply, solve problems, and take pride in myself and my work.

First and foremost, I would like to express my deepest gratitude to my supervisor, Maarten de Rijke. I still remember the very first time we spoke online, when you carefully tried to understand my work and my research plans, just as you have done throughout my Ph.D. You've shown me what truly good research looks like, taught me not to get discouraged by poor results, and reminded me that analyzing failures can often contribute more to the community than straightforward successes. You've instilled in me a rigorous way of thinking, always encouraging me to ask "why" at every step. You have been much more than an academic supervisor; you have been a role model for my life. I feel truly fortunate to have walked this journey with you as my supervisor.

Second, I want to thank my co-promotor, Jin Huang. You are not only a mentor but also a dear friend. We argued, debated, and challenged each other until one of us was convinced by the other, and I learned so much from those moments. I'm not always a careful person, but you accepted my occasional oversights and patiently helped me work through problems. Thank you for your guidance, for your perspectives, and for the support that you gave to my Ph.D. experience.

I am also deeply grateful to the individuals who laid the foundation for my research career and guided me before I joined IRLab. Thank you, Xiangnan and Xiang, if it weren't for you, I never would have delved into recommender systems or had the chance to pursue my Ph.D. at UvA.

It is a great honor to have Mohammad Aliannejadi, Paul Groth, Evangelos Kanoulas, Mounia Lalmas, and Min Zhang as my Ph.D. committee members. I sincerely appreciate your valuable time devoted to reading and discussing my thesis.

I am sincerely thankful to everyone I met at IRLab: Ali, Ana, Andrew, Antonios, Arezoo, Barrie, Catherine, Chen, Chuan, Clara, Clemencia, Daniel, David, Evangelos, Fatemeh, Gabriel, Gabrielle, Georgios, Gionata, Hongyi, Ivana, Jia-Hong, Jia-Huei, Jasmin, Jiahuan, Jin, Jingfen, Jingwei, Julien, Kidist, Lu, Maarten, Maartje, Maik, Maria, Mariya, Maryam, Maurits, Maxime, Ming, Mohammad, Mohanna, Mounia, Mozhdeh, Olivier, Panagiotis, Petra, Philipp, Pooya, Romain, Roxana, Ruben, Ruqing, Sam, Samarth, Sami, Shashank, Siddharth Mehrotra, Siddharth Singh, Simon, Songgaojun, Teng, Thilina, Thong, Vaishali, Weijia, Xinyi, Yibin, Yixing, Yongkang, Yougang, Yuanna, Yubao, Zahra, Zhaochun, Zihan, Zhirui and Ziyi. You've all made the lab feel like a warm, welcoming place where I could truly enjoy myself. I loved our Soos talks, research meetings, SEA talks, reading groups, discussion groups, and all the social events, like Friday drinks, barbecues, dumpling parties, and Christmas gatherings. It's

been a real privilege to share this journey with you.

Finally, I want to thank my parents, Zhijun and Hairong. Papa and Mama, thank you for always standing by my side no matter what choices I made, and for your unconditional support. I wish you good health and happiness forever.

Yuyue Zhao
China
September 1, 2025

Contents

1 Introduction

Recommender systems (RSs) have become essential components of modern digital ecosystems, delivering personalized suggestions that enhance user satisfaction across diverse applications [29, 38, 52, 95, 121, 134, 135, 145]. These systems power recommendations in e-commerce, streaming services, and news platforms, predicting user preferences to suggest items, such as products, movies, or articles, tailored to individual interests. Conventional RSs predominantly rely on neural representations learned from behavioral logs such as clicks and ratings [13, 74, 95, 155]. While effective, this paradigm faces a critical limitation: it often lacks commonsense knowledge, which restricts the system's ability to interpret nuanced content and deliver contextually relevant recommendations, particularly in domains that require deeper semantic understanding [70, 170].

Large language models (LLMs) have recently demonstrated strong capacities for commonsense reasoning, open-world knowledge, and nuanced understanding and generation of text [70, 136]. Trained on large-scale datasets, models such as GPT [80] and Llama [112] learn complex knowledge and language patterns that are beyond the reach of conventional RSs embeddings. These models are great at generating text and understanding huge amounts of unstructured data, making them promising for enhancing RSs. In text-rich domains such as news recommendations, LLMs can use article content and users' historical interactions to improve the semantic understanding of items and refine preference modeling [71, 165]. By integrating LLMs, RSs can address the limits of traditional neural approaches, potentially yielding more accurate and context-aware recommendations [70, 137].

Nevertheless, incorporating LLMs into RSs introduces significant challenges arising from the mismatch between language-generation and recommendation objectives [70, 137]. One prominent issue is *hallucination*, in which LLMs generate irrelevant or incorrect outputs that do not match user preferences or item characteristics [7]. This problem becomes more serious in large item catalogs, as it is important to identify items correctly. Additionally, LLMs may suffer from *under-representation*, a misalignment between user behavior (*e.g.,* clicks or ratings) and the item semantics inferred from textual data [120, 165]. For example, a user might click on a news article out of simple curiosity rather than strong interest, yet an LLM may over-emphasize textual content, leading to skewed recommendations. Existing literature on LLM-enhanced RSs highlights these issues, exploring fine-tuning strategies and hybrid models to bridge the

gap between semantic and behavioral spaces [68, 76, 118]. Despite these efforts, how to better use LLMs to improve recommendation accuracy remains an open question, requiring further exploration of their practical effectiveness.

Beyond their potential for enhancement, LLMs expose RSs to new vulnerabilities, particularly in text-rich domains like news recommendation [147]. News RSs rely on textual content to rank and recommend articles, making them more vulnerable to adversarial textual attacks [63]. In textual attacks, malicious content providers can exploit LLMs' advanced text-generation capabilities to rewrite news articles, boosting their recommendation rankings while preserving semantic similarity [123, 153]. These attacks carry serious implications, potentially exposing users to biased or misleading information, and their text-level modifications make them difficult to detect, amplifying the risks in domains where trust is important. Unlike traditional shilling or injection attacks, textual attacks are harder to detect because of their content-level nature and the authority of publishers to edit articles [147, 153]. The reliance of news RSs on textual information further amplifies this risk, underscoring the need for robust defenses.

Existing textual attack methods primarily manipulate news content to boost rankings, but they typically overlook a critical dimension of news: *media bias orientation*. Media bias orientation refers to the ideological stance of an article, often categorized as left, center, or right, shown through its tone, word choice, affecting how readers understand it [96, 103]. Ignoring media bias orientation can reduce an attack's efficacy, as rewritten content that conflicts with a platform's or user's ideological stance is more likely to be detected or disregarded [37, 58, 104]. Preserving the original bias may be crucial when rewriting news, as it helps maintain the trust and engagement of user groups with original views; changing the bias orientation can weaken that trust [4, 77]. Consequently, maintaining media bias orientation during a textual attack has the potential to produce stealthier and more effective manipulations, allowing the modified content to remain consistent with the original article's ideological stance. However, most existing methods fail to account for this factor, limiting their effectiveness.

In this thesis, we examine the dual role of LLMs in RSs, exploring both their capacity for enhancement and their potential for exploitation. On the enhancement side, we will focus on improving RSs by alleviating problems like hallucination and under-representation, using innovative tool-learning methods and examining fine-tuning strategies. By aligning text semantics more closely with user preferences, recommendation reliability and accuracy can be improved. On the exploitation side, we will investigate how LLMs can be used to expose vulnerabilities in RSs through adversarial textual attacks. We will further discuss how preserving media bias orientation enhances attack stealth and efficacy. By analyzing both how LLMs can be used for textual attacks and how bias preservation can aid attacks, this side aims to reveal the need for defenses to protect the integrity of recommendations.

In a nutshell, the thesis contributes to the responsible use of LLMs in RSs, offering insights into both their benefits and risks.

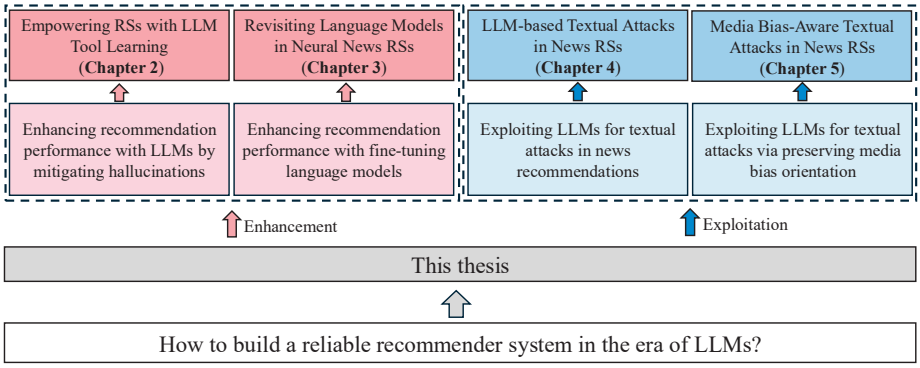


Figure 1.1: Helicopter view of the research agenda underlying this thesis.

1.1 Research Outline and Questions

Designing RSs that remain reliable in the era of LLMs involves many dimensions, such as hallucinations, privacy, bias, adversarial attacks, and more [22, 94, 167]. Rather than attempting to exhaust this extensive potential direction, this thesis zooms in on two key themes in the integration of LLMs with RSs, as illustrated in Figure 1.1:

(1) Enhancing recommendation performance by using LLMs to address issues of hallucination and under-representation.

(2) Examining how LLMs can be exploited to perform adversarial textual attacks that manipulate recommendation outcomes.

Together, these themes highlight the dual role of LLMs in RSs, as both enhancers and potential threats, and contribute to our key scientific question: *How to build a reliable recommender system in the era of large language models?*

1.1.1 Enhancing Recommendation Performance with LLMs by Mitigating Hallucinations

As noted above, conventional RSs rely heavily on historical interaction data to predict user preferences and recommend items [74, 98]. While this approach has proven effective, it faces significant limitations, such as a lack of commonsense knowledge and struggles to capture the nuanced intent of users, which may result in recommendations that do not fully align with user preferences or contextual needs [70]. LLMs, with their advanced capabilities in commonsense reasoning, open-world knowledge, and text understanding, offer a promising solution to address these shortcomings [136]. By using their rich semantic understanding and inference capabilities, LLMs can enhance the controllability, precision, and flexibility of recommendation systems [70, 70, 137, 170]. They can analyze user profiles, recent behaviors, and contextual factors to infer latent interests and generate recommendations that are both personalized and diverse. However, deploying LLMs for recommendations introduces a critical challenge: the risk of *hallucinations* [7]. LLMs are trained on language tasks, with limited knowledge of

user-item interactions in recommendation scenarios. Therefore, LLMs may generate recommendations that are factually incorrect or misaligned with user needs. This motivates the following research question:

RQ1 How can LLMs be used to enhance recommendation performance while mitigating hallucinations in general recommendation scenarios?

To answer **RQ1**, we propose a novel framework, ToolRec, which uses LLMs as surrogate users to simulate human-like decision-making in recommendation scenarios. The LLM is initialized with the user's behavior history and interacts adaptively with attribute-oriented tools to explore and refine the item space. The framework integrates three components: a decision simulation module, where the LLM assesses the scenario and makes choices based on historical preferences; attribute-oriented tools, which guide retrieval and ranking of items according to specific attributes such as genre or release year; and a memory strategy that tracks tool interactions and intermediate results, enabling consistent and informed refinement of recommendations. Through iterative interactions between the surrogate user and these tools, the system generates a user-centric item list that aligns closely with the user's preferences.

Our experiments show that the proposed framework consistently outperforms baselines such as SASRec and BERT4Rec on the ML-1M and Amazon-Book benchmarks, confirming the effectiveness of simulating user decision-making and using attribute-oriented tools to enhance recommendation diversity and relevance. The results highlight the value of combining LLM reasoning with structured tool interactions, which not only improves recommendation quality but also reduces the risk of hallucinations by grounding outputs in concrete item attributes and minimizing irrelevant suggestions.

1.1.2 Enhancing Recommendation Performance by Fine-tuning Language Models

Unlike general recommendation scenarios, such as e-commerce or movie recommendations, news RSs rely on analyzing the textual content of news articles to generate recommendations [46, 63]. These systems need to deeply understand article semantics to align recommendations with user preferences, a task that goes beyond what traditional behavioral data can provide. A key challenge in integrating LLMs into news RSs is *under-representation*, where user behavior, such as clicks, does not fully align with the representation extracted from textual data by LLMs [120, 165]. This issue is particularly severe in news RSs due to the complexity and variability of textual information, creating a unique challenge compared to the *hallucinations* mentioned earlier. To address under-representation, fine-tuning LMs on news-specific tasks provides a solution [129, 165]. By training models on a combination of article content and user interaction behaviors, we can improve their ability to match the meaning of articles with what users want. This approach makes news recommendations more accurate and relevant, ensuring the system understands both the content and user preferences well. This motivates our investigation into optimizing LLMs to better align with user preferences in news RSs, leading to the following research question:

RQ2 How can fine-tuning language models improve the capture of user preferences

and enhance recommendations in semantic-rich news recommendation scenarios while avoiding under-representation?

To address **RQ2**, we investigate the impact of fine-tuning various language models, including small language models like GloVe, pre-trained language models such as BERT and RoBERTa, and large language models like Llama3.1-8B, on the performance of news RSs. Our approach involves fine-tuning these LMs on the MIND-small dataset [133] to enhance their ability to generate semantic representations that align with user preferences. We explore a range of fine-tuning strategies, such as adjusting the number of layers fine-tuned (*e.g.,* full-model versus layer-specific fine-tuning) and incorporating domain-specific tasks like predicting user engagement based on article semantics. These strategies aim to improve recommendation relevance and address challenges like under-representation of diverse user interests.

Our experiments suggest that fine-tuning enhances the LLMs' capacity to interpret semantic-rich news content accurately, leading to improved recommendation relevance. Furthermore, by addressing under-representation, fine-tuned models are anticipated to reduce biases in preference modeling, such as over-reliance on textual signals that do not align with user intent, thereby delivering more robust and context-aware news recommendations.

1.1.3 Exploiting LLMs for Textual Attacks in News Recommendations

As previously noted, news RSs are crucial for delivering personalized content. They rely heavily on the text in articles to rank and suggest news to users. However, this dependence on text increases their vulnerability to adversarial attacks that manipulate article content to influence recommendations [44, 78]. Textual attacks, where malicious content providers rewrite news articles to enhance their visibility in RSs, pose significant risks to the trustworthiness of news platforms [78, 123, 153]. Such manipulations can expose users to misleading or biased information, potentially shaping public opinion in unintended ways. The advanced text-generation capabilities of LLMs make them particularly effective for crafting these attacks, enabling content perturbations while preserving semantic similarity to the original articles [153].

Existing textual attack methods face limitations when applied to the dynamic and semantically rich context of news recommendations. Many approaches, such as those injecting fake interactions or mimicking popular item traits, require extensive prior knowledge of the target RS or struggle to adapt to the rapidly evolving nature of news content [123, 153]. This highlights the need to explore LLMs for crafting textual attacks that identify rewrite patterns capable of successfully boosting news rankings. This leads to the following research question:

RQ3 Can LLMs be exploited to conduct textual attacks in semantic-rich news recommendation scenarios?

To address **RQ3**, we propose LANCE, a two-stage framework that uses LLMs to generate rewritten news content designed to boost the ranking of target articles in news RSs. In the first stage, an *explorer* component employs an LLM to create diverse

rewritten versions of a news article. Then the generated rewrites are evaluated against a target RS to identify successful attacks that improve the article's ranking. In the second stage, a *reflector* module is fine-tuned using triplets of the original article, a successful rewrite, and a failed rewrite. This fine-tuning trains the model to generate effective attacks that boost news rankings.

Experiments conducted on three news RSs demonstrate that LANCE achieves state-of-the-art attack performance. Notably, our study reveals that negative and neutral rewrites outperform positive ones in the news domain, uncovering a unique vulnerability in news RSs. Additionally, LANCE exhibits strong generalization, with a model trained on one RS effectively attacking other news RS systems.

1.1.4 Exploiting LLMs for Textual Attacks by Preserving Media Bias Orientation

In the news domain, media bias refers to the ideological stance (*e.g.,* left, center, or right) reflected in an article's tone and word choice, which influences content delivery and user engagement [96]. Textual attacks, which manipulate article content to boost rankings, could benefit from preserving this bias to stay stealthy and effective. Changing the article's original bias orientation risks detection by platforms, as it may conflict with expected ideological patterns, or disengage users whose preferences align with specific stances [4, 77]. By maintaining the article's bias, attackers can rewrite news to increase its visibility while keeping the ideological tone consistent, making the attack harder to detect [83, 106].

Current textual attack methods focus on boosting rankings but often ignore preserving media bias [78, 123, 153]. This oversight can reduce the attack's effectiveness. A media bias-aware approach is essential to keep the attack discreet and ensure it blends naturally with the platform's content. This need motivates exploring advanced techniques, leading to the following research question:

RQ4 How can LLMs be used to conduct textual attacks that preserve media bias orientation in news recommender systems?

To address **RQ4**, we propose BALANCE, a new framework that uses LLMs to rewrite news articles, aiming to boost their rankings in RSs while keeping their original media bias orientation. BALANCE employs a progressive fine-tuning strategy to overcome the trade-off between ranking improvement and bias preservation. Initially, we train specialized LLM experts independently: one for optimizing ranking using rank-focused data, and another for maintaining bias using bias-specific data. Subsequently, we integrate their capabilities by fine-tuning on a combined dataset, which includes articles rewritten to achieve both objectives. This approach mitigates the scarcity of combined data and aligns the conflicting goals (rank improvement and bias maintenance), enabling effective attacks.

Experiments show that BALANCE outperforms existing methods in boosting news rankings while preserving media bias orientation. Evaluated on MIND datasets, BALANCE consistently maintains bias across ideological groups (*e.g.,* strongly preserving right-leaning bias) with minimal impact on recommendation quality. Simulated user studies also indicate that bias-aware rewrites are harder to detect, increasing attack

stealth. These results highlight BALANCE's ability to discreetly exploit vulnerabilities in news RSs, emphasizing the need for strong defenses against such attacks.

1.2 Main Contributions

In this section, the main contributions of this thesis are summarized.

1.2.1 Algorithmic Contributions

The algorithmic contributions of this thesis come in the form of four models:

(1) *A tool-augmented LLM framework for recommendation*: a new interactive approach that treats LLMs as surrogate users to simulate human-like decision-making (ToolRec) (Chapter 2). The framework introduces a multi-round decision process in which the LLM iteratively selects item attributes, retrieves candidates using external tools, and refines the item set through reasoning. To enable precise and controllable retrieval, two types of attribute-oriented tools are designed: retrieval tools, which extract items based on specific attributes, and rank tools, which reorder candidate sets according to user-aligned semantics. To improve reliability and facilitate multi-tool coordination, a memory strategy is implemented that verifies, records, and contextualizes tool outputs.

(2) *A language model-enhanced framework for news recommendation*: a systematic framework that evaluates SLMs, PLMs, and LLMs as news encoders within a standard neural recommender architecture (Chapter 3). Each LM is evaluated in both non-fine-tuned and fine-tuned modes to explore its effect on modeling textual features of news. SLMs provide static embeddings, PLMs use contextualized representations via transformer encoders, and LLMs use prompt-based embeddings refined through a two-stage fine-tuning process. Fine-tuning strategies are carefully designed to balance performance gains with computational cost.

(3) *A LLM-based textual attack framework for news recommendation*: a novel textual attack framework introduces a two-stage architecture (LANCE) (Chapter 4). The *explorer* prompts an LLM to generate diverse rewrites of news articles by varying writing style, sentiment, and persona, simulating editorial changes. It then filters effective rewrites that significantly improve item rankings in the victim RS using a filtering mechanism. These collected pairs of successful and failed rewrites form training data for the *reflector*, which is fine-tuned using direct preference optimization to produce optimized attack texts. At inference time, the fine-tuned *reflector* rewrites target news articles to promote their ranking within the RS.

(4) *A progressive fine-tuning framework for bias-aware textual attacks*: a LLM-based approach to rewriting news content for bias-aware textual attack in news RSs (BALANCE) (Chapter 5). The proposed framework introduces a progressive fine-tuning architecture designed to generate rewritten news articles that both improve ranking and preserve the original media bias orientation. The method begins by using an LLM to rewrite original news across four textual dimensions—bias

orientation, writing style, sentiment, and author persona. These rewrites are labeled into three categories: Rank data (rewrites that improve ranking), Bias data (rewrites that preserve bias orientation), and Combined data (rewrites achieving both). Given the scarcity of high-quality Combined data, the framework adopts a progressive fine-tuning strategy. In the first stage, two task-specific LLMs are trained using direct preference optimization: a rank expert fine-tuned on Rank data and a bias expert on Bias data. Using LoRA, both adaptations are merged into a unified model, which is then refined in the second stage using the Combined data. A new set of LoRA parameters is fine-tuned while the merged model remains frozen, enabling the final model to generate rewrites that balance both attack goals.

1.2.2 Experimental Contributions

(1) The experimental contributions made in Chapter 2 are: (a) An empirical comparison of ToolRec cross three real-world datasets and eight baselines. (b) An ablation study on core components: user decision simulation, termination rounds, and attribute-oriented retrieval tools. And (c) analysis of the LLM's reasoning capabilities, round-wise user decision dynamics.

(2) In Chapter 3 our experimental contributions are: (a) A comprehensive evaluation of different LMs (SLMs, PLMs, and LLMs) as news encoders. (b) A detailed analysis of the effects of fine-tuning on LM-based news RSs. (c) An investigation into the impact of LM size and architecture on recommendation accuracy. And (d) an empirical study on cold-start users.

(3) The experimental contributions made in Chapter 4 are: (a) An empirical comparison of LANCE with eight baseline textual attack methods across three news RSs. (b) An ablation study analyzing the impact of rewriting dimensions (style, sentiment, persona) and rank thresholds on attack effectiveness. (c) An empirical assessment of the DPO-trained *reflector*. And (d) an analysis of cross-system generalization.

(4) And, finally, in Chapter 5 the experimental contributions are: (a) A comprehensive evaluation of BALANCE on MIND datasets with three news RS models. (b) An empirical analysis of the trade-off between ranking and bias preservation. (c) An ablation study of rank expert, bias expert, and unified adaptation. And (d) a simulated user study showing that BALANCE produces more stealthy, bias-consistent rewrites with lower detectability.

1.3 Thesis Overview

We provide a brief overview of the content of each chapter in this thesis. In this thesis, we investigate the integration of LLMs with RSs to enhance recommendation performance while tackling challenges like hallucinations and under-representation, and to explore their potential for exploitation through textual attacks, as illustrated in the research outline (Figure 1.1).

In Chapter 2, we address hallucinations in general recommendation scenarios by proposing ToolRec, a framework that uses LLMs as surrogate users to simulate human-like decision-making. This approach employs attribute-oriented tools to refine recommendations, improving relevance and diversity.

In Chapter 3, we focus on news recommendation, exploring fine-tuning language models to better capture user preferences and mitigate under-representation, ensuring recommendations align with semantic-rich content.

In Chapter 4, we investigate vulnerabilities in news RS by introducing LANCE, a two-stage framework that uses LLMs to conduct textual attacks, rewriting news content to manipulate recommendation outcomes and expose risks.

In Chapter 5, we propose BALANCE, a framework for textual attacks that preserve media bias orientation in news RS, enhancing attack effectiveness while highlighting the need for robust defenses.

Finally, in Chapter 6, we summarize the thesis, discuss its limitations, and outline future research directions. Through this work, we contribute to the responsible use of LLMs in RSs, providing insights into their benefits and risks.

1.4 Origins

The research chapters in this thesis are built upon the following publications.

Chapter 2 is based on the following paper:

- Y. Zhao, J. Wu, X. Wang, W. Tang, D. Wang, and M. de Rijke. Let me do it for you: Towards LLM empowered recommendation via tool learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 1796–1806. ACM, 2024.

  YZ: Conceptualization, Data Curation, Formal Analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing. JW: Investigation, Software, Formal Analysis, Writing – Review & Editing. XW: Investigation, Methodology, Writing – Review & Editing. WT: Data Curation, Discussion. DW: Validation, Discussion. MdR: Conceptualization, Methodology, Supervision, Writing – Review & Editing.

Chapter 3 is based on the following paper:

- Y. Zhao, J. Huang, D. Vos, and M. de Rijke. Revisiting language models in neural news recommender systems. In *Advances in Information Retrieval - 47th European Conference on Information Retrieval, ECIR 2025, Lucca, Italy, April 6-10, 2025, Proceedings, Part IV*, volume 15575 of *Lecture Notes in Computer Science*, pages 161–176. Springer, 2025.

  YZ: Conceptualization, Data Curation, Formal Analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – Original Draft

Preparation, Writing – Review & Editing. JH: Conceptualization, Methodology, Validation, Writing – Review & Editing. DV: Formal Analysis, Investigation. MdR: Conceptualization, Methodology, Supervision, Writing – Review & Editing.

Chapter 4 is based on the following paper:

- Y. Zhao, J. Huang, S. Liu, J. Wu, X. Wang, and M. de Rijke. LANCE: Exploration and reflection for LLM-based textual attacks on news recommender systems. In *RecSys 2025: 19th ACM Conference on Recommender Systems*. ACM, September 2025.

  YZ: Conceptualization, Data Curation, Formal Analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing. JH: Conceptualization, Methodology, Supervision, Validation, Writing – Review & Editing. SL: Investigation, Writing – Review & Editing. JW: Investigation, Discussion. XW: Methodology, Discussion. MdR: Conceptualization, Methodology, Supervision, Writing – Review & Editing.

Chapter 5 is based on the following paper:

- Y. Zhao, Y. Li, J. Huang, X. Wang, and M. de Rijke. Unseen threats: Media bias-aware textual attacks on news recommender systems. *Submitted for review*, 2025.

  YZ: Conceptualization, Data Curation, Formal Analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing. YL: Formal Analysis, Investigation, Writing – Review & Editing. JH: Conceptualization, Methodology, Supervision, Validation, Writing – Review & Editing. XW: Methodology, Investigation, Discussion. MdR: Conceptualization, Methodology, Supervision, Writing – Review & Editing.

The writing of the thesis also benefited from work on the following publications:

- Y. Zhao, X. Wang, J. Chen, Y. Wang, W. Tang, X. He, and H. Xie. Time-aware path reasoning on knowledge graph for recommendation. *ACM Transactions on Information Systems*, 41(2):26:1–26:26, 2023.

- Y. Zhao, J. Huang, and M. de Rijke. Can LLMs serve as user simulators for recommender systems? *The Search Futures Workshop at ECIR 2024, SIGIR Forum*, 58(1):1–41, 2024.

- W. Tang, B. Xu, Y. Zhao, Z. Mao, Y. Liu, Y. Liao, and H. Xie. UniRel: Unified representation and interaction for joint relational triple extraction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 7087–7099. Association for Computational Linguistics, 2022.

- W. Tang, Y. Cao, J. Ying, B. Wang, Y. Zhao, Y. Liao, and P. Zhou. A + B: A general generator-reader framework for optimizing LLMs to unleash synergy potential. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 3670–3685. Association for Computational Linguistics, 2024.
- W. Tang, Y. Cao, Y. Deng, J. Ying, B. Wang, Y. Yang, Y. Zhao, Q. Zhang, X. Huang, Y. Jiang, and Y. Liao. EvoWiki: Evaluating LLMs on evolving knowledge. In *Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27-August 1st, 2025*. Association for Computational Linguistics, 2025.

Part I

Leveraging LLMs to Enhance Recommendation Quality

2

Empowering RSs with LLM Tool Learning

In this chapter, we address **RQ1**: How can LLMs be used to enhance recommendation performance while mitigating hallucinations in general recommendation scenarios? We propose ToolRec, a novel framework that uses LLMs as surrogate users to simulate human-like decision-making, integrating attribute-oriented tools and a memory strategy to refine recommendations iteratively and ground outputs in structured item attributes.

2.1 Introduction

Recommender systems (RSs) are typically designed to identify user preferences and subsequently suggest potential items of interest [29, 38, 52, 95, 121, 134, 135, 145]. This strategy has two important limitations. First, the capabilities of existing RSs to accurately capture a user's true preferences are limited when relying solely on historical interaction data. Second, conventional RSs are often "narrow experts," lacking commonsense knowledge about users and items, which leads to a restricted scope of recommendations [36].

Inspired by the capabilities of large language models (LLMs) to perform commonsense reasoning and use knowledge, there have been several attempts to integrate LLMs with RSs and mitigate their inherent limitations [70, 170]:

- **LLMs as RSs**: Here, LLMs, whether initially trained or further fine-tuned using user-item interaction data, are adapted to serve as RSs [76, 118, 138] and directly generate candidate items in text. This approach easily suffers from the hallucination problem [7], especially given large item catalogue sizes and extensive item names [45].

- **LLMs enhance RSs:** Here, RSs are enhanced with world knowledge and reasoning abilities of LLMs [68, 76, 90, 118, 138, 154]. This category limits LLMs to

This chapter was published as Y. Zhao, J. Wu, X. Wang, W. Tang, D. Wang, and M. de Rijke. Let me do it for you: Towards LLM empowered recommendation via tool learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 1796–1806. ACM, 2024.

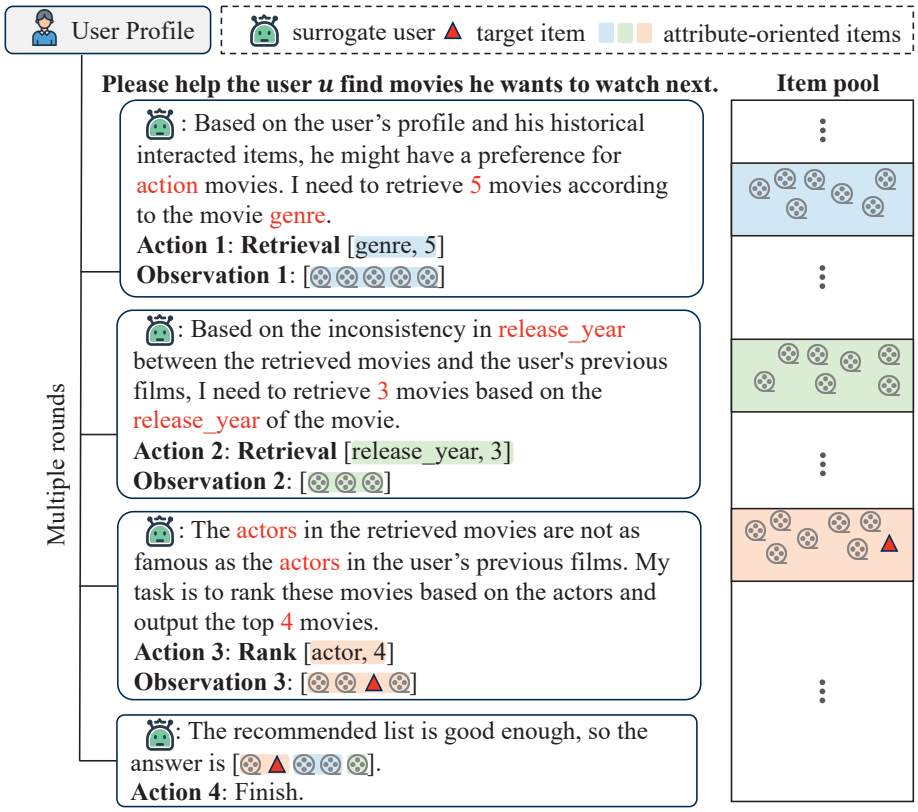


Figure 2.1: Illustrating how ToolRec works. The LLM-based *surrogate user* learns the real user's preferences and decides to employ attribute-oriented tools to explore areas of items. This process leads to a broad view of items, which, in turn, leads to the successful retrieval of the target item. It is important to note that the areas representing different attribute-oriented items that are retrieved according to a specific attribute may contain overlapping elements.

offering semantic information within a conventional recommendation paradigm, sometimes leading to inconsistencies between the semantic and behavior spaces.

- **LLMs control RSs:** Here, LLMs are used to monitor and control the recommendation pipeline. Existing controllers either have simple control strategies [31, 39] or necessitate active user involvement [28]. Their decisions are rarely human-like, which hinders their effectiveness in applications.

Tool Learning for Recommendations. Motivated by recent advancements in tool learning with foundation models [*e.g.,* 99, 101, 146], we propose to use LLMs as a surrogate user to emulate her/his decision-making process along with the use of tools. At the core of our proposal is the task of learning to adaptively select appropriate recommender tools and curate a user-centric item list that is aligned with the user's preferences. Figure 2.1 illustrates a four-round example of how ToolRec works. The

LLM starts the simulation by focusing on the movie genre and selects an initial set of 5 movies; satisfied with the genre of the returned movies, it then aims to complement the set based on the release_year, and retrieves 3 additional movies; subsequently, the LLM refines its focus towards the actors, leading to an adjustment of four movies in the list. Such iterative refinements continue until the simulator deems the movie list satisfactory enough to include the item of interest, thus finishing the recommendation process. Notice how the LLM directs the recommendation through multiple decision-making rounds with attribute signals, contrasting with conventional RSs. By adopting this approach, we aim to move beyond relying solely on users' historical interactions, resulting in a more tailored set of recommended items (*cf.* the red area on the right of Figure 2.1).

Using LLM tool learning to simulate users' decision-making processes presents distinct challenges. The first challenge concerns the recommendation ability of LLMs. Although LLMs are pretrained on extensive datasets [5, 80, 112, 113], improving their ability to produce quality recommendations remains a challenge, particularly in domain-specific scenarios. The second challenge is developing appropriate attribute-oriented tools. Attribute-oriented tools should not only be capable of exploring facets of the item pool (*e.g.,* genre, release year of movies) but also need to be effective in handling different attribute choices during decision simulation. Lastly, using LLMs to refine the set of candidate items in each round presents a significant challenge. This step could ensure that the final results benefit from an LLM's open-world knowledge and are not limited by a single "narrow expert."

A New Proposal for Tool Learning for Recommendations. To address the challenges listed above, we introduce ToolRec, for LLM-empowered recommendations via tool learning, which is aimed at aligning the emergent abilities of LLMs with the demands of recommender systems. ToolRec comprises three key components: (i) A **user decision simulation module**: We use LLMs initialized with user behavior history, acting as a *surrogate user* to evaluate user preferences against the current scenario. (ii) **Attribute-oriented tools**: We develop two distinct sets of attribute-based tools: rank tools and retrieval tools. The ranking tools are operationalized by LLMs with attribute-oriented ranking instructions, while the retrieval tools are operationalized by merging a frozen backbone with additional fine-tuned attribute encoders. These tools are activated when the *surrogate user* identifies unsatisfactory attributes, and then fetches corresponding candidate items. (iii) A **memory strategy**: This component checks the presence of intermediate results and stores them with associated tool marks, aiding the LLM in using open-world knowledge to refine the final recommendation list. The latter two components are governed by the *surrogate user*'s decisions, and conclude the recommendation process once a satisfactory candidate item list has been found. ToolRec's iterative framework integrates LLMs into recommender systems while enhancing the quality of recommendations.

Contributions. Our main contributions in this chapter are as follows:

- We propose ToolRec, a framework that deploys LLMs to enhance recommendations via tool learning. It employs LLMs to closely emulate user preferences, thereby improving the accuracy of recommendations generated during user decision simulation.

- To better meet the surrogate user's needs, we incorporate attribute-oriented tools and a memory strategy. Those components address the challenge of effective item retrieval based on identified attributes, ensuring the recommendations are well-aligned with user preferences.
- Experimental results on three real-world datasets demonstrate the effectiveness of ToolRec, especially in domains enriched by world knowledge.

2.2 Related Work

We review tool learning with LLMs and the use of LLMs for recommendation.

2.2.1 LLMs with Tool Learning

There is a growing trend to employ LLMs to construct autonomous agents to achieve decision-making capabilities [18, 89, 99, 101, 169]. These LLM-based agents often fall short in domains that demand extensive expert knowledge and suffer from hallucination issues [119]. To alleviate these problems, these agents are enhanced with the ability to invoke external tools for action execution. Previous external tools can be grouped into three types: APIs [64, 88, 89, 99, 101], Databases & Knowledge Bases [32, 40, 53], and External Models [107, 143, 168].

The use of APIs as tools has become a popular approach. For example, HuggingGPT [101] employs models on HuggingFace to accomplish complex user tasks. API-Bank [64] serves as an LLM-based API recommendation agent, and autonomously searches and generates suitable API calls across various programming languages and domains. ToolBench [88] is an LLM-based tool generation system that creates various tools based on natural language requirements. Connecting LLMs to external databases or knowledge bases enables access to domain-specific information, thus generating more realistic actions. For example, ChatDB [40] uses SQL statements to query databases, enabling logical action execution by agents. Similarly, in the recommendation scenario, RecMind [122] and InteRecAgent [45] engage various expert systems such as MySQL and planners to retrieve detailed item information. Employing external models can expand the range of feasible actions. For example, MemoryBank [168] employs two language models to enhance text retrieval capabilities: one for encoding input text and the other for matching query statements. MM-REACT [143] integrates various vision models to improve its performance in visual understanding tasks.

Our attribute-oriented rank tools can be summarized into APIs, while retrieval tools can be summarized into external models. This approach helps our surrogate user to explore different interest areas and return a better user preference-aligned item set.

2.2.2 LLMs for Recommendation

Methods using LLMs in RSs come in three groups, based on the role of the LLMs: LLMs as RSs [9, 33, 34, 43, 140, 144, 152], LLMs as assistants [76, 118, 138], and LLMs as pipeline controllers [28, 31, 45, 122].

LLMs as RSs involve using LLMs to generate candidate items. For instance, P5 [33] is fine-tuned on T5 [92] with a collection of personalized prompts to achieve zero-shot generalization. Following this, UP5 [43] and VIP5 [34] extend the paradigm to fairness and multimodal tasks, respectively. InstructRec [152] views recommendation as an instruction following task for LLMs, tuning T5 with user-specific instructions. TALLRec [9] trains LLaMA [112] with LoRA [41] to follow instructions and respond to a binary query provided within the contextual information. LLaRA [68] uses hybrid prompting to bridge the modality gap between traditional recommendation systems and LLMs.

When LLMs serve as Assistants, LLMs are used for their factual knowledge or reasoning capabilities to generate or encode auxiliary textual features, assisting conventional RSs with semantic information. For example, Various studies use auxiliary textual features from BERT [23] for document ranking [172], while others aim to leverage these features for news recommendation [90, 129, 154]. KAR [138] extracts reasoning and factual knowledge from LLMs to serve as augmented features, enhancing recommendation models in a model-agnostic manner.

In the setting where LLMs act as Pipeline Controllers, models such as ChatREC [31] use ChatGPT to understand user preferences, decide whether to use the backend recommender system, and refine the suggested items before showing them to the user. RecLLM [28] and InteRecAgent [45] propose frameworks for integrated conversational recommender systems with LLMs managing dialogues, understanding user preferences. InteRecAgent further determines the usage of various tools (i.e., information query tools, retrieval tools, and rank tools) to support the candidate items. RecMind [122], inspired by P5's various-task experiment setup, manages tool usage through a Self-Inspiring mechanism. Agent4Rec [149] simulates and infers user-personalized preferences and behavior patterns using a LLM-based movie recommendation simulator.

While these pipeline controller efforts address challenges similar to ours, there are several aspects that make our approach different from theirs. First, we focus on the sequential recommendation task and top-N recommendation setting. Second, we introduce attribute-oriented tools designed specifically for the recommendation exploration journey. Lastly, we propose a proactive analysis of preference mismatches by using an LLM as a surrogate user.

2.3 Methodology

We propose ToolRec, as illustrated in Figure 2.2. We first formalize the recommendation process as a process exploring users' interests. Then, we introduce a general framework that adapts LLMs as surrogate users to enhance the recommendation mechanism through attribute-oriented tools.

2.3.1 Problem Formulation

In the context of sequential recommendation, as illustrated in Figure 2.1, given user u with a historical interaction sequence $\mathcal{H} = \{i^1, i^2, \ldots, i^{n-1}\}$, the goal is to predict the next item of interest $i^n \in \mathcal{I}$, where $\mathcal{I}$ is the complete item pool. Conventional RSs retrieve a subset $\mathcal{I}_c \subseteq \mathcal{I}$ based on the predicted preferences of user u. However, this

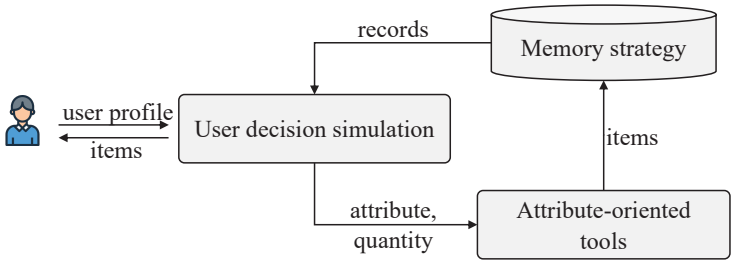


Figure 2.2: An overview of the proposed LLM-based recommendation method via tool learning.

easily results in the desired item i^n being absent from $\mathcal{I}_c$.

In this work, we position LLMs as the central controller for recommendation, simulating the user exploration w.r.t. item attributes in a multi-round manner. The interaction history $\mathcal{H}$ of user u is fed into the LLM, so as to initialize the profile of surrogate user $\hat{u}$. In the first round, $\hat{u}$ identifies a key attribute a_1 and uses external tools to fetch the related item set $\mathcal{I}_{\hat{u}}^{a_1}$. For clarity and brevity, we omit the subscript $\hat{u}$ when the user is clear from the context. In the second round, $\hat{u}$ contrasts the preferences between $\mathcal{I}^{a_1}$ and $\mathcal{H}$, selects another attribute a_2, and retrieves $\mathcal{I}^{a_2}$. Such iterative refinements continue until $\hat{u}$ is satisfied with the retrieved items, and outputs the final set $\mathcal{I}_{\hat{u}}$ ranked by preference. Each derived item set reflects the interest emerging from the respective round.

2.3.2 User Decision Simulation

To validate LLMs' capability in simulating user preferences and using external tools, we draw inspiration from prior tool-learning studies [88, 89, 99, 146] and propose a user decision simulation process. Here is an example of the simulation prompt in the movie recommendation scenario:

User Decision Simulation Prompt Example

"Assuming you are an online movie recommender, your task is to help users find movies they would like to watch next based on their interests. To effectively recommend movies, you should follow three steps: Thought, Action, Observation.
During the Thought step, your objective is to consider how to make the final movie list match the user's preferences, and decide the best course of action $< \mathcal{D} >$. If the movie list is good enough or no better action to take, you will finish early with the candidate movie list. For the Action step, you have three options: Retrieval, Rank, and Finish. $< \mathcal{T} >$"

where $<\mathcal{D}>$ represents the collection of demonstrations that show LLMs what a good movie list looks like, and $<\mathcal{T}>$ denotes the detailed description of the tools. The foundational elements of the user decision simulation methodology are twofold:

Chain-of-thought Prompting (CoT). We use CoT [125] to synergize reasoning and action. Here, "reasoning" refers to the "thinking procedure" for how to recom-

mend suitable items to users, and decide the details of the subsequent action. Meanwhile, "action" entails executing the directives derived from reasoning, making "observations," and yielding the corresponding outcome. At each step t, the LLM-driven surrogate user $\hat{u}$ receives an observation o^{t-1} from the last step and derives a thought g_t to take an action $\mathcal{A}_t$ following some policy $\pi(g_t, \mathcal{A}_t|c_t)$, where π could be implemented by any LLMs, specifically ChatGPT, in our ToolRec, and $c_t = (o_0, g_1, \mathcal{A}_1, o_1, \ldots, g_{t-1}, \mathcal{A}_{t-1}, o_{t-1})$ is the *context* to the $\hat{u}$. The primary objective is to deduce the policy and mapping $c_t \rightarrow (g_t, \mathcal{A}_t)$. As shown in the left of Figure 2.1, within the context c_t, $\hat{u}$ observes a discrepancy in the movie actors between the retrieved movies and the user's previous movies. Consequently, it decides to rank the movie list based on their actors (*i.e.,* $\mathcal{A}_t$) and obtain o_t. If $\hat{u}$ is satisfied within the context (*i.e.,* c_N), by comparing the candidate items to the user's history, then $\hat{u}$ will conclude the process and present the final set of candidate items, denoted as $\mathcal{I}_{\hat{u}}$.

Tool Learning. Tool learning technology refers to combine the strength of LLMs and specialized tools, as discussed in previous studies [88]. Here, our tools are activated by the generated action $\mathcal{A}_t$ [89, 99, 146]. For each context c_t, the initial observation $o_0 = (\mathcal{H}, \mathcal{T}, \mathcal{D})$ is composed of the user u's historical interactions $\mathcal{H}$, tool description $\mathcal{T}$, and demonstration $\mathcal{D}$. The tools can be categorized into two types based on $\mathcal{T}_{type}$, which can be either "retrieve tools" or "rank tools." The description $\mathcal{T}$ elucidates the impact of using the tools, and $\mathcal{D}$ offers practical demonstrations of their application. Together, these components promote the effective use of external tools. Tool actions are formulated as $\mathcal{T}_{type}[a_t, \$K]$, where $\mathcal{T}_{type} \in \{Retrieval, Rank\}$, a_t indicates the chosen attribute based on $\hat{u}$'s decision at time t, and $\$K$ specifies the number of items to be returned. For instance, as shown in Figure 2.1, at the first step, $\hat{u}$ opts to use the retrieval tools conditioned on the attribute "genre" and retrieves 5 items. Meanwhile, at step 3, $\hat{u}$ decides to employ rank tools conditioned on the attribute "actor", returning the top 4 items. This phase is constructed to emulate a human-like approach of leveraging tools to broaden their choices through linguistic reasoning.

The CoT prompting phase controls the iterative decision process, determining when to employ external tools or finalize the recommendations. Concurrently, feedback from these tools enhances item exploration and further refines the recommendation process.

2.3.3 Attribute-oriented Tools

By empowering an LLM to use tools, we can explore different parts of the item pool, uncovering target items that remain latent. To achieve this, we have designed two types of tools.

Rank Tools. For attribute-oriented rank tools, we incorporate a ranking instruction template and employ LLMs to order the candidate items. Given that LLMs have demonstrated proficiency in both zero-shot and few-shot scenarios [39], their capabilities are essential for returning item sets that align more closely with the user's latent intent. Our instruction for ranking is framed as: "{*User Historical Record*} {*Prior Retrieved Item Set*}. *Please rank the above recommended movies by measuring the possibilities*

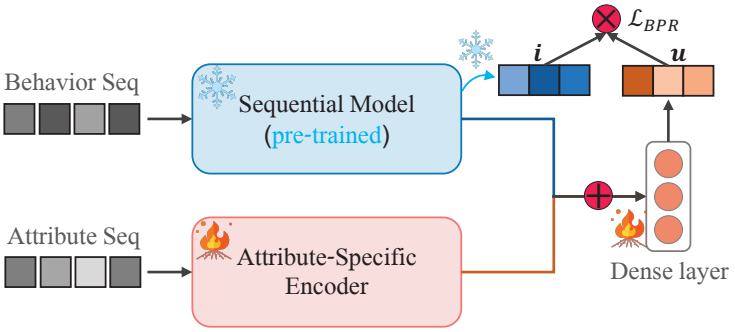


Figure 2.3: Fine-tuning stage of attribute-oriented retrieval tools. The parameters in the blue 'ice' section remain fixed, while those in the red 'flame' section are exclusively fine-tuned.

that the user would like to watch next most according to the movie [*Attribute Pattern* a_t] *attribute and the given movie history records, and output top* [*Output Size Pattern* $\$K$] *movies except user's historical movies*."

Retrieval Tools. For each user, our attribute-oriented retrieval tools accept an attribute pattern a_t and a specified item set size $\$K$, subsequently returning the matching candidate item set. An intuitive way is creating dedicated models tailored to each attribute pattern. However, it is markedly inefficient to fully train and store separate models for every possible attribute permutation. To address this, we introduce a two-stage method for managing attribute-specific variations:

Pre-training. The pre-training stage is for the original model with no attribute-specific consideration. Without loss of generality, we incorporate the attribute-specific switches into the popular sequential recommendation model, SASRec [52]. For a historical behavior sequence $\mathcal{H}$, the l-th layer behavior representation matrix is denoted as $\mathbf{H}^l = \{\mathbf{h}_1^l, \mathbf{h}_2^l, \ldots, \mathbf{h}_{|\mathcal{H}|}^l\}$, where $\mathbf{h}_i^l$ is the l-th layer's representation of the i-th behavior within the sequence $\mathcal{H}$. Then in the final layer L, the pre-training behavior representation $\mathbf{H}$ is derived:

$$\mathbf{H} = f_{seq}(\mathcal{H}|\beta) = \mathbf{h}_{|\mathcal{H}|}^L,\ \mathbf{H}^{l+1} = \text{Transformer}_h^l(\mathbf{H}^l), \tag{2.1}$$

where f_{seq} represents the sequential model, β is the pre-training parameter, Transformer denotes the Transformer architecture encoder, and $\mathbf{H}$ is considered as the user representation at the pre-training phase for sequential recommendation.

Attribute-specific Encoder in Tuning. As illustrated in Figure 2.3, after learning the pretrained model, we freeze the pre-training parameter β. Our goal is to fine-tune the attribute-specific encoder. To achieve this, we construct an additional attribute encoder f_{attr}. This encoder takes the user's historical item attribute sequence, a_u, as input. Importantly, $\mathcal{H}$ and a_u are aligned based on items in the historical sequence. Subsequently, we define the l-th layer attribute representation matrix as $\boldsymbol{a}_u^l = \{\boldsymbol{a}_1^l, \boldsymbol{a}_2^l, \cdots, \boldsymbol{a}_{|a_u|}^l\}$. Similar to the pre-training approach, the attribute

sequence representation $\boldsymbol{a}_u$ is learned as:

$$\boldsymbol{a}_u = f_{attr}(a_u|\gamma) = \boldsymbol{a}^L_{|a_u|},\ \boldsymbol{a}^{l+1}_u = \text{Transformer}^l_a(\boldsymbol{a}^l_u). \tag{2.2}$$

We then incorporate a dense layer to encode the combination of attribute representation $\boldsymbol{a}_u$ and the frozen behavior representation $\mathbf{H}$, resulting in a new user representation:

$$\mathbf{u} = \text{Dense}(\boldsymbol{a}_u \oplus \mathbf{H}, \theta), \tag{2.3}$$

where γ and θ are the trainable parameters in the tuning phase.

The sequential recommender is trained by minimizing the BPR loss function:

$$\mathcal{L}_{BPR} = - \sum_{(\mathcal{H},\, v)\in\mathcal{O}^+,(\mathcal{H},w)\in\mathcal{O}^-,} \log\sigma(\phi(\mathbf{u}, v) - \phi(\mathbf{u}, w)), \tag{2.4}$$

where $\mathcal{O}^+$ and $\mathcal{O}^-$ denote the positive samples and negative samples, $\phi(\cdot)$ represents the inner-product layer, and $\sigma(\cdot)$ refers to the Sigmoid activation function. During the pre-training phase, $\mathbf{u}$ is represented by $\mathbf{H}$. The finetuned user embedding $\mathbf{u}$ is designed to be sensitive to the specific attribute, while maintaining the personalized sequential recommendation learned in the pre-training phase.

2.3.4 Memory Strategy

The vast number of items and the complex item names&IDs pose a challenge for LLMs when generating control commands or tool usages. Additionally, items retrieved from various tools should be systematically ordered to aid the LLM-based surrogate user $\hat{u}$ in making decisions. Therefore, we introduce a memory strategy, ensuring the correctness of generated items and cataloging candidate items with their respective tool annotations.

The memory strategy is initialized with the item pool directory. Whenever external tools return candidate items, particularly from attribute-oriented rank tools, the strategy verifies the presence of these items in the initial directory. If there are any discrepancies, the tools are prompted to re-run with additional incorrect details attached behind. Once validated, the candidate items are recorded alongside their associated tool marks, in order to serve the subsequent tool calls. As an illustration, a typical prompt might be "*Here's the top* [*Output Size Pattern* $\$K$] *movie ID, movie name, and the recommendation confidence score from the recommender system with* [*Attribute Pattern* $a_{\hat{u}}$] *type*. {*Candidate Item Set*}."

2.4 Experiments

In this section, we report on extensive experiments aimed at evaluating the performance of our proposed ToolRec. Our experiments focus on answering the following research questions:

RQ1.1 How does ToolRec compare to conventional RSs and LLM-based RSs in the sequential recommendation setting?

Table 2.1: The statistics of the datasets used.

Datasets	#Users	#Items	#Interactions	Sparsity (%)
ML-1M	6,041	3,884	1,000,209	95.74
Amazon-Book	158,349	97,566	3,876,663	99.97
Yelp2018	77,278	45,582	2,102,836	99.94

RQ1.2 How do different components (i.e., *user decision simulation, termination round, attribute-oriented retrieval tools*) influence our ToolRec?

RQ1.3 Are LLMs capable of using their inherent knowledge to cater to the recommendation task's needs?

2.4.1 Experimental Settings

Datasets

To evaluate the effectiveness of our methods, we conduct experiments on three real-world datasets: Ml-1M, Amazon-Book, and Yelp2018.

- **ML-1M**. This dataset is derived from the MovieLens-1M[1] benchmark, which contains user ratings for movies with timestamps. We take the movies' *genre* and *release year* as the attribute information.
- **Amazon-Book**. This dataset[2] is extracted from *Amazon.com* platform. We adopt the Book category to evaluate our method. We take the books' *Price* and *Sales rank* as the attribute information. A 10-core setting is applied to maintain dataset quality.
- **Yelp2018**. This dataset is collected from the 2018 edition of the Yelp Challenge.[3] We use local businesses' *Categories*, *City* and *Stars* as attributes. We employ the 10-core setting to ensure a minimum of ten interactions for each user and item.

For each dataset, we organize users' interactions chronologically based on timestamps, allowing us to create the corresponding historical interaction sequences. Items are described using their product IDs&Names. We summarize the statistics of our datasets in Table 2.1.

Evaluation Protocols

We apply the leave-one-out strategy [52, 105] and employ timestamps to set the sequence order, dividing the interaction data into training, validation, and test sets. The attribute-oriented retrieval tools are trained on the training and validation sets. To measure the recommendation performance, we adopt two widely used metrics *NDCG@N* and *Recall@N* to evaluate the results within the top-*N* positions, with $N = 10$ in our

[1] https://grouplens.org/datasets/movielens/
[2] https://nijianmo.github.io/amazon/
[3] https://www.yelp.com/dataset

experiments. Due to budget constraints, following prior work [31, 39], we randomly sample 200 users and their historical behaviors from the test set for each dataset. A similar, and similarly-sized, setting has been adopted in other recent LLM-related recommendation researches [149, 151]. To enhance the robustness and credibility of our results, we repeated the experiments three times, each with a different sample of 200 users. The average results and standard deviations from these trials are presented in Table 2.2.

Baselines

We compare ToolRec[4] against two traditional approaches (the first two below), one that uses LLMs as RSs (the following one), two that enhance RSs with LLMs (the next two), and two that use LLMs to control RSs (the remaining two).

- **SASRec** [52]. A self-attention-based sequential recommender, which employs the encoder of the Transformer architecture to generate representations of users' behavior sequences.
- **BERT4Rec** [105]. A bidirectional self-attention-based sequential recommender. It uses the Transformer encoder to predict randomly masked items in a sequence by conditioning on both their left and right context, thereby capturing user historical behaviors.
- **P5** [33]. An encoder–decoder Transformer-based approach that unifies different recommendation related tasks into a single generative LLM. For our sequential recommendation downstream task, we adopt the personalized prompts from OpenP5 [140] and apply the same random indexing method to items as used in our ToolRec. This model is fully fine-tuned using these personalized prompts on the pre-trained T5-small [92].
- **SASRec**$_{BERT}$ [23]. An attention-based method that modifies the single interaction sequence encoder by adding a feature encoder (structured as shown in Figure 2.3). This model is fully fine-tuned using semantic representations pre-trained with BERT.
- **BERT4Rec**$_{BERT}$. A variation of the BERT4Rec sequential recommender enhanced with BERT's pre-trained representations.
- **Chat-REC** [31]. The first work was on using an LLM as a controller. For our sequential recommendation task, we adopt the recommendation prompt based on the original paper, choose SASRec to supply the candidate items, and modify the output format for parsing.
- **LLMRank** [39]. A LLM-based ranking model. In our full-ranking experimental setup, we retrieve thirty candidate items (according to the original paper) from SASRec and adjust the output format for parsing.

[4]Our code is available at `https://github.com/Go0day/ToolRec-Code`.

- **ToolRec**. This is the method that we propose. We primarily evaluate two versions of ToolRec: ToolRec that is implemented using external attribute-oriented retrieval tools with frozen $SASRec$ parameters; and ToolRec$_B$ that is developed using BERT4Rec as the backbone for external attribute-oriented retrieval tools.

Implementation Details

We use the gpt-3.5-turbo-16k (ChatGPT for short) model as our primary LLM within ToolRec. This model is responsible for parsing user preferences and assisting in tool learning. We retain the default hyperparameters of ChatGPT without modifications. To enable ToolRec to emulate user decisions, we incorporate decision demonstrations into the prompt for in-context-learning. For attribute-oriented tools, retrieval tools are enhanced with corresponding additional item attributes specific to each dataset. Attributes are represented using word embeddings from GloVe [86]. These tools are developed on two widely recognized backbones: SASRec and BERT4Rec. Additionally, our rank tools are constructed with instructions on ChatGPT. We limit our decision processes to a maximum of eight rounds.

2.4.2 Performance Comparison (RQ1.1)

We compare our proposal with conventional sequential recommenders and LLM-enhanced sequential recommenders, as detailed in Section 2.4.1. The results are reported in Table 2.2, from which we observe:

- In general, ToolRec outperforms all baselines on the ML-1M and Amazon-Book datasets. The performance improvement can be attributed to the efficacy of our designed ToolRec framework. By delving into various facets of the item pool and using LLMs to guide the exploration process, it can more effectively align with the user's intent.

- ToolRec and ToolRec$_B$ consistently demonstrate improved performance compared to their respective underlying models (*i.e.,* SASRec and BERT4Rec). This confirms the superiority of the ToolRec framework and underscores its adaptability. The results suggest that our approach, which harnesses LLMs for recommendation through tool learning, has the potential to be integrated as a supplementary module in various recommender systems.

- ToolRec exhibits subpar performance on the Yelp2018 dataset. We attribute this to the LLM's limited knowledge of local businesses. Since LLMs are primarily trained on widely available web data, they might possess a more robust understanding of topics like movies and books than local (niche) businesses. In more specialized domains, ToolRec, which involves multiple interactions between the LLM and external tools, has the potential to make more incorrect decisions and may exhibit heightened deficiencies when compared to other LLM-based approaches such as LLMRank and SASRec$_{\text{BERT}}$.

- In the approach where LLMs serve as RSs, P5 demonstrates strong performance on the ML-1M. This is exciting, considering we use datasets on the scale of

Table 2.2: The test performance comparison on three real-world datasets. The bold font denotes the winner in that column. The row "Improvement" indicates the relative performance gain of our ToolRec and the suboptimal method.

	ML-1M		Amazon-Book		Yelp2018	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
SASRec	0.203±0.047	<u>0.1017</u>±0.016	0.047±0.015	0.0205±0.006	0.030±0.005	0.0165±0.006
BERT4Rec	0.158±0.024	0.0729±0.008	0.042±0.015	0.0212±0.009	<u>0.033</u>±0.021	**0.0218**±0.016
P5	<u>0.208</u>±0.021	0.0962±0.009	0.006±0.003	0.0026±0.002	0.012±0.005	0.005±0.001
SASRec$_{BERT}$	0.192±0.015	0.0967±0.006	0.042±0.003	0.0194±0.002	0.032±0.016	0.0131±0.007
BERT4Rec$_{BERT}$	0.202±0.013	0.0961±0.009	0.045±0.023	0.0233±0.012	**0.040**±0.028	<u>0.0208</u>±0.015
Chat-REC	0.185±0.044	0.1012±0.016	0.033±0.015	0.0171±0.007	0.022±0.003	0.0121±0.001
LLMRank	0.183±0.049	0.0991±0.020	<u>0.047</u>±0.013	<u>0.0246</u>±0.004	0.030±0.005	0.0140±0.004
ToolRec	**0.215**±0.044	**0.1171**±0.018	**0.053**±0.013	**0.0259**±0.005	0.028±0.003	0.0159±0.001
ToolRec$_B$	0.185±0.018	0.0895±0.002	0.043±0.013	0.0223±0.008	0.025±0.005	0.0136±0.009
Improvement	3.36%	15.10%	14.28%	5.14%	–29.16%	–27.32%

millions (Table 2.1), and accurately generating items for users is challenging. However, P5 shows weaker performance on the Amazon-Book and Yelp2018 datasets, potentially due to the fact that the sparsity of those datasets is higher than ML-1M, thus leading to more severe hallucination issues. The development of more carefully designed item indexing methods [42] may help alleviate this problem, but this falls beyond the scope of our current research.

- For LLM-enhanced RS approaches, such as SASRec$_{BERT}$ and BERT4Rec$_{BERT}$, it is not surprising to see improvements over their backbone models in most cases. However, on the ML-1M dataset, SASRec$_{BERT}$ performs much worse than SASRec. Meanwhile, on the Amazon-Book dataset, BERT4Rec$_{BERT}$'s performance is comparable to its baseline version. While language models can generally augment recommendation tasks, addressing the disparity between semantic and behavioral spaces remains challenging.

- For LLM-controlled RS approaches, such as Chat-REC and LLMRank, they couldn't achieve consistent improvements over SASRec in the same way as ToolRec. This suggests that merely using basic control strategies or employing LLM as a ranker might not be the most effective way forward for recommendation tasks. A more comprehensive strategy, as demonstrated by ToolRec, appears better suited to harnessing LLMs to enhance recommendations.

2.4.3 Decomposing ToolRec (RQ1.2)

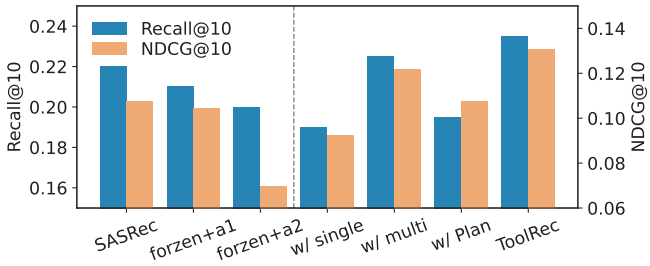
Next, we delve deeper into ToolRec. We examine the user decision simulation, revealing how multi-round interactions enhance recommendation quality. We evaluate the efficiency of our attribute-oriented retrieval tools, highlighting the balance between rich information and computational practicality.

Effectiveness of User Decision Simulation

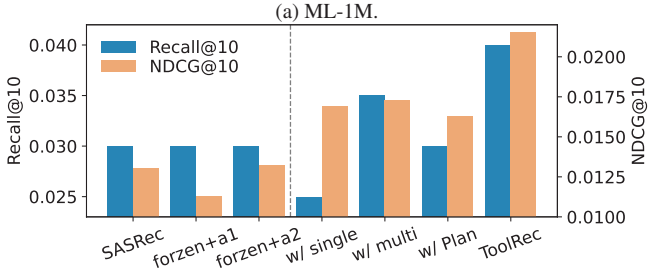
To verify the contribution of the user decision simulation component (*cf.* Section 2.3.2), we conducted an ablation study considering three variants of ToolRec: (i) Disabling the CoT and tool learning components of ToolRec, we forced the LLM to rank the candidate items from SASRec and output the result. This variant is denoted as "w/ single"; (ii) Unlike "w/ single", we had the LLM rank candidate items using both SASRec and the attribute-oriented retrieval tools, termed "w/ multi"; (iii) We disable the CoT component, and instead, the LLM was instructed to generate all the steps of tool-calling at once and then strictly follow the execution plan. This setup is termed "w/ Plan". To ensure fairness in our comparisons, both the "w/ single" and "w/ multi" variants use the same number of candidate items: thirty items in total.

We demonstrate the experimental results in Figure 2.4 and have the following findings:

- Removing the CoT and tool learning component degrades the model's performance. The "w/ single" variant consistently underperforms both "w/ multi" and "w/ Plan", and its performance is even subpar compared to SASRec. This decline



(a) ML-1M.



(b) Amazon-Book.

Figure 2.4: Performance of ToolRec and its variants. The right side of the dividing line indicating the methods involving LLMs.

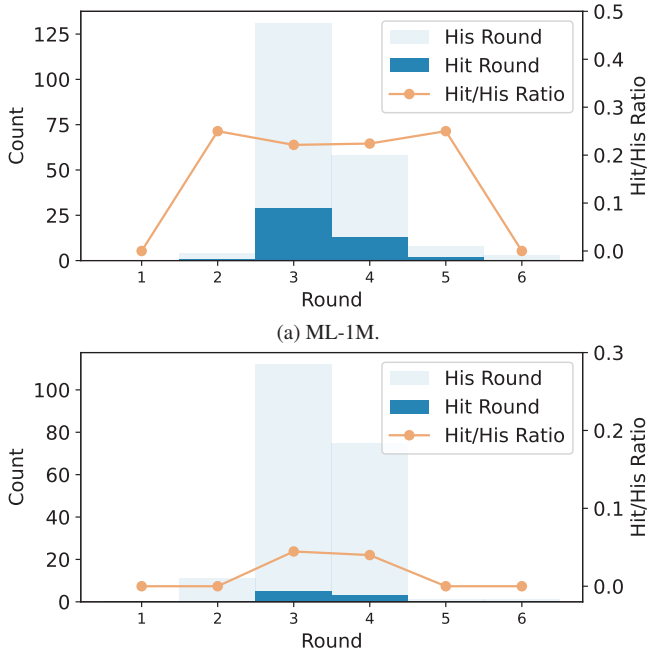
in performance can be attributed to "w/ single" relying exclusively on the LLM's zero-shot ranking without using additional information to refine the results.

- It is important to note that the performance of attribute-oriented retrieval tools, namely "frozen + $a1$" and "frozen + $a2$", is inferior to SASRec in dataset ML-1M. However, results refined by 'w/ multi' not only surpass 'w/ single' but also outperform SASRec. This improvement suggests that the strength of our approach is not solely due to the broader item size, but also derives from the assistance of the additional attribute information.
- When comparing "w/ Plan" with "w/ multi", the latter consistently achieves superior results. One potential reason might be that the "generate-then-execute" approach in "w/ Plan" lacks explicit performance guidance, causing it to miss the target. This further underscores the importance of the user decision simulation in our approach.

Analysis of Round Termination in ToolRec

Figure 2.5 illustrates the distribution of the number of termination rounds, N, in ToolRec.

Based on the data, we make the following observations: (i) The majority of processes conclude within three or four rounds. This suggests that after a few iterations, our LLM-based surrogate user, denoted as $\hat{u}$, develops a good understanding of whether the user's preferences have been adequately addressed. (ii) While the majority of successful hits also occur within three or four rounds, an interesting trend emerges in the ML-1M



(a) ML-1M.

(b) Amazon-Book.

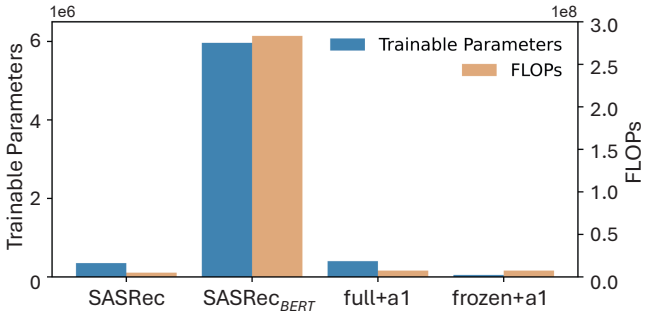
Figure 2.5: Distribution of termination rounds for ToolRec. "His Round" indicates the distribution of termination rounds for all users, while "Hit Round" highlights the termination round where the recommended list accurately contains the user's target item.

dataset: both shorter and longer processes tend to be more successful in reaching the target item. One interpretation is that shorter rounds signify tasks that are more straightforward for the surrogate user $\hat{u}$. But for users with diverse interests or nuanced tastes, $\hat{u}$ might need additional rounds to gather more information and determine if the recommendation process has been satisfactorily completed.

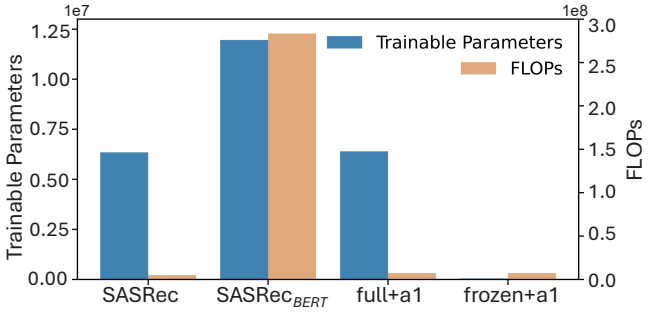
Efficiency and Scalability of Attribute-oriented Retrieval Tools

As elaborated in Section 2.3.3, the attribute-oriented retrieval tools are designed to adeptly follow diverse attribute choices. Figure 2.6 compares the number of trainable parameters and FLOPs on the ML-1M and Amazon-Book datasets. Here, 'forzen+$a1$' represents our attribute-oriented retrieval tool with frozen backbone parameters, while 'full+$a1$' denotes its full fine-tuning variant.

We have the following observations: (i) The SASRec$_{BERT}$ model is considerably larger than the original SASRec. This increase in size can be attributed to the enriched semantic information present in the BERT embedding. (ii) The trainable parameter count for 'full+$a1$' aligns closely with that of SASRec. Essentially, this is akin to training an entirely new model. Such an approach becomes impractical as the number of attributes escalates. (iii) While both 'forzen+$a1$' and 'full+$a1$' exhibit identical FLOPs,



(a) ML-1M.



(b) Amazon-Book.

Figure 2.6: Comparison of trainable parameters and FLOPs for various retrieval model configurations.

the former boasts a significantly reduced count of trainable parameters compared to the latter. Moreover, since our 'forzen+a1' employs the shared backbone of SASRec, storage is primarily reserved for the additive parameters from the attribute-specific encoder and the dense layer.

2.4.4 Surprises and Limitations (RQ1.3)

ToolRec benefits from LLMs to reason about users' preferences across attributes, enabling multi-round interaction processes and improved recommendations. However, language models are not inherently designed for recommendation tasks. Below, we discuss the failures and limitations this gives rise to.

Unanticipated Outcomes: Beyond Conventional Failures

For a rigorous comparison with existing baselines, we employed the Memory strategy (*cf.* Section 2.3.4). This strategy ensures that returned items align with the dataset directory, prompting a re-run if discrepancies arise. However, beyond the traditional recommendation paradigm, some samples that are typically classified as "failures" deserve a second look.

Figure 2.7 shows an example. At *Thought 5*, when our LLM-based surrogate user,

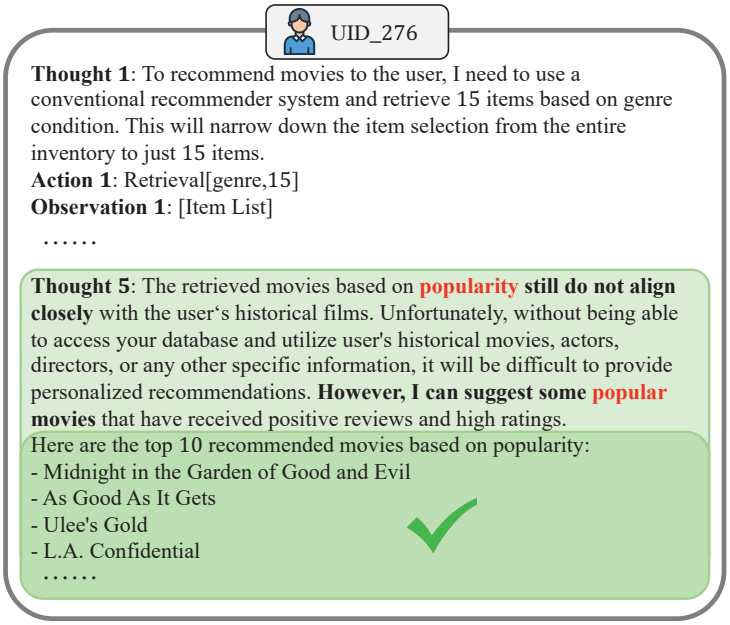


Figure 2.7: Case study of parsing errors in output samples. Highlighted samples have been manually verified to exist in the real world.

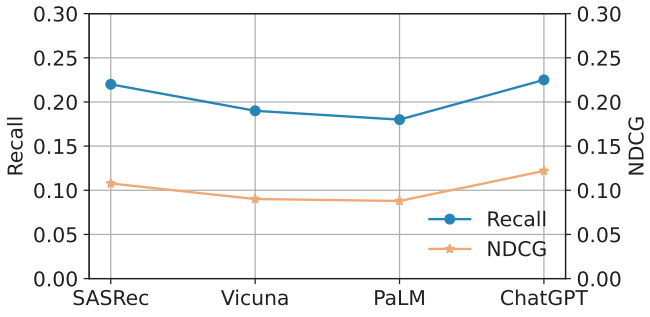
$\hat{u}$, is unsatisfied with the retrieved movies, it typically evaluates the unsatisfactory attributes and uses external tools to uncover additional options, thereby refining the recommendation.

Yet, in this scenario, rather than using external tools or settling for the current list of candidates, $\hat{u}$ decides to suggest "some popular movies", providing a top-N recommendation on its own. Since we never trained the LLM on any of our recommendation datasets, it is evident that not all of the movies it suggests are found in the ML-1M dataset. As a result, this action is labeled as a "failure", prompting a re-run. However, from another perspective, this could be viewed as a successful recommendation; the only limitation is our inability to evaluate it within the current dataset constraints.

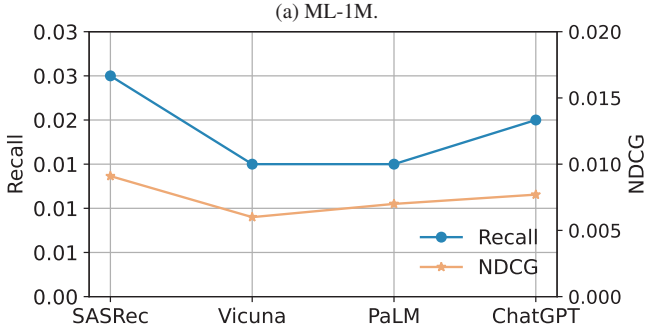
Influence of LLM Selection on Recommendation Performance

Following the experimental setup described in Section 2.4.1, we replace the base LLM with two alternative LLMs: Vicuna1.5-13B-16k [20] (Vicuna in short), an open-source chatbot fine-tuned on Llama2 [113]; and PaLM [21], a commercial LLM developed by Google AI. However, both variants yielded subpar outcomes, either failing to adhere to task instructions, misconstruing tool usage, or misinterpreting user preferences (some representative failures are cataloged in the Appendix). Notably, even though Vicuna and PaLM use the same prompt template as ChatGPT, they are unable to consistently generate recommendations across all test users, despite several attempts. This suggests that ChatGPT has better reasoning capabilities than both Vicuna and PaLM.

Beyond reasoning, a ranking experiment was conducted (*cf.* Section 2.4.3, w/ multi)



(a) ML-1M.



(b) Yelp2018.

Figure 2.8: Ranking performance across different LLMs.

to examine the open-world knowledge of various LLMs. As revealed in Figure 2.8, on datasets like ML-1M (and similar outcomes on Amazon-Book), both Vicuna and PaLM underperformed SASRec, whereas ChatGPT exhibited superior results. These findings align with insights presented in LLMRank [39]. However, all three models lagged behind SASRec on the Yelp2018 dataset, suggesting a possible limitation of LLMs in contexts like local businesses, where open-world knowledge is limited and of more limited use than on the other datasets.

2.5 Conclusion

In this work, we zoomed in on the tool-learning capacities of LLMs, using them as controllers to guide the exploration of item spaces in a recommendation scenario. Specifically, by treating LLMs as surrogate users, they can adeptly capture the nuances of a current context alongside user preferences. Subsequently, we employ attribute-oriented tools for precise item retrieval. We developed two types of attribute-oriented tools: rank tools and retrieval tools, each fetching the corresponding candidate items. To enhance accurate item retrieval, items that appear in the process are verified and stored using the memory strategy. Extensive experiments on real-world datasets rich in knowledge demonstrate the effectiveness and rationality of ToolRec.

The idea of using LLMs for simulation, either under the hood as in our approach or for counterfactual explorations while interacting with users, holds great potential

for combining the strengths of LLMs and recommendation models. In the short term, companies operating their own recommender systems may find it impractical to switch to LLM-based RSs. However, ToolRec could enhance recommendation performance by integrating LLMs with their current systems. In the long-term, users are increasingly relying on LLMs for various daily tasks, including recommendations. ToolRec does not require extensive fine-tuning of the LLM, which can lead to additional costs and potential delays due to outdated information. Furthermore, the results from ToolRec are more reliable than those from zero-shot or few-shot LLM recommendations, as they are augmented by traditional recommender system outputs.

Achieving strong recommendation performance hinges on a dataset with rich semantic knowledge and the robust capabilities of the LLM. In future work, we plan to incorporate recommendation knowledge into LLMs to enhance domain-specific tool learning, such as retrieval-augmented generation or fine-tuning LLMs, potentially reducing the reliance on rich semantic knowledge. Additionally, we intend to explore different types of tools, including search engines and databases, along with a self-reflection strategy, to achieve even more personalized recommendations.

3

Revisiting Language Models in Neural News Recommender Systems

In the previous chapter, we enhanced recommendation performance in general scenarios by using LLMs with tool learning to mitigate hallucinations. In this chapter, we shift focus to semantic-rich news recommendation, investigating how fine-tuning language models can better capture user preferences from textual content. The proposed approach, involving fine-tuning strategies across various language models like GloVe, BERT, RoBERTa, and Llama on news recommendation tasks, provides an answer to the following research question posed in Chapter 1: **RQ2**: How can fine-tuning language models improve the capture of user preferences and enhance recommendations in semantic-rich news recommendation scenarios while avoiding under-representation?

3.1 Introduction

News recommender systems (RSs) help deliver relevant news articles to users. Unlike RSs in other domains, such as e-commerce and music, that primarily focus on modeling interactions between users and items, news RSs rely heavily on modeling text-based news articles with rich textual information [63]. Therefore, natural language processing techniques, particularly methods based on language models (LMs), are widely used to generate news representations in news RSs.

Among the early LM-based approaches to news representation are shallow language models (SLMs) such as GloVe [86], a model that generates word representations based on corpus co-occurrence statistics. In news RSs, GloVe embeddings are used to initialize word embeddings, which are later employed to model news articles and interactions [3, 126, 127]. Following progress in language modeling, pre-trained language models (PLMs) such as BERT [23] and RoBERTa [72] have also been integrated into news RSs to generate embeddings for news articles. Compared to SLMs, PLMs are typically larger, featuring complex architectures with more layers, thus contributing to a greater number of parameters. Large language models (LLMs) such as Llama [112] are also

This chapter was published as Y. Zhao, J. Huang, D. Vos, and M. de Rijke. Revisiting language models in neural news recommender systems. In *Advances in Information Retrieval - 47th European Conference on Information Retrieval, ECIR 2025, Lucca, Italy, April 6-10, 2025, Proceedings, Part IV*, volume 15575 of *Lecture Notes in Computer Science*, pages 161–176. Springer, 2025.

used to enhance news modeling in RSs because of their ability to capture context and to generalize [67, 71].

Most prior work in news RSs shows that news RSs using larger PLMs outperform those using SLMs [49, 61, 129–131, 139, 148, 154]. Work using LLMs has shown better recommendation performance than PLMs [67, 71]. But the findings are not consistent: some work reports that PLMs sometimes perform worse than SLMs in news RSs [46, 47]. Are larger models worth the additional computational resources? We examine (i) whether larger LMs truly improve news recommendation performance, and (ii) what size LMs (as news encoders) provides a reasonable trade-off between performance and resource consumption.

To answer these questions, we compare the impact of using eight LMs – across different LM families, i.e., GloVe (SLM), BERT and RoBERTa (PLMs), and Llama3.1-8B (LLM), as well as multiple sizes within the BERT family (tiny, mini, small, medium, and base) – on the performance of three well-known news recommendation models: NAML [126], NRMS [127], and LSTUR [3]. Consistent with widely adopted practices [12, 67, 71, 159], our experiments are based on the small version of the real-world MIND dataset [133].

We focus on the following research questions:

RQ2.1 Does using a larger LM in news RSs consistently lead to better recommendation accuracy?

RQ2.2 How does fine-tuning LMs affect the performance of LM-based news RSs?

LMs may enhance the performance of news RSs for cold-start users with limited or no user interaction history by analyzing the textual content of news articles and recommending relevant content based on extracted semantic information [97, 117]. Therefore, our third research question concerns the recommendation performance of different LMs for cold-start users:

RQ2.3 Do news RSs based on larger LMs provide better performance for cold-start users?

Larger LMs in news RSs do not always lead to improved performance of recommendations. The performance of LM-based news RSs depends heavily on whether the LMs are fine-tuned. For example, without fine-tuning, NRMS using the SLM GloVe outperforms NRMS using PLMs BERT and RoBERTa, and even performs comparable to NRMS using LLM Llama. Moreover, while larger LMs require more comprehensive fine-tuning, such as searching for the optimal number of fine-tuned layers, they tend to achieve better performance for cold-start users. LM-enhanced news encoders alleviate the dependency on user interaction history, making recommendations more reliant on news content itself.

3.2 Related Work

Selection Criteria for Related Work. We follow the guidelines in [55] to select relevant literature on LMs as news encoders for news recommendation. Sources are

chosen from top venues and journals in the fields of artificial intelligence (AI) and information retrieval (IR). Papers are included if they (i) propose a definition of text modeling in the context of news recommendation, (ii) introduce approaches to improve news recommendation performance, or (iii) present experimental results comparing the performance of different-sized LMs as news encoders using the same benchmark. Papers are excluded if (i) their approaches are not tested on an English news recommendation dataset or (ii) they fall outside the date range of October 2014 (the release of GloVe) to October 2024. We identified over 200 studies, 24 of which are highly relevant to our work. Below, we introduce these studies to provide context for our research.

LMs in News RSs. News content modeling is a crucial component of news RSs, as news articles contain rich textual information that can be effectively encoded using LMs [132]. Following the categorization criteria in [70, 137], methods using LMs in news RSs can be grouped based on the role of the LM: (i) LMs as news recommenders, which generate candidate news items [65, 67], (ii) LMs as news encoders, which encode news content to support news RSs [46, 49, 61, 129–131, 139, 148, 154], and (iii) LMs as news enhancers, which generate additional textual features that assist news RSs [71, 141]. In this study, we focus on the largest group, where LMs are used as news encoders to explore the impact of different LMs in news RSs in relation to their effectiveness and efficiency.

LMs as News Encoders. Early LM-based approaches to news RSs learn representations on SLMs, such as GloVe. NAML [126] uses GloVe embeddings to initialize word representations and employs a word-level and view-level attention mechanism, along with convolutional neural networks, to capture important words for news representation. NRMS [127] uses GloVe embeddings for initialization and adopts multi-head self-attention to learn news representation. Prior work has argued that such shallow models may not be sufficient to capture the semantic information in news articles, and has explored PLMs based on the transformer architecture [115], such as BERT, for news modeling. For example, PLM-NR [129] uses PLMs to enhance news representation and observes improvements over SLMs as a news encoder model. MINER [61] employs a pre-trained BERT as the news encoder and uses a poly-attention mechanism to extract multiple aspect interest vectors for users. More recently, LLMs have been explored for news modeling. ONCE [71] uses both open- and closed-source LLMs to enrich training data and enhance content representation. Yada and Yamana [141] improve news recommendations by using LLMs to generate category descriptions. PGNR [67] employs LLMs to frame news recommendation as a text-to-text generation task, performing recommendation through generation.

According to the studies listed above, transitioning from SLMs to PLMs and then to LLMs results in clear improvements in news recommendation performance. However, some studies report different findings. NewsRecLib [46], a widely used news recommendation benchmark, reports that NAML and LSTUR, which originally used GloVe, performs worse when GloVe is replaced by the PLM BERT. Additionally, xMIND [47], a publicly available multilingual benchmark for news recommendation, indicates that NAML using PLM-based embeddings performs worse than the version using randomly-initialized embeddings. These contradictory results highlight the need for further investigation into the effectiveness of LMs in news recommendation, which

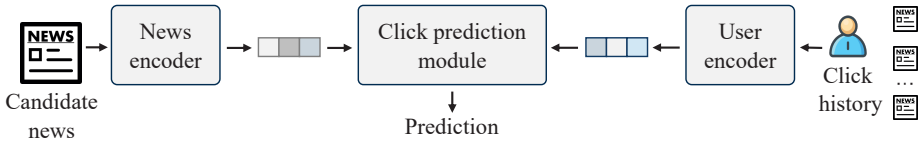


Figure 3.1: The typical structure of neural news recommendation methods.

motivates this study.

3.3 Reproducibility Methodology

3.3.1 Problem Formulation

Let $\mathcal{V}$ represent the set of news articles, where each news article $v \in \mathcal{V}$ consists of its textual feature $f_t(v)$ (*e.g.,* title or abstract) and other features $f_d(v)$ (*e.g.,* news categories or subcategories). Let $\mathcal{U}$ represent the set of users. Each user $u \in \mathcal{U}$ has a click history $H_u = \{v_1^h, v_2^h, \ldots, v_n^h\}$ in chronological order, denoting the sequence of n news articles previously clicked by the user. Given a candidate news article $v_c \in \mathcal{V}$, the goal of a news recommendation method is to predict the probability $\hat{y}_{u,v_c}$ that user u will click on v_c.

A typical (neural) news recommendation method has three components: a news encoder, a user encoder, and a click prediction module, as shown in Figure 3.1. The news encoder, primarily based on LMs, is responsible for encoding the textual features $f_t(v)$ and/or other features $f_d(v)$ associated with news article v, ultimately producing the news representation $\boldsymbol{q}_v$. We focus solely on the textual features $f_t(v)$ to compare the ability of different LMs in news modeling within news RSs. The user encoder generates the user preference representation $\boldsymbol{p}_u$ based on the user's click history H_u, summarizing the representations of news articles they have browsed. Using these representations, the click prediction module estimates the click probability $\hat{y}_{u,v_c}$ for candidate news article v_c.

3.3.2 News Recommendation Methods

Following [46, 49, 61, 129, 139, 148], we select NAML [126], NRMS [127], and LSTUR [3] as (neural) news recommender systems; all involve attention mechanisms for news recommendation. In terms of the news encoder, all three use attention for news modeling; NAML incorporates different types of news information, such as titles, bodies, categories, and subcategories, while NRMS focuses solely on learning news representations from titles. LSTUR models news representations based on titles and topic categories. NAML and NRMS employ attention mechanisms to learn user representations, whereas LSTUR uses a GRU network.

As depicted in Figure 3.1, the news encoder takes news features as input and then yields the news representation $\boldsymbol{q}_v$. The user encoder learns the user representation $\boldsymbol{p}_u$ based on the user's click history H_u, and the click score $\hat{y}$ is computed following the

(a) SLMs in news encoders.

(b) PLMs in news encoders.

Figure 3.2: SLMs and PLMs as building blocks of news encoders. Each LM can be used either in its non-fine-tuned form, shown in the left plots, or in its fine-tuned form, shown in the right plots. The parameters/embeddings in the blue "ice" section are fixed, while those in the red "flame" section are fine-tuned.

click prediction module $\mathcal{F}^{\text{RS}}(\cdot)$:

$$\hat{y}_{u,v} = \mathcal{F}^{\text{RS}}([\boldsymbol{p}_u, \boldsymbol{q}_v]). \tag{3.1}$$

For model training, the loss function minimizes the negative log-likelihood of all positive news articles in the ground truth:

$$\mathcal{L} = -\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{V}^+} \log \frac{\exp(\hat{y}_{u,i})}{\exp(\hat{y}_{u,i}) + \sum_{j \in \mathcal{V}^-} \exp(\hat{y}_{u,j})}, \tag{3.2}$$

where $\mathcal{V}^+$ is the set of positive new articles for user u in the training dataset, and $\mathcal{V}^-$ is the sampled negative news set corresponding to user u and the i-th positive news. This optimization encourages the model to differentiate between clicked and non-clicked news articles.

3.3.3 Language Models as News Encoders

To investigate the effect of different LMs as news encoders on the performance of news RSs, we compare three types of LMs based on model size: SLMs, PLMs, and LLMs. Each LM, when used as a news encoder, can either be used in its *non-fine-tuned* form, relying on its pre-trained knowledge, or it can be *fine-tuned* with additional training on news-specific data to improve the performance on the recommendation task.

SLMs as News Encoders. Given a news article $v = [v_1, v_2, \ldots, v_n]$, where v_i represents the i-th word in article v, SLMs generate static, non-contextualized word embeddings $\boldsymbol{e}_{v_i}$ by aggregating global word co-occurrence statistics, with each word having its embedding regardless of context: $\boldsymbol{e}_{v_i} = \mathcal{M}^{\text{SLM}}(v_i; \theta^{\text{SLM}})$. The news representation $\boldsymbol{q}_v$ is obtained by concatenating the word embeddings of the news content and then applying a fully-connected layer (FC): $\boldsymbol{q}_v = \text{FC}([\boldsymbol{e}_{v_1} \parallel \boldsymbol{e}_{v_2} \parallel \cdots \parallel \boldsymbol{e}_{v_n}]; \theta^{\text{FC}})$, where $\parallel$ denotes the concatenation operator.[1]

Non-fine-tuned mode. As illustrated in the left part of Figure 3.2a, in the non-fine-tuned mode, SLMs map each word in a news article to its corresponding embedding

[1]All FC layers in the paper share a similar architecture, differing primarily in the size of the first layer, which varies depending on the input size to enable the news RS to process varying input dimensions.

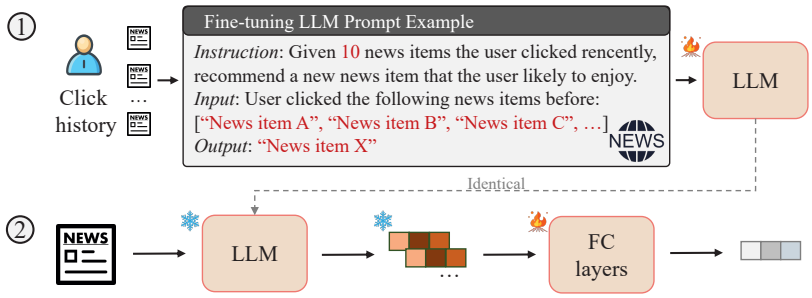


Figure 3.3: Fine-tuning LLMs as news encoders. In step 1, the LLMs are fine-tuned on news data presented in a natural language format. In step 2, the fine-tuned LLMs generate news embeddings, which are used for the recommendation task.

$\boldsymbol{e}_{v_i}$, which are then concatenated for further processing. The parameters of the fully connected layer θ^{FC} are tuned according to Eq. 3.2.

Fine-tuned mode. As illustrated in the right part of Figure 3.2a, in the fine-tuned mode, the word embeddings $\boldsymbol{e}_{v_i}$ and fully-connected layer parameters θ^{FC} are updated based on the recommendation signal (see Eq. 3.2).

PLMs as News Encoders. For a news article v, PLMs first tokenize the news text into tokens $\mathcal{T}^{\text{PLM}}(v) = [v^t_{[\text{CLS}]}, v^t_1, \ldots, v^t_m, v^t_{[\text{SEP}]}]$. The PLMs then generate contextualized token embeddings by passing the token sequence through transformer encoder layers: $[\boldsymbol{e}_{v^t_{[\text{CLS}]}}, \boldsymbol{e}_{v^t_1}, \ldots, \boldsymbol{e}_{v^t_m}, \boldsymbol{e}_{v^t_{[\text{SEP}]}}] = \mathcal{M}^{\text{PLM}}([v^t_{[\text{CLS}]}, v^t_1, \ldots, v^t_m, v^t_{[\text{SEP}]}]; \theta^{\text{PLM}})$. Following [61, 129, 139], we use the embedding of the `[CLS]` token, which appears at the start of every input sequence, and apply a fully connected layer to represent the entire news article: $\boldsymbol{q}_v = \text{FC}(\boldsymbol{e}_{v^t_{[\text{CLS}]}}; \theta^{\text{FC}})$.

Non-fine-tuned mode. As shown in the left part of Figure 3.2b, a PLM models the news content and outputs the news embedding. Similar to SLMs, the parameters of the fully-connected layer θ^{FC} are trained using the loss in Eq. 3.2.

Fine-tuned mode. As illustrated in the right part of Figure 3.2b, both the PLM parameters θ^{PLM} and the fully-connected layer parameters θ^{FC} are updated during the recommendation process.

LLMs as News Encoders. For LLMs, given a news article v, we follow the approach in [50], using fill-in-the-blank prompts, *i.e.,* "*This news: [v] means in one word:*" to create a prompted version of the news, denoted as v'. We tokenize it with $\mathcal{T}^{\text{LLM}}(v') = [v'^t_1, v'^t_2, \ldots, v'^t_L]$. Then the LLM generates token embeddings: $[\boldsymbol{e}_{v'^t_1}, \boldsymbol{e}_{v'^t_2}, \ldots, \boldsymbol{e}_{v'^t_L}] = \mathcal{M}^{\text{LLM}}([v'^t_1, v'^t_2, \ldots, v'^t_L]; \theta^{\text{LLM}})$. We use the embeddings of the last l tokens[2] and apply a fully-connected layer, representing the news article: $\boldsymbol{q}_v = \text{FC}([\boldsymbol{e}_{v'^t_{L-l}}, \ldots, \boldsymbol{e}_{v'^t_L}]; \theta^{\text{FC}})$.

Non-fine-tuned mode. Similar to SLMs and PLMs, the parameters of the fully-connected layer θ^{FC} are trained in this mode.

Fine-tuned mode. Due to a large number of parameters and high computational costs, as illustrated in Figure 3.3, we adopt a two-step process inspired by [8]. First, news recommendation data is transformed into a natural language prompt format, and the

[2] l is set to 10 in practice due to computational efficiency.

Table 3.1: Statistics of the MIND dataset.

#users	#news	#words in title	#words in abs	#pos clicks	#neg clicks
94,057	65,238	11.79	38.17	347,727	8,236,715

LLM parameters θ^{LLM} are updated using cross-entropy loss to let the LLM learn news recommendation-specific information. Second, the LLM, which is fine-tuned in the first step and fixed in the second step, outputs the news embedding for the recommendation process in the same way as in the non-fine-tuned mode, with the fully-connected layer parameters θ^{FC} being updated according to the recommendation objective.

3.4 Experimental Setup

Below, we detail the dataset and implementation; resources to reproduce our results are available at `https://github.com/Go0day/LM4newsRec`.

3.4.1 The MIND Dataset

Following [3, 12, 67, 71, 126, 127, 159], we conduct experiments using the MIND [133] dataset, a public news recommendation dataset collected from the Microsoft News website. Table 3.1 provides descriptive statistics for the dataset. We use the small version of the original MIND dataset, which is widely adopted in academic research and consists of randomly sampled users and their behavior logs. Impressions from November 9 to 14, 2019 are used for training, and those from November 15, 2019 are used for testing [67, 133].

3.4.2 Implementation Details

We use four representative LMs: *GloVe.840B.300d*[3] (referred to as GloVe), *bert-base-uncased*[4] (BERT base version, 110M parameters), *roberta-base*[5] (RoBERTa, 125M), and *Llama 3.1-8B*[6] (Llama for short). These models are selected to cover different families of LMs. To examine the impact of varying model sizes within the same family, we further explore the BERT family by comparing different versions: BERT$_{tiny}$ (4.4M parameters), BERT$_{mini}$ (11.3M), BERT$_{small}$ (29.1M), and BERT$_{medium}$ (41.7M).[7]

Among the selected models, *GloVe.840B.300d* is an SLM, *Llama 3.1-8B* an LLM, and the rest are PLMs. For the PLMs, we fine-tune varying numbers of layers (from none to all) and select the optimal configuration based on recommendation performance. For Llama, we apply LoRA [41] for fine-tuning in step 1, then pre-compute and store news embeddings in advance for recommendation in step 2 (see Figure 3.3). Specifically, NAML [126], NRMS [127], and LSTUR [3] were originally equipped with GloVe,

[3] `https://nlp.stanford.edu/projects/glove`
[4] `https://huggingface.co/google-bert`
[5] `https://huggingface.co/FacebookAI`
[6] `https://huggingface.co/meta-llama`
[7] `https://huggingface.co/prajjwal1`

while PLM-NR [129] was originally equipped with the BERT base version. We re-implement these foundational publications, standardize them within a unified news recommendation setting (including consistent datasets and model structures), and extend their evaluation by incorporating different LMs. For all recommendation methods, the maximum length of news titles is set to 20 tokens, and for news abstracts, it is set to 50 tokens; we search the size of the negative clicked news set $|\mathcal{V}^-|$ in Eq. 3.2 from $\{1, 2, 3, 4\}$, the dropout ratio from $\{0.1, 0.2, 0.3, 0.4, 0.5\}$, and the learning rate from $\{0.0001, 0.00001\}$. We use AUC, MRR, nDCG@5 (N@5), and nDCG@10 (N@10) as our evaluation metrics.

3.5 Results

3.5.1 Impact of LMs on News Recommendation Accuracy (RQ2.1)

To answer RQ2.1, we train news RS methods with different sizes of LMs, as detailed in Section 3.3.3. The results are reported in Table 3.2. We observe:

(1) GloVe generally yields the lowest performance across different LM families, which is expected given its shallow structure. However, it surpasses BERT variants (BERT$_{tiny}$, BERT$_{mini}$, and BERT$_{small}$), showing that larger LMs do not inherently guarantee superior performance as news encoders.

(2) Comparing BERT and RoBERTa, BERT outperforms RoBERTa in most cases, except on LSTUR. This suggests that BERT may offer more effective news encoding, despite RoBERTa's higher parameter count.

(3) The performance of Llama does not significantly exceed that of other LMs, despite its considerably larger parameter count. Thus, an increase in parameters alone does not necessarily translate to better performance.

(4) Within the BERT family, larger models generally achieve better performance than smaller variants. There is one exception: BERT$_{small}$ does not consistently outperform BERT$_{mini}$, even with a higher parameter count.

Our findings for RQ2.1 indicate that, across different LM families, larger LMs do *not* consistently improve news recommendation performance. Within the BERT family, models with more parameters generally perform better; however, this trend is not absolute, as seen in the performance of BERT$_{mini}$ versus BERT$_{small}$.

3.5.2 Impact of Fine-tuning LMs on Performance and Efficiency (RQ2.2)

To investigate the role of fine-tuning, we compare the news recommendation accuracy of non-fine-tuned vs. fine-tuned models and evaluate the computational efficiency of different LMs. This section highlights the trade-off between improved performance and computational feasibility.

Table 3.2: Performance comparison of different LMs as news encoders deployed across three news recommendation methods on the MIND dataset. "BERT" in the left section denotes BERT base version. Results are averaged over three runs and reported as percentages (%). Bold font indicates the winner in that column.

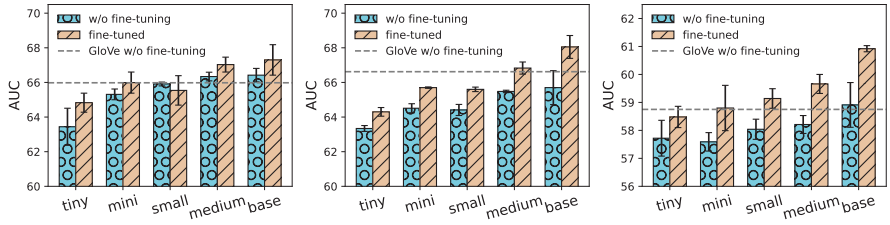
	Performance					Performance			
Model	LM	AUC	MRR	N@5	N@10	LM	AUC	MRR	N@5	N@10
NAML	GloVe	66.29	31.61	34.93	41.22	BERT_{tiny}	64.83	30.71	33.89	40.24
	BERT	67.30	32.62	36.04	42.19	BERT_{mini}	65.99	31.58	34.76	41.02
	RoBERTa	66.73	32.10	35.52	41.64	BERT_{small}	65.91	31.70	34.89	41.24
	Llama	**68.39**	**33.20**	**36.88**	**43.06**	BERT_{medium}	**67.03**	**32.50**	**35.97**	**42.06**
NRMS	GloVe	66.62	31.34	34.83	41.04	BERT_{tiny}	64.30	29.10	31.76	38.56
	BERT	**68.05**	**31.80**	**35.30**	**41.72**	BERT_{mini}	65.70	30.34	33.32	39.82
	RoBERTa	67.22	31.59	34.94	41.35	BERT_{small}	65.60	30.17	33.19	39.83
	Llama	66.64	31.66	35.05	41.33	BERT_{medium}	**66.83**	**31.20**	**34.49**	**40.95**
LSTUR	GloVe	60.43	26.26	28.62	35.11	BERT_{tiny}	58.48	24.47	26.41	33.09
	BERT	60.92	26.74	29.14	35.55	BERT_{mini}	58.80	24.81	27.03	33.62
	RoBERTa	**61.25**	**27.15**	**29.60**	**35.84**	BERT_{small}	59.14	24.65	26.66	33.55
	Llama	60.88	26.83	29.21	35.54	BERT_{medium}	**59.66**	**25.39**	**27.93**	**34.38**

Table 3.3: Performance comparison between fine-tuned and non-fine-tuned settings. The column "FT ?" marks whether an LM is fine-tuned (Y) or not (N). "Change" denotes AUC gain of fine-tuned LMs over non-fine-tuned ones.

Model	LM	FT ?	AUC	MRR	N@5	N@10	Change
NAML	GloVe	Y	66.29	31.61	34.93	41.22	+ 0.46%
		N	65.98	31.67	35.15	41.10	
	BERT	Y	67.30	32.62	36.04	42.19	+ 1.32%
		N	66.42	31.53	34.71	41.06	
	RoBERTa	Y	66.73	32.10	35.52	41.64	+ 5.07%
		N	63.51	29.25	32.29	38.70	
	Llama	Y	67.90	32.99	36.61	42.72	− 0.72%
		N	68.39	33.20	36.88	43.06	
NRMS	GloVe	Y	65.96	30.86	34.09	40.54	− 0.99%
		N	66.62	31.34	34.83	41.04	
	BERT	Y	68.05	31.80	35.30	41.72	+ 3.58%
		N	65.70	30.56	33.33	40.13	
	RoBERTa	Y	67.22	31.59	34.94	41.35	+ 9.11%
		N	61.61	26.44	29.02	35.77	
	Llama	Y	66.64	31.66	35.05	41.33	+ 0.11%
		N	66.56	31.71	35.13	41.30	
LSTUR	GloVe	Y	60.43	26.26	28.62	35.11	+ 2.86%
		N	58.75	25.66	27.89	34.20	
	BERT	Y	60.92	26.74	29.14	35.55	+ 3.41%
		N	58.91	25.17	27.19	33.91	
	RoBERTa	Y	61.25	27.15	29.60	35.84	+ 6.09%
		N	57.74	24.45	26.66	32.97	
	Llama	Y	60.88	26.83	29.21	35.54	+ 2.35%
		N	59.48	26.00	28.36	34.62	

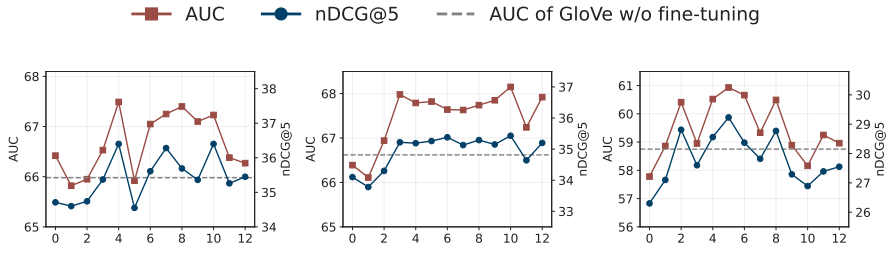
Effectiveness. Table 3.3 and Figure 3.4 show that fine-tuning LMs generally improves news recommendation performance, highlighting the effectiveness of fine-tuning. However, for Llama, the performance benefits of fine-tuning decline in NAML. A plausible reason is that NAML uses both title and abstract text, which may introduce redundancy and noise during Llama's two-step fine-tuning process.

Within the BERT family (see Figure 3.4), we observe that all models benefit from fine-tuning. Notably, in non-fine-tuned settings, most BERT models do not outperform GloVe. This may be due to GloVe's pre-training on the Common Crawl web data [86], likely making it more suited to news content than BERT models trained on BookCorpus and Wikipedia [23]. The effectiveness of GloVe in representing news content could also explain why fine-tuning leads to a slight performance drop when used in NRMS (see



(a) NAML. (b) NRMS. (c) LSTUR.

Figure 3.4: Effect of fine-tuning versus no fine-tuning in the BERT family.



(a) NAML. (b) NRMS. (c) LSTUR.

Figure 3.5: Effect of varying the number of fine-tuned layers in BERT.

Table 3.3).

Additionally, when analyzing BERT (base) by fine-tuning different numbers of layers as shown in Figure 3.5, we observe that the optimal number of layers varies significantly across different RS methods. Interestingly, in some cases, fine-tuned BERT does not outperform non-fine-tuned GloVe, further underscoring GloVe's strength in capturing news representations.

Efficiency. Figure 3.6 provides parameter statistics within the NAML framework, with similar trends in NRMS and LSTUR. "Total parameters" represents all parameters involved in the recommendation process, while "trainable parameters" includes only those updated during training (see Section 3.3.3). Generally, as LM size increases, both total and trainable parameters grow. However, for Llama, the two-step fine-tuning strategy and the use of pre-computed news embeddings significantly reduce its trainable parameters. And for GloVe, the concatenation of word embeddings results in a higher number of trainable parameters than BERT_{tiny} and is comparable to BERT_{mini}.

Overall, these findings underscore different trade-offs when selecting a LM as news encoder (answering RQ2.2): GloVe is an efficient option for cases with limited fine-tuning resources, offering effective performance with minimal computational demands. For static datasets without frequent news updates, precomputing Llama embeddings and using them for inference offers a high-performance alternative. Finally, when both computational resources and performance are priorities, fine-tuning the BERT base version for the news recommendation task provides a balanced, high-performing solution.

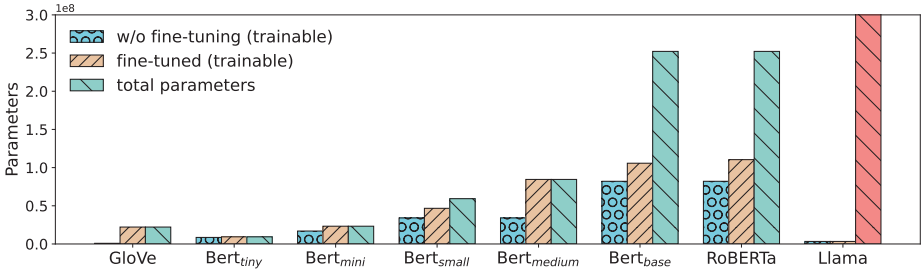


Figure 3.6: Comparison of trainable and total parameters for different LMs in fine-tuned and non-fine-tuned modes within the NAML framework. Llama's total parameter count (over 8 billion) is highlighted in red as it significantly exceeds the scale of other models and cannot be visually included within the same figure.

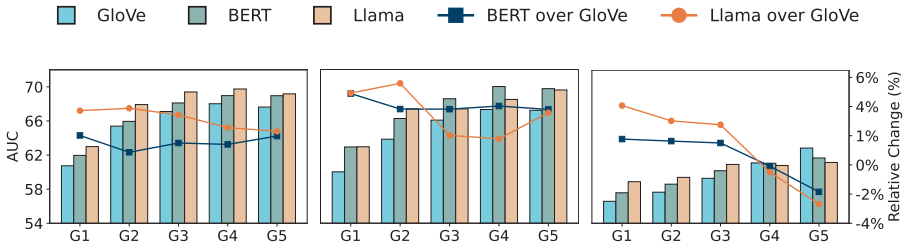
3.5.3 Impact of LMs on Cold-start User Performance (RQ2.3)

Given the inconsistent results regarding performance gains with larger LMs as news encoders (see Section 3.5.1), we further investigate whether larger LMs benefit specific user groups in news RSs. To examine this, we test three representative LMs: GloVe, BERT (base), and Llama. The users are sorted by click history length and categorized into five engagement levels:

- Group 1 (0–20%),
- Group 2 (20%–40%),
- Group 3 (40%–60%),
- Group 4 (60%–80%), and
- Group 5 (80%–100%),

representing different levels of engagement based on the distribution of click history length. The average click lengths for each group, *i.e.,* the number of news articles previously clicked by users in each group, are 4.01, 9.27, 16.57, 29.60, and 48.58, respectively. In Figure 3.7, the bars show the AUC scores for various LMs as news encoders across user groups, while the line plot illustrates each LM's relative change over GloVe.

We find that Llama provides the greatest improvement in Group 1, which includes the "coldest" users with the smallest amount of click history. This improvement may be due to the limited interaction data available for these users, where larger LMs can leverage richer text-based representations to alleviate sparse click signals. As users' click history expands (*e.g.,* Group 5), the relative benefit of larger LMs diminishes, indicating that user engagement itself provides a strong signal for modeling. Interestingly, in the LSTUR model (see Figure 3.7c), the relative improvement from larger LMs decreases more sharply and even turns negative in Group 5. This may be attributed to LSTUR's use of GRU for modeling user preferences, which, unlike the attention mechanisms used in NAML and NRMS, is less effective at capturing evolving user interests [52]. As news representations become more comprehensive with larger LMs, the limitations



(a) NAML. (b) NRMS. (c) LSTUR.

Figure 3.7: Effect of LMs across user groups with varying click history lengths. User groups '1' through '5' represent progressively longer click histories. The 'Relative Change' indicates each LM's performance improvement ratio compared to GloVe.

in modeling dynamic user preferences likely contribute to the observed performance declines.

In response to RQ2.3, these findings suggest that larger LMs enhance performance for cold-start users. Their effectiveness decreases as click history increases, especially in news RSs with limited user modeling capabilities.

3.6 Limitations and Broader Impact

Our study has several limitations. First, we only use the MIND-small dataset for news recommendation due to resource constraints. Additionally, we limit our analysis to English news datasets to focus on evaluating model size effectiveness, leaving out datasets in other languages, such as EB-NeRD [57], Adressa [35], and Plista [54]. Investigating the impact of LMs in non-English news recommendation would be an interesting and valuable direction. Second, we examine LMs with a maximum of 8 billion parameters (Llama) as news encoders, as evaluating larger models (*e.g.,* 13 billion, 70 billion, etc.) exceeded our resource capacity. We expect that larger models might offer further gains, particularly for cold-start users. Third, our study explores three news recommendation methods: NAML, NRMS, and LSTUR, which are commonly used as benchmarks [see, *e.g.,* 129, 132, 148]. In news recommendation, a significant proportion of news articles that are awaiting recommendations do not appear in the logged data. Specifically, in the MIND dataset we used, approximately 32.9% of the news articles in the test set never appear in the training set. This makes ID-based collaborative filtering methods, such as matrix factorization and graph-based approaches, unsuitable for our setting. Therefore, we focus on these three representative content-based news recommendation methods and leave the exploration of other techniques for future work.

Beyond limitations, our study has broader impacts. It provides a reference point for both academia and industry regarding the role of LMs in news recommendation, showing that larger models do not always translate to better performance. Our findings demonstrate that deploying LMs can help address the cold-start problem for new users,

enhancing recommendation reliability for underrepresented groups. We believe our work has the potential to contribute to advancing socially responsible and reliable news recommendation systems.

3.7 Conclusion

In this chapter, we have revisited the role of language models (LMs) as news encoders within neural news RSs on the MIND dataset. We have investigated the effects of varying LM sizes, assessing the impact of fine-tuning on recommendation performance and analyzing model performance across different user groups.

Our main finding is that larger LMs as news encoders do not consistently yield better recommendation results, contrasting with previous studies [129, 148]. Additionally, we observe that larger LMs require more precise fine-tuning and greater computational resources, prompting a trade-off consideration based on performance needs and resource availability.

Notably, we identify an interesting tendency: larger LMs show more significant improvements in recommendations for cold-start users, suggesting potential benefits in modeling user interests with limited click history. A promising future direction is to investigate the stability of LM-based RSs as the news RS domain evolves. Additionally, exploring the design of larger LMs to better meet the dynamic needs of diverse user groups would be valuable.

Part II

Exploiting LLMs to Uncover Vulnerabilities in News Recommender Systems

4

LLM-based Textual Attacks in News Recommender Systems

In Chapter 3, we improved news recommendations by fine-tuning language models to align semantic representations with user interactions, addressing under-representation. In this chapter, we move to the second theme, exploring the adversarial potential of LLMs in news recommender systems (RSs) by examining how they can manipulate textual content to alter recommendation outcomes. We propose a two-stage framework with an explorer for generating diverse rewrites and a reflector for fine-tuning effective attacks, which provides an answer to the following research question from Chapter 1: **RQ3**: Can LLMs be exploited to conduct textual attacks in semantic-rich news recommendation scenarios?

4.1 Introduction

News recommender systems (RSs) rely on rich textual content of news articles and play a unique role in supporting users' participation in a democratic society by recommending news articles. This makes them different from other RS scenarios, *e.g.,* those for products, movies, and books [93, 116]. Neural news RSs often use language models (LMs) to enhance their understanding of the content of news articles and improve recommendation accuracy [165]. These news RSs have been shown to be vulnerable to malicious attacks [44, 78]. The goal of attacks is to produce recommendations as the attacker desires, *e.g.,* an attacker-chosen target news article is recommended to many users. This could lead to severe threats to the trustworthiness of news RSs and significant social consequences, *e.g.,* manipulating users' opinions and spreading misleading information.

Textual Attacks. A widely studied type of attacks in general recommendation scenarios is namely data poison attacks [44], a.k.a. shilling attacks, which commonly inject fake users [102] or fake interactions [17] into the RS to increase the exposure of a target item set. Such attacks are relatively easy to defend against, *e.g.,* by fake user detection [1]

This chapter was published as Y. Zhao, J. Huang, S. Liu, J. Wu, X. Wang, and M. de Rijke. LANCE: Exploration and reflection for LLM-based textual attacks on news recommender systems. In *RecSys 2025: 19th ACM Conference on Recommender Systems*. ACM, September 2025.

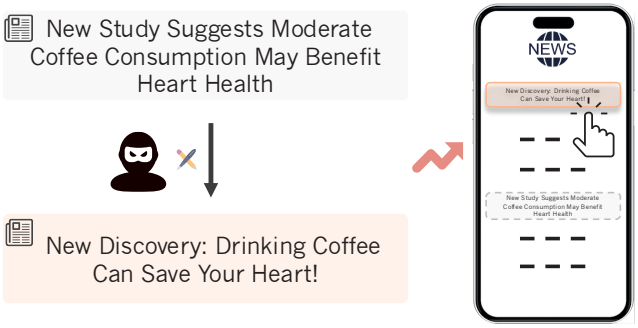


Figure 4.1: Illustrating how textual attack works.

and adversarial training [128]. In contrast, considering the unique nature of news RSs, which relies on textual information, attack strategies w.r.t. news content perturbation have gradually gained attention [78, 158]. In this chapter, we consider a specific, widely studied scenario, where an adversarial content provider wants to boost the ranking of a specific article for all users by textual content perturbation techniques (see Figure 4.1). Accordingly, following Oh et al. [78], we define the *textual attack* in this scenario as an act of the news provider rewriting the news article to increase its exposure to users while keeping the content similar to the original.

Current Limitations. Given the powerful capabilities of large language models (LLMs) in understanding and generating text, it is both straightforward and interesting to explore their use in textual attacks. Zhang et al. [153] mentioned a method to prompt LLMs to rewrite news to make it more "attractive." Due to the inherent difference in objectives between LLMs and textual attacks, as well as the lack of guidance (*e.g.,* indicators of whether the rewritten news will boost its ranking in a specific RS), this approach is not effective in textual attacks (see Section 4.4.2). Wang et al. [123] alter the textual information of target items by simulating the characteristics of popular items. However, this approach is not effective when applied to news RSs, as popular news topics change very frequently. To address these limitations and present an effective attack approach designed for news RSs, Oh et al. [78] propose ATR-2FT, which fine-tunes a small-size LM [*e.g.,* OPT-350M, 156], following a joint learning objective that simultaneously optimizes item ranking and content quality. Although ATR-2FT is implemented using a small-size LM, it can be extended to use LLMs. Still, we identify three limitations of ATR-2FT, even when extended to LLMs:

(1) ATR-2FT requires prior knowledge of the RS to be attacked.[1]

(2) Due to training efficiency constraints, ATR randomly select a subset of users and items to optimize the item ranking promotion objective, which reduces its effectiveness.

(3) ATR optimizes rewritten content quality by updating the text embeddings from the fine-tuned OPT rather than the language space, which reduces naturalness

[1]Even in the black-box setting, ATR-2FT assumes prior knowledge of the type of RS method (*e.g.,* sequential or collaborative filtering-based).

and makes the attack easier to detect.

Proposed Method. To address these limitations and explore the LLMs' ability in textual attacks on news RSs, we propose LANCE, a two-stage **LA**rge language model-based **N**ews **C**ontent r**E**writing framework. In the first stage, we propose an *explorer* component to effectively generate the diverse rewritten content, which can be used to guide the optimization of our LLM-based attack method. We use a powerful closed-source LLM (*e.g.,* GPT-4o) capable of accessing online information to generate rewritten versions of a given news article and identify which rewrites successfully boost its ranking on the news RSs and which do not. To explore potential textual factors that might influence ranking in the news scenario [2, 114, 142], we employ diverse prompts—covering five writing styles, three sentiments, and six personas—to produce varied rewrites, and implement a filtering mechanism to ensure quality.

In the second stage, we fine-tune a *reflector* module to learn effective textual attacks based on the explored rewrites. For each original article, we form a triplet consisting of the original text, a successful rewrite, and a failed rewrite. During fine-tuning, the original article serves as the instruction. We adopt a DPO [91] training procedure that prioritizes successful rewrites over failed ones, ensuring the model learns how to generate more effective attacks.

LANCE operates entirely at the textual level. We explicitly instruct the LLMs to rewrite the news while preserving its original meaning, aiming to avoid semantic mismatches and maintain consistency with the original article. By combining systematic exploration, targeted reflection, our approach addresses current limitations and effectively achieves ranking manipulation.

Main Findings. Extensive experiments show that LANCE achieves state-of-the-art attack performance on three news RSs. By using a small fraction (4.59%) of input news articles and their corresponding rankings, the fine-tuned *reflector* can rewrite news content and effectively boost its rank during inference, causing the RS to promote it above the original version. Our analysis of diverse prompts in the *explorer* indicates that in contrast to attacks on other types of RSs [153], which often insert positive words to raise item ranking, negative and neutral rewrites tend to outperform positive ones in the news domain, revealing a unique attack preference in news RSs. When trained on rewritten data from a single news RS, LANCE can successfully generate rewrites that enhance the target news rankings on unseen news RSs, demonstrating its generalization capabilities.

Contributions. To summarize our contributions:

(1) We introduce LANCE and, using it, show that textual attacks can pose a significant vulnerability for news RSs, requiring low attack costs and limited system information knowledge.

(2) We highlight the unique effectiveness of negative rewrites in news RSs, showing how they differ from attacks on other types of RSs in a news context.

(3) We demonstrate the generalization capability of LANCE by showing that a model trained on one news RS can generate successful attacks on unseen news RSs, showing shared vulnerabilities across news RSs.

(4) We propose an intuitive defense strategy by measuring token probabilities in news text. Although it cannot fully detect rewritten text, it highlights the need and potential for developing more robust defenses in the future.

4.2 Related Work

4.2.1 Attack on Recommender Systems

Posioning attacks on RSs have been shown to be effective in manipulating RSs predictions [124]. Most existing work on poisoning attacks involves injecting fake user interactions into training or test data to control the recommended items. For example, RAPU-R [150] identifies incomplete and perturbed data, and then crafts fake user-item interactions to influence the recommendations. With the rise of content-based RSs, textual attacks have emerged. These attacks focus on manipulating the textual content associated with items, without requiring fake user interactions. For example, ARG [19] introduces a reinforcement learning framework to generate fake reviews that target review-based RSs.

From a seller's perspective, promoting items by inserting fake reviews still requires the creation of fake user accounts to post the reviews. To address this limitation, ATR [78] is a two-phase fine-tuning method to rewrite item descriptions, enabling sellers to unfairly boost their product rankings without needing fake user accounts. Similarly, TextRecAttack [153] targets LLM-based RSs and uses adversarial textual attacks in NLP tasks [30, 51] by perturbing and searching the item text to increase item exposure. It iteratively modifies each text until a stopping criterion is met, requiring repeated system feedback for every item.

These methods often rely on knowledge of the victim RS, such as its parameters or embeddings, or require input-output pairs to train a surrogate victim RS. Inspired by the correlation between popular items and their ranking positions, TextSimu [123] exploits LLMs to simulate the textual characteristics of popular items. Because news RSs are constantly evolving, the attributes that made older news articles popular may no longer be relevant for rewriting new content or enhancing its ranking. Hence, the applicability of existing textual attack models to news RSs is limited.

4.2.2 News Recommender Systems

News RSs provide personalized recommendations by encoding news articles using LMs [63]. Early systems like LSTUR [3] use GloVe embeddings [86] to represent news content and employed GRU networks to learn user representations from their browsing history. NRMS [127] and NAML [126] also use GloVe, with NRMS using multi-head attention and NAML adopting multi-view learning for unified news representations. With the success of pre-trained LMs, models like BERT [23] and RoBERTa [72] have been employed in news RSs. For example, MINER [61] uses BERT for news encoding and introduces poly-attention for user representation. PLM-NR [129] explores multiple pre-trained models to improve news representation. Recently, LLMs have been adopted for encoding news content. ONCE [71] uses both closed and open-source LLMs for news encoding, and Zhao et al. [165] show that LLMs excel in cold-start user scenarios

in news RSs. Unlike other RSs domains, such as e-commerce or movies, most news articles that show up during inference do not appear during training [165]. This makes it difficult for news RSs to use item ID information during inference, forcing them to rely on content to capture relationships between news and users. This reliance exposes news RSs to vulnerabilities from textual attacks based on news content.

4.3 Methodology

4.3.1 Problem Definition

News Recommender Systems. In a news RSs, let $\mathcal{V}$ denote the set of news items and $\mathcal{U}$ denote the set of users. Each news item $v \in \mathcal{V}$ has its textual content t_v, which is encoded by the system's news encoder into a news representation $\boldsymbol{q}_v$. Each user $u \in \mathcal{U}$ has a click history $H_u = \{v_1, v_2, \ldots, v_n\}$. The user encoder processes H_u to produce the user representation $\boldsymbol{p}_u$. The goal of news RSs is to learn a scoring/rank function as follows:

$$\mathcal{R}(t_v, u; f^{RS}) := f^{RS}(t_v, H_u) \tag{4.1}$$

where $f^{RS}(\cdot)$ is the pre-trained and frozen news RS model, and the $\mathcal{R}(\cdot)$ denotes the rank function.

Attack Scenario. A textual attack on news RSs is conducted by a news content provider seeking to promote a target news $t_g \in \mathcal{V}_g$. The attacker rewrites the original content t_g as $t_g{}'$, with the aim of achieving a higher rank (*i.e.,* a lower value in the rank function):

$$\mathcal{R}(t_g{}', u; f^{RS}) < \mathcal{R}(t_g, u; f^{RS}). \tag{4.2}$$

The task is to find a rewriting transformation $t_g \rightarrow t_g{}'$ that improves the news's ranking in $f^{RS}(\cdot)$.

4.3.2 Explorer: Rewriting and Filtering

To explore how rewriting news articles can promote their rankings, we draw inspiration from LLM-based data augmentation [24, 160] and propose a rewriting process. In particular, we consider textual factors that might influence ranking in the news domain [2, 114, 142]. Our *explorer* prompts an LLM to generate rewritten versions of news content across three dimensions: writing styles (*e.g.,* formal to colloquial), sentiment polarity (*e.g.,* positive to negative), and author personas (*e.g.,* objective to opinionated). Detailed descriptions of these dimensions are provided in our repository.[2] Formally, for a news content $t_e \in \mathcal{V}_e$, the *explorer* gets a diverse set of rewritten variants:

$$\mathcal{S}_e = \mathcal{T}_{style}(t_e) \cup \mathcal{T}_{sentiment}(t_e) \cup \mathcal{T}_{persona}(t_e), \tag{4.3}$$

where $\mathcal{T}_{style}(\cdot)$, $\mathcal{T}_{sentiment}(\cdot)$ and $\mathcal{T}_{persona}(\cdot)$ denote rewriting operations that perturb the original text t_e by altering its writing style, sentiment, and persona, respectively. $\mathcal{S}_e$ collects all rewritten versions of $t_e \in \mathcal{V}_e$. Here, $\mathcal{V}_e$ is sampled from the training-stage

[2] `https://github.com/Go0day/LANCE`.

news, while the target news $t_g \in \mathcal{V}_g$ is sampled from the inference-stage news to evaluate the attack effectiveness against the deployed news RS. Below is an example of the rewriting prompt used by the *Explorer* to apply different writing styles:

> **Task**: *Rewrite the provided news title and abstract into 5 different versions, each reflecting a specific writing style. The output should maintain the original core information while adhering to the designated tone and length constraints.*
>
> **Writing Styles**: *Narrative, Persuasive, Journalistic, Humorous and Conversational.* $<\mathcal{D}>$
>
> **Original News**: $<t_e>$

where $<\mathcal{D}>$ denotes the detailed description of different writing styles [142], and $<t_e>$ represents the original news article to be rewritten.

While the rewritten data $\mathcal{S}_e$ can be used to fine-tune the *reflector*, we apply additional filtering to ensure the quality of rewrites. Consider two news articles t_e^A and t_e^B, each with rewritten versions s^A and s^B. *Case 1*: t_e^A improves from rank 100 to 60 with s^A. *Case 2*: t_e^B improves from rank 50 to 10 with s^B. Though both rewrites achieve a 40-rank improvement, s^B is more impactful: users typically browse only the top-ranked news (*e.g.,* top 50), so s^B gains visibility while s^A remains largely unseen. Therefore, we filter the rewritten news $s_e \in \mathcal{S}_e$ and construct the successful rewrites S_e^+ as follows:

$$S_e^+ = S_e^+ \cup \{s_e\} \quad \text{if} \begin{cases} \mathcal{R}(s_e, u; f^{RS}) < K, \\ \mathcal{R}(t_e, u; f^{RS}) > K. \end{cases} \tag{4.4}$$

where K represents the top-K ranking threshold and serves as a hyperparameter. For unsuccessful rewrites S_e^-, we identify explicit cases where the rewritten news fails to improve ranking:

$$S_e^- = S_e^- \cup \{s_e\} \quad \text{if } \mathcal{R}(s_e, u; f^{RS}) < \mathcal{R}(t_e, u; f^{RS}). \tag{4.5}$$

Finally, the selected rewritten news samples (t_e, S_e^+, S_e^-) are collected for training the *reflector*.

4.3.3 Reflector: Fine-Tuning for Textual Attack

Based on the explored rewritten news, we now illustrate how the *reflector* learns to rewrite news content to improve its ranking. Recent studies have shown that human-labeled, pairwise data can serve as reward signals to align LMs with human preferences, such as RLHF [81] and DPO [91]. RLHF uses a preference model to model preference distributions, whereas DPO directly learns the optimal policy from pairwise preference data and is often more practical for preference alignment. Therefore, we employ DPO to train the *reflector* on the rewritten data, enabling it to perform textual attacks on news RSs more effectively. Below is the fine-tuning template used for preference alignment:

> **Task**: *You are an expert in news content optimization for recommender systems. Your goal is to rewrite the given news title and abstract to maximize their chances of ranking higher in a typical news recommender system. The output should maintain the original core information while adhering to the goals and length constraints.* $<t_e>$
> *Please rewrite the title and abstract according to two optimization goals: (1) Focused on maximizing user clicks. (2) Focused on improving algorithmic ranking based on relevance and engagement.*
>
> **Chosen rewrite**: $s_e^+ \in S_e^+$
>
> **Rejected rewrite**: $s_e^- \in S_e^-$

where $<t_e>$ is the original news content, s_e^+ and s_e^- denote rewrites that respectively succeed or fail to promote the t_e's ranking. We implement the *reflector* using Llama 3.1-8B, parameterized by θ, and fine-tune it with a DPO loss to maximize the probability of the chosen rewrite and minimize the probability of the rejected one:

$$\begin{aligned} L_{\text{DPO}}(\theta) = \; & -\mathbb{E}_{(t_e, s_e^+, s_e^-)} \left[\log \sigma \left(\beta \log \frac{p_\theta(s_e^+|t_e)}{p_{\text{ref}}(s_e^+|t_e)} \right.\right. \\ & \left.\left. -\beta \log \frac{p_\theta(s_e^-|t_e)}{p_{\text{ref}}(s_e^-|t_e)} \right) \right], \end{aligned} \tag{4.6}$$

where $\sigma(\cdot)$ is the sigmoid function, β is a temperature parameter, $p_\theta(s_e|t_e)$ represents the probability of the *reflector* generating s_e given t_e, and $p_{\text{ref}}(s_e|t_e)$ denotes the probability of generating s_e given t_e under the reference model, which is the pre-trained LLM before preference fine-tuning.

Final Attack Generation. Finally, we describe the attack generation process. Given a target news t_g, the fine-tuned *reflector* transforms and rewrites it into a new version: $t_g \xrightarrow{p_\theta(\cdot)} t_g{}'$. The rewritten $t_g{}'$ is then submitted to the victim news RSs to promote its ranking, thereby achieving the attack goal.

4.4 Experiments

In this section, we conduct extensive experiments to answer the following research questions:[3]

RQ3.1 How does the proposed LANCE approach perform compared to existing models in attacking news RSs?

RQ3.2 How do variations in diverse news content (*i.e.,* writing style, sentiment, and persona) affect the performance of rewrite attacks on news RSs?

RQ3.3 How does fine-tuning a single LLM perform in attacking different news RSs?

RQ3.4 How does the LANCE attack affect wider dimensions (*i.e., recommendation performance*, *semantic preservation*, and *detectability*) of news RSs ?

[3]Our code is available at: `https://github.com/Go0day/LANCE`.

Table 4.1: Statistics of the MIND dataset.

#users	#news	$\lvert\mathcal{V}_e\rvert$	$\lvert\mathcal{V}_g\rvert$	$\frac{\lvert\mathcal{V}_e\rvert}{\#\text{news}}$
94,057	65,238	3,000	300	4.59%

4.4.1 Experimental Setup

Dataset

Benchmark Datasets. We conduct experiments on the MIND dataset [133], a publicly available news recommendation dataset from Microsoft News. The dataset's statistics are detailed in Table 4.1. We employ impressions from November 9–14, 2019 for training the RSs and impressions from November 15, 2019 for testing [67, 165].

Target News. Previous textual attack methods on recommender systems [78, 153] typically select a random set of target items and perform word perturbation search or rewrite learning directly on them. In the news domain, where information changes rapidly, such approaches lack robustness and may fail to effectively rewrite and promote unseen news articles. To address this limitation, we derive attack training news t_e from news that appears in the MIND training impressions and derive target news t_g from news in the MIND test impressions. To ensure that the textual attack generalizes across news with varying popularity levels [73, 165], we randomly select 1,000 training news items and 100 test news items from three popularity levels based on frequency: (0–20%), (40–60%), and (80–100%). These selections form $\mathcal{V}_e$ and $\mathcal{V}_g$, with final sizes of $|\mathcal{V}_e| = 3,000$ and $|\mathcal{V}_g| = 300$, respectively.

News Recommender Systems

We select three mainstream news RS models as victim models: NAML [126], NRMS [127], and LSTUR [3]. Following [129, 165], we re-implement these models using the BERT-base version as the news encoder, with its parameters fine-tuned on recommendation signals. Their architectures are summarized below:

- **NAML** [126]: NAML models news representations using both titles and abstracts. It applies a multi-view learning mechanism to integrate titles, bodies, categories, and subcategories. User representations are learned through an attention mechanism based on browsing history.

- **NRMS** [127]: NRMS models news representations using only titles. It employs a multi-head self-attention mechanism to capture semantic features. User representations are learned through self-attention on browsing history.

- **LSTUR** [3]: LSTUR models news representations using titles and topic categories. It employs a GRU network to learn user representations from browsing history. The model captures both long-term and short-term user interests.

Table 4.2: Attack performance comparison across three news RSs. Bold values indicate the best performance in each column, while underlined values represent the second-best. Results are averaged over five runs. The raw "%Impv" column shows the relative performance improvement of LANCE over the second-best method. For the Rank metric, "%Impv" denotes the average rank improvement compared to the original target news ranking.

Model	Context?	NAML				NRMS				LSTUR			
		BSR↑	Rank↓	Expo↑	Appear↑	BSR↑	Rank↓	Expo↑	Appear↑	BSR↑	Rank↓	Expo↑	Appear↑
Original	-	-	21,001	0.29	0.0058	-	20,970	0.11	0.0022	-	20,811	0.04	0.0007
GPT-4o	Y	0.44	22,097	0.46	0.0093	0.43	22,557	0.08	0.0016	0.46	21,808	0.03	0.0006
	N	0.43	22,193	0.43	0.0086	0.41	22,901	0.09	0.0019	0.43	22,241	0.03	0.0007
GPT-3.5	Y	0.42	22,317	0.48	0.0095	0.43	22,429	0.09	0.0018	0.47	21,317	0.06	0.0012
	N	0.42	22,304	0.48	0.0097	0.41	22,692	0.09	0.0018	0.43	21,983	0.03	0.0007
Llama-3.1	Y	0.45	21,800	0.43	0.0087	0.45	22,107	0.09	0.0018	0.46	21,562	0.04	0.0008
	N	0.45	21,828	0.50	0.0101	0.44	22,153	0.10	0.0020	0.45	21,674	0.05	0.0010
Llama-FT$_{rec}$	-	0.45	21,742	0.51	0.0103	0.46	21,972	0.07	0.0014	0.45	21,806	0.05	0.0009
ATR-2FT	-	0.53	20,604	0.52	0.0104	0.47	21,182	0.17	0.0034	0.49	21,013	0.06	0.0015
LANCE	-	**0.69**	**18,125**	**0.79**	**0.0159**	**0.56**	**18,677**	**0.41**	**0.0084**	**0.57**	**19,375**	**0.08**	**0.0016**
% Impv	-	30.2%	2,479	51.9%	52.9%	19.2%	2,293	141.2%	147.0%	16.3%	1,436	33.3%	6.7%

Baseline Rewriting Methods

Textual attacks rewrite news content to improve its ranking. We compare LANCE with eight LLM-based baselines, including the state-of-the-art ATR [78]; we exclude shilling attacks from the baselines because they require generating fake users and ratings, which falls outside the scope of our work. Similarly, we exclude adversarial textual attack methods from NLP tasks, such as TextRecAttack [153], because they rely on repeated feedback from a RS for each item. This approach is impractical for our target news set, $\mathcal{V}_g$, where no information is available at this stage.

- **GPT-4o**, **GPT-3.5** and **Llama-3.1** (without context). We follow the implementation described in existing work [153]. In this baseline group, we prompt these LLMs to rewrite news items without incorporating any news RSs data as contextual input.

- **GPT-4o**, **GPT-3.5** and **Llama-3.1** (with context). We follow the implementation from prior work [123]. In this baseline group, we use popular news articles from the same category as the target news as context. The LLMs are then prompted to rewrite the news, using contextual information to improve ranking performance.

- **Llama-FT$_{\text{rec}}$**. We fine-tune Llama-3.1-8B on news recommendation data. This fine-tuning injects domain-specific knowledge into the LM. The fine-tuned model then performs the rewriting task without additional contextual input to enhance the target news ranking.

- **ATR-2FT** [78]. ATR-2FT is a recent textual attack method that uses a fine-tuned LM [OPT-350M, 156]. To adapt ATR-2FT to our experimental setting, we isolate the news text encoder from the news RS and perform embedding alignment between the LM embeddings and the news text encoder. The attack process is optimized using a promotion loss and a text generation loss. To ensure a fair comparison, we train ATR-2FT on our sampled news dataset $\mathcal{V}_e$ and test it on the target news dataset $\mathcal{V}_g$.

Implementation Details

We use GPT-4o to explore rewrites, as it can access external knowledge and perform well in the news domain. We use Llama-3.1-8B in LANCE for *Reflector*. This open-source model has strong reasoning capabilities for textual attacks. All attack models and news RSs are implemented in PyTorch. For DPO fine-tuning, we use Lora (Low-Rank Adaptation) [41] to efficiently fine-tune the model. We set $\beta = 0.1$, and the learning rate is selected from {1e-5, 5e-5, 1e-4, 5e-4}. The number of fine-tuning epochs is fixed at 3. For news RSs, we use the same hyperparameters as in [165]. All training is performed on three NVIDIA RTX A6000 GPUs, each with 49,140M of memory.

Evaluation Metrics

Attack Performance. We evaluate attacks using four metrics: (i) Boost Success Rate (BSR): The percentage of target news articles successfully boosted above the average

Table 4.3: Comparison of naturalness performance. LSTUR is excluded due to a similar title rewriting method as NRMS. The yellow background in the PPL column indicates that ATR-2FT has worse perplexity than the original text.

Model	Context?	NAML				NRMS			
		PPL↓	BLEU↑	RougeL↑	BertS↑	PPL↓	BLEU↑	RougeL↑	BertS↑
Original	-	144.3	-	-	-	257.1	-	-	-
GPT-4o	Y	112.5	0.090	0.319	0.653	253.6	0.064	0.364	0.644
	N	124.3	0.097	0.333	0.660	223.7	0.068	0.388	0.652
GPT-3.5	Y	127.8	0.139	0.384	0.690	293.5	0.135	0.450	0.698
	N	145.9	0.127	0.372	0.678	290.5	0.138	0.445	0.687
Llama-3.1	Y	83.6	0.137	0.355	0.666	209.0	0.089	0.395	0.652
	N	74.6	0.140	0.350	0.660	200.4	0.083	0.391	0.647
Llama-FT$_{\text{rec}}$	-	79.2	0.141	0.357	0.666	199.2	0.087	0.399	0.653
ATR-2FT	-	179.7	0.090	0.313	0.636	323.9	0.070	0.411	0.657
LANCE	-	79.6	0.120	0.328	0.649	195.8	0.075	0.401	0.649

rank across all users. (ii) Average Predicted Rank (Rank): The mean rank of target news articles across all users, with and without adversarial rewrites. (iii) Exposure@K (Expo): The proportion of users who see the target news articles in their top-K recommendations. (iv) Appear@K (Appear): The frequency of target news articles appearing in the top-K recommendations across all users. We set $K = 50$.

Naturalness Performance. We evaluate the naturalness of rewritten news articles using the following metrics: (i) Language model perplexity (PPL), which measures how well a language model predicts the rewritten text. (ii) BLEU [82], which evaluates the overlap of n-grams to assess the quality of the rewritten content compared to the original. (iii) ROUGE-L [69], which measures recall-oriented similarity based on the longest common subsequence between the rewritten and original news. (iv) BERTScore (BertS) [157], which uses contextual embeddings to capture semantic similarity between the rewritten and original news.

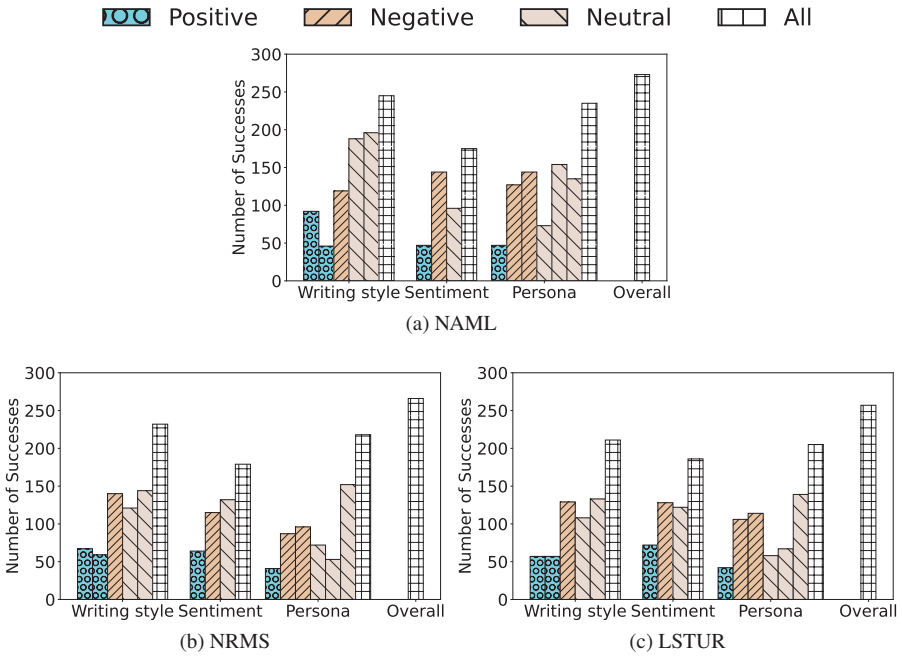
4.4.2 Performance Comparison (RQ3.1)

We compare LANCE with several baseline rewriting methods, as outlined in Section 4.4.1, to evaluate their effectiveness in manipulating news RSs while preserving the naturalness of the rewritten text. The baselines consist of eight methods: six are not fine-tuned (GPT-4o, GPT-3.5 and Llama-3.1, each with and without context), and two are fine-tuned (Llama-FT$_{\text{rec}}$ and ATR-2FT).

Attack Performance

The results of the attack performance are presented in Table 4.2. Below, we make the following three observations:

- LANCE outperforms all baselines across the three news RSs. This improvement is attributed to the diverse rewrites generated by the *explorer* and the quality



(a) NAML

(b) NRMS

(c) LSTUR

Figure 4.2: Impact of diverse rewrites on attacking news RSs. 'All' represents the combined successful rewrites within each group, while 'Overall' denotes the total number of successful rewrites across all versions.

control mechanism for successful rewrites. Fine-tuning on these collected rewrites enables the *reflector* to effectively align with the attack task.

- ATR-2FT gets the second-best attack performance in improving news rankings, demonstrating the effectiveness of its two-phase fine-tuning framework. However, it fails on NRMS and LSTUR. This is likely because both news RSs systems encode only title text, which limits the impact of content-based rewrites.

- All LLM-based baselines perform poorly on BSR and Rank but show improvements in Expo and Appear. This suggests that prompting LLMs without clear rewriting directions causes performance variance. Some rewrites boost rankings, but others result in significant rank drops. Additionally, using popularity-based context does not help, likely because dynamic popularity shifts make simulated popularity-based rewrites ineffective. Fine-tuning Llama on recommendation text also fails to help, as injecting recommendation information without guiding the attack objective is ineffective for the attack task.

Naturalness

The results regarding naturalness are shown in Table 4.3. We make the following observations:

- On naturalness metrics, LANCE performs competitively across all baselines. It achieves a lower perplexity (PPL) than the original text, indicating improved fluency. Although its BLEU, RougeL, and BertS scores are not the highest, they remain reasonable, suggesting that rewrites retain sufficient similarity to the original text.

- In contrast, ATR-2FT increases perplexity compared to the original text, resulting in less natural output. This degradation likely results from its joint fine-tuning approach with a promotion loss mechanism, which updates the LM OPT parameters to prioritize attack success over text quality.

- The LLM-based baselines generally produce fluent and similar text, reflected in low PPL and high BLEU, RougeL, and BertS scores. However, their lack of attack-specific optimization limits their utility for improving rankings.

4.4.3 Diverse Rewrite Effectiveness (RQ3.2)

To investigate how different rewrite styles affect attack performance, we collect diverse rewrites of the target news articles (cf. Section 4.3.2) and measure the number of successful rank improvements for each rewrite version. To ensure a natural classification, we use GPT-4o to label each rewrite into three categories based on style: Positive, Negative, and Neutral (details can be found in our code repository[4]). Figure 4.2 shows the distribution of successful rewrites by category. We have the following observations:

- Diverse rewrites complement each other in attacks. Although the prompts differ in style, all versions of rewrites achieve successful attacks, contributing to a high overall success rate. This suggests that diverse rewrites can complement each other when attacking different news articles. Such diversity has the potential to enhance the *reflector*'s ability to generate effective attacks.

- Negative and neutral rewrites are more effective in attacking news RSs. Negative news often triggers curiosity, leading to more clicks and higher rankings. Neutral news, which emphasizes factual content, is typically more informative and straightforward. Such content is easier for recommender systems to classify and surface based on relevance and clarity.

- Positive rewrites show lower success rates. Positive news often lacks urgency and fails to spark curiosity, resulting in lower engagement metrics such as clicks and shares. Additionally, since recommender systems prioritize content with high engagement signals (*e.g.,* click-through rates, comments, and shares), they may rank positive news lower due to its relatively weaker interaction rates. This finding contrasts with attacks observed in other RSs domains, such as e-commerce [153], where positive descriptions are more effective. It highlights a unique vulnerability in the news recommendation scenario.

[4] `https://github.com/Go0day/LANCE`.

Table 4.4: Generalization comparison of attack performance across different news recommender systems.

Model	NAML				NRMS				LSTUR			
	BSR↑	Rank↓	Expo↑	Appear↑	BSR↑	Rank↓	Expo↑	Appear↑	BSR↑	Rank↓	Expo↑	Appear↑
Original	-	21,001	0.29	0.0058	-	20,970	0.11	0.0022	-	20,811	0.04	0.0007
w/ $\mathcal{V}_e^{\text{NAML}}$	-	-	-	-	0.53	20,193	0.16	0.0032	0.58	19,416	0.10	0.0020
w/ $\mathcal{V}_e^{\text{NRMS}}$	0.53	20,518	0.44	0.0088	-	-	-	-	0.55	19,700	0.07	0.0015
w/ $\mathcal{V}_e^{\text{LSTUR}}$	0.58	20,018	0.74	0.0149	0.52	20,562	0.20	0.0040	-	-	-	-
w/ $\mathcal{V}_e^{\text{ALL}}$	0.56	19,941	0.94	0.0187	0.54	20,174	0.27	0.0054	0.58	19,289	0.09	0.0018
LANCE	0.69	18,125	0.79	0.0159	0.56	18,677	0.41	0.0084	0.57	19,375	0.08	0.0016

Table 4.5: News recommendation performance before and after the LANCE attack. The "Change" column denotes the relative performance change.

	Mode	AUC	MRR	nDCG5	nDCG10
NAML	Before	68.49	33.30	36.99	42.98
	After	68.41	33.28	36.97	42.93
	Change	-0.117%	-0.060%	-0.054%	-0.116%
NRMS	Before	66.39	31.24	34.10	40.85
	After	66.37	31.24	34.09	40.84
	Change	-0.030%	0.000%	-0.029%	-0.024%
LSTUR	Before	60.41	26.38	28.81	35.35
	After	60.34	26.36	28.79	35.32
	Change	-0.116%	-0.076%	-0.069%	-0.085%

4.4.4 Generalization Capability in Cross-System Attacks (RQ3.3)

To evaluate the generalization capability of LANCE, we consider four versions of LANCE with different training and evaluation settings: (i) w/ $\mathcal{V}_e^{\text{NAML}}$ – fine-tuned on rewrites from NAML and evaluated on NRMS and LSTUR; (ii) w/ $\mathcal{V}_e^{\text{NRMS}}$ – fine-tuned on NRMS and evaluated on NAML and LSTUR; (iii) w/ $\mathcal{V}_e^{\text{LSTUR}}$ – fine-tuned on LSTUR and evaluated on NAML and NRMS; and (iv) w/ $\mathcal{V}_e^{\text{ALL}}$ – fine-tuned on all rewrites from the three news RSs and evaluated on all models.

We observe the following:

- *Cross-system generalization*: Table 4.4 shows that all versions of LANCE improve the target items' rankings and exposure rates, demonstrating the effectiveness of our framework. Notably, w/ $\mathcal{V}_e^{\text{NAML}}$, w/ $\mathcal{V}_e^{\text{NRMS}}$ and w/ $\mathcal{V}_e^{\text{LSTUR}}$ achieve performance gains on unseen models they were never trained on. This indicates that the textual attack model can generalize to attack unseen news RSs.

- *Effect of training on combined data*: The w/ $\mathcal{V}_e^{\text{ALL}}$ model performs worse than individually trained versions on NAML and NRMS but achieves better results on LSTUR. This suggests that while fine-tuning on specific models yields strong performance, training on combined data has the potential to enhance generalization, particularly on models like LSTUR.

- *Impact of data quality*: We observe that w/ $\mathcal{V}_e^{\text{ALL}}$ performs worse than w/ $\mathcal{V}_e^{\text{LSTUR}}$ when attacking NAML. This may be because NRMS employs multi-head self-attention to identify key words, making it more sensitive to semantic rather than stylistic rewrites. As a result, mixing data from NRMS causes the model trained on $\mathcal{V}_e^{\text{NRMS}}$ to become confused, leading to subpar performance on NAML. This highlights the critical role of data quality in textual attacks.

4.4.5 Effect on Wider Dimensions (RQ3.4)

Recommendation Performance

We assessed the impact of our LANCE attack on recommendation performance by replacing targeted news items ($\mathcal{V}_g$) with rewritten versions and evaluating key metrics: Area Under the Curve (AUC), Mean Reciprocal Rank (MRR), and normalized Discounted Cumulative Gain (nDCG) at 5 and 10. The results, shown in Table 4.5, indicate slight performance drops, with a maximum AUC decrease of 0.117% for NAML. These minimal changes demonstrate that LANCE disrupts recommendation rankings—our primary goal—while maintaining stealthiness. By altering the content of target news $\mathcal{V}_g$, our approach ensures minimal degradation, avoids detection by platform owners, and promotes the rank of target news (see Section 4.4.2), achieving a successful attack.

Semantic Preservation

In addition to evaluating naturalness metrics like BLEU, ROUGE-L, and BERTScore (as mentioned in Section 4.4.2), we conducted a simulated user study using LLM (GPT-4o) to directly assess how well LANCE preserves textual semantics compared to ATR-2FT. The methodology is outlined below:

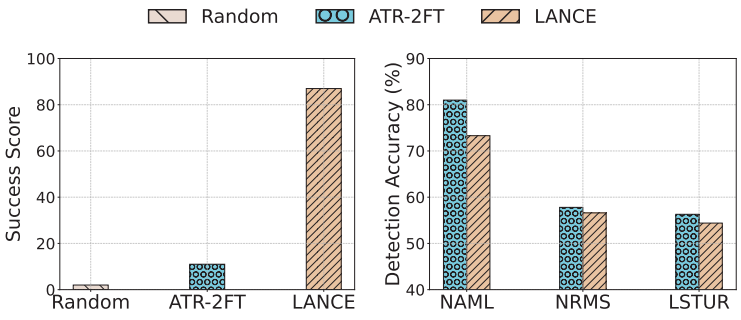
- Candidate Preparation: We sampled 100 news articles from the test set as original targets. For each article, we created three candidate lists: (i) rewritten by LANCE, (ii) rewritten by ATR-2FT, both under identical attack conditions, and (iii) a random article from the same category.
- User Simulation: GPT-4o evaluated the three candidates for each article, selecting the one that best preserved the original semantics. This process was repeated five times per article, with the candidate receiving the most votes across the five runs earning one point. The Success Score reflects the total points accumulated across all 100 articles. List positions were randomized for each presentation to ensure fairness.

The results, shown in Figure 4.3a, indicate that LANCE achieves the highest Success Score, outperforming both ATR-2FT and the random baseline. This demonstrates LANCE's superior ability to retain semantic integrity during rewriting.

Detectability of Rewritten Content

As noted in Section 4.4.2, LANCE-generated attack text exhibits high naturalness, complicating human detection. However, its perplexity scores are notably lower than those of original news text, suggesting a detectable difference between LANCE-rewrited and original news content. To investigate potential defenses, we conducted a preliminary detection experiment, detailed as follows:

- Data Preparation: We combined original news text with rewritten versions from LANCE, labeling each as original (0) or rewritten (1). The dataset was split into 70% training and 30% testing sets. We did the same to ATR-2FT, providing a reference about the detection accuracy.



(a) Success Scores (b) Detection Accuracies

Figure 4.3: (a) Success scores in simulated user study for semantic preservation, and (b) Detection accuracies of rewritten content across victim models.

- Text Processing: Using GPT-2 (which we used to calculate perplexity scores in Section 4.4.2), we computed probabilities for each token in the text sequences. These sequences were truncated or padded to a uniform length for consistent feature representation.
- Detection Model Training: A three-layer multilayer perceptron (MLP) was trained on the labeled training set to classify texts as original or rewritten. Separate models were trained for LANCE and ATR-2FT outputs.

Detection accuracies across victim models are presented in Figure 4.3b. The MLP achieves more than 70% detection accuracy for LANCE on NAML, indicating that language model probabilities can effectively identify rewritten text. LANCE proves stealthier than ATR-2FT, with consistently lower detection rates. However, accuracy drops on NRMS and LSTUR for LANCE and ATR-2FT, likely because these models encode news titles for recommendations, limiting the token probability information for detection. These findings highlight LANCE's stealth advantage, and reveal a promising direction for refining detection strategies.

4.5 Limitations and Broader Impact

Our work has several limitations. First, we use only the MIND dataset due to the limited availability of news recommendation datasets. Additionally, we limit our textual attacks to English news rewrites to focus on attack effectiveness, excluding other languages. Investigating performance on non-English news recommendation datasets [35, 57] is an important direction for future work. Second, we conduct attacks using Llama-3.1-8B and fine-tune it with the DPO method. Exploring other LLM versions, model sizes, and designing a specialized fine-tuning mechanism to better align the LLM with the attack task has the potential to yield further improvements. This would further highlight the vulnerability of news RSs to textual attacks. Third, we evaluate attacks on three news RSs models: NAML, NRMS, and LSTUR, which are commonly used benchmarks in the news recommendation domain [129, 165]. Due to the frequent updates in news domain, ID-based collaborative filtering methods are less

suitable for news recommendation [165]. We leave the exploration of attacks on other recommendation techniques for future work.

Beyond limitations, our study has broader impacts. It reveals the vulnerabilities present in news RSs, which are shared across systems, thereby allowing textual attacks like LANCE to be effectively generalized. In today's world, and potentially more in the future, everyone can be not only a consumer of information but also a content provider. This means that users can easily conduct textual attacks, leading to severe consequences for user trust, platform integrity, and even societal cohesion. This underscores the urgent need for research into defense strategies to counter such attacks. We believe our work can contribute to the development of more secure and socially responsible news RSs.

4.6 Conclusion

In this chapter, we propose LANCE, an LLM-based news rewriting framework for textual attacks on news RSs. With a diverse *explorer* and fine-tuned *reflector*, LANCE generates rewrites that effectively boost rankings while preserving text naturalness and semantic meaning. Our results reveal a unique vulnerability in news RSs compared to other RSs domains (*e.g.,* e-commerce), as negative and neutral rewrites consistently outperform positive ones, highlighting a distinct ranking preference. Additionally, we demonstrate a shared vulnerability across different news RSs, as LANCE successfully attacks unseen systems. Notably, although our attack text achieves high naturalness – making it difficult for humans to detect – its perplexity scores are significantly lower than those of the original news text. This indicates a detectable difference between LLM-generated content and human-written text, which could be used to identify such attacks. A promising future direction is to investigating performance on non-English news recommender systems, using, *e.g.,* the recently released EB-NeRD dataset [57]. Another important direction is to develop defense mechanisms that exploit these differences between LLM and human text generation, helping news RSs mitigate textual attacks more effectively.

5

Media Bias-Aware Textual Attacks in News Recommender Systems

In Chapter 4, we demonstrated the capability of LLMs to perform textual attacks in news recommender systems (RSs) through rewrites that boost rankings. In this chapter, we extend this work by incorporating media bias, exploring attacks that preserve media bias orientation. We propose a framework that uses progressive fine-tuning of rank and bias experts to unify these two objectives, which addresses the research question introduced in Chapter 1: **RQ4**: How can LLMs be used to conduct textual attacks that preserve media bias orientation in news recommender systems?

5.1 Introduction

Large language models (LLMs) are increasingly exploited in offensive security applications, such as user-level attacks (*e.g.,* misinformation and social engineering), due to their ability to generate human-like text [147]. Recent advances in LLMs have demonstrated impressive text rewriting capabilities, applicable to various scenarios such as paraphrasing, style transfer, and simplification. This capability makes LLMs powerful tools for conducting *textual attacks* on news recommender systems (RSs), which rely heavily on textual content to deliver personalized articles. In a textual attack on news RSs, an adversarial content provider subtly perturbs the text of a news article to boost its ranking across all users while preserving semantic similarity to the original [78]. Such attacks pose serious societal risks, potentially exposing users to shocking or extreme content, inciting panic, or prompting irrational actions.

Textual attacks are highly feasible, as news content providers typically possess the authority to edit articles. Unlike shilling or injection attacks [19, 102, 124], which involve creating fake users, items, interactions, or reviews—actions often beyond providers' capabilities and detectable through methods such as fake user detection or adversarial training—textual attacks are more practical and less noticeable. These attacks involve subtly altering article text to influence RSs, making them harder to detect than conventional attack approaches. However, current textual attack methods face

This chapter is currently under review as Y. Zhao, Y. Li, J. Huang, X. Wang, and M. de Rijke. Unseen threats: Media bias-aware textual attacks on news recommender systems. *Submitted for review*, 2025.

limitations in news RSs. For instance, TextRecAttack [153] adapts natural language processing (NLP) techniques, such as TextFooler [51], to perturb item text and increase item exposure. This approach requires extensive system interaction with the target item to produce effective attacks. Oh et al. [78] introduce ATR-2FT, a two-phase fine-tuning method for rewrite attacks. Its joint training to optimize item ranking and content quality faces efficiency constraints and is limited to small-scale language models (*e.g.,* OPT-350M [156]). TextSimu [123] modifies textual content by mimicking popular items, but this method is less effective in news RSs due to the rapidly changing nature of news, where previously popular news often loses relevance over time. Critically, these methods prioritize item exposure while overlooking the media bias implications of modified content, risking misalignment with the original article's ideological stance.

5.1.1 Media Bias

Media bias in news refers to the selective presentation of information shaped by ideological leanings [96, 103], which may originate from the platform, journalist, or content itself. For example, labeling someone an "illegal immigrant" versus an "undocumented worker" implies differing associations. Media bias orientation (also referred to as bias orientation) can be systematically categorized as left, center, or right-leaning directions [37, 58, 96, 104]. Ignoring media bias in textual attacks has several negative consequences. On the platform side, rewritten content conflicting with the platform's ideological stance risks detection, leading to attack failure. On the user side, content misaligned with their beliefs may reduce engagement and click-through rates due to confirmation bias. For attackers, failing to account for media bias orientation limits their ability to tailor attacks to specific platforms or user groups, reducing the potential to influence societal perceptions effectively. Thus, incorporating bias-aware strategies is essential for enhancing the stealth and efficacy of textual attacks on news RSs.

Although important, the problem of producing textual news attacks that preserve ideological stances remains underexplored in the literature. The primary challenge stems from the absence of clear signals for preserving media bias orientation while improving ranking. Attackers can only rewrite news content and assess whether it enhances ranking or maintains bias orientation, without insight into which factors promote both objectives. To investigate this topic further, we conduct an empirical analysis of real-world datasets to examine how the tasks of preserving the original article's ideological stance and improving its ranking affect each other. Our analysis yields the following observations:

(1) *Textual attacks aimed at rank improvement often unintentionally alter the original article's ideological stance*, indicating that these objectives are not inherently aligned.

(2) *Preserving ideological stance direction constrains rank improvement*, revealing a trade-off between the tasks.

(3) *Context-enhanced LLMs show potential to mitigate unintended bias orientation shifts*, highlighting the promise of LLMs in reducing unintended bias orientation changes during textual attacks.

5.1.2 Proposed Method

Inspired by these insights, we introduce a novel framework for **B**i**A**s-aware, **L**LM-based textual **A**ttacks on **N**ews re**C**omm**E**nder systems (BALANCE). Our approach employs LLMs to rewrite news content, and aims to boost its ranking while preserving its original media bias orientation. To overcome the challenge of limited signals, we explore three types of dataset—rank data, bias data, and combined data (which contains articles altered to simultaneously improve ranking and preserve bias orientation)—from which we derive insights into effective rewrite patterns for rank improvement, bias orientation preservation, or both objectives. To address the conflicting goals, we propose a progressive fine-tuning strategy that first trains specialized LLM experts for ranking and bias tasks independently, then integrates their capabilities using the combined data to achieve both objectives. This strategy can help overcome the scarcity problem inherent in the combined dataset. By combining these strengths, BALANCE delivers superior performance in generating effective, bias orientation-preserving textual attacks, offering a robust solution to this underexplored problem.

Our experimental results demonstrate that BALANCE outperforms baseline methods in enhancing news rankings while preserving media bias orientation. The progressive unified fine-tuning framework effectively integrates task-specific components, resulting in superior performance for both ranking improvement and bias orientation maintenance. Notably, BALANCE maintains a consistent bias orientation across ideological groups, with strong performance in preserving right-leaning bias orientation on MIND-large datasets. Additionally, the attack causes negligible disruption to recommendation quality, and rewritten news that preserves bias orientation is less noticeable to users, as shown in simulated user studies. These findings highlight BALANCE's dual capability to effectively attack news RSs while maintaining bias orientation and minimizing detectability.

5.1.3 Our Contributions

In summary, the contributions of this chapter are as follows:

- We introduce textual attacks on news RSs and highlight the critical need to preserve media bias orientation, demonstrating that neglecting bias orientation risks platform detection and reduced user engagement due to ideological misalignment.
- Through empirical analyses, we show that preserving media bias orientation and improving news rankings are conflicting tasks, as rank-focused textual attacks often unintentionally alter bias orientation, establishing a fundamental trade-off in attack design.
- We propose BALANCE, an LLM-based framework for bias-aware textual attacks, using rank, bias, and combined datasets with a progressive fine-tuning strategy to balance ranking improvement and bias orientation preservation, achieving superior performance in both objectives.
- Our approach ensures stealth and practicality, with BALANCE generating subtle, bias orientation-preserving rewrites that minimally disrupt recommendation

quality and remain less noticeable to users, as validated by simulated user studies, enhancing the attack's effectiveness and discretion.

5.2 Related Work

In this section, we review the background of media bias, with a particular emphasis on its role in news RSs and the risks posed by adversarial attacks.

5.2.1 Media Bias

Media bias occurs when news outlets present information in ways that reflect their ideological, political, or economic interests, rather than providing neutral and objective coverage [96, 103]. This bias appears in several forms, including selective reporting, where certain events or perspectives are emphasized or ignored; factual cherry-picking, where specific facts are highlighted to support a particular narrative; and emotionally charged news coverage that often reflects the media sources' ideologies [37, 104]. Such bias orientation is evident across media outlets, from left- to right-leaning, and shapes how stories about societal issues are presented [37, 58, 104]. For example, McKeever et al. [75] find that biased reporting can fuel anti-immigrant sentiment in regions with shifting demographics. Media bias fosters confirmation bias, deepens societal divisions, and erodes trust in news [11, 14, 15, 27, 87]. Additionally, news RSs can worsen these effects by creating echo chambers that limit exposure to diverse viewpoints [83].

In the context of textual attacks, ignoring media bias orientation can undermine malicious strategies. Aligning manipulated news content with a platform's media bias orientation can help to avoid detection. A mismatch may flag content as inauthentic, causing the attack to fail [4]. From a user perspective, content that contradicts their beliefs is less engaging due to confirmation bias, reducing click-through rates [77]. Conversely, attackers who exploit ideological bias orientation can target platforms and users effectively, amplifying their impact by using information bubbles and group polarization [83, 106]. Understanding media bias orientation is thus critical for both offensive and defensive strategies in digital information ecosystems.

Detecting media bias orientation is challenging but essential. Traditional methods analyze reported speech, such as the use of quotes, which can create a false sense of balance or favor certain perspectives [84]. Advanced approaches use deep learning models, like recurrent neural networks (RNNs) and transformers, to identify bias orientation at the sentence or document level by capturing subtle linguistic and contextual cues [6, 48].

Existing textual attack methods often overlook ideological media bias orientation, thereby limiting their real-world effectiveness. These methods typically alter content without accounting for the ideological context of the target platform or audience, risking detection or reduced impact [10]. As media bias orientation shapes user perceptions and platform dynamics, attack strategies must incorporate bias-aware approaches to succeed. Attack research can integrate bias orientation detection into frameworks, using advancements in NLP and deep learning to align with the ideological nuances of media ecosystems [37, 56].

5.2.2 News Recommender Systems

News RSs have evolved significantly over the past two decades, driven by the exponential growth of online news and the demand for personalized content delivery. Early methods included content-based filtering, which recommended articles based on textual similarity to user profiles [85], but this approach struggled to capture dynamic user interests. Collaborative filtering was also employed, recommending articles by using patterns in user-item interactions and the behaviors of similar users [98]. However, these approaches proved less effective in the news domain due to the rapidly changing nature of news, where articles quickly become outdated.

In recent years, the advent of deep learning has profoundly transformed news RSs [63]. Early deep learning models demonstrated superior performance over traditional methods by employing techniques like sequence modeling and attention mechanisms to capture user interests [79, 126]. These models typically relied on static word embeddings, such as GloVe [86], for news representation. For instance, LSTUR [3] used a GRU network over GloVe embeddings to model user browse history, while NRMS [127] and NAML [126] integrated attention mechanisms and multi-view learning, respectively, to enhance news and user representations. A significant leap forward came with the success of pre-trained language models (LMs). By using powerful contextualized embeddings from models like BERT [23] and RoBERTa [72], subsequent research further improved recommendation accuracy. Representative works include MINER [61], which uses BERT for news encoding and a poly-attention mechanism for user representation, and PLM-NR [129], which systematically investigates various PLMs for news recommendation and enhances models like NAML, NRMS, and LSTUR for better performance. More recently, the research frontier has shifted towards adopting LLMs for even more sophisticated content understanding. Models like ONCE [71] have begun to use both closed and open-source LLMs as news encoders. Furthermore, initial studies show that LLMs particularly excel in addressing challenges like the cold-start user problem in news recommendation [165].

A critical challenge distinguishing news recommendations from other domains, such as e-commerce or movies, is the ephemeral nature of its items. The vast majority of articles encountered during inference are novel and thus unseen during the training phase [165]. This renders ID-based features largely ineffective, compelling news RSs to rely almost exclusively on semantic content to model the relationships between news and users. Consequently, this strong dependence on content makes these systems inherently vulnerable to textual adversarial attacks.

5.2.3 Adversarial Attacks on Recommender Systems

Adversarial attacks on RSs aim to manipulate model outputs—such as rankings or recommendations—by introducing carefully crafted inputs that exploit model vulnerabilities. In the context of RSs, early adversarial efforts primarily focused on collaborative filtering models through data poisoning [26, 59, 60, 124, 150], where attackers injected fake user interactions to promote or demote specific items. For example, RAPU-R [150] generates fake user-item interactions to influence recommendations based on incomplete or perturbed data. However, such attacks are less effective against modern news RSs,

which rely heavily on news content rather than solely on user interaction histories. For instance, ARG [19] proposes a reinforcement learning framework to generate fake reviews for review-based RSs, demonstrating the potential of content-level manipulation in undermining recommendation integrity.

This shift towards content-centric architectures has given rise to textual adversarial attacks, where the focus is on manipulating the content of news items themselves. Inspired by adversarial techniques from the NLP domain, recent attacks like TextRecAttack [153] apply word- and character-level perturbations (*e.g.,* TextFooler [51], DeepWordBug [30] and BERT-Attack [62]) to item text in order to influence recommendations from LLM-based systems. However, these methods often depend on access to model parameters, embedding spaces, or repeated system feedback, which is unrealistic in practical deployments where interaction budgets and system transparency are limited. Similarly, ATR-2FT [78] uses a two-phase fine-tuning approach to rewrite item descriptions; however, it requires access to RSs model parameters, and its joint training paradigm limits it to smaller LM models like OPT-350M [156]. To address the reliance on knowledge of victim RSs, such as parameters or embeddings, TextSimu [123] exploits LLMs to simulate the textual characteristics of popular items. However, because news RSs are constantly evolving, the attributes that made older news articles popular may no longer be relevant. This temporal shift limits the effectiveness of directly applying existing textual attack models to rewriting new content or enhancing its ranking in news RSs.

Besides these limitations, a crucial gap in existing work is the lack of attention to media bias orientation when crafting adversarial text. As discussed earlier, media bias orientation strongly shapes both user engagement and platform dynamics. Textual attacks that disregard ideological alignment risk detection or reduced effectiveness, as they may introduce content that conflicts with the platform's typical bias orientation or the audience's expectations. Our work addresses this limitation by proposing a new form of adversarial attack: bias-aware content rewriting. We exploit the generation capabilities of LLMs to rewrite news articles in a way that both improves their ranking in news RSs and preserves their original media bias orientation, introducing a novel and realistic threat to news RSs.

5.3 Media Bias in Attacking News Recommender Systems

In this section, we investigate the role of media bias in adversarial attacks on news RSs. We first propose the problem formulation of our "media bias-aware textual attack," then we propose an intuitive context-enhanced rewrite prototype to assess its effectiveness, and finally we conduct an empirical study to evaluate its effect and gain insights.

5.3.1 Problem Formulation

News Recommender Systems. Consider a news RS that recommends news items from a set of news articles $\mathcal{V}$ to users in a user set $\mathcal{U}$ based on users' click history. Each news article $v \in \mathcal{V}$ is associated with textual content t_v. A typical new RS consist

of the following components: (i) a news encoder that transforms news article content t_v into a news representation $\mathbf{q}_v$, and (ii) a user modeling component that processes the click history of a user $u \in \mathcal{U}$, denoted as $H_u = \{v_1, v_2, \ldots, v_n\}$, to yield a user representation $\mathbf{p}_u$. This news RSs predicts a rank/relevance score for a given news item v and user u as follows:

$$\mathcal{R}(t_v, u; f^{RS}) = f^{RS}(\mathbf{p}_u, \mathbf{q}_v), \tag{5.1}$$

where f^{RS} is a pre-trained, fixed news RS model. News items are ranked for each user in descending order of these relevance scores, and $\mathcal{R}(\cdot)$ is the rank function.

Media Bias Orientation Detector. Media bias orientation in news occurs when the content reflects ideological leanings, shaped by factors like the news outlet or the way the story is presented [96, 103]. For a news item $v \in \mathcal{V}$ with content t_v, we define its bias orientation as $m_v \in \{\text{left}, \text{center}, \text{right}\}$, based on Allsides:[1]

- **Left**: Emphasize liberal, progressive, or left-wing ideas and policies, reflecting values commonly associated with left-leaning news sources.
- **Center**: Maintain neutrality, avoiding strong political bias, or balance perspectives from both left and right viewpoints.
- **Right**: Highlight conservative, traditional, or right-wing ideas and policies, reflecting values commonly associated with right-leaning news sources.

A media bias orientation detector evaluates the bias orientation of a news item v based on its textual content t_v, assigning it a bias direction:

$$m_v \leftarrow \mathcal{M}(t_v; f^{\text{bias}}), \tag{5.2}$$

where f^{bias} typically denotes a pre-trained LM encoder, and $\mathcal{M}(\cdot)$ maps the encoded representation to a bias category.

Media Bias-aware Textual Attack. In a typical textual attack on a news RSs, a malicious news provider seeks to increase the exposure of a target news item $v_g \in \mathcal{V}$ by rewriting its original content t_g to t'_g, aiming to boost its rank:

$$\mathcal{R}(t_g{}', u; f^{RS}) < \mathcal{R}(t_g, u; f^{RS}). \tag{5.3}$$

However, such attacks may inadvertently alter the news item's bias orientation. A media bias-aware textual attack aims to improve the ranking of a target news item v_g while preserving its bias orientation. The attacker modifies t_g into t'_g to achieve a higher ranking while maintaining the original bias orientation:

$$\mathcal{R}(t_g{}', u; f^{RS}) < \mathcal{R}(t_g, u; f^{RS}) \text{ such that } m_{t_g{}'} = m_{t_g}, \tag{5.4}$$

where $m_{t_g{}'} = \mathcal{M}(t'_g; f^{\text{bias}})$ and $m_{t_g} = \mathcal{M}(t_g; f^{\text{bias}})$. This attack involves finding a transformation $t_g \rightarrow t'_g$ that enhances the ranking of v_g within $f^{RS}(\cdot)$ while ensuring the modified content retains its ideological stance, as evaluated by the bias orientation detector $\mathcal{M}(\cdot\,; f^{bias})$.

[1] https://www.allsides.com/media-bias

5.3.2 Context-Enhanced Rewrite Prototype

Recent advances in generative artificial intelligence have introduced LLMs, such as GPT [80] and Llama [112], which exhibit impressive capabilities for reasoning and problem-solving. Inspired by these advancements, we propose an intuitive approach that uses the reasoning abilities of LLMs through in-context learning (ICL) [16] to enhance news ranking in news RSs while preserving the original media bias orientation.

In an ICL rewrite attack, an LLM receives a prompt containing a task instruction and k input-output example pairs $\{(t_1, t'_1), (t_2, t'_2), \ldots, (t_k, t'_k)\}$, where each pair comprises an original news article t_i and its rewritten version t'_i. The LLM then predicts a rewritten version t'_g for a target news article t_g. This process involves two key steps: (i) the LLM implicitly approximates a function f^{ICL} that maps inputs to outputs based on the provided examples, and (ii) it generalizes this rewrite function $f^{ICL}(\cdot)$ to new target news articles t_g within the same context. Unlike traditional learning paradigms (*e.g.,* gradient descent), this approach does not involve explicit parameter updates; instead, the LLM relies on its pre-trained parameters to infer patterns learned during training. Below is an example of the ICL prompt used to conduct the rewrite attack:

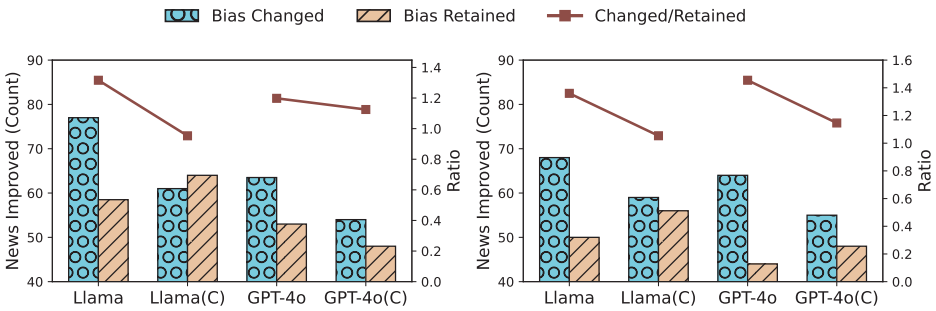
> **Task**: *You are an expert in optimizing news content for recommender systems. Your goal is to rewrite the provided news title and abstract to improve their ranking in news recommender systems while preserving the original media bias orientation (left, center, or right).*
> **Bias Definition:** $<\mathcal{B}>$
> **Rewrite Examples**: *The following k pairs of news articles (original and rewritten title and abstract) can be used as references during the rewrite process.* $<\mathcal{R}>$
> **Original News**: $<t_g>$

Here, $<\mathcal{B}>$ refers to the detailed bias description in Section 5.3.1, $<\mathcal{R}>$ represents k pairs of news articles before and after rewriting, and $<t_g>$ denotes the target news article to be rewritten.

To generate rewrite examples, we use an LLM (*i.e.,* Llama 3.1) to transform a news article t_v into a rewritten version t'_v. We then filter the resulting (t_v, t'_v) pairs to include only those where t'_v achieves a higher ranking in the news RS while preserving the original media bias orientation. Specifically, we select articles originally ranked below the top 50 that, after rewriting, climb into the top 50, indicating a successful rank improvement. The bias orientation of the rewritten articles is verified using a bias orientation detector, with details provided in Section 5.5. For each target news article t_g, we randomly select k ($k = 5$ or 10, as suggested by [16, 78]) pairs of rewrite examples $<\mathcal{R}>$ that share the same bias orientation as t_g to serve as context and promote the rewrite on t_g.

5.3.3 Empirical Analyses

In this section, we investigate the impact of ignoring media bias orientation during textual attacks by conducting empirical analyses on two real-world news recommendation datasets: MIND-small and MIND-large. These datasets include both click data and news



(a) MIND-small (NAML). (b) MIND-large (NAML).

Figure 5.1: Comparison of bias-changed versus bias-retained news articles for improved news ranking.

content for all articles. We selected NAML, a widely used news RS, as the target model and randomly sampled 300 news articles to form the target set $\mathcal{V}_g$. Four LLM-based attack methods were employed: two without context information—Llama 3.1 (abbreviated as Llama) and GPT-4o—which rewrote news articles to enhance their ranking while preserving media bias orientation; and two with context information—Llama(C) and GPT-4o(C). Detailed experimental settings are provided in Section 5.5. After collecting the attacked news content, we evaluated the average rank across all users post-attack. A media bias orientation detector classified the original articles' bias orientation (left, center, or right) based on their textual content, and the same process was applied to the rewritten news.

We performed a statistical analysis on rewritten news articles that successfully improved their rank. For each attack method, we divided the rank-improved articles into two groups:

- **Bias Changed**: News articles whose bias orientation shifted after the attack.
- **Bias Retained**: News articles whose bias orientation remained unchanged.

We calculated the number of articles in each group, and the results for all four attack methods are presented in Figure 5.1. Three key observations emerged from this analysis.

Observation 1. *Textual attacks aimed at rank improvement tend to alter media bias orientation.* As illustrated in Figure 5.1a and 5.1b, for both pure LLM-based methods (Llama and GPT-4o) and most context-enhanced methods (Llama(C) and GPT-4o(C)), the number of rank-improved articles with changed bias orientations consistently exceeds those with retained bias orientation. This suggests that rewrite attacks, even when not explicitly designed to modify bias orientation, tend to result in a shift in media bias orientation.

Observation 2. *Preserving media bias orientation limits rank improvement in rewrite attacks.* As shown in Figure 5.1a and 5.1b, the number of rank-improved articles with retained bias orientation is consistently lower than those with changed bias orientation across pure LLM-based and most context-enhanced methods. This indicates that improving rank through rewriting and maintaining media bias orientation are potentially conflicting objectives, necessitating a strategy to balance or integrate these tasks.

Observation 3. *LLM-based methods show potential in mitigating unintended bias alterations.* In Figure 5.1a and 5.1b, the ratio of articles with changed bias orientation to those with retained bias orientation is lower for context-enhanced methods than for pure LLM-based methods. Notably, on the MIND-small dataset using the NAML model, the context-enhanced Llama(C) method resulted in more articles retaining their original bias orientation than changing it. This demonstrates that LLMs, especially when augmented with contextual information, have the potential to reduce unintended shifts in media bias orientation during textual attacks. However, context-enhanced methods, such as Llama(C) and GPT-4o(C), do not consistently outperform pure LLM-based methods in improving articles. This happens because news content changes rapidly, and older high-ranking articles may lose their position over time. With only limited ICL examples, LLMs struggle to understand these shifting rankings, which hinders the performance of context-enhanced approaches. Therefore, a more effective approach is needed to integrate the tasks of rank improvement and maintaining media bias orientation.

Based on the above analyses, we propose the following assumption, which lays the foundation for our textual attack method:

Assumption 1. *Achieving a successful textual attack and maintaining bias orientation consistency are two conflicting tasks. LLM-based methods have the potential to balance these tasks, which could be attributed to their reasoning capability in understanding the task requirements.*

Building on this assumption, our method uses LLMs to design an approach that achieves textual attack and maintains bias orientation consistency, as detailed in the subsequent sections.

5.4 Methodology

This section details the proposed BALANCE framework, illustrated in Figure 5.3, for media bias-aware textual attacks. Our approach effectively rewrites news content to improve its ranking while preserving its original media bias orientation. We begin by describing the process of exploring diverse ranking and Bias data to build the datasets. Next, we present the rewrite pattern learning process, which fine-tunes LLMs to effectively rewrite news based on the extracted data. We then introduce the unified attack pipeline, which further fine-tunes LLMs to address both ranking and bias objectives. Finally, we describe the attack process, demonstrating how our model targets specific news content.

5.4.1 Data Preparation

As discussed in Section 5.3.2, exploring rewritten examples $\mathcal{R}$ that achieve both higher rankings and maintain the same bias orientation as the original news (termed "combined data") could ideally yield a high-quality dataset. This dataset includes both patterns, and effective optimization on this data could perfectly align with our task requirements. However, fine-tuning LLMs on combined data faces limitations due to the following challenges: (i) According to Observation 1, successful textual attacks often alter bias

orientation, resulting in a limited number of combined data samples. Fine-tuning on such a small dataset typically fails to teach LLMs how to rewrite news effectively, leading to model training collapse. (ii) Per Observation 2, preserving the media bias orientation constrains rank improvement, which limits the attack's effectiveness and may result in failed textual attacks. These limitations are further confirmed in our experimental section below.

To address these challenges, we examine the rewriting process in detail. See Figure 5.2 (left), where we consider a left-biased original news article $t_e \in \mathcal{V}_e$. Our explorer prompts an LLM to generate multiple rewritten versions of t_e. In the explored rewritten space (see Figure 5.2, right), rewritten news articles with higher rankings (red area) or with the same bias orientation (left bias, blue area) are more abundant, as they only need to excel in one aspect. However, most higher-ranked rewritten news struggle to retain the left media bias orientation, and most same-bias orientation rewritten news fail to improve rankings, resulting in a limited number of rewritten news samples qualifying for the combined dataset (orange area). Given the challenges of dataset scarcity and quality, we adopt a progressive approach instead of conditioning on both tasks simultaneously, enabling better usage of the explored data.

Rewrite Exploration. To investigate the impact of news text on news ranking and bias orientation, we conducted a rewrite exploration phase to generate rewrite samples. Using recent advancements in LLM-based data augmentation [24, 160], we developed an LLM-driven approach to augment data, focusing on textual factors that may influence media bias orientation and article ranking within the news domain [2, 114, 142]. Inspired by this, our *explorer* uses an LLM to generate rewritten news articles by manipulating four key dimensions:

- **Bias orientation**: Highlighting specific story elements to influence reader perception, framing the narrative from a distinct ideological viewpoint (*e.g.,* "left," "center," or "right").
- **Writing style**: Modifying the tone and formality of the text (*e.g.,* "formal" to "conversational").
- **Sentiment polarity**: Adjusting the emotional tone of the narrative (*e.g.,* "positive" to "negative").
- **Author persona**: Varying the author's perspective or stance (*e.g.,* "objective" to "opinionated").

Further details on these dimensions are accessible in our repository[2].

For a given news article $t_e \in \mathcal{V}_e$, the *explorer* generates a set of rewritten variants as follows:

$$\mathcal{S}_e = \mathcal{T}_{bias}(t_e) \cup \mathcal{T}_{style}(t_e) \cup \mathcal{T}_{sentiment}(t_e) \cup \mathcal{T}_{persona}(t_e), \tag{5.5}$$

where $\mathcal{T}_{\text{bias}}(\cdot)$, $\mathcal{T}_{\text{style}}(\cdot)$, $\mathcal{T}_{\text{sentiment}}(\cdot)$, and $\mathcal{T}_{\text{persona}}(\cdot)$ represent transformations that modify the original text t_e according to bias orientation, writing style, sentiment polarity, and

[2] `https://github.com/Go0day/BALANCE`.

author persona, respectively. Here, $\mathcal{S}_e$ denotes the collection of all rewritten versions of t_e, with $\mathcal{V}_e$ representing the set of news articles drawn from the training dataset. To illustrate, we provide an example prompt used by the *explorer* to incorporate various types of media bias orientation:

> **Task**: *Rewrite the given news title and abstract into 3 different versions, to exhibit distinct media bias orientation types.*
>
> **bias orientation definition**: *Left, Center and Right.* $<\mathcal{B}>$
>
> **Original News**: $<t_e>$

In this context, $<\mathcal{B}>$ refers to the detailed bias orientation definitions outlined in Section 5.3.1, and $<t_e>$ represents the input news article.

Rank Data. For Rank data, we select original news articles $t_e \in \mathcal{V}_e$ with $\mathcal{R}(t_e, u; f^{RS}) > K$, where K is the ranking threshold, indicating they are initially ranked outside the top K. The Rank data consists of pairs (t_e, s_e^{rank}) where the rewritten version $s_e \in \mathcal{S}_e$ achieves a ranking within the top K, *i.e.,* $\mathcal{R}(s_e, u; f^{RS}) \leq K$, indicating a successful rank improvement. For example, with $K = 50$, if an original article t_e has a rank of 100 (> 50) and a rewritten version s_e^a achieves a rank of 20 (≤ 50), then $s_e^a \in S_{e^+}^{\text{rank}}$. Conversely, if another version s_e^b for the same t_e has a rank of 60 (> 50), then $s_e^b \in S_{e^-}^{\text{rank}}$, as it remains outside the top 50 and is less visible to users. The Rank data collection $(t_e, S_{e^+}^{\text{rank}}, S_{e^-}^{\text{rank}})$ is defined as:

$$\begin{aligned} S_{e^+}^{\text{rank}} &= \{s_e \in \mathcal{S}_e \mid \mathcal{R}(s_e, u; f^{RS}) \leq K\} \\ S_{e^-}^{\text{rank}} &= \{s_e \in \mathcal{S}_e \mid \mathcal{R}(s_e, u; f^{RS}) > K\}, \end{aligned} \tag{5.6}$$

where $\mathcal{R}(\cdot; f^{RS})$ is defined in Section 5.3.1, and K is a hyperparameter. $S_{e^+}^{\text{rank}}$ contains rewritten versions s_e that successfully improve ranking, while $S_{e^-}^{\text{rank}}$ includes those that do not.

Bias Data. The Bias data includes pairs (t_e, s_e^{bias}) where the rewritten version $s_e \in \mathcal{S}_e$ preserves the media bias orientation of the original news article $t_e \in \mathcal{V}_e$. For example, if t_e has a left-leaning bias, *i.e.,* $\mathcal{M}(t_e; f^{\text{bias}}) = \text{left}$, and a rewritten version s_e^a has $\mathcal{M}(s_e^a; f^{\text{bias}}) = \text{left}$, then $s_e^a \in S_{e^+}^{\text{bias}}$. If another version s_e^b has $\mathcal{M}(s_e^b; f^{\text{bias}}) = \text{center/right}$, then $s_e^b \in S_{e^-}^{\text{bias}}$ because the bias orientation differs. The Bias data collection $(t_e, S_{e^+}^{\text{bias}}, S_{e^-}^{bias})$ is defined as:

$$\begin{aligned} S_{e^+}^{\text{bias}} &= \{s_e \in \mathcal{S}_e \mid \mathcal{M}(s_e; f^{\text{bias}}) = \mathcal{M}(t_e; f^{\text{bias}})\} \\ S_{e^-}^{\text{bias}} &= \{s_e \in \mathcal{S}_e \mid \mathcal{M}(s_e; f^{\text{bias}}) \neq \mathcal{M}(t_e; f^{\text{bias}})\}, \end{aligned} \tag{5.7}$$

where $\mathcal{M}(\cdot; f^{\text{bias}})$ is the media bias orientation detector that assigns a bias orientation category $m \in \{\text{left}, \text{center}, \text{right}\}$, as defined in Section 5.3.1. $S_{e^+}^{\text{bias}}$ includes rewritten versions s_e that preserve the bias orientation of the original article t_e, while $S_{e^-}^{\text{bias}}$ contains those with a different bias orientation.

Combined Data. The combined data comprises pairs (t_e, s_e^{comb}) where the rewritten version $s_e \in \mathcal{S}_e$ both rises to the top K ranks and preserves the media bias orientation of

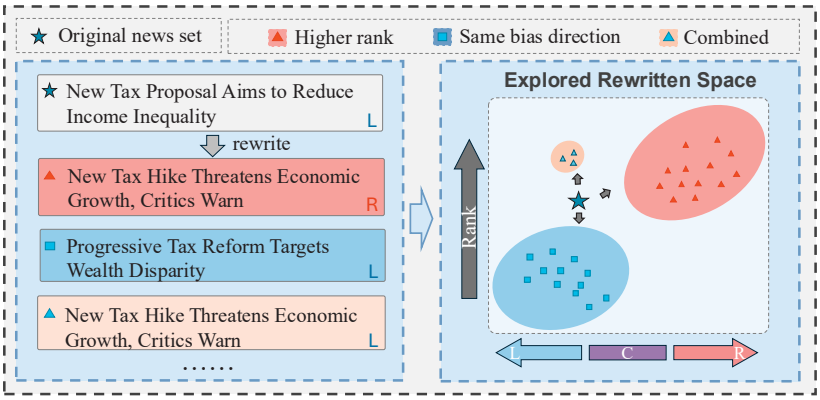


Figure 5.2: Data preparation process for the BALANCE framework. L, R, C represent left, right, and center bias orientations, respectively. The left part shows a left-biased original news article as an example. The right part displays higher-ranked news (red), same-bias news (blue), and overlapping samples achieving both (orange), with larger areas indicating more samples.

the original news article $t_e \in \mathcal{V}_e$ with $\mathcal{R}(t_e, u; f^{RS}) > K$. The tuple $(t_e, S_{e^+}^{\text{comb}}, S_{e^-}^{\text{comb}})$ is defined as:

$$\begin{aligned} S_{e^+}^{\text{comb}} &= \{s_e \in \mathcal{S}_e \mid \mathcal{R}(s_e, u; f^{RS}) \leq K \text{ and } \mathcal{M}(s_e; f^{\text{bias}}) = \mathcal{M}(t_e; f^{\text{bias}})\}, \\ S_{e^-}^{\text{comb}} &= \{s_e \in \mathcal{S}_e \mid \mathcal{R}(s_e, u; f^{RS}) > K \text{ and } \mathcal{M}(s_e; f^{\text{bias}}) \neq \mathcal{M}(t_e; f^{\text{bias}})\}. \end{aligned} \tag{5.8}$$

Here, $S_{e^+}^{\text{comb}}$ includes rewritten versions s_e that achieve a top-K rank and retain the bias orientation of t_e, aligning with the bias-aware media bias textual attack framework, while $S_{e^-}^{\text{comb}}$ contains those that fail to meet these conditions.

5.4.2 Progressive Fine-tuning Approach

With the prepared datasets described in Section 5.4.1 above, we aim to fine-tune a LLM to rewrite news articles, promoting their ranking in news RSs while maintaining their original media bias orientation (left, center, or right). However, directly fine-tuning the LLM on these datasets poses challenges due to their inherent complexities and limitations. For instance, a rewritten article that successfully preserves bias orientation might fail to improve ranking, and vice versa, leading to potential confusion in the training data. To illustrate, consider an original left-leaning news article t_e with a rank of 100. A rewritten version s_e^a might achieve a rank of 20 but shift the bias orientation to center, while another version s_e^b retains the left bias but only reaches a rank of 60. The intersection of successful ranking improvement and bias orientation maintenance—represented by the Combined data—is thus limited in quantity and quality, as noted in Section 5.3.2.

To overcome these challenges, we propose a **progressive fine-tuning approach**, as illustrated in Figure 5.3. This method first fine-tunes the LLM on the Rank data and Bias data separately to create specialized experts: a rank expert excelling at ranking improvement and a bias expert adept at maintaining bias orientation. Subsequently,

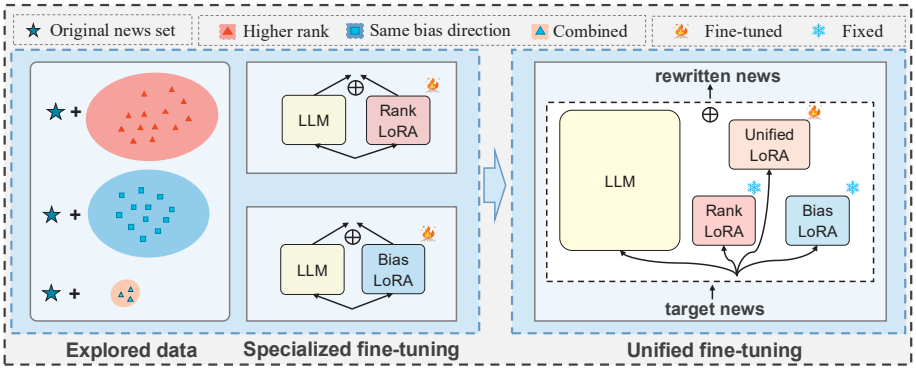


Figure 5.3: Overview of the progressive fine-tuning approach in the BALANCE framework, showing separate training on Rank data for ranking improvement and Bias data for bias orientation maintenance, followed by unified fine-tuning on Combined data to combine and enhance two objectives in a single model.

we use the high-quality but scarce combined data to balance and enhance these two objectives in a unified model.

Specialized Fine-tuning. In the first phase, we employ Low-Rank Adaptation (LoRA) to fine-tune the LLM on Rank data and Bias data independently, creating specialized experts. LoRA is a parameter-efficient fine-tuning technique that adapts the pre-trained model by introducing low-rank weight updates, reducing computational overhead while maintaining performance [41].

For the rank and bias datasets, we use direct preference optimization (DPO) [91] to train the LLM, as DPO excels at learning from pairwise preference data, effectively distinguishing between successful and unsuccessful rewrites. The instruction template for fine-tuning is as follows:

> **Task**: You are an expert in news content optimization. Your goal is to rewrite the given news title and abstract to achieve [specific goal]. The output should maintain the original core information while adhering to the goals and length constraints. $<t_e>$
>
> **Chosen rewrite**: s_e^+
>
> **Rejected rewrite**: s_e^-

For Rank data, the objective is to maximize ranking improvement, where s_e^+ belongs to $S_{e^+}^{\text{rank}}$ and s_e^- belongs to $S_{e^-}^{\text{rank}}$. For Bias data, the objective is to preserve the original media bias orientation, where s_e^+ belongs to $S_{e^+}^{\text{bias}}$ and s_e^- belongs to $S_{e^-}^{\text{bias}}$. The DPO loss function is defined as:

$$L_{\text{DPO}}(\theta) = -\mathbb{E}_{(t_e, s_e^+, s_e^-)} \left[\log \sigma \left(\beta \log \frac{p_\theta(s_e^+|t_e)}{p_{\text{ref}}(s_e^+|t_e)} - \beta \log \frac{p_\theta(s_e^-|t_e)}{p_{\text{ref}}(s_e^-|t_e)} \right) \right], \quad (5.9)$$

where $\sigma(\cdot)$ is the sigmoid function, β is a temperature parameter controlling preference strength, $p_\theta(s_e|t_e)$ is the probability of generating rewrite s_e given t_e under the model

with parameters θ, $p_{\text{ref}}(s_e|t_e)$ is the probability under the pre-trained reference model. This loss encourages the model to favor the chosen rewrite s_e^+ (*e.g.,* a rank-improved or bias-maintained version) over the rejected rewrite s_e^- (*e.g.,* a version failing the respective goal).

LoRA Adaptation. In LoRA, the pre-trained weight matrix W is updated with a low-rank decomposition BA, where B and A are low-rank matrices. The adapted weight is thus:

$$W' = W + BA. \tag{5.10}$$

For Rank data, we fine-tune the LLM to obtain LoRA parameters B^{rank} and A^{rank}, yielding a rank expert model. The LoRA merge weight is set to 1 in this work, meaning the full contribution of the learned adaptation is applied. Similarly, for Bias data, we obtain B^{bias} and A^{bias}, resulting in a bias expert model. These parameters capture task-specific adaptations, enabling the LLM to specialize in each domain efficiently.

Unified Fine-tuning. After obtaining the rank expert ($B^{\text{rank}}A^{\text{rank}}$) and bias expert ($B^{\text{bias}}A^{\text{bias}}$), we proceed to the second phase: unified fine-tuning. In this phase, we first merge the LoRA parameters of both experts into the base LLM, integrating their specialized capabilities into the model. The merged model incorporates adaptations for both ranking improvement and bias maintenance, with the weights defined as:

$$W_{\text{merged}} = W + B^{\text{rank}}A^{\text{rank}} + B^{\text{bias}}A^{\text{bias}}, \tag{5.11}$$

where W represents the original pre-trained weight matrix of the LLM, and $B^{\text{rank}}A^{\text{rank}}$ and $B^{\text{bias}}A^{\text{bias}}$ are the LoRA adaptations from the rank and bias experts, respectively.

Next, we introduce a new set of trainable LoRA parameters, B^{unified} and A^{unified}, to fine-tune the merged model using the high-quality but quantity-limited Combined data. During this step, only B^{unified} and A^{unified} are updated, while the merged parameters W_{merged} remain fixed. The unified model's adapted weights are thus:

$$W_{\text{unified}} = W_{\text{merged}} + B^{\text{unified}}A^{\text{unified}}. \tag{5.12}$$

We fine-tune $B^{\text{unified}}A^{\text{unified}}$ using the combined data with the DPO loss, which prioritizes rewrites that simultaneously improve ranking and maintain the original media bias orientation. This process uses the small but high-quality combined data to enhance the model's performance on both tasks. Therefore, for the final rewrite attack, we take a target news article t_g and use the fine-tuned unified model to create a new version t'_g. This rewritten article is crafted to rank higher in the news RS while keeping the same bias orientation as the original. The rewritten t'_g is then submitted to the news RS, boosting its visibility without changing its ideological stance, thus achieving our attack goals.

5.4.3 Rationale for Unified Fine-tuning

The unified fine-tuning enhances both tasks through the following mechanisms:

(1) **Knowledge Transfer**: Merging $B^{\text{rank}}A^{\text{rank}}$ and $B^{\text{bias}}A^{\text{bias}}$ into the base LLM equips the unified model with pre-learned expertise in ranking improvement and bias orientation maintenance. This ensures the model begins with strong, task-specific capabilities before further refinement.

(2) **Objective Balancing**: The Combined data consists of rewrites that achieve both objectives (*e.g.,* s_e with $\mathcal{R}(s_e, u; f^{RS}) \leq K$ and $\mathcal{M}(s_e; f^{\text{bias}}) = \mathcal{M}(t_e; f^{\text{bias}})$). Fine-tuning $B^{\text{unified}}A^{\text{unified}}$ on these samples adjusts the model to prioritize rewrites that excel in both ranking and bias preservation, enhancing its dual-task performance.

(3) **Refinement via Adaptation**: Training $B^{\text{unified}}A^{\text{unified}}$ specifically on Combined data allows the model to optimize the interplay between ranking and bias orientation maintenance. For example, it can further improve ranking in bias orientation-preserving rewrites or enhance bias orientation maintenance in high-ranking rewrites, surpassing the performance of the individual experts.

Formally, the unified model's output benefits from the merged adaptations of the rank and bias experts, while the DPO loss increases the probability $p_\theta(s_e^+|t_e)$ for rewrites s_e^+ that excel in both tasks. This approach combines the strengths of the merged experts and the trainable unified parameters to achieve superior overall performance.

5.4.4 The BALANCE Framework

Our BALANCE is a media bias-aware textual attack framework that explicitly models the media bias factor overlooked by most existing methods. It begins by using an LLM-based explorer to generate rewritten news across multiple dimensions, forming a pool of candidate rewrites. These rewrites are then filtered into three task-specific datasets: Rank data, Bias data, and Combined data. We train ranking and bias experts separately, merge their adaptations, and finish with a unified fine-tuning on the Combined data. This progressive fine-tuning pipeline ensures that BALANCE generates rewrites that enhance news ranking in RSs while preserving the original media bias orientation, enabling targeted, realistic attacks in news RSs.

5.5 Experiments

In this section, we present a series of experiments designed to evaluate the attack performance of our proposed method, BALANCE . These experiments aim to address the following research questions:

RQ4.1 How does the proposed approach BALANCE compare to existing models when attacking news RSs?

RQ4.2 What is the impact of individual components of BALANCE (*e.g.,* the fine-tuning data and fine-tuning strategies) on the effectiveness of the proposed attack method?

RQ4.3 How does accounting for news bias affect different ideological groups in textual attacks on news RSs?

RQ4.4 How does our textual attack impact broader dimensions (*e.g.,* recommendation performance, noticeability) of news RSs?

Table 5.1: Statistics of the datasets used.

Dataset	#News	#Categories	$\lvert\mathcal{V}_e\rvert$	$\lvert\mathcal{V}_g\rvert$	$\frac{\lvert\mathcal{V}_e\rvert}{\#\text{News}}$
MIND-small	65,238	18	1,500	300	2.29%
MIND-large	161,013	20	1,500	300	0.93%

5.5.1 Experimental Setup

Dataset

We conduct experiments on the MIND dataset [133], a publicly available English news recommendation dataset from Microsoft News.[3] The dataset has two versions: MIND-large and MIND-small. MIND-large includes approximately 160,000 English news articles and over 15 million impression logs from 1 million users. MIND-small is a subset of MIND-large, randomly sampling 50,000 users and their behavior logs. The dataset's statistics are detailed in Table 5.1.

Target News

Previous textual attack methods for RSs [78, 153] typically select random target items and apply word perturbation or rewrite learning directly to them. In the rapidly evolving news domain, these methods lack robustness and often fail to effectively rewrite or promote unseen articles. To address this limitation, we select the attack training news set $\mathcal{V}_e$ from articles appearing in the MIND training impressions and the target news set $\mathcal{V}_g$ from articles in the MIND test impressions. To enable textual, bias-aware media bias attacks, we focus exclusively on the *Politics* news category, as other categories—such as *Weather* news—rarely exhibit media ideology. To ensure that our textual attack generalizes across news articles with varying popularity levels [73, 165], we randomly select 500 news items for training and 100 news items for testing from each of three popularity ranges, determined by frequency: 0–20% (least popular), 40–60% (medium popularity), and 80–100% (most popular). This process yields $|\mathcal{V}_e| = 1,500$ and $|\mathcal{V}_g| = 300$, respectively. The ideological distribution of these sets is as follows: for $\mathcal{V}_e$, the proportions of left, center, and right-leaning news are 26.2%, 35.73%, and 38.06%, respectively; for $\mathcal{V}_g$, the proportions are 24.33%, 36.33%, and 39.33%, respectively, indicating more right-leaning articles in the MIND dataset.

News Recommender Systems

For our study, we assess three widely used news RSs as victim models: NAML [126], NRMS [127], and LSTUR [3]. Consistent with prior work [129, 165], we re-implement these models using the BERT-base framework as the news encoder, with parameters optimized based on recommendation signals. Below, we provide a concise overview of their architectures:

- **NAML** [126]: NAML constructs news representations by combining titles and abstracts. It employs a multi-view learning strategy to synthesize features from

[3] Accessible at `https://msnews.github.io/`

titles, article bodies, categories, and subcategories. User representations are generated through an attention mechanism that processes the user's browsing history.

- **NRMS** [127]: NRMS relies solely on news titles to build news representations. A multi-head self-attention mechanism extracts semantic features from these titles, while user representations are derived by applying self-attention to the sequence of articles a user has browsed.
- **LSTUR** [3]: LSTUR integrates news titles and topic categories to create news representations. It uses a GRU network to model user representations from browsing history, effectively capturing both sustained and transient user interests.

Compared Methods

We compare our method, BALANCE, with seven LLM-based baselines, evaluating both attack performance and bias orientation consistency. Shilling attacks are excluded from consideration, as they involve generating fake users and ratings rather than manipulating textual content, which is beyond the scope of this study. Similarly, adversarial textual attack methods from NLP tasks, such as Deepwordbug and BertAttack in TextRecAttack [153], are omitted. These methods depend on repeated feedback from a RS for each target news item $t_g \in \mathcal{V}_g$, an approach impractical for our target news set $\mathcal{V}_g$, where such feedback is unavailable.

- **GPT-4o** and **Llama 3.1** (without context). Following the implementation in [153], we prompt these LLMs to rewrite news items to improve their ranking compared to the original news while maintaining the news direction, without incorporating contextual data. This method relies solely on the LLM to generate rewritten content based on the provided prompt.
- **GPT-4o** and **Llama 3.1** (with context). Based on the methodology in [78, 123] (see Section 5.3.2), we provide these LLMs with pairs of original and rewritten news articles as context. These pairs have demonstrated effectiveness in promoting news ranking while preserving media bias orientation. The LLMs use this context to enhance ranking and bias orientation-preserving performance during rewriting.
- **Llama-sft**. We conducted supervised fine-tuning of Llama on a dataset of successfully rewritten news articles (pairs of original and successfully rewritten news). This process embeds effective rewriting strategies into the LLM, excluding unsuccessful rewrite samples. The fine-tuned model then rewrites news items, aiming to boost the ranking of target news.
- **ATR-2FT** [78]. ATR-2FT, a recent textual attack method, employs a fine-tuned OPT-350M language model [156]. To adapt it to our setting, we isolate the news text encoder from the RS and align the LM embeddings with those of the encoder. The attack is optimized using promotion and text generation losses. For fair comparison, ATR-2FT is trained on our sampled news dataset $\mathcal{V}_e$ and tested on $\mathcal{V}_g$.

- **LANCE** [164]. LANCE, a variant of our method, is trained solely on Rank data (see Section 5.4.1), including original news, successfully rewritten news, and failed rewrites. Using DPO, it learns from comparison data to refine rewriting strategies. LANCE corresponds to the *w/ Rank* configuration in the subsequent section.

Implementation Details

In this work, we use the Llama-3.1-8B model (Llama), an open-source model optimized for reasoning, to generate rewritten news content and build the training set $\mathcal{V}_e$. For the progress fine-tuning component, we employ Llama in textual attack scenarios. Both the attack models and news recommender systems (RS) are implemented using PyTorch. We apply LoRA [41] with a regularization parameter $\beta = 0.1$ during DPO fine-tuning. The learning rate is chosen from the set {1e-5, 3e-5, 5e-5, 1e-4, 3e-4, 5e-4}, and fine-tuning is conducted over 3 epochs. The hyperparameters for the news RS are aligned with those established in [165]. Training is conducted on three NVIDIA RTX A6000 GPUs, each equipped with 48GB of memory.

Evaluation Metrics

To assess the effectiveness of our textual attack on news RSs, we employ a suite of metrics that evaluate both the attack's success in promoting target news and its ability to preserve the original media bias orientation. These metrics are categorized into attack performance and bias preservation, as detailed below:
Attack performance. We evaluate the attack's ability using two metrics:

(i) **Boost Success Rate (BSR):** This metric measures the proportion of (user, target news) pairs where the rank of the target news article improves after the attack. A rank is considered improved if its numerical value decreases (*e.g.,* from rank 5 to rank 3), indicating a higher position in the recommendation list. Formally, it is defined as:

$$\text{BSR} = \frac{1}{|\mathcal{U}| \cdot |\mathcal{V}_g|} \sum_{u \in \mathcal{U}} \sum_{t_g \in \mathcal{V}_g} \mathbb{I}[\mathcal{R}(t'_g, u; f^{RS}) < \mathcal{R}(t_g, u; f^{RS})], \tag{5.13}$$

where $\mathcal{U}$ is the set of users, $\mathcal{V}_g$ is the set of target news articles, $\mathcal{R}(t_g, u; f^{RS})$ and $\mathcal{R}(t'_g, u; f^{RS})$ are the ranks of news article v for user u before and after the textual attack, respectively, and $\mathbb{I}$ is the indicator function (1 if true, 0 otherwise).

(ii) **Exposure@K (Expo@K):** This metric calculates the proportion of users who have at least one target news article in their top-K recommendations after the attack, reflecting the visibility of the targeted content. It is expressed as:

$$\text{Expo@}K = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbb{I}[\exists t_g \in \mathcal{V}_g : \mathcal{R}(t'_g, u; f^{RS}) \leq K]. \tag{5.14}$$

Bias Orientation Preservation. To evaluate the preservation of media bias orientation in rewritten news articles, we employ PoliticalBiasBERT [6], a robust article-level bias

orientation detector trained on a balanced dataset of 34,737 manually annotated news articles labeled for political ideology (left, center, or right), as described in Section 5.3.1. We use this model to assess the ideological consistency of rewritten news articles through the following metrics:

(i) **Bias Consistency Ratio of Title (BiasCR-T)**: This metric measures the proportion of rewritten news articles whose titles retain the original political bias orientation (left, center, or right) as predicted by PoliticalBiasBERT. Let $m^{\mathrm{T}}_{t'_g}$ and $m^{\mathrm{T}}_{t_g}$ denote the predicted bias orientation of the original and rewritten titles for article i, respectively. BiasCR-T is formulated as:

$$\text{BiasCR-T} = \frac{1}{|\mathcal{V}_g|} \sum_{t_g \in \mathcal{V}_g} \mathbb{I}\left(m^{\mathrm{T}}_{t'_g} = m^{\mathrm{T}}_{t_g}\right), \tag{5.15}$$

where $\mathcal{V}_g$ is the set of target news articles.

(ii) **Bias Consistency Ratio of Title and Abstract (BiasCR-TA)**: This metric extends BiasCR-T to consider both the title and abstract of news articles. Using PoliticalBiasBERT, we predict the combined bias orientation of the original and rewritten title-abstract pairs. BiasCR-TA is defined as:

$$\text{BiasCR-TA} = \frac{1}{|\mathcal{V}_g|} \sum_{t_g \in \mathcal{V}_g} \mathbb{I}\left(m^{\mathrm{TA}}_{t'_g} = m^{\mathrm{TA}}_{t'_g}\right), \tag{5.16}$$

where $m^{\mathrm{TA}}_{t'_g}$ and $m^{\mathrm{TA}}_{t'_g}$ represent the bias orientation of the original and rewritten title-abstract pairs, respectively.

5.5.2 Performance Comparison (RQ4.1)

We compare our BALANCE with several baseline rewriting methods, as outlined in Section 5.5.1, to evaluate their performance in attacking news RSs while maintaining media bias orientation. The baselines comprise seven methods: four non-fine-tuned approaches (GPT-4o and Llama 3.1, each with and without context) and three fine-tuned approaches (ATR-2FT, Llama-sft, and LANCE). The results are presented in Table 5.2. Below, we outline the following key observations:

Attack Performance

We evaluate the ability of BALANCE and baseline methods to enhance rewritten news rankings in news RSs. The following observations highlight their attack performance:

- The fine-tuned methods (ATR-2FT, Llama-sft, LANCE, and BALANCE) generally exhibit superior performance in attacking news RSs. This suggests that obtaining attack samples and fine-tuning them into the LLM is an effective strategy. Non-fine-tuned methods struggle to achieve comparable performance, suggesting that sufficient guidance is critical for effective rewriting.

Table 5.2: Performance comparison of rewriting methods on the MIND-small and MIND-large datasets across NAML, NRMS, and LSTUR recommender systems, evaluating attack performance (BSR, Expo) and bias orientation preservation (BiasCR-T, BiasCR-TA). Column 'Ctx?' indicates whether contextual information is provided (Y) or not (N). Bold and underlined values indicate the best and second-best performances.

			MIND-small				MIND-large			
		Ctx?	BSR	Expo	Bias-CR-T	Bias-CR-TA	BSR	Expo	Bias-CR-T	Bias-CR-TA
NAML	Original	-	-	0.4357	-	-	-	0.0686	-	-
	GPT-4o	Y	0.3547	0.2527	0.4333	0.4600	0.4153	0.0359	0.4044	0.4256
		N	0.4020	0.3574	0.3976	0.4233	0.4166	0.0570	0.3976	0.4233
	Llama	Y	0.4320	0.3158	0.4367	**0.4889**	0.4280	0.0167	0.3600	0.4155
		N	0.4568	0.3827	0.3333	0.4122	0.4400	0.0226	0.3333	0.4122
	ATR-2FT	Y	0.5433	0.6732	0.3667	0.2737	0.4479	0.0597	0.3233	0.4043
	Llama-sft	Y	0.5197	0.4307	0.3697	0.3864	0.4499	0.0578	0.3110	0.3913
	LANCE	Y	0.6656	0.8919	0.4407	0.2746	0.5037	0.0739	0.3087	0.3255
	BALANCE	Y	**0.8607**	**0.9671**	**0.5933**	0.4622	**0.5220**	**0.0891**	**0.5800**	**0.4333**
NRMS	Original	-	-	0.2586	-	-	-	0.1681	-	-
	GPT-4o	Y	0.4014	0.1104	0.4033	0.4267	0.3714	0.0821	0.4038	0.4344
		N	0.3860	0.1281	0.3976	0.4233	0.3808	0.1079	0.3976	0.4233
	Llama	Y	0.4049	0.1304	0.4300	0.4167	0.3791	0.0271	0.4100	**0.4389**
		N	0.4366	0.1531	0.3333	0.4122	0.3959	0.0954	0.3333	0.4122
	ATR-2FT	Y	0.4667	0.1545	0.3885	0.3667	0.4143	0.1134	0.3937	0.3653
	Llama-sft	Y	0.4394	0.1700	0.3933	0.4233	0.3734	0.0037	0.3833	0.3633
	LANCE	Y	0.5680	0.3255	0.5433	0.4567	0.5290	0.1688	0.4885	0.3585
	BALANCE	Y	**0.5953**	**0.4431**	**0.6733**	**0.5467**	**0.5615**	**0.1812**	**0.5604**	0.4347
LSTUR	Original	-	-	0.0018	-	-	-	0.0933	-	-
	GPT-4o	Y	0.4175	0.0044	0.4274	0.4289	0.4185	0.0382	0.4033	0.4356
		N	0.3962	0.0034	0.3976	0.4233	0.4154	0.0490	0.3976	0.4233
	Llama	Y	0.4474	0.0054	0.3967	0.4322	0.4201	0.0350	0.4133	0.4245
		N	0.4566	0.0045	0.3333	0.4122	0.4377	0.0435	0.3333	0.4122
	ATR-2FT	Y	0.4943	0.0055	0.3182	0.3729	0.4247	0.0454	0.4014	0.3787
	Llama-sft	Y	0.4670	0.0055	0.3734	0.3618	0.4415	0.0478	0.3255	0.3591
	LANCE	Y	0.5287	0.0068	0.4900	0.3627	0.5120	0.0949	0.4815	0.3973
	BALANCE	Y	**0.5664**	**0.0089**	**0.5663**	**0.4827**	**0.5454**	**0.1067**	**0.5567**	**0.5133**

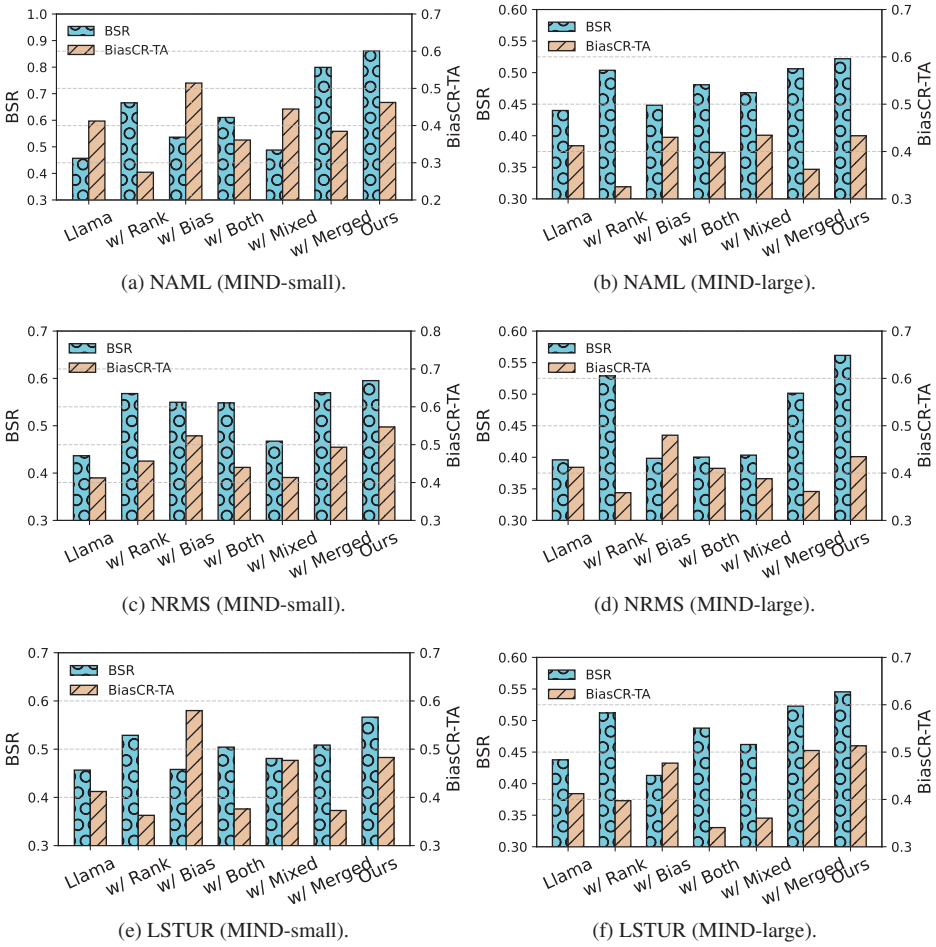
- BALANCE surpasses all baselines in attack performance across three news RSs in both the MIND-small and MIND-large datasets. LANCE achieves the second-best performance, underscoring the efficacy of the DPO fine-tuning pipeline. Moreover, the progressive fine-tuning pipeline employed by our BALANCE further enhances attack performance, demonstrating its effectiveness.
- Llama-sft underperforms compared to BALANCE and LANCE, likely due to its supervised fine-tuning strategy, which relies solely on successful rewrite samples. In contrast, DPO fine-tuning, which incorporates both successful and unsuccessful samples, enables better learning of ranking-enhancing rewrite patterns.
- ATR-2FT demonstrates strong performance in attacking NAML within the MIND-small dataset, but underperforms in other scenarios. This may result from NRMS and LSTUR encoding only news titles, thus restricting the impact of content rewrites. Additionally, to operate within reasonable computational resources (5 NVIDIA A100 GPUs [78]), ATR-2FT employs randomly sampled subsets in its promotion loss. We fixed these subsets to 10% of users in MIND-small. However, MIND-large, with over 14 times the user count of MIND-small, likely contributes to ATR-2FT's limited performance in that dataset.
- Among non-fine-tuned methods, the inclusion of context does not consistently improve attack performance. This may arise from the rapid evolution of news content, where older high-ranking articles in context samples lose relevance over time. This finding aligns with observations in Section 5.3.3.

Preservation of Media Bias Orientation

We analyze the extent to which BALANCE and baselines maintain the original media bias orientation during rewriting. The following observations summarize their bias maintenance performance:

- Fine-tuned methods generally exhibit reduced performance in preserving bias orientation compared to non-fine-tuned methods. This suggests that the task of improving rankings may compromise an LLM's ability to maintain media bias orientation.
- For non-fine-tuned methods, providing bias-consistent examples in the context version yields a higher bias consistency ratio compared to the non-context version. This indicates that LLMs, using their reasoning capabilities, can learn to preserve bias orientation when guided by relevant examples, consistent with findings in Section 5.3.3.
- BALANCE achieves the highest bias consistency ratio in most cases. This demonstrates that, through progressive unified fine-tuning, BALANCE enhances performance in both ranking improvement and media bias maintenance, highlighting the effectiveness of our framework.

Overall, these findings (answering RQ4.1) demonstrate that our proposed approach, BALANCE, outperforms existing models in attacking news RSs while effectively maintaining the original media bias orientation. Specifically, BALANCE achieves superior



(a) NAML (MIND-small).

(b) NAML (MIND-large).

(c) NRMS (MIND-small).

(d) NRMS (MIND-large).

(e) LSTUR (MIND-small).

(f) LSTUR (MIND-large).

Figure 5.4: Ablation study results for BALANCE on the NAML, NRMS, and LSTUR model with MIND-small and MIND-large datasets. The figure shows the attack performance (BSR, left y-axis) and bias orientation preservation (BiasCR-TA, right y-axis) for different variants.

attack performance across multiple news RSs and datasets. This highlights the effectiveness of our progressive unified fine-tuning framework, which not only enhances the model's ability to improve news rankings but also preserves media bias orientation more consistently than other methods. These results underscore the dual capability of BALANCE to balance ranking improvement with bias orientation preservation, addressing a critical challenge in textual attacks on news RSs.

5.5.3 Decomposing BALANCE (RQ4.2)

Next, we conduct a detailed analysis of BALANCE. We examine the Rank data, Bias data, and Combined data, and perform experiments to investigate how the LLM performs when fine-tuned on different datasets and to determine the effectiveness of the progressive fine-tuning framework. To evaluate the contributions of the data types and the framework (cf. Section 5.4), we conduct an ablation study involving five variants of BALANCE: (i) Fine-tuning Llama solely on Rank data to enhance its ability to improve news ranking without considering bias orientation, denoted as "w/ Rank." (ii) Fine-tuning Llama solely on Bias data to learn how to rewrite news while preserving the original bias orientation, termed "w/ Bias." (iii) Fine-tuning Llama on Combined data to rewrite news for higher ranking while maintaining the bias orientation, named "w/ Both." (iv) Fine-tuning Llama on a random mixture of Rank, Bias, and Combined data, which ideally encompasses all necessary information, named "w/ Mixed." (v) Merging the LoRA adapters from Rank LoRA and Bias LoRA without further fine-tuning on Combined data, called "w/ Merged." We present the experimental results in Figure 5.4 and offer the following observations:

- *Task-specific fine-tuning enhances targeted performance.* From Figure 5.4, fine-tuning the LLM exclusively on Rank data or Bias data significantly improves attack performance (for "w/ Rank") and bias orientation maintenance performance (for "w/ Bias"), respectively. This aligns with our expectation that specialized fine-tuning enhances performance in the targeted task. Specifically, "w/ Rank" exhibits high attack performance but poor bias orientation maintenance, which is anticipated since it was not trained on bias-related data. Conversely, "w/ Bias" shows poor attack performance due to the absence of rank-related training data.

- *Constraints of Combined data training.* As shown in Figure 5.4, when trained solely on Combined data, "w/ Both" occasionally outperforms the baseline LLM Llama. However, it sometimes yields reduced performance in maintaining bias orientation across the LSTUR, MIND-small, and MIND-large datasets (see Figures 5.4e and 5.4f), and it does not significantly enhance attack performance on the NRMS and MIND-large datasets (see Figure 5.4d). This may be attributed to the limited availability of high-quality data, which hinders the LLM's ability to learn both ranking and bias orientation preservation patterns effectively.

- *Challenges with mixed data.* According to Figure 5.4, although Mixed data contains sufficient information for rank improvement and bias orientation maintenance, "w/ Mixed" underperforms. This may be attributed to conflicting examples within the mixed data—*e.g.,* a rewrite deemed positive in one instance may be negative in another—leading to confusion during LLM fine-tuning.

- *Limitations of merged LoRA adapters.* As evidenced in Figures 5.4d and 5.4e, "w/ Merged" occasionally fails to achieve a balanced performance in rank improvement and bias orientation maintenance. This could result from the simplistic merging of parameters from Rank LoRA and Bias LoRA without a strategic approach or guiding signal, hindering optimal balance.

- *Advantages of progressive unified fine-tuning.* From Figure 5.4, BALANCE achieves the highest attack performance and bias orientation maintenance comparable to "w/ Bias". This underscores the superiority of our progressive unified fine-tuning strategy, which integrates the strengths of Rank LoRA and Bias LoRA, further enhancing performance on both tasks with Combined data.

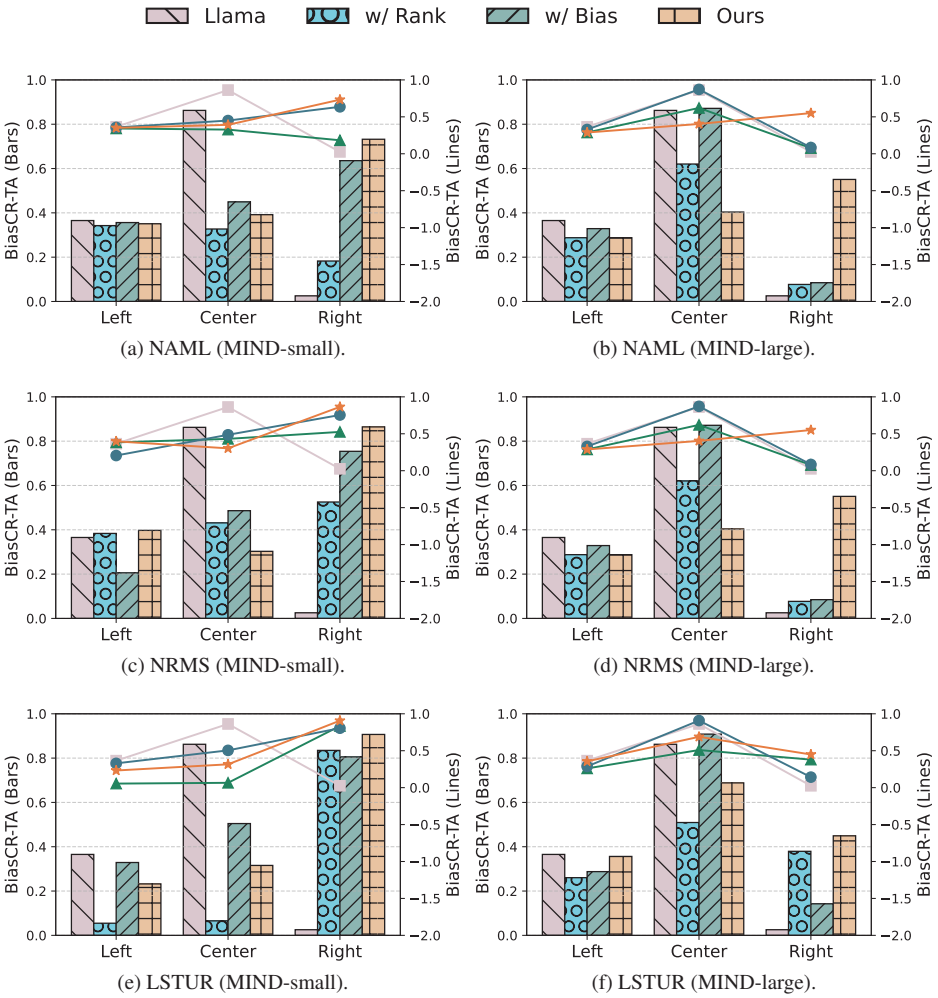
Our findings for RQ4.2 indicate that individual components of the proposed attack method play a critical role in its effectiveness. Specifically, task-specific fine-tuning on Rank data significantly enhances the model's ability to improve news rankings, while fine-tuning on Bias data is essential for preserving the original media bias orientation. However, relying solely on Combined data or a random mixture of datasets leads to suboptimal performance due to data constraints and conflicting examples. The progressive unified fine-tuning framework of BALANCE effectively integrates the strengths of both Rank and Bias fine-tuning, achieving superior attack performance while maintaining bias consistency. These results demonstrate that a careful integration of task-specific components and the strategic use of fine-tuning pipelines are crucial for balancing ranking improvement and bias orientation preservation in textual attacks on news RSs.

5.5.4 Effect of News Media Bias Orientation on Ideological Groups (RQ4.3)

To investigate the effectiveness of our rewrite attack in maintaining media bias orientation across ideological groups, we evaluate the performance of our proposed method, BALANCE, alongside three variants: a baseline LLM ("Llama"), a rank-focused fine-tuned model ("w/ Rank"), and a bias-focused fine-tuned model ("w/ Bias"). The results, shown in Figure 5.5, plot the ratio of news articles that retain their original ideological media bias orientation (left, center, or right) after rewriting, with the x-axis representing bias orientation and the y-axis indicating the consistency ratio. We present the following observations:

- *Baseline LLM prioritizes center bias preservation.* As depicted in Figure 5.5, the baseline "Llama" model preserves center media bias orientation most effectively across all ideological groups. This suggests that, in the absence of ideological bias influence, the LLM inherently favors center-aligned bias, likely due to its pre-training on large datasets.

- *Fine-tuning enhances right-leaning bias preservation.* All fine-tuned variants ("w/ Rank," "w/ Bias," and BALANCE) outperform the baseline "Llama" in preserving right-leaning bias, as shown in Figures 5.5c, 5.5e, and 5.5f. This aligns with the media bias orientation dynamics discussed in Section 5.1, where platforms with a specific ideological leaning, have a greater need to preserve content consistent with that bias orientation. Notably, "w/ Rank," despite lacking explicit bias-related training data, achieves improved right-leaning bias preservation in specific cases. This may be attributed to a higher proportion of right-leaning news in our sampled dataset (see Section 5.5.1), $\mathcal{V}_e$, which provides "w/ Rank" with more right-biased information during fine-tuning.



(a) NAML (MIND-small). (b) NAML (MIND-large).

(c) NRMS (MIND-small). (d) NRMS (MIND-large).

(e) LSTUR (MIND-small). (f) LSTUR (MIND-large).

Figure 5.5: Effect of BALANCE on different media bias orientation groups. The bars represent the ratio of news articles that retain their original ideological media bias orientation (Left, Center, or Right) after rewriting. The lines connect these ratios for each method across the bias orientation groups to illustrate performance trends.

- *Superior performance of BALANCE.* On the MIND-large dataset, as shown in Figures 5.5b, 5.5d, and 5.5f, the variants ("Llama", "w/ Rank", and "w/ Bias") exhibit limited performance in preserving right-leaning bias. In contrast, BALANCE achieves the highest preservation ratio for right-leaning bias and demonstrates a more balanced trend (orange line) across all ideological groups (left, center, right), indicating robust performance in maintaining bias consistency.

Our findings for RQ4.3 indicate that accounting for news bias orientation significantly influences the attack's impact across ideological groups in news RSs. The baseline

Table 5.3: News-recommendation performance on the **MIND-small** dataset before and after the BALANCE attack. The "Change" row denotes the relative performance change.

		AUC	MRR	nDCG5	nDCG10
NAML	Before	68.49	33.30	36.99	42.98
	After	68.48	33.29	36.97	42.97
	Change	–0.015%	–0.030%	–0.054%	–0.023%
NRMS	Before	66.39	31.24	34.10	40.85
	After	66.38	31.23	34.09	40.85
	Change	–0.015%	–0.032%	–0.029%	0.000%
LSTUR	Before	60.41	26.38	28.81	35.35
	After	60.41	26.38	28.81	35.34
	Change	0.000%	0.000%	0.000%	–0.028%

Table 5.4: News-recommendation performance on the **MIND-large** dataset before and after the BALANCE attack. The "Change" row denotes the relative performance change.

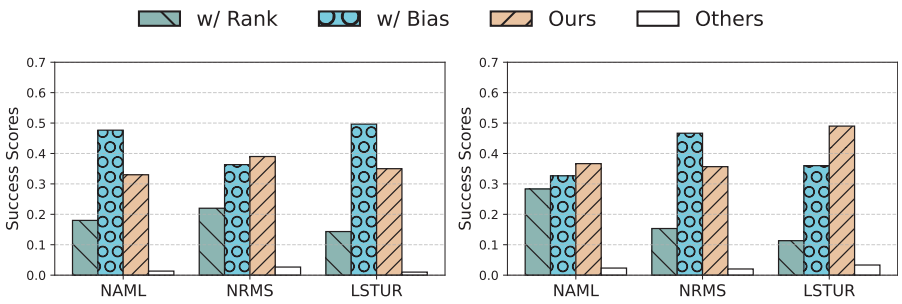
		AUC	MRR	nDCG5	nDCG10
NAML	Before	64.67	30.57	33.54	39.90
	After	64.67	30.57	33.53	39.90
	Change	0.000%	0.000%	–0.030%	0.000%
NRMS	Before	65.81	30.81	33.81	40.41
	After	65.81	30.81	33.81	40.40
	Change	0.000%	0.000%	0.000%	–0.025%
LSTUR	Before	56.90	24.46	26.23	32.76
	After	56.89	24.46	26.23	32.76
	Change	–0.018%	0.000%	0.000%	0.000%

LLM tends to preserve center bias, reflecting its neutral pre-training tendencies. Fine-tuning substantially improves right bias orientation preservation, likely due to dataset composition and strategic fine-tuning. Notably, BALANCE outperforms all variants by maintaining bias consistency across all ideological directions, especially on large-scale datasets like MIND-large.

5.5.5 Impact on Recommendation Performance and Noticeability (RQ4.4)

Recommendation Performance Evaluation

To assess the impact of BALANCE, we tested its performance by replacing targeted news items ($\mathcal{V}_g$) with rewritten news generated by BALANCE. We evaluated the recom-



(a) MIND-small. (b) MIND-large.

Figure 5.6: Success Scores from a GPT-4o-based simulated user study evaluating news articles rewritten by BALANCE. The 'Others' indicates instances where the simulated user was unable to determine which rewritten article best aligned with the original.

mendation performance using the following key metrics: Area Under the Curve (AUC), Mean Reciprocal Rank (MRR), normalized Discounted Cumulative Gain at 5 (nDCG5), and normalized Discounted Cumulative Gain at 10 (nDCG10). The evaluation was conducted on two datasets, MIND-small and MIND-large, using three recommendation models: NAML, NRMS, and LSTUR. The results are presented in Table 5.3 and Table 5.4. We have the following observations:

(1) The performance degradation is minimal, with the largest change being a decrease of 0.032% in the MRR metric for the NRMS model on the MIND-small dataset. This indicates that our method has a negligible impact on recommendation quality, ensuring stealthiness and reducing the likelihood of detection by news platforms.

(2) Moreover, the influence on the larger dataset, MIND-large, is even less noticeable, with most metrics showing 0% change (*e.g.,* AUC, MRR, and nDCG5 for NAML and NRMS). Since only a small portion of targeted news items ($\mathcal{V}_g$) is altered, the overall recommendation performance remains stable, supporting the effectiveness of our targeted approach.

Simulated User Study on Bias and Semantics

We conducted a simulated user study using GPT-4o to investigate whether users can identify rewritten news articles that align with the original media bias orientation and preserve semantic content. We used sampled news articles, $\mathcal{V}_g$, from the MIND dataset. For each news article $t_g \in \mathcal{V}_g$, three rewritten candidates were generated: one using ranking optimization ("w/ Rank"), one with bias alignment ("w/ Bias"), and one using our proposed method, BALANCE (Ours). GPT-4o evaluated each candidate set five times, selecting the candidate that best corresponded to the original news article. The candidate receiving the most votes per article earned a point, with results aggregated as Success Scores and normalized to [0,1], as shown in Figure 5.6.

We observe that when bias information is incorporated, the "w/ Bias" and Ours methods are selected more frequently than the "w/ Rank" method. This indicates the effectiveness of our media bias design in the proposed method. Furthermore, this finding

provides partial support for our assumption in Section 5.3.3, suggesting that users are more likely to notice and prefer news rewrites that maintain the same bias orientation.

In response to RQ4.4, our textual attack has a minimal impact on recommendation performance, ensuring the attack remains stealthy. Furthermore, by maintaining the original bias orientation in rewritten news, the attack is likely to be less noticeable to users. Our simulated user study using GPT-4o demonstrates that rewritten articles preserving bias orientation are more frequently selected as corresponding to the original, suggesting that users are less likely to perceive them as altered. This suggests that our approach is both unnoticeable to users and maintains the intended bias orientation, ensuring the attack's effectiveness without sacrificing the performance of news RSs.

5.6 Limitations and Future Directions

In this study, we focus on the MIND dataset, specifically its MIND-large and MIND-small variants, which are exclusively English-language datasets. We deliberately exclude multilingual datasets to avoid the complexities associated with multilingual news encoders and LLMs, as incorporating multiple languages introduces numerous variables that are challenging to control and exceed the scope of our current research.

We restrict our textual attack to news RSs, as these systems predominantly rely on text representations to generate recommendations. Furthermore, we target news articles within the political category exclusively. While we recognize that other forms of bias may also require consideration, our present investigation centers on media bias, which is well-documented in existing literature [96, 103]. Additionally, we focus on three widely adopted benchmark models in the news recommendation domain: NAML, NRMS, and LSTUR. Alternative recommender systems, such as ID-based collaborative filtering methods, are less suitable for news recommendation due to the frequent updates inherent to news content. We defer the exploration of these alternative approaches to future research.

For fine-tuning, we adopt DPO with LoRA adapters, updating only a small fraction of the parameters for efficient optimization. Preliminary experiments on fewer than $1{,}500$ labeled Rank data revealed that both *full-parameter fine-tuning* and *mixture-of-experts* (MoE) variants suffered from over-fitting and even degraded textual-attack performance. These findings are consistent with recent studies [25, 100, 171]. Consequently, we focus the remainder of this work on DPO + LoRA and leave a systematic exploration of full-scale or MoE fine-tuning to future research. Additionally, investigating other LLM variants, model sizes, or specialized fine-tuning mechanisms could further enhance performance and deserves future investigation.

Although our progressive fine-tuning pipeline introduces additional complexity by training the rank and bias experts separately before unifying them, ablation studies (see Section 5.5.3) reveal that this two-step strategy consistently outperforms the single-stage approach in attack performance and bias orientation preservation, while keeping GPU memory usage manageable thanks to LoRA's low-rank updates. The staged design enables each expert to specialize without mutual interference, and the final unified step integrates their strengths. Although more complex, this offers a path to balance and further enhance multiple, potentially conflicting objectives. We believe it is worthwhile,

and our future work could further refine the approach in the textual attack task, for example, through effective MoE fine-tuning.

To assess bias orientation, we use the open-source PoliticalBiasBERT model [6] to classify news articles into three categories: left, center, and right, reflecting the primary forms of media bias orientation [37, 58, 104]. Nevertheless, an intriguing direction for future work involves exploring subtle shifts in bias orientation, such as modifying a mildly left-leaning article to exhibit a moderate left stance, and assessing how rewrite attacks might facilitate such transformations. Moreover, developing methods to rewrite news articles to achieve a specific bias orientation in a bias-aware manner represents a promising direction for more sophisticated textual attacks.

5.7 Conclusion

In this chapter, we proposed BALANCE, a framework that uses LLMs for bias-aware textual attacks on news RSs. Our approach rewrites news content to boost the ranking of a news article while preserving its original media bias orientation, addressing a critical gap in prior methods that overlooked bias implications. Through an empirical analysis, we identified a challenge in aligning ranking improvements with bias orientation preservation, which BALANCE mitigates using a progressive fine-tuning strategy. Experiments on the MIND-small and MIND-large datasets with the NAML, NRMS, and LSTUR news recommendation models demonstrated that our proposed attack method BALANCE outperforms baselines in both attack performance and bias orientation consistency, with simulated user studies suggesting lower detectability.

Our findings reveal a significant vulnerability in news RSs: textual manipulations can enhance rankings stealthily, potentially amplifying exposure to biased or extreme news content. By preserving media bias orientation of the target articles, BALANCE aligns rewritten content with platform and user expectations, reducing detection risks and increasing attack efficacy. This poses societal risks, such as reinforcing echo chambers or influencing perceptions, highlighting the feasibility of such attacks given content providers' editing capabilities. Our work underscores the urgent need for robust defenses against LLM-driven attacks in news RSs.

Our own future work will focus on developing defense strategies to safeguard recommendation integrity and maintain user trust in an increasingly LLM-influenced media environment.

6 Conclusions

In the previous chapters, we have described how we addressed the research questions raised in Chapter 1 as well as the answers that we have obtained. In this chapter, we first look back to our research questions and summarize the main findings and implications of our work in Section 6.1. We then spell out some future research directions that follow the work in this thesis in Section 6.2.

6.1 Main Findings

6.1.1 Enhancing Recommendation Performance with LLMs by Mitigating Hallucinations

We first consider the challenge of hallucinations when using LLMs for recommendation and pose the following question:

RQ1 How can LLMs be used to enhance recommendation performance while mitigating hallucinations in general recommendation scenarios?

To answer this question, we proposed ToolRec, a tool-learning framework that adapts LLMs for recommendation tasks. ToolRec simulates user decision-making through a multi-round reasoning process, guided by chain-of-thought prompting and external attribute-oriented tools. We designed specialized retrieval and ranking tools to enable dynamic item exploration, along with a memory strategy to manage tool outputs and ensure item consistency. Our approach decomposes recommendation into iterative attribute-based refinements, where the LLM interacts with tools to progressively identify items aligned with user preferences. To support efficient retrieval, we introduced a two-stage learning scheme that fine-tunes attribute-specific encoders while preserving pre-trained sequential behavior representations. Together, these components enable ToolRec to generate controllable and diverse recommendations by grounding LLM reasoning in structured recommendation feedback.

We evaluated the effectiveness of ToolRec through extensive experiments. The results demonstrate that: (i) ToolRec consistently outperforms traditional recommenders and LLM-based baselines on datasets rich in semantic knowledge, such as ML-1M and Amazon-Book; (ii) ToolRec benefits significantly from its multi-round user decision simulation and attribute-oriented tool design, which allow LLMs to refine item selection

based on dynamic user intent; (iii) our attribute-oriented retrieval tools effectively balance performance and parameter efficiency. However, our results also indicate that ToolRec's performance declines on datasets like Yelp2018, where LLMs have limited domain knowledge of local businesses, and that the choice of LLM critically affects success, with ChatGPT outperforming alternatives like Vicuna and PaLM.

6.1.2 Enhancing Recommendation Performance by Fine-tuning Language Models

Next, we investigated the challenge of under-representation in news RSs and posed the following question:

RQ2 How can fine-tuning language models improve the capture of user preferences and enhance recommendations in semantic-rich news recommendation scenarios while avoiding under-representation?

To evaluate the effectiveness of different language models (LMs) as news encoders for neural news recommendation systems, we conducted a comprehensive analysis across static LMs (SLMs), pre-trained LMs (PLMs), and large LMs (LLMs). We systematically explored each LM type in both non-fine-tuned and fine-tuned modes and examined their performance within three representative news recommendation models: NAML, NRMS, and LSTUR. By isolating textual features and unifying the experimental setup, we assessed the semantic modeling capabilities of various LMs in a controlled manner.

The experimental results confirm the effectiveness of using LMs as news encoders in neural news RSs. We found that: (i) increasing LM size does not consistently improve recommendation performance; (ii) fine-tuning is beneficial for most pre-trained LMs, though the extent of improvement varies by model and architecture; (iii) GloVe demonstrates competitive performance and efficiency, making it a strong baseline in low-resource settings. Furthermore, we found that larger LMs, such as Llama, offer notable gains for cold-start users with sparse click histories, but their advantages diminish for highly active users.

6.1.3 Exploiting LLMs for Textual Attacks in News Recommendations

For the exploitation side, we investigate how textual content can be rewritten to manipulate the ranking outcomes of news RSs and asked:

RQ3 Can LLMs be exploited to conduct textual attacks in semantic-rich news recommendation scenarios?

To address this research question, we proposed LANCE, a two-stage textual attack framework comprising an Explorer and a Reflector. The Explorer prompts large language models (LLMs) to rewrite news articles across multiple stylistic dimensions, including writing style, sentiment, and author persona, to generate diverse rewrite variants. These rewrites are filtered based on their effectiveness in improving ranking positions within a target recommender system, producing a dataset of pairs of successful

and unsuccessful attacks. The Reflector is then fine-tuned using direct preference optimization to learn rewriting patterns that consistently enhance news rankings. Finally, the fine-tuned model generates adversarial rewrites that promote the ranking of news items, thus implementing a textual attack mechanism.

Our experimental results confirm the effectiveness of LANCE: (i) LANCE consistently outperforms LLM-based and fine-tuned baselines in boosting target news rankings while maintaining high text naturalness and semantic preservation; (ii) ablation studies demonstrate that LANCE benefits from diverse rewriting strategies and task-specific fine-tuning; (iii) LANCE achieves attack performance gains even in unseen news RSs, whose information was not included in LANCE's training data. Moreover, we uncovered a unique vulnerability in news RSs: negative and neutral rewrites are more successful than positive ones, contrary to trends observed in other domains, such as e-commerce.

6.1.4 Exploiting LLMs for Textual Attacks by Preserving Media Bias Orientation

Finally, we took a step toward the media bias in news scenarios and answered the following question:

RQ4 How can LLMs be used to conduct textual attacks that preserve media bias orientation in news recommender systems?

To answer this question, we proposed BALANCE, a media bias-aware textual attack framework. We modeled the textual attack as a dual-objective problem: enhancing article ranking while preserving media bias orientation. Our empirical analyses revealed that textual attacks often cause unintended bias shifts, and that maintaining bias consistency can hinder rank improvement. Motivated by these insights, we developed a progressive fine-tuning approach that uses LLMs to decouple and then unify the two objectives. Specifically, we trained a rank expert and a bias expert using LoRA and direct preference optimization, then merged their capabilities through unified fine-tuning on combined data that satisfies both constraints.

Our experimental results demonstrate the effectiveness of BALANCE in conducting bias-aware textual attacks on news RSs. The experimental results show that: (i) BALANCE consistently outperforms baselines in both boosting news rankings and preserving original media bias orientation. (ii) ablation studies reveal the critical role of progressive fine-tuning in balancing ranking improvement and bias consistency; and (iii) simulated user studies and performance evaluations confirm the attack's stealthiness and minimal impact on overall recommendation quality. Furthermore, BALANCE maintains bias orientation across ideological groups more effectively than all tested variants, demonstrating robustness in real-world political news settings.

6.2 Future Work

In this section, we first outline several potential directions for future work rooted in our two main research themes, and then we broaden the scope by discussing other lines that lie outside the present thesis.

6.2.1 Improving Domain Adaptability and User Modeling

We have answered **RQ1** by proposing ToolRec, a framework that uses LLMs and attribute-oriented tools for enhanced recommendation. However, our results show that ToolRec heavily depends on the LLM's underlying domain knowledge and performs less effectively on datasets like Yelp2018, where such knowledge is sparse. Future work could improve domain adaptability by incorporating recommendation-specific knowledge into LLMs through retrieval-augmented generation (RAG) or lightweight domain-specific fine-tuning. Additionally, our current toolset relies on static attribute-based retrieval. Exploring more dynamic or hybrid tools, such as knowledge graphs, online databases, or search engines, could further enhance the diversity and relevance of generated recommendations. Lastly, we observed performance variability across different LLMs, indicating that our approach can progressively benefit from more powerful LLMs as they become available. These directions collectively aim to improve the controllability, reliability, and generalization of LLM-enhanced recommender systems.

Regarding the under-representation issue, our study on **RQ2** shows that fine-tuning LMs as news encoders can improve the performance of neural news RSs. However, several future directions remain open. First, while LLMs like LLaMA perform well in cold-start scenarios, their effectiveness diminishes for highly active users. Future work could explore methodologies to better model highly active users, whose more complex behaviors may require more sophisticated designs when using LLMs. Second, although our experiments use static datasets and pre-computed embeddings, real-world news platforms involve constant updates. Future work may explore incremental embedding updates or lightweight continual fine-tuning to adapt to evolving content. Finally, since news recommendation is semantically rich, a promising direction is to integrate the ToolRec idea: using multiple LLMs to simulate user preferences and refining recommendations based on agreement. While this hybrid approach holds promise, managing the complexity of multiple LLMs and controlling their fine-tuning strategies remains a key challenge for future research.

6.2.2 Enhancing Attack Realism and Defense Strategies

We answered **RQ3** by proposing LANCE, a novel LLM-based textual attack framework that manipulates news rankings in recommender systems via content rewriting. In more severe scenarios, attackers may lack access to immediate ranking feedback, such as when news platforms delay rank updates. Although this limits the direct applicability of our method, a promising future direction is to develop surrogate news RSs. These surrogate systems do not need to exactly replicate the target RSs but should exhibit similar ranking tendencies. This would ensure that LANCE can still receive feedback for training. We plan to investigate how such surrogate systems can be built using limited information, such as a small set of users and their received recommendations, thus extending our method to more realistic attack settings.

In addition to LANCE, we introduced BALANCE, a media bias-aware textual attack framework, to address **RQ4**. BALANCE rewrites political news articles to improve their rankings in RSs while preserving their original ideological bias orientation. However, ideological bias is just one form of media bias. Other dimensions, such as selection bias

or cognitive bias, may also influence RSs or user perceptions. Future work could explore more comprehensive or multi-dimensional bias categories to enhance the realism and controllability of attacks, allowing them to target specific bias groups more effectively.

Additionally, to mitigate emerging risks and promote more reliable RSs, our future research should explore robust defense mechanisms against LLM-based textual manipulation. These may include adversarial training, media integrity scoring, or AI-generated text detection. Such efforts are crucial to protect the integrity of RSs and ensure the responsible use of LLMs in sensitive domains like news recommendation, highlighting the growing importance of reliability in future RS design.

6.2.3 Other Potential Directions

In addition to the directions tied to our core research themes, we identify broader opportunities to advance reliable RSs in the era of LLMs. These directions complement earlier subsections by addressing challenges in conversational interaction, multi-LLM collaboration, and ethical considerations.

LLMs enable conversational RSs that engage users through multi-turn dialogues, clarifying intents and adapting to evolving preferences [70, 136]. Future work could focus on creating unified user representations that integrate behavioral signals, such as clicks, with dialogue-based signals, including sentiment shifts and explicit feedback. To evaluate the quality of these conversational RSs, new metrics are needed to balance task success, user satisfaction, and computational efficiency, ensuring reliability in open-ended interactions.

Another promising direction is the development of collaborative LLM teams to enhance RS reliability and performance [66, 108, 149]. Unlike systems relying on a single LLM, teams of specialized models can distribute tasks, improve robustness, and maintain functionality even if one model fails. Potential approaches include: (i) enabling models to discuss and vote on optimal recommendations; (ii) routing subtasks to the most confident model; and (iii) implementing cross-verification to detect errors, biases, or privacy risks. Realizing these ideas requires defining clear protocols for model cooperation, efficient task scheduling, and fair mechanisms for resolving conflicts.

Beyond improving interaction and architecture, reliable RSs need to address ethical challenges, particularly in user privacy and fairness [167]. Integrating LLM-based personalization with privacy-preserving techniques, such as federated learning or differential privacy, can ensure sensitive data remains on user devices. Additionally, detecting and mitigating emergent biases in dynamically generated content is critical to ensuring fairness across diverse demographics. Addressing these challenges will advance RSs toward holistic reliability, balancing personalization with ethical considerations.

Bibliography

[1] M. Aktukmak, Y. Yilmaz, and I. Uysal. Quick and accurate attack detection in recommender systems through user attributes. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 348–352, 2019. (Cited on page 51.)

[2] M. Alam, A. Iana, A. Grote, K. Ludwig, P. Müller, and H. Paulheim. Towards analyzing the bias of news recommender systems using sentiment and stance detection. In *Companion of The Web Conference 2022, Virtual Event / Lyon, France, April 25 - 29, 2022*, pages 448–457. ACM, 2022. doi: 10.1145/3487553.3524674. URL https://doi.org/10.1145/3487553.3524674. (Cited on pages 53, 55, and 79.)

[3] M. An, F. Wu, C. Wu, K. Zhang, Z. Liu, and X. Xie. Neural news recommendation with long- and short-term user representations. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 336–345. Association for Computational Linguistics, 2019. doi: 10.18653/V1/P19-1033. URL https://doi.org/10.18653/v1/p19-1033. (Cited on pages 35, 36, 38, 41, 54, 58, 73, 85, and 86.)

[4] S. E. Asch. Effects of group pressure upon the modification and distortion of judgments. In *Organizational Influence Processes*, pages 295–303. Routledge, 2016. ISBN 9781315290614. (Cited on pages 2, 6, and 72.)

[5] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, N. Joseph, S. Kadavath, J. Kernion, T. Conerly, S. E. Showk, N. Elhage, Z. Hatfield-Dodds, D. Hernandez, T. Hume, S. Johnston, S. Kravec, L. Lovitt, N. Nanda, C. Olsson, D. Amodei, T. B. Brown, J. Clark, S. McCandlish, C. Olah, B. Mann, and J. Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022. (Cited on page 17.)

[6] R. Baly, G. D. S. Martino, J. R. Glass, and P. Nakov. We can detect your bias: Predicting the political ideology of news articles. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4982–4991. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.404. URL https://doi.org/10.18653/v1/2020.emnlp-main.404. (Cited on pages 72, 87, and 98.)

[7] Y. Bang, S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung, Q. V. Do, Y. Xu, and P. Fung. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. In *IJCNLP (1)*, pages 675–718, 2023. (Cited on pages 1, 3, and 15.)

[8] K. Bao, J. Zhang, W. Wang, Y. Zhang, Z. Yang, Y. Luo, C. Chen, F. Feng, and Q. Tian. A bi-step grounding paradigm for large language models in recommendation systems. *arXiv preprint arXiv:2308.08434*, 2023. (Cited on page 40.)

[9] K. Bao, J. Zhang, Y. Zhang, W. Wang, F. Feng, and X. He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *RecSys*, pages 1007–1014, 2023. (Cited on pages 18 and 19.)

[10] W. L. Bennett and S. Iyengar. A new era of minimal effects? the changing foundations of political communication. *Journal of Communication*, 58(4):707–731, 12 2008. ISSN 0021-9916. doi: 10.1111/j.1460-2466.2008.00410.x. (Cited on page 72.)

[11] D. Bernhardt, S. Krasa, and M. Polborn. Political polarization and the electoral effects of media bias. *Journal of Public Economics*, 92(5-6):1092–1104, 2008. ISSN 0047-2727. doi: https://doi.org/10.1016/j.jpubeco.2008.01.006. (Cited on page 72.)

[12] Q. Bi, J. Li, L. Shang, X. Jiang, Q. Liu, and H. Yang. Mtrec: Multi-task learning over bert for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2663–2669, 2022. (Cited on pages 36 and 41.)

[13] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. Recommender systems survey. *Knowledge-based systems*, 46:109–132, 2013. (Cited on page 1.)

[14] M. T. Boykoff. Flogging a dead norm? newspaper coverage of anthropogenic climate change in the united states and united kingdom from 2003 to 2006. *Area*, 39(4):470–481, 2007. doi: 10.1111/j.1475-4762.2007.00769.x. (Cited on page 72.)

[15] H. Brandenburg. Party strategy and media bias: A quantitative analysis of the 2005 uk election campaign. *Journal of Elections, Public Opinion and Parties*, 16(2):157–178, 2006. doi: 10.1080/13689880600716027. (Cited on page 72.)

[16] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess,

J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html. (Cited on page 76.)

[17] H. Chen and J. Li. Data poisoning attacks on cross-domain recommendation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 2177–2180. ACM, 2019. doi: 10.1145/3357384.3358116. URL https://doi.org/10.1145/3357384.3358116. (Cited on page 51.)

[18] X. Chen, S. Li, H. Li, S. Jiang, Y. Qi, and L. Song. Generative adversarial user model for reinforcement learning based recommendation system. In *ICML*, pages 1052–1061, 2019. (Cited on page 18.)

[19] H. Chiang, Y. Chen, Y. Song, H. Shuai, and J. S. Chang. Shilling black-box review-based recommender systems through fake review generation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 286–297. ACM, 2023. doi: 10.1145/3580305.3599502. URL https://doi.org/10.1145/3580305.3599502. (Cited on pages 54, 69, and 74.)

[20] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/. (Cited on page 32.)

[21] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel. Palm: Scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24:240:1–240:113, 2023. (Cited on page 32.)

[22] Y. Deldjoo, N. Mehta, M. Sathiamoorthy, S. Zhang, P. Castells, and J. McAuley. Toward holistic evaluation of recommender systems powered by generative models. *arXiv preprint arXiv:2504.06667*, 2025. (Cited on page 3.)

[23] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1423. URL https://doi.org/10.18653/v1/n19-1423. (Cited on pages 19, 25, 35, 44, 54, and 73.)

[24] B. Ding, C. Qin, R. Zhao, T. Luo, X. Li, G. Chen, W. Xia, J. Hu, A. T. Luu, and S. Joty. Data augmentation using LLMs: Data perspectives, learning paradigms and challenges. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 1679–1705. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-ACL.97. URL https://doi.org/10.18653/v1/2024.findings-acl.97. (Cited on pages 55 and 79.)

[25] G. Dong, H. Yuan, K. Lu, C. Li, M. Xue, D. Liu, W. Wang, Z. Yuan, C. Zhou, and J. Zhou. How abilities in large language models are affected by supervised fine-tuning data composition. In L. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 177–198. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.12. URL https://doi.org/10.18653/v1/2024.acl-long.12. (Cited on page 97.)

[26] M. Fang, G. Yang, N. Z. Gong, and J. Liu. Poisoning attacks to graph-based recommender systems. In *Proceedings of the 34th Annual Computer Security Applications Conference, ACSAC 2018, San Juan, PR, USA, December 03-07, 2018*, pages 381–392. ACM, 2018. doi: 10.1145/3274694.3274706. URL https://doi.org/10.1145/3274694.3274706. (Cited on page 73.)

[27] C. Fisher, T. Flew, S. Park, J. Y. Lee, and U. Dulleck. Improving trust in news: Audience solutions. *Journalism Practice*, 15(10):1497–1515, 2021. doi: 10.1080/17512786.2020.1787859. (Cited on page 72.)

[28] L. Friedman, S. Ahuja, D. Allen, Z. Tan, H. Sidahmed, C. Long, J. Xie, G. Schubiner, A. Patel, H. Lara,

B. Chu, Z. Chen, and M. Tiwari. Leveraging large language models in conversational recommender systems. *CoRR*, abs/2305.07961, 2023. (Cited on pages 16, 18, and 19.)

[29] C. Gao, S. Wang, S. Li, J. Chen, X. He, W. Lei, B. Li, Y. Zhang, and P. Jiang. CIRS: bursting filter bubbles by counterfactual interactive recommender system. *ACM Trans. Inf. Syst.*, 42(1):14:1–14:27, 2024. (Cited on pages 1 and 15.)

[30] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops, SP Workshops 2018, San Francisco, CA, USA, May 24, 2018*, pages 50–56. IEEE Computer Society, 2018. doi: 10.1109/SPW.2018.00016. URL https://doi.org/10.1109/SPW.2018.00016. (Cited on pages 54 and 74.)

[31] Y. Gao, T. Sheng, Y. Xiang, Y. Xiong, H. Wang, and J. Zhang. Chat-REC: Towards interactive and explainable LLMs-augmented recommender system. *CoRR*, abs/2303.14524, 2023. (Cited on pages 16, 18, 19, and 25.)

[32] Y. Ge, W. Hua, K. Mei, J. Ji, J. Tan, S. Xu, Z. Li, and Y. Zhang. OpenAGI: When LLM meets domain experts. In *NeurIPS*, 2023. (Cited on page 18.)

[33] S. Geng, S. Liu, Z. Fu, Y. Ge, and Y. Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *RecSys*, pages 299–315, 2022. (Cited on pages 18, 19, and 25.)

[34] S. Geng, J. Tan, S. Liu, Z. Fu, and Y. Zhang. VIP5: towards multimodal foundation models for recommendation. In *EMNLP (Findings)*, pages 9606–9620, 2023. (Cited on pages 18 and 19.)

[35] J. A. Gulla, L. Zhang, P. Liu, Ö. Özgöbek, and X. Su. The Adressa dataset for news recommendation. In *Proceedings of the International Conference on Web Intelligence*, pages 1042–1048, 2017. (Cited on pages 47 and 67.)

[36] Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, and Q. He. A survey on knowledge graph-based recommender systems. *IEEE Trans. Knowl. Data Eng.*, 34(8):3549–3568, 2020. (Cited on page 15.)

[37] F. Hamborg, K. Donnay, and B. Gipp. Automated identification of media bias in news articles: An interdisciplinary literature review. *International Journal on Digital Libraries*, 20(4):391–415, 2019. doi: 10.1007/s00799-018-0261-y. (Cited on pages 2, 70, 72, and 98.)

[38] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*, pages 639–648, 2020. (Cited on pages 1 and 15.)

[39] Y. Hou, J. Zhang, Z. Lin, H. Lu, R. Xie, J. J. McAuley, and W. X. Zhao. Large language models are zero-shot rankers for recommender systems. In *ECIR (2)*, volume 14609, pages 364–381, 2024. (Cited on pages 16, 21, 25, and 33.)

[40] C. Hu, J. Fu, C. Du, S. Luo, J. Zhao, and H. Zhao. ChatDB: Augmenting LLMs with databases as their symbolic memory. *CoRR*, abs/2306.03901, 2023. (Cited on page 18.)

[41] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9. (Cited on pages 19, 41, 60, 82, and 87.)

[42] W. Hua, S. Xu, Y. Ge, and Y. Zhang. How to index item ids for recommendation foundation models. In *SIGIR-AP*, pages 195–204, 2023. (Cited on page 28.)

[43] W. Hua, Y. Ge, S. Xu, J. Ji, Z. Li, and Y. Zhang. UP5: unbiased foundation model for fairness-aware recommendation. In *EACL (1)*, pages 1899–1912, 2024. (Cited on pages 18 and 19.)

[44] H. Huang, J. Mu, N. Z. Gong, Q. Li, B. Liu, and M. Xu. Data poisoning attacks to deep learning based recommender systems. *arXiv preprint arXiv:2101.02644*, 2021. (Cited on pages 5 and 51.)

[45] X. Huang, J. Lian, Y. Lei, J. Yao, D. Lian, and X. Xie. Recommender AI agent: Integrating large language models for interactive recommendations. *CoRR*, abs/2308.16505, 2023. (Cited on pages 15, 18, and 19.)

[46] A. Iana, G. Glavaš, and H. Paulheim. NewsRecLib: A pytorch-lightning library for neural news recommendation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 296–310, 2023. (Cited on pages 4, 36, 37, and 38.)

[47] A. Iana, G. Glavaš, and H. Paulheim. Mind your language: A multilingual dataset for cross-lingual news recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 553–563, 2024. (Cited on pages 36 and 37.)

[48] M. Iyyer, P. Enns, J. L. Boyd-Graber, and P. Resnik. Political ideology detection using recursive neural networks. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 1113–1122. The Association for Computer Linguistics, 2014. doi: 10.3115/V1/P14-1105. URL

https://doi.org/10.3115/v1/p14-1105. (Cited on page 72.)

[49] J. Jiang. Tadi: Topic-aware attention and powerful dual-encoder interaction for recall in news recommendation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15647–15658, 2023. (Cited on pages 36, 37, and 38.)

[50] T. Jiang, S. Huang, Z. Luan, D. Wang, and F. Zhuang. Scaling sentence embeddings with large language models. *arXiv preprint arXiv:2307.16645*, 2023. (Cited on page 40.)

[51] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press, 2020. doi: 10.1609/AAAI.V34I05.6311. URL https://doi.org/10.1609/aaai.v34i05.6311. (Cited on pages 54, 70, and 74.)

[52] W.-C. Kang and J. McAuley. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE, 2018. (Cited on pages 1, 15, 22, 24, 25, and 46.)

[53] E. Karpas, O. Abend, Y. Belinkov, B. Lenz, O. Lieber, N. Ratner, Y. Shoham, H. Bata, Y. Levine, K. Leyton-Brown, D. Muhlgay, N. Rozen, E. Schwartz, G. Shachaf, S. Shalev-Shwartz, A. Shashua, and M. Tennenholtz. MRKL systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning. *CoRR*, abs/2205.00445, 2022. (Cited on page 18.)

[54] B. Kille, F. Hopfgartner, T. Brodt, and T. Heintz. The plista dataset. In *Proceedings of the 2013 International News Recommender Systems Workshop and Challenge*, pages 16–23, 2013. (Cited on page 47.)

[55] B. Kitchenham et al. Guidelines for performing systematic literature reviews in software engineering. Technical report, Keele University, 2007. Version 2.3. (Cited on page 36.)

[56] J. Krieger, T. Spinde, T. Ruas, J. Kulshrestha, and B. Gipp. A domain-adaptive pre-training approach for language bias detection in news. In *JCDL '22: The ACM/IEEE Joint Conference on Digital Libraries in 2022, Cologne, Germany, June 20 - 24, 2022*, page 3. ACM, 2022. doi: 10.1145/3529372.3530932. URL https://doi.org/10.1145/3529372.3530932. (Cited on page 72.)

[57] J. Kruse, K. Lindskow, S. Kalloori, M. Polignano, C. Pomo, A. Srivastava, A. Uppal, M. R. Andersen, and J. Frellsen. EB-NeRD: A large-scale dataset for news recommendation. *arXiv preprint arXiv:2410.03432*, 2024. (Cited on pages 47, 67, and 68.)

[58] V. Kulkarni, J. Ye, S. Skiena, and W. Y. Wang. Multi-view models for political ideology detection of news articles. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3518–3527. Association for Computational Linguistics, 2018. doi: 10.18653/V1/D18-1388. URL https://doi.org/10.18653/v1/d18-1388. (Cited on pages 2, 70, 72, and 98.)

[59] S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th international conference on World Wide Web, WWW 2004, New York, NY, USA, May 17-20, 2004*, pages 393–402. ACM, 2004. doi: 10.1145/988672.988726. URL https://doi.org/10.1145/988672.988726. (Cited on page 73.)

[60] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1885–1893, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/83fa5a432ae55c253d0e60dbfa716723-Abstract.html. (Cited on page 73.)

[61] J. Li, J. Zhu, Q. Bi, G. Cai, L. Shang, Z. Dong, X. Jiang, and Q. Liu. MINER: multi-interest matching network for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 343–352. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.FINDINGS-ACL.29. URL https://doi.org/10.18653/v1/2022.findings-acl.29. (Cited on pages 36, 37, 38, 40, 54, and 73.)

[62] L. Li, R. Ma, Q. Guo, X. Xue, and X. Qiu. BERT-ATTACK: adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6193–6202. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.500. URL https://doi.org/10.18653/v1/2020.emnlp-main.500. (Cited on page 74.)

[63] M. Li and L. Wang. A survey on personalized news recommendation technology. *IEEE Access*,

7:145861–145879, 2019. doi: 10.1109/ACCESS.2019.2944927. URL https://doi.org/10.1109/ACCESS.2019.2944927. (Cited on pages 2, 4, 35, 54, and 73.)
[64] M. Li, Y. Zhao, B. Yu, F. Song, H. Li, H. Yu, Z. Li, F. Huang, and Y. Li. API-Bank: A comprehensive benchmark for tool-augmented LLMs. In *EMNLP*, pages 3102–3116, 2023. (Cited on page 18.)
[65] X. Li, Y. Zhang, and E. C. Malthouse. Exploring fine-tuning chatgpt for news recommendation. *arXiv preprint arXiv:2311.05850*, 2023. (Cited on page 37.)
[66] X. Li, S. Wang, S. Zeng, Y. Wu, and Y. Yang. A survey on llm-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth*, 1(1):9, 2024. (Cited on page 103.)
[67] X. Li, Y. Zhang, and E. C. Malthouse. Prompt-based generative news recommendation (PGNR): accuracy and controllability. In *Advances in Information Retrieval - 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24-28, 2024, Proceedings, Part II*, volume 14609 of *Lecture Notes in Computer Science*, pages 66–79. Springer, 2024. doi: 10.1007/978-3-031-56060-6_5. URL https://doi.org/10.1007/978-3-031-56060-6_5. (Cited on pages 36, 37, 41, and 58.)
[68] J. Liao, S. Li, Z. Yang, J. Wu, Y. Yuan, X. Wang, and X. He. Large language-recommendation assistant. In *SIGIR*, 2024. (Cited on pages 2, 15, and 19.)
[69] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004. (Cited on page 61.)
[70] J. Lin, X. Dai, Y. Xi, W. Liu, B. Chen, X. Li, C. Zhu, H. Guo, Y. Yu, R. Tang, and W. Zhang. How can recommender systems benefit from large language models: A survey. *CoRR*, abs/2306.05817, 2023. (Cited on pages 1, 3, 15, 37, and 103.)
[71] Q. Liu, N. Chen, T. Sakai, and X. Wu. ONCE: boosting content-based recommendation with both open- and closed-source large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining, WSDM 2024, Merida, Mexico, March 4-8, 2024*, pages 452–461. ACM, 2024. doi: 10.1145/3616855.3635845. URL https://doi.org/10.1145/3616855.3635845. (Cited on pages 1, 36, 37, 41, 54, and 73.)
[72] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692. (Cited on pages 35, 54, and 73.)
[73] Y. Liu, R. Zhang, J. Guo, M. de Rijke, Y. Fan, and X. Cheng. Multi-granular adversarial attacks against black-box neural ranking models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 1391–1400. ACM, 2024. doi: 10.1145/3626772.3657704. URL https://doi.org/10.1145/3626772.3657704. (Cited on pages 58 and 85.)
[74] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang. Recommender system application developments: A survey. *Decis. Support Syst.*, 74:12–32, 2015. (Cited on pages 1 and 3.)
[75] B. W. McKeever, D. Riffe, and F. D. Carpentier. Perceived hostile media bias, presumed media influence, and opinions about immigrants and immigration. *Southern Communication Journal*, 77(5): 420–437, 2012. doi: 10.1080/1041794X.2012.691602. (Cited on page 72.)
[76] S. Mysore, A. McCallum, and H. Zamani. Large language model augmented narrative driven recommendations. *RecSys*, pages 777–783, 2023. (Cited on pages 2, 15, and 18.)
[77] R. S. Nickerson. Confirmation bias: A ubiquitous phenomenon in many guises. *Review of General Psychology*, 2(2):175–220, 1998. doi: 10.1037/1089-2680.2.2.175. (Cited on pages 2, 6, and 72.)
[78] S. Oh, G. Verma, and S. Kumar. Adversarial text rewriting for text-aware recommender systems. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, CIKM 2024, Boise, ID, USA, October 21-25, 2024*, pages 1804–1814. ACM, 2024. doi: 10.1145/3627673.3679592. URL https://doi.org/10.1145/3627673.3679592. (Cited on pages 5, 6, 51, 52, 54, 58, 60, 69, 70, 74, 76, 85, 86, and 90.)
[79] S. Okura, Y. Tagami, S. Ono, and A. Tajima. Embedding-based news recommendation for millions of users. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 1933–1942. ACM, 2017. doi: 10.1145/3097983.3098108. URL https://doi.org/10.1145/3097983.3098108. (Cited on page 73.)
[80] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. (Cited on pages 1, 17, and 76.)
[81] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022. (Cited on page 56.)
[82] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine

translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002. (Cited on page 61.)

[83] E. Pariser. *The Filter Bubble: What The Internet Is Hiding From You*. Penguin Books Limited, 2011. ISBN 9780141969923. (Cited on pages 6 and 72.)

[84] S. Park, K. S. Lee, and J. Song. Contrasting opposing views of news articles on contentious issues. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 340–349. The Association for Computer Linguistics, 2011. URL https://aclanthology.org/P11-1035/. (Cited on page 72.)

[85] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2007. doi: 10.1007/978-3-540-72079-9_10. URL https://doi.org/10.1007/978-3-540-72079-9_10. (Cited on page 73.)

[86] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014. doi: 10.3115/V1/D14-1162. URL https://doi.org/10.3115/v1/d14-1162. (Cited on pages 26, 35, 44, 54, and 73.)

[87] U. Peters. What is the function of confirmation bias? *Erkenntnis*, 87(3):1351–1376, 2022. doi: 10.1007/s10670-020-00252-1. (Cited on page 72.)

[88] Y. Qin, S. Hu, Y. Lin, W. Chen, N. Ding, G. Cui, Z. Zeng, Y. Huang, C. Xiao, C. Han, Y. R. Fung, Y. Su, H. Wang, C. Qian, R. Tian, K. Zhu, S. Liang, X. Shen, B. Xu, Z. Zhang, Y. Ye, B. Li, Z. Tang, J. Yi, Y. Zhu, Z. Dai, L. Yan, X. Cong, Y. Lu, W. Zhao, Y. Huang, J. Yan, X. Han, X. Sun, D. Li, J. Phang, C. Yang, T. Wu, H. Ji, Z. Liu, and M. Sun. Tool learning with foundation models, 2023. (Cited on pages 18, 20, and 21.)

[89] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian, S. Zhao, R. Tian, R. Xie, J. Zhou, M. Gerstein, D. Li, Z. Liu, and M. Sun. ToolLLM: Facilitating large language models to master 16000+ real-world apis. *CoRR*, abs/2307.16789, 2023. (Cited on pages 18, 20, and 21.)

[90] Z. Qiu, X. Wu, J. Gao, and W. Fan. U-bert: Pre-training user representations for improved recommendation. In *AAAI*, volume 35, pages 4320–4327, 2021. (Cited on pages 15 and 19.)

[91] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024. (Cited on pages 53, 56, and 82.)

[92] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. (Cited on pages 19 and 25.)

[93] S. Raza and C. Ding. News recommender system: a review of recent progress, challenges, and opportunities. *Artificial Intelligence Review*, pages 1–52, 2022. (Cited on page 51.)

[94] X. Ren, W. Wei, L. Xia, L. Su, S. Cheng, J. Wang, D. Yin, and C. Huang. Representation learning with large language models for recommendation. In *Proceedings of the ACM web conference 2024*, pages 3464–3475, 2024. (Cited on page 3.)

[95] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009. (Cited on pages 1 and 15.)

[96] F. Rodrigo-Ginés, J. Carrillo-de-Albornoz, and L. Plaza. A systematic review on media bias detection: What is media bias, how it is expressed, and how to detect it. *Expert Syst. Appl.*, 237(Part C):121641, 2024. doi: 10.1016/J.ESWA.2023.121641. URL https://doi.org/10.1016/j.eswa.2023.121641. (Cited on pages 2, 6, 70, 72, 75, and 97.)

[97] S. Sanner, K. Balog, F. Radlinski, B. Wedin, and L. Dixon. Large language models are competitive near cold-start recommenders for language-and item-based preferences. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 890–896, 2023. (Cited on page 36.)

[98] J. B. Schafer, D. Frankowski, J. L. Herlocker, and S. Sen. Collaborative filtering recommender systems. In *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*, pages 291–324. Springer, 2007. doi: 10.1007/978-3-540-72079-9_9. URL https://doi.org/10.1007/978-3-540-72079-9_9. (Cited on pages 3 and 73.)

[99] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom. Toolformer: Language models can teach themselves to use tools. In *NeurIPS*, 2023. (Cited on pages 16, 18, 20, and 21.)

[100] S. Shen, L. Hou, Y. Zhou, N. Du, S. Longpre, J. Wei, H. W. Chung, B. Zoph, W. Fedus, X. Chen,

T. Vu, Y. Wu, W. Chen, A. Webson, Y. Li, V. Y. Zhao, H. Yu, K. Keutzer, T. Darrell, and D. Zhou. Mixture-of-experts meets instruction tuning: A winning combination for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=6mLjDwYte5. (Cited on page 97.)

[101] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang. Hugginggpt: Solving AI tasks with chatgpt and its friends in hugging face. In *NeurIPS*, 2023. (Cited on pages 16 and 18.)

[102] J. Song, Z. Li, Z. Hu, Y. Wu, Z. Li, J. Li, and J. Gao. PoisonRec: An adaptive data poisoning framework for attacking black-box recommender systems. In *2020 IEEE 36th international conference on data engineering (ICDE)*, pages 157–168. IEEE, 2020. (Cited on pages 51 and 69.)

[103] T. Spinde, S. Hinterreiter, F. Haak, T. Ruas, H. Giese, N. Meuschke, and B. Gipp. The media bias taxonomy: A systematic literature review on the forms and automated detection of media bias. *CoRR*, abs/2312.16148, 2023. doi: 10.48550/ARXIV.2312.16148. URL https://doi.org/10.48550/arXiv.2312.16148. (Cited on pages 2, 70, 72, 75, and 97.)

[104] M. Spliethöver, M. Keiff, and H. Wachsmuth. No word embedding model is perfect: Evaluating the representation accuracy for social bias in the media. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 2081–2093. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.FINDINGS-EMNLP.152. URL https://doi.org/10.18653/v1/2022.findings-emnlp.152. (Cited on pages 2, 70, 72, and 98.)

[105] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*, pages 1441–1450. ACM, 2019. (Cited on pages 24 and 25.)

[106] C. R. Sunstein. *Going to Extremes: How Like Minds Unite and Divide*. Oxford University Press, 2009. ISBN 9780199754120. (Cited on pages 6 and 72.)

[107] D. Surís, S. Menon, and C. Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *ICCV*, pages 11854–11864, 2023. (Cited on page 18.)

[108] Y. Talebirad and A. Nadiri. Multi-agent collaboration: Harnessing the power of intelligent LLM agents. *arXiv preprint arXiv:2306.03314*, 2023. (Cited on page 103.)

[109] W. Tang, B. Xu, Y. Zhao, Z. Mao, Y. Liu, Y. Liao, and H. Xie. UniRel: Unified representation and interaction for joint relational triple extraction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 7087–7099. Association for Computational Linguistics, 2022.

[110] W. Tang, Y. Cao, J. Ying, B. Wang, Y. Zhao, Y. Liao, and P. Zhou. A + B: A general generator-reader framework for optimizing LLMs to unleash synergy potential. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 3670–3685. Association for Computational Linguistics, 2024.

[111] W. Tang, Y. Cao, Y. Deng, J. Ying, B. Wang, Y. Yang, Y. Zhao, Q. Zhang, X. Huang, Y. Jiang, and Y. Liao. EvoWiki: Evaluating LLMs on evolving knowledge. In *Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27-August 1st, 2025*. Association for Computational Linguistics, 2025.

[112] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. (Cited on pages 1, 17, 19, 35, and 76.)

[113] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. Canton-Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. (Cited on pages 17 and 32.)

[114] L. Van den Bogaert, D. Geerts, and J. Harambam. Putting a human face on the algorithm: Co-designing recommender personae to democratize news recommender systems. *Digital Journalism*, 12 (8):1097–1117, 2024. doi: 10.1080/21670811.2022.2097101. (Cited on pages 53, 55, and 79.)

[115] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

(Cited on page 37.)

[116] S. Vrijenhoek, G. Bénédict, M. Gutierrez Granada, D. Odijk, and M. de Rijke. Radio–Rank-aware divergence metrics to measure normative diversity in news recommendations. In *Proceedings of the 16th ACM conference on recommender systems*, pages 208–219, 2022. (Cited on page 51.)

[117] J. Wang, H. Lu, J. Caverlee, E. H. Chi, and M. Chen. Large language models as data augmenters for cold-start item recommendation. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 726–729, 2024. (Cited on page 36.)

[118] L. Wang and E. Lim. Zero-shot next-item recommendation using large pretrained language models. *CoRR*, abs/2304.03153, 2023. (Cited on pages 2, 15, and 18.)

[119] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, W. X. Zhao, Z. Wei, and J. Wen. A survey on large language model based autonomous agents. *Frontiers Comput. Sci.*, 18(6):186345, 2024. (Cited on page 18.)

[120] R. Wang, V. Liesaputra, and Z. Huang. A survey on LLM-based news recommender systems. *arXiv preprint arXiv:2502.09797*, 2025. (Cited on pages 1 and 4.)

[121] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua. Kgat: Knowledge graph attention network for recommendation. In *SIGKDD*, pages 950–958, 2019. (Cited on pages 1 and 15.)

[122] Y. Wang, Z. Jiang, Z. Chen, F. Yang, Y. Zhou, E. Cho, X. Fan, X. Huang, Y. Lu, and Y. Yang. Recmind: Large language model powered agent for recommendation. *CoRR*, abs/2308.14296, 2023. (Cited on pages 18 and 19.)

[123] Z. Wang, M. Gao, J. Yu, X. Gao, Q. V. H. Nguyen, S. Sadiq, and H. Yin. LLM-powered text simulation attack against ID-free recommender systems. *CoRR*, abs/2409.11690, 2024. doi: 10.48550/ARXIV.2409.11690. URL https://doi.org/10.48550/arXiv.2409.11690. (Cited on pages 2, 5, 6, 52, 54, 60, 70, 74, and 86.)

[124] Z. Wang, J. Yu, M. Gao, W. Yuan, G. Ye, S. Sadiq, and H. Yin. Poisoning attacks and defenses in recommender systems: A survey. *CoRR*, abs/2406.01022, 2024. doi: 10.48550/ARXIV.2406.01022. URL https://doi.org/10.48550/arXiv.2406.01022. (Cited on pages 54, 69, and 73.)

[125] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022. (Cited on page 20.)

[126] C. Wu, F. Wu, M. An, J. Huang, Y. Huang, and X. Xie. Neural news recommendation with attentive multi-view learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3863–3869. ijcai.org, 2019. doi: 10.24963/IJCAI.2019/536. URL https://doi.org/10.24963/ijcai.2019/536. (Cited on pages 35, 36, 37, 38, 41, 54, 58, 73, and 85.)

[127] C. Wu, F. Wu, S. Ge, T. Qi, Y. Huang, and X. Xie. Neural news recommendation with multi-head self-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6388–6393. Association for Computational Linguistics, 2019. doi: 10.18653/V1/D19-1671. URL https://doi.org/10.18653/v1/D19-1671. (Cited on pages 35, 36, 37, 38, 41, 54, 58, 73, 85, and 86.)

[128] C. Wu, D. Lian, Y. Ge, Z. Zhu, and E. Chen. Triple adversarial learning for influence based poisoning attack in recommender systems. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1830–1840, 2021. (Cited on page 52.)

[129] C. Wu, F. Wu, T. Qi, and Y. Huang. Empowering news recommendation with pre-trained language models. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1652–1656. ACM, 2021. doi: 10.1145/3404835.3463069. URL https://doi.org/10.1145/3404835.3463069. (Cited on pages 4, 19, 36, 37, 38, 40, 42, 47, 48, 54, 58, 67, 73, and 85.)

[130] C. Wu, F. Wu, Y. Yu, T. Qi, Y. Huang, and Q. Liu. NewsBERT: Distilling pre-trained language model for intelligent news application. *arXiv preprint arXiv:2102.04887*, 2021.

[131] C. Wu, F. Wu, T. Qi, and Y. Huang. Are big recommendation models fair to cold users? *arXiv preprint arXiv:2202.13607*, 2022. (Cited on pages 36 and 37.)

[132] C. Wu, F. Wu, Y. Huang, and X. Xie. Personalized news recommendation: Methods and challenges. *ACM Transactions on Information Systems*, 41(1):1–50, 2023. (Cited on pages 37 and 47.)

[133] F. Wu, Y. Qiao, J. Chen, C. Wu, T. Qi, J. Lian, D. Liu, X. Xie, J. Gao, W. Wu, and M. Zhou. MIND: A large-scale dataset for news recommendation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 3597–3606. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.ACL-MAIN.331. URL

https://doi.org/10.18653/v1/2020.acl-main.331. (Cited on pages 5, 36, 41, 58, and 85.)

[134] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie. Self-supervised graph learning for recommendation. In *SIGIR*, pages 726–735, 2021. (Cited on pages 1 and 15.)

[135] J. Wu, X. Wang, X. Gao, J. Chen, H. Fu, T. Qiu, and X. He. On the effectiveness of sampled softmax loss for item recommendation. *ACM Trans. Inf. Syst.*, 42(4), 2024. (Cited on pages 1 and 15.)

[136] J. Wu, S. Yang, R. Zhan, Y. Yuan, L. S. Chao, and D. F. Wong. A survey on LLM-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, pages 1–65, 2025. (Cited on pages 1, 3, and 103.)

[137] L. Wu, Z. Zheng, Z. Qiu, H. Wang, H. Gu, T. Shen, C. Qin, C. Zhu, H. Zhu, Q. Liu, et al. A survey on large language models for recommendation. *World Wide Web*, 27(5):60, 2024. (Cited on pages 1, 3, and 37.)

[138] Y. Xi, W. Liu, J. Lin, J. Zhu, B. Chen, R. Tang, W. Zhang, R. Zhang, and Y. Yu. Towards open-world recommendation with knowledge augmentation from large language models. *CoRR*, abs/2306.10933, 2023. (Cited on pages 15, 18, and 19.)

[139] S. Xiao, Z. Liu, Y. Shao, T. Di, B. Middha, F. Wu, and X. Xie. Training large-scale news recommenders with pretrained language models in the loop. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4215–4225, 2022. (Cited on pages 36, 37, 38, and 40.)

[140] S. Xu, W. Hua, and Y. Zhang. Openp5: Benchmarking foundation models for recommendation. *CoRR*, abs/2306.11134, 2023. (Cited on pages 18 and 25.)

[141] Y. Yada and H. Yamana. News recommendation with category description by a large language model. *arXiv preprint arXiv:2405.13007*, 2024. (Cited on page 37.)

[142] Y. Yang, J. Cao, M. Lu, J. Li, and C. Lin. How to write high-quality news on social network? predicting news quality by mining writing style. *CoRR*, abs/1902.00750, 2019. URL http://arxiv.org/abs/1902.00750. (Cited on pages 53, 55, 56, and 79.)

[143] Z. Yang, L. Li, J. Wang, K. Lin, E. Azarnasab, F. Ahmed, Z. Liu, C. Liu, M. Zeng, and L. Wang. MM-REACT: prompting chatgpt for multimodal reasoning and action. *CoRR*, abs/2303.11381, 2023. (Cited on page 18.)

[144] Z. Yang, J. Wu, Y. Luo, J. Zhang, Y. Yuan, A. Zhang, X. Wang, and X. He. Large language model can interpret latent space of sequential recommender. *CoRR*, abs/2310.20487, 2023. (Cited on page 18.)

[145] Z. Yang, J. Wu, Z. Wang, X. Wang, Y. Yuan, and X. He. Generate what you prefer: Reshaping sequential recommendation via guided diffusion. In *NeurIPS*, 2023. (Cited on pages 1 and 15.)

[146] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models. In *ICLR*, 2023. (Cited on pages 16, 20, and 21.)

[147] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang. A survey on large language model (LLM) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, 4(2):100211, 2024. ISSN 2667-2952. doi: https://doi.org/10.1016/j.hcc.2024.100211. URL https://www.sciencedirect.com/science/article/pii/S266729522400014X. (Cited on pages 2 and 69.)

[148] J. Yi, F. Wu, C. Wu, R. Liu, G. Sun, and X. Xie. Efficient-FedRec: Efficient federated learning framework for privacy-preserving news recommendation. *arXiv preprint arXiv:2109.05446*, 2021. (Cited on pages 36, 37, 38, 47, and 48.)

[149] A. Zhang, Y. Chen, L. Sheng, X. Wang, and T.-S. Chua. On generative agents in recommendation. In *SIGIR*, 2024. (Cited on pages 19, 25, and 103.)

[150] H. Zhang, C. Tian, Y. Li, L. Su, N. Yang, W. X. Zhao, and J. Gao. Data poisoning attack against recommender system using incomplete and perturbed data. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 2154–2164. ACM, 2021. doi: 10.1145/3447548.3467233. URL https://doi.org/10.1145/3447548.3467233. (Cited on pages 54 and 73.)

[151] J. Zhang, Y. Hou, R. Xie, W. Sun, J. J. McAuley, W. X. Zhao, L. Lin, and J. Wen. Agentcf: Collaborative learning with autonomous language agents for recommender systems. *CoRR*, abs/2310.09233, 2023. (Cited on page 25.)

[152] J. Zhang, R. Xie, Y. Hou, W. X. Zhao, L. Lin, and J. Wen. Recommendation as instruction following: A large language model empowered recommendation approach. *CoRR*, abs/2305.07001, 2023. (Cited on pages 18 and 19.)

[153] J. Zhang, Y. Liu, Q. Liu, S. Wu, G. Guo, and L. Wang. Stealthy attack on large language model based recommendation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 5839–

5857. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.318. URL https://doi.org/10.18653/v1/2024.acl-long.318. (Cited on pages 2, 5, 6, 52, 53, 54, 58, 60, 63, 70, 74, 85, and 86.)

[154] Q. Zhang, J. Li, Q. Jia, C. Wang, J. Zhu, Z. Wang, and X. He. UNBERT: User-news matching BERT for news recommendation. In *International Joint Conference in Artificial Intelligence*, volume 21, pages 3356–3362, 2021. (Cited on pages 15, 19, 36, and 37.)

[155] S. Zhang, L. Yao, A. Sun, and Y. Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.*, 52(1):5:1–5:38, 2019. (Cited on page 1.)

[156] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. T. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer. OPT: open pre-trained transformer language models. *CoRR*, abs/2205.01068, 2022. doi: 10.48550/ARXIV.2205.01068. (Cited on pages 52, 60, 70, 74, and 86.)

[157] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019. (Cited on page 61.)

[158] X. Zhang, Z. Wang, J. Zhao, and L. Wang. Targeted data poisoning attack on news recommendation system by content perturbation. *arXiv preprint arXiv:2203.03560*, 2022. (Cited on page 52.)

[159] Z. Zhang and B. Wang. Prompt learning for news recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 227–237, 2023. (Cited on pages 36 and 41.)

[160] H. Zhao, H. Chen, T. A. Ruggles, Y. Feng, D. Singh, and H.-J. Yoon. Improving text classification with large language model-based data augmentation. *Electronics*, 13(13):2535, 2024. doi: 10.3390/electronics13132535. (Cited on pages 55 and 79.)

[161] Y. Zhao, X. Wang, J. Chen, Y. Wang, W. Tang, X. He, and H. Xie. Time-aware path reasoning on knowledge graph for recommendation. *ACM Transactions on Information Systems*, 41(2):26:1–26:26, 2023.

[162] Y. Zhao, J. Huang, and M. de Rijke. Can LLMs serve as user simulators for recommender systems? *The Search Futures Workshop at ECIR 2024, SIGIR Forum*, 58(1):1–41, 2024.

[163] Y. Zhao, J. Wu, X. Wang, W. Tang, D. Wang, and M. de Rijke. Let me do it for you: Towards LLM empowered recommendation via tool learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 1796–1806. ACM, 2024.

[164] Y. Zhao, J. Huang, S. Liu, J. Wu, X. Wang, and M. de Rijke. LANCE: Exploration and reflection for LLM-based textual attacks on news recommender systems. In *RecSys 2025: 19th ACM Conference on Recommender Systems*. ACM, September 2025. (Cited on page 87.)

[165] Y. Zhao, J. Huang, D. Vos, and M. de Rijke. Revisiting language models in neural news recommender systems. In *Advances in Information Retrieval - 47th European Conference on Information Retrieval, ECIR 2025, Lucca, Italy, April 6-10, 2025, Proceedings, Part IV*, volume 15575 of *Lecture Notes in Computer Science*, pages 161–176. Springer, 2025. (Cited on pages 1, 4, 51, 54, 55, 58, 60, 67, 68, 73, 85, and 87.)

[166] Y. Zhao, Y. Li, J. Huang, X. Wang, and M. de Rijke. Unseen threats: Media bias-aware textual attacks on news recommender systems. *Submitted for review*, 2025.

[167] Z. Zhao, W. Fan, J. Li, Y. Liu, X. Mei, Y. Wang, Z. Wen, F. Wang, X. Zhao, J. Tang, et al. Recommender systems in the era of large language models (LLMs). *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6889–6907, 2024. (Cited on pages 3 and 103.)

[168] W. Zhong, L. Guo, Q. Gao, H. Ye, and Y. Wang. Memorybank: Enhancing large language models with long-term memory. In *AAAI*, pages 19724–19731, 2024. (Cited on page 18.)

[169] X. Zhu, Y. Chen, H. Tian, C. Tao, W. Su, C. Yang, G. Huang, B. Li, L. Lu, X. Wang, et al. Ghost in the minecraft: Generally capable agents for open-world enviroments via large language models with text-based knowledge and memory. *CoRR*, abs/2305.17144, 2023. (Cited on page 18.)

[170] Y. Zhu, H. Yuan, S. Wang, J. Liu, W. Liu, C. Deng, Z. Dou, and J. Wen. Large language models for information retrieval: A survey. *CoRR*, abs/2308.07107, 2023. (Cited on pages 1, 3, and 15.)

[171] B. Zoph, I. Bello, S. Kumar, N. Du, Y. Huang, J. Dean, N. Shazeer, and W. Fedus. St-moe: Designing stable and transferable sparse expert models, 2022. URL https://arxiv.org/abs/2202.08906. (Cited on page 97.)

[172] L. Zou, S. Zhang, H. Cai, D. Ma, S. Cheng, S. Wang, D. Shi, Z. Cheng, and D. Yin. Pre-trained language model based ranking in baidu search. In *KDD*, pages 4014–4022. ACM, 2021. (Cited on page 19.)

Summary

In the era of large language models (LLMs), recommender systems (RSs) face a double-edged challenge: harnessing the generative power of LLMs to enrich recommendation performance while confronting the threat of LLM-enabled manipulation. Motivated by this tension, this thesis addresses a key scientific question: how to build a reliable recommender systems in the era of LLMs? We explore this question through two complementary themes: (i) enhancing recommendations by mitigating hallucinations and under-representation, and (ii) exposing vulnerabilities via LLM-driven textual attacks.

We first address the problem of hallucinations in general, interaction-based recommendation domains such as movie and e-commerce platforms. We introduce a framework that treats an LLM as a *surrogate user* who iteratively queries external tools to refine candidate items. Specifically, the model integrates attribute-oriented tools and a memory strategy, guided by a multi-round reasoning process, to refine the item space and generate recommendations that are both relevant and diverse. Experiments in Chapter 2 show that the proposed framework improves accuracy and diversity while reducing hallucinated recommendations.

Next, we continue our research in the news domain, addressing the challenge of under-representation, where user behavior misaligns with rich textual representations. Chapter 3 conducts a systematic study of language models (GloVe, BERT, RoBERTa, and Llama) as news encoders. By exploring strategies such as layer-specific fine-tuning and domain-specific tasks, we achieve enhanced recommendation relevance. The results underscore the effectiveness of tailored fine-tuning in bridging the gap between textual data and user preferences, offering a robust solution for news RSs.

We then move to the second theme of LLM-powered exploitation by introducing a two-stage framework that uses LLMs to rewrite news content and manipulate recommendation outcomes. The Explorer generates diverse rewrite variants, while the Reflector fine-tunes the model to identify effective attack patterns. Experiments in Chapter 4 demonstrate the effectiveness of the proposed framework and reveal that negative and neutral rewrites prove particularly effective in the news domain. This work exposes critical vulnerabilities in news RSs and demonstrates the power of LLMs for adversarial manipulation.

Deepening the threat analysis, Chapter 5 extends the investigation of textual attacks by proposing a framework to boost news rankings while preserving media bias orientation. Using a progressive fine-tuning strategy, we train specialized LLM experts to balance ranking improvement and bias consistency, merging their capabilities to produce stealthy, bias-aware rewrites. We conduct extensive experiments, and the results show that our model achieves superior rank gains alongside strong bias preservation across all ideological segments. These findings emphasize the complexity of LLM-driven attacks and the urgent need for defenses.

Finally, in Chapter 6, we summarize our findings, highlight their practical implications, and propose future directions. On the enhancement side, we outline pathways for improving domain adaptability and user modeling. On the defense side, we identify directions for increasing attack realism and devising defense strategies.

Samenvatting

In het tijdperk van grote taalmodellen (LLMs) bevinden aanbevelingssystemen (RSs) zich in een spanningsveld tussen enerzijds het benutten van de generatieve kracht van LLMs om de aanbevelingsprestaties te verbeteren, en anderzijds het risico van ongewilde manipulaties door LLMs. Gemotiveerd door dit spanningsveld behandelt dit proefschrift de volgende onderzoeksvraag: hoe kunnen we betrouwbare aanbevelingssystemen bouwen in het tijdperk van LLMs? We benaderen deze vraag met twee complementaire oogmerken: (i) het verbeteren van aanbevelingen door hallucinaties en ondervertegenwoordiging te verminderen, en (ii) het blootleggen van kwetsbaarheden die onstaan in aanbevelingssystemen door tekstuele manipulaties van LLMs.

Allereerst richten we ons op het probleem van hallucinaties in algemene, interactiegebaseerde aanbevelingsdomeinen, zoals film- en e-commerceplatforms. We introduceren een framework waarin een LLM wordt behandeld als een surrogaatgebruiker die iteratief externe tools raadpleegt om items die potentieel aanbevolen kunnen worden te selecteren. Het model integreert attribuutgerichte tools en een geheugenstrategie, aangestuurd door een redeneerproces van meerdere stappen, om de selectie van items te verfijnen en aanbevelingen te genereren die zowel relevant als divers zijn. Experimenten in Hoofdstuk 2 tonen aan dat het voorgestelde framework de nauwkeurigheid en diversiteit van aanbevelingen verbetert, terwijl het aantal gehallucineerde aanbevelingen afneemt.

We zetten ons onderzoek voort in het nieuwsaanbevelingsdomein, gericht op het probleem van ondervertegenwoordiging, waarbij gebruikersgedrag niet goed aansluit bij rijke tekstuele representaties. Hoofdstuk 3 voert een systematische studie uit naar taalmodellen (GloVe, BERT, RoBERTa, en Llama) als nieuwsencoders. Door strategieën zoals het fine-tunen van specifieke lagen en het verkennen van domeinspecifieke taken, bereiken we een hogere kwaliteit van aanbevelingen. De resultaten benadrukken de effectiviteit van het gericht fine-tunen van nieuwsencoders om de kloof tussen tekstuele data en gebruikersvoorkeuren te overbruggen. Dit resulteert in een robuuste oplossing voor nieuwsaanbevelingssystemen.

Hierna gaan we over naar het tweede thema van deze thesis: het misbruiken van LLMs voor manipulatie doeleinden. We introduceren een tweeledig framework dat LLMs inzet om nieuwsartikelen te herschrijven en aanbevelingsresultaten te manipuleren. Enerzijds genereert de *Explorer* diverse alternatieven voor nieuwsartikelen, terwijl anderzijds de *Reflector* het model leert om effectieve aanvalspatronen te herkennen. Experimenten in Hoofdstuk 4 tonen de effectiviteit van het voorgestelde framework aan en onthullen dat vooral negatieve en neutrale herschrijvingen bijzonder effectief zijn in het nieuwsdomein. Dit werk legt kritieke kwetsbaarheden bloot in aanbevelingssystemen voor nieuws, en toont de kracht van LLMs aan voor het manipuleren van artikelen.

Hoofdstuk 5 breidt het onderzoek naar manipulatie van aanbevelingssystemen door tekstuele herschrijvingen uit door een framework voor te stellen dat nieuwsartikelen hoger rankt terwijl de mediabias behouden blijft. Met een progressieve *fine-tuning* strategie trainen we gespecialiseerde LLMs die rankingverbetering en biasconsistentie in balans houden. Op deze manier kunnen subtiele, bias-bewuste herschrijvingen worden gegenereerd. Uitgebreide experimenten tonen aan dat ons model aanzienlijke

verbeteringen in rankings behaalt, terwijl de mediabias minimaal verandert binnen alle ideologische segmenten. Deze bevindingen onderstrepen de complexiteit van LLM-gestuurde manipulaties en de dringende noodzaak tot verdediging.

Tot slot vatten we in Hoofdstuk 6 onze bevindingen samen, benadrukken we de praktische implicaties en stellen we richtingen voor toekomstig onderzoek voor. Als mogelijke verbeteringen stellen we voor om domeinadaptiviteit en gebruikersmodellering te verder te ontwikkelen. Aan de verdedigingskant identificeren we richtingen om het realisme van aanvallen te vergroten en doeltreffende verdedigingsstrategieën te ontwerpen.