UNIVERSITY OF CALIFORNIA SAN DIEGO

Accuracy, Explainability and Interactivity: Towards Conversational Recommender Systems

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Zhankui He

Committee in charge:

Professor Julian McAuley, Chair
Professor Sicun Gao
Professor Zhiting Hu
Professor Jingbo Shang

2024

The Dissertation of Zhankui He is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

DEDICATION

To my beloved family and my dreams.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I am profoundly grateful to my advisor, Prof. Julian McAuley, for his unwavering guidance, support, and invaluable mentorship throughout my PhD journey. His insights and support have fundamentally shaped my academic and professional development.

My sincere thanks extend to my dissertation committee members, Prof. Sicun Gao, Prof. Zhiting Hu, and Prof. Jingbo Shang, for their indispensable feedback on my research approach.

I am eternally grateful for my parents, whose sacrifices and encouragement empowered me to pursue my dreams. Special thanks to my wife, Shihan Ran; her love and support have been my bedrock. Through every challenge, she has been my constant inspiration, helping me stay focused and overcome obstacles. I also express gratitude to my youngest family member, Chuaichuai, the cutest golden retriever who brings us sunshine and happiness.

I have had the honor of collaborating with incredibly talented individuals from both academia and industry: Handong Zhao, Tong Yu, Sungchul Kim, Du Fan, Zhaowen Wang, Zhe Lin, Ajinkya Kale (Adobe Research); Harald Steck, Rahul Jha, Dawen Liang, Yesu Feng, Nathan Kallus (Netflix); Wang-Cheng Kang, Jianmo Ni, Noveen Sachdeva, Benjamin Coleman, Derek Zhiyuan Cheng (Google DeepMind). Many thanks to my inspiring labmates Jiacheng Li, Zexue He, An Yan, Yupeng Hou, Zhouhang Xie, Canwen Xu, Yu Wang, Se-eun Yoon, Nafis Sadeq, Bodhisattwa Prasad Majumder, Junda Wu, Jessica Maria Echterhoff and all other members from my lab. And I extend my sincere thanks to other amazing external collaborators as well. Thanks to Zhenrui Yue (UIUC), Huimin Zeng (UIUC), Yueqi Wang (Berkeley), Xiusi Chen (UCLA), Yashar Deldjoo (Polytechnic University of Bari), Hengchang Hu (NUS), Min-Yen Kan (NUS), Shuchang Liu (Kuaishou), Qingpeng Cai (Kuaishou), Yu Xia (SJTU), Tianyang Zhang (Google), Zheng Chen (Amazon), Ziyan Jiang (Amazon), Fan Yang (Amazon), Zeyuan Zhang (UCSD), Tanmay Laud (Megagon Labs), Zihang He (UCSD), Xiaojie Chen (UCSD), Xinshuang Liu (UCSD), Taiqi Liu (Umich), Haozhe Xu (Fudan), Haitian Jiang (NYU), Xin Peng (NYU) and all the other collaborators. I am also immensely grateful for the inspiration, and generous support in discussions with Xiangnan He (USTC), Fuli Feng (USTC), Vito Walter Anelli (Polytechnic

University of Bari), Xiaolei Wang (RUC), Yisong Miao (NUS), and Ke Chen (UCSD). I truly appreciate the wisdom, friendships, and help from all of them and more.

Finally, I am grateful to my co-authors for their kind approval of the publications and material included in my dissertation.

Chapter 2, in part, is a reprint of the material as it appears in "Locker: Locally constrained self-attentive sequential recommendation." by Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley, in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management. 2021*. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in part, is a reprint of the material as it appears in "Query-Aware Sequential Recommendation." by Zhankui He, Handong Zhao, Zhaowen Wang, Zhe Lin, Ajinkya Kale, and Julian McAuley, in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management. 2022*. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in part, is a reprint of the material as it appears in "UCEpic: Unifying Aspect Planning and Lexical Constraints for Generating Explanations in Recommendation." by Jiacheng Li*, Zhankui He*, Jingbo Shang, and Julian McAuley, in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2023. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in part, is a reprint of the material as it appears in "Personalized Showcases: Generating multi-modal explanations for recommendations." by An Yan*, Zhankui He*, Jiacheng Li*, Tianyang Zhang, and Julian McAuley in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2023*. The dissertation author was the primary investigator and author of this paper.

Chapter 6, in part, is a reprint of the material as it appears in "Bundle MCR: Towards Conversational Bundle Recommendation." by Zhankui He, Handong Zhao, Tong Yu, Sungchul Kim, Fan Du, and Julian McAuley, in *Proceedings of the 16th ACM Conference on Recommender*

Systems. 2022. The dissertation author was the primary investigator and author of this paper.

Chapter 7, in part, is a reprint of the material as it appears in "Large language models as zero-shot conversational recommenders." by Zhankui He*, Zhouhang Xie*, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley in *Proceedings of the 32nd ACM International Conference on Information & knowledge management. 2023*. The dissertation author was the primary investigator and author of this paper.

VITA

2015–2019 B.S., Fudan University

2019–2021 M.S., University of California San Diego

2019–2024 Ph.D., University of California San Diego

PUBLICATIONS

Zhankui He*, Zhouhang Xie*, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. "Large language models as zero-shot conversational recommenders."The 32nd ACM international conference on information and knowledge management, 2023.

Zhankui He, Handong Zhao, Tong Yu, Sungchul Kim, Fan Du, and Julian McAuley. "Bundle MCR: Towards Conversational Bundle Recommendation." The 16th ACM Conference on Recommender Systems, 2022.

Zhankui He, Handong Zhao, Zhaowen Wang, Zhe Lin, Ajinkya Kale, and Julian Mcauley. "Query-Aware Sequential Recommendation." The 31st ACM International Conference on Information & Knowledge Management, 2022.

Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley. "Locker: Locally constrained self-attentive sequential recommendation." The 30th ACM International Conference on Information & Knowledge Management, 2021.

Jiacheng Li*, Zhankui He*, Jingbo Shang, and Julian McAuley. "Ucepic: Unifying aspect planning and lexical constraints for generating explanations in recommendation." The 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023.

An Yan*, Zhankui He*, Jiacheng Li*, Tianyang Zhang, and Julian McAuley. "Personalized Showcases: Generating multi-modal explanations for recommendations." The 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2023.

Zhenrui Yue*, Zhankui He*, Huimin Zeng, and Julian McAuley. "Black-box attacks on sequential recommenders via data-free model extraction." The 15th ACM Conference on Recommender Systems, 2021.

Zhiqiang Shen*, Zhankui He*, and Xiangyang Xue. "Meal: Multi-model ensemble via adversarial learning." The AAAI Conference on Artificial Intelligence, 2019.

Se-eun Yoon, Zhankui He, Jessica Echterhoff, Julian McAuley. "Evaluating large language

models as generative user simulators for conversational recommendation." The North American Chapter of the Association for Computational Linguistics, 2024.

Hou, Yupeng, Zhankui He, Julian McAuley, and Wayne Xin Zhao. "Learning vector-quantized item representation for transferable sequential recommenders." The ACM Web Conference, 2023.

Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. "Adversarial personalized ranking for recommendation." The 41st International ACM SIGIR conference on research & development in information retrieval, 2018.

Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. "NAIS: Neural attentive item similarity model for recommendation." IEEE Transactions on Knowledge and Data Engineering, 2018.

Yue, Zhenrui, Yueqi Wang, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. "Linear Recurrent Units for Sequential Recommendation." The 17th ACM International Conference on Web Search and Data Mining, 2024

Yu Xia, Tong Yu, Zhankui He, Handong Zhao, Julian McAuley, Shuai Li. "Aligning as debiasing: Causality-aware alignment via reinforcement learning with interventional feedback." The North American Chapter of the Association for Computational Linguistics, 2024.

Yu Wang, Zexue He, Zhankui He, Hao Xu, and Julian McAuley. "Deciphering Compatibility Relationships with Textual Descriptions via Extraction and Explanation." The 38th Annual AAAI Conference on Artificial Intelligence, 2024.

Liu, Shuchang, Qingpeng Cai, Zhankui He, Bowen Sun, Julian McAuley, Dong Zheng, Peng Jiang, and Kun Gai. "Generative flow network for listwise recommendation." The 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2023.

Canwen Xu, Zexue He, Zhankui He, and Julian McAuley. "Leashing the Inner Demons: Self-Detoxification for Language Models." The AAAI Conference on Artificial Intelligence, 2022.

Zeyuan Zhang*, Tanmay Laud*, Zihang He*, Xiaojie Chen, Xinshuang Liu, Zhouhang Xie, Julian McAuley, Zhankui He. "RecWizard: A Toolkit for Conversational Recommender Systems with Modular, Portable Models and Interactive User Interfaces." The AAAI Conference on Artificial Intelligence, 2024, Demo Track.

ABSTRACT OF THE DISSERTATION

Accuracy, Explainability and Interactivity: Towards Conversational Recommender Systems

by

Zhankui He

Doctor of Philosophy in Computer Science

University of California San Diego, 2024

Professor Julian McAuley, Chair

Recommender systems are fundamental to numerous personalization services. However, traditional systems often lack the ability to proactively explore user interests, explain their choices, or adapt immediately to user feedback, leaving users unsatisfied with the recommendation results. Conversational recommender systems, instead, offer a promising future with the capability of understanding both explicit textual feedback and implicit behavior patterns. Such systems accurately model user sequential interactions, proactively engage with users to explore preferences and provide explanations and context for their recommendations through conversations. The research on conversational recommender systems marks a crucial step in the evolution from traditional recommender systems to truly personalized intelligent agents.

In this dissertation, our research focuses on three pillars in developing effective conversational recommender systems: (1) Accuracy: modeling the dynamic and evolving nature of user preferences within sequential behaviors to ensure recommendations better match the user's needs. (2) Explainability: improving the quality and expressiveness of explanations that accompany recommendations to enhance user understanding and trust. (3) Interactivity: exploring multi-round conversational capabilities to support complex recommendation scenarios, including managing bundled item recommendations and seamlessly integrating with Large Language Models for a natural-language-driven user experience.

By advancing conversational recommender systems with the components for accuracy, explainability, and interactivity, this research paves the way for future research to create powerful conversational recommender systems that reshape how users discover and interact with products and services.

Chapter 1

Introduction

Recommender systems hold the potential to serve as *personalized intelligent agents* capable of comprehending human preferences and interests, subsequently providing personalized products or information tailored to their enjoyment. These systems enable users to effectively manage information overload, clarify their needs, and easily access desired items. Meanwhile, platforms employing recommender systems can adeptly understand user needs, manage product supply, enhance user engagement, and ultimately boost business profits. The proficiency of recommender systems in understanding user preferences and delivering pertinent suggestions renders them invaluable tools for both individuals and organizations.

However, a gap persists between the envisioned personalized intelligent agents and current recommender systems. Currently, mainstream recommender systems often exhibit several limitations. First, they are mostly *static*, meaning they learn static user representations from historical user-item interaction data. They lack the adaptability to accurately model users' evolving preferences from their sequential actions. Second, they are mostly *silent*, as they often solely offer a list of recommended items (e.g., movies) but fail to provide additional information (such as justifications) or to engage meaningfully with direct user feedback (e.g., textual comments like "I don't like this recommendation because of the color, please change it.") beyond simple clicks or ratings. Third, they are mainly based on *one-shot paradigms* because the focus is on achieving successful recommendations in a single attempt. Such systems generally

lack the capacity for multi-round interactions that allow them to actively solicit feedback or pose questions. Such limitations collectively impede recommender systems in several key ways. They hinder efforts to enhance recommendation accuracy, build trust with the user, handle more complex and challenging recommendation scenarios, thus struggle to fully satisfy user needs.

In this dissertation, our core idea to mitigate or even overcome the issues outlined above is to let recommender systems and users *converse*. In other words, we are interested in a new research direction, namely *Conversational Recommender Systems (CRS)*. Considering the limitations of traditional recommender systems discussed above, our expected properties for the proposed CRS should encompass the following:

1. Beyond static user representations: Accurately modeling users' dynamic (mostly sequential) actions, effectively capturing historical user preferences and emerging user dynamics within current sessions;

2. Beyond item lists only: Generating more evidence to serve as explanations for the recommended item lists, supporting users' decision-making processes;

3. Beyond one-shot paradigms: Making recommendations with single- or multiple-round interactions, such as asking questions or answering users' questions.

1.1 Our Framework

Motivated by those three properties, our research in this dissertation towards Conversational Recommendation System is naturally focusing on three different and interlinked pillars: *accuracy*, *explainability*, and *interactivity*, which are illustrated in Figure 1.1.

- **Accuracy**: At the heart of recommender systems lies the ability to model user dynamics and deliver accurate recommendations, including the new CRS paradigm. Due to the inherent nature of CRS, we have a particular interest in modeling sequential user behaviors, capturing both long-term preferences and short-term interests in the given sessions. Here,

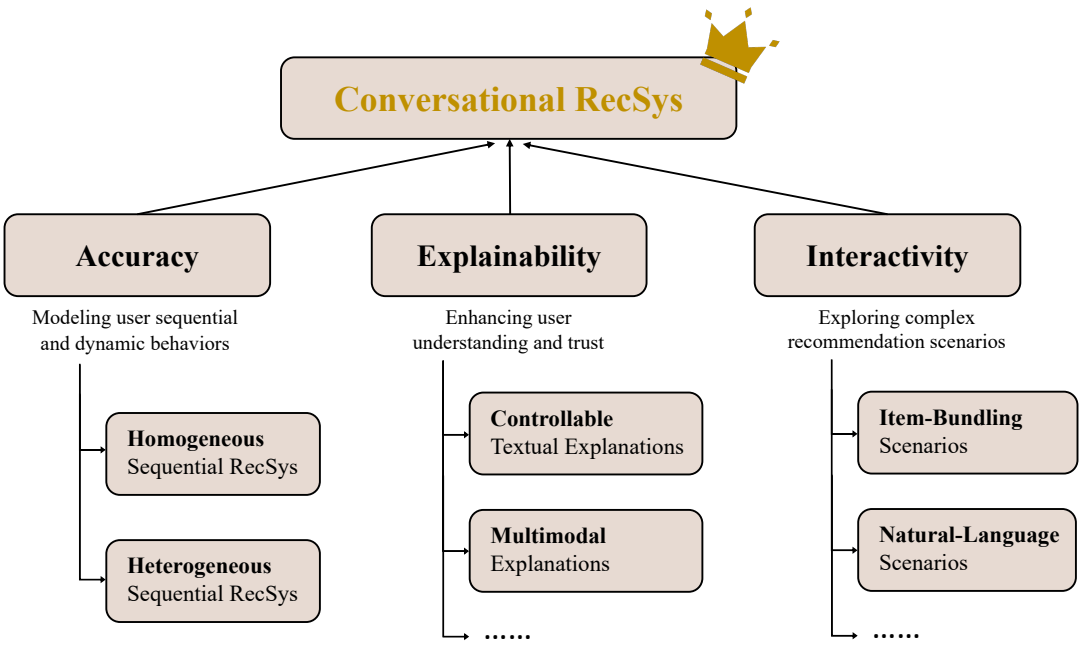


Figure 1.1. Our research framework towards conversational recommender systems.

we mainly investigate them with the testbed in sequential recommendations and consider two cases where the user historical sequences may be homogeneous (i.e., single-type historical interactions) or heterogeneous (i.e., multi-type historical interactions) with the technical contributions about how to improve the model recommendation accuracy.

- **Explainability**: To enhance the expressive nature of recommendations and offer additional information to support users' decision-making processes, we aim to generate personalized explanations or justifications that go beyond simply presenting recommended item lists. This pillar focuses on how to generate accurate and informative explanations for different users. To this end, we explored two directions about explanation generations. The first direction is about how to incorporate highly specific product information with controllable generation, and the second direction is about how to generate visually grounded results and serve as multimodal explanations.

- **Interactivity**: We explore how CRS unlocks new and engaging recommendation scenarios that cannot be handled by traditional recommender systems, by introducing multi-round

conversational mechanisms. This pillar includes two different recommendation scenarios we explored. The first scenario is about generating recommendations with complex structures like item bundles, and the second scenario is about understanding natural language dialogues and making recommendations by Large Language Models. This pillar explores the boundaries of CRS by leveraging interactivity and new emerging techniques.

1.2 Dissertation Organization

In Chapter 1, we present the background, motivations, and framework of focusing on accuracy, explainability, and interactivity towards conversational recommender systems. We explain what are the limitations of the current mainstream recommender systems, and why directions we need to work to improve such recommender systems and overcome those limitations. Then we discuss the research directions in the following three parts, in detail.

The first part is about *accuracy*, i.e., accurately modeling user sequential and dynamic behaviors. We investigate this topic in Chapter 2 and Chapter 3, mainly focusing on two examples of how to effectively capture long-term user preferences and short-term dynamics given homogeneous and heterogeneous user interaction sequences.

In Chapter 2, we first show that, for homogeneous sequential recommender systems, although self-attentive [Vaswani et al., 2017a, Devlin et al., 2019, Kang and McAuley, 2018, Sun et al., 2019] models are the *de facto* options for modeling user sequential behaviors, the standard self-attention mechanisms overly focus on distant historical user behaviors. This is because, without additional locality inductive bias, self-attention mechanisms fail to sufficiently attend to recent user preferences from limited training user sequences. However, we reveal that such recency signals play important roles in the accuracy of sequential recommenders. Motivated by this, we propose a framework, LOCKER to explicitly introduce a focus on recent user interactions and boost model recommendation accuracy.

In Chapter 3, we investigate a new setting for heterogeneous sequential recommender systems, where user action sequences (e.g., clicks) are punctuated by an additional signal – *user*

textual queries, namely *Query-Aware Sequential Recommendation*. Compared to traditional recommendations, this setting moves a step towards CRS where models are required to handle heterogenous signals from user actions and textual inputs. In this chapter, we show that *user queries* are strong clues to uncover the user dynamic interests. Our proposed model, QUERYSR exhibits superior recommendation performance where *user queries* are incorporated, as validated by offline experiments and in-depth case studies.

The second part is about *explainability*, to enhance user understanding and trust. We studied controllable and multi-modal explanation generations in Chapter 4 and Chapter 5 respectively, where those techniques share the same objective for generating more accurate and informative explanations or justifications for users to make decisions.

In Chapter 4, we reveal that many existing explanation generators struggle to include faithful and informative keyphrases in explanation generation, which hurts users' trust. Inspired by the recent advancement of insertion-based language generators, we propose a framework, UCEPIC, to combine aspect planning and lexical constrained generation into a single language model. This model provides more personalized, informative, and faithful textual explanations along with recommendation items, and is validated by automatic evaluations as well as human evaluation.

In Chapter 5, we argue that generated explanations should go beyond text only, thus we introduce a new visual modality in explanation generation. In detail, we identify the first bottleneck is the lack of datasets with visual and textual explanations as ground truth. We thus collect a large-scale dataset from Google Local Reviews and construct explanation pairs from the data dump for restaurant recommendations. We further design a framework, P-SHOWCASE, to present the images of interest for users, as well as generate relevant and informative textual generations at the same time.

The third part is about *interactivity*, to explore complex recommendation scenarios that cannot be handled by traditional recommender systems. CRS facilitates intricate recommendation scenarios challenging for traditional systems. Chapter 6 delves into conversational bundle

recommendations, while Chapter 7 explores natural-language-based conversational recommendations.

In Chapter 6, we investigate the challenging task of bundle recommendation within the context of conversational recommendations. Bundle recommendation involves suggesting multiple dependent items (e.g., outfits, playlists) to users rather than individual items. This scenario offers additional benefits to users and platforms but poses exceptional challenges to traditional recommendation methods due to its complexity and limited user signals. We reexamine this task in a conversational setting, referred to as BUNDLEMCR, and demonstrate that the conversational mechanism breaks down the bundling process, easing tasks across multiple real-world datasets, and opening up new opportunities for research in this field.

In Chapter 7, we focus on a recommendation scenario with free-form natural language inputs and outputs from both the user and system sides. This is a challenging setting historically due to the limitations of models' abilities to understand and generate high-quality natural language content. However, recent LLMs demonstrate impressive abilities in language-related tasks. Hence, in this chapter, we systematically study how to use LLMs in natural-language-based CRS settings and with related insights, and further uncover the limitations for future research.

In Chapter 8 we describe the research works related to these three pillars of conversational recommender systems. In Chapter 9, we summarize our research contributions and discuss future directions.

Part I

Accuracy: Modeling User Sequential and Dynamic Behaviors

Chapter 2

Homogeneous Sequential Recommender Systems

In this chapter, we introduce a model design to improve the recommendation accuracy of homogeneous sequential recommender systems. Homogeneous sequential recommender systems aim to model user interaction sequences consisting of single-type actions, such as clicking. This serves as a popular and fundamental sequential recommendation setting and remains challenging since we lack more auxiliary data to help understand users' long-term preferences and short-term dynamics. Recently, self-attentive models have demonstrated significant potential in the homogeneous sequential recommendation. Their ability to simultaneously capture long-term user preferences and short-term dynamics from user click signals makes them particularly effective. However, we argue that the non-local nature of self-attention modules hinders their ability to accurately model short-term user dynamics in homogeneous sequential recommender systems due to a lack of inductive local bias.

To investigate this hypothesis, we conduct a controlled analytical experiment focusing on "short-term" scenarios. The results reveal a substantial performance gap between self-attentive recommenders with and without local constraints. This finding supports our hypothesis that existing self-attentive recommenders may not sufficiently model short-term user dynamics. Motivated by this observation, we propose LOCKER, a simple plug-and-play framework designed to enhance self-attentive recommenders. LOCKER combines local encoders with existing global

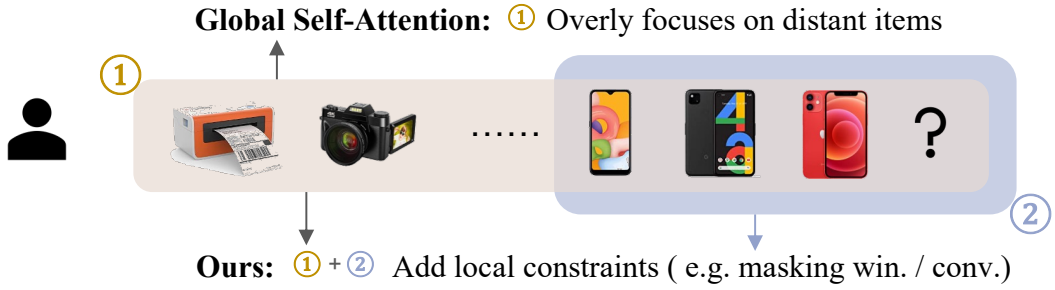


Figure 2.1. Global self-attention tends to overly focus on distant items.

attention heads, improving short-term user dynamics modeling while preserving the long-term semantics captured by standard self-attentive encoders. We evaluate LOCKER using five different local methods and find that it consistently outperforms state-of-the-art self-attentive recommenders on three datasets, which makes LOCKER stand as one of the state-of-the-art sequential recommender systems in terms of the recommendation accuracy according to user homogeneous interactions.

2.1 Introduction

Sequential recommenders seek to recommend accurate items based on user historical actions in chronological order. Most sequential recommender studies are conducted on homogeneous user interaction sequences (e.g., interacted items only), where one important research question is how to capture both long-term user preferences (e.g., preference for action movies) and short-term context of recent actions(e.g., the last movie watched) without other auxiliary data. Many approaches, such as Markov Chain (MC) based [Rendle et al., 2010, He and McAuley, 2016a] and RNN/CNN based methods [Li et al., 2017a, Tang and Wang, 2018, Yuan et al., 2019, Devooght and Bersini, 2017], are proposed in this research direction to enhance recommendation accuracy.

Recently, self-attentive recommenders have become the state-of-the-art to model both *long-* and *short-term* user preferences [Kang and McAuley, 2018, Sun et al., 2019, Li et al., 2020d, Wu et al., 2020] in sequential recommendation tasks with homogeneous (item-only)

sequential data. These recommenders leverage self-attention mechanisms [Vaswani et al., 2017a] to calculate item-to-item attention weights across the entire user behavior sequence. Some recent enhancements [Wu et al., 2020, Lin et al., 2020] introduce user models for long-term semantics, assuming that self-attentive recommenders can effectively capture short-term user dynamics. However, in this dissertation, we argue that the existing "vanilla" self-attention (referred to as *global self-attention*) in self-attentive recommenders falls short in adequately capturing the significance of short-term user dynamics.

We begin with a motivating experiment on a "short-term" dataset. While global self-attention theoretically has the capacity to learn correct semantics with *sufficient* homogeneous sequential data [Yun et al., 2020], our experimental findings demonstrate that in real-world sequential recommendation tasks with limited data of interacted items, global self-attention, serving as a non-local operator, tends to excessively focus on distant historical items. This leads to a decline in performance. As illustrated in Figure 2.1, some distant items (e.g., printer, camera) are less relevant to users' short-term interests (e.g., a mobile phone). In practice, global self-attention often fails to accurately capture short-term dynamics and instead overly fixates on distant items. Recent developments in linguistics indicate that incorporating inductive local and other biases can enhance self-attention's generalization ability [Guo et al., 2020, Yang et al., 2018, Li et al., 2018a, Guo et al., 2019]. However, this concept has not been widely investigated in the context of recommendation systems.

In this dissertation, we introduce a novel framework, *Locally Constrained Self-attentive Recommender* (LOCKER), which extends self-attentive networks for sequential recommendation tasks. LOCKER enhances the capture of short-term user dynamics through local constraints (implemented in the local encoder) while preserving the ability to model long-term user preferences. We explore various local operators (e.g., model- or masking-based local encoders) for the local encoder, and for the global encoder, we leverage existing global self-attention networks. We conducted experiments to compare the model accuracy with other baselines on multiple datasets with item interaction sequences. Our experimental results demonstrate that

LOCKER with diverse local encoders surpasses existing self-attentive recommenders with a small computational overhead, standing as one of the state-of-the-art homogeneous sequential recommender systems.

2.2 Preliminaries

2.2.1 Task Formulation

We formulate the homogenous sequential recommendation task as following: Given a user set $\mathscr{U}$, an item set $\mathscr{I}$, and user behavior sequences $\mathscr{S} = \{S_1, S_2, \ldots, S_{|\mathscr{U}|}\}$, where each user sequence S_u is chronologically ordered as $S_u = (s_1^{(u)}, s_2^{(u)}, \ldots, s_{N_u}^{(u)})$, with $S_u \in \mathscr{S}$, $u \in \mathscr{U}$, $s_i^{(u)} \in \mathscr{I}$, and N_u representing the sequence length. Here $s_i^{(u)}$ implies that we only consider a single type of user interaction on the item $s_i^{(u)}$, such as purchasing or clicking. The objective is to predict the next interacted item $s_{N_u+1}^{(u)}$ based on the interaction history S_u. Without loss of generality, we omit the user identifier u to simplify the notations below.

2.2.2 Self-Attentive Recommenders

Self-attentive recommenders [Kang and McAuley, 2018, Sun et al., 2019, Zhang et al., 2019, Wu et al., 2020, Li et al., 2020d] heavily leverage global self-attention, with variations in inputs, training, and masking strategies. The core idea of global self-attention is to learn pairwise item-to-item "attention" weights from the sequential data, thereby identifying "relevant" items from users' complete interaction sequences to make predictions for future interactions.

Formally, let us place the position of the i-th item in the user interaction sequence. For self-attention recommenders, $\mathbf{H}_i^l \in \mathbb{R}^{1\times d}$ represents the item embedding for s_i after the l^{th} self-attention layer. This item embedding is fed into a multi-head (let us set #Heads=M) self-attention module [Vaswani et al., 2017b] to first generate related vectors for the "attention" weights calculation, including the query vector $\mathbf{Q}_i^{(m)} = \mathbf{H}_i^l \mathbf{W}_Q^{(m)}$, key vector $\mathbf{K}_i^{(m)} = \mathbf{H}_i^l \mathbf{W}_K^{(m)}$ and value vector $\mathbf{V}_i^{(m)} = \mathbf{H}_i^l \mathbf{W}_V^{(m)}$ for the m-th multi-head attention module, where $\mathbf{W}_Q^{(m)}, \mathbf{W}_K^{(m)}, \mathbf{W}_V^{(m)} \in$

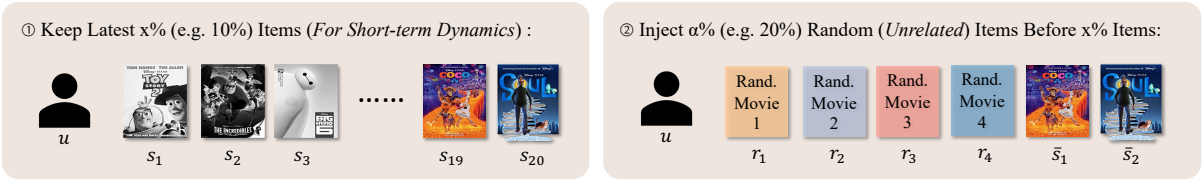


Figure 2.2. Short-term data processing example.

$\mathbb{R}^{d\times d/M}$ are the m-th learnable projection matrices. With those vectors, the new value vector $\tilde{\mathbf{V}}_i$ from this multi-head global self-attention module is calculated as:

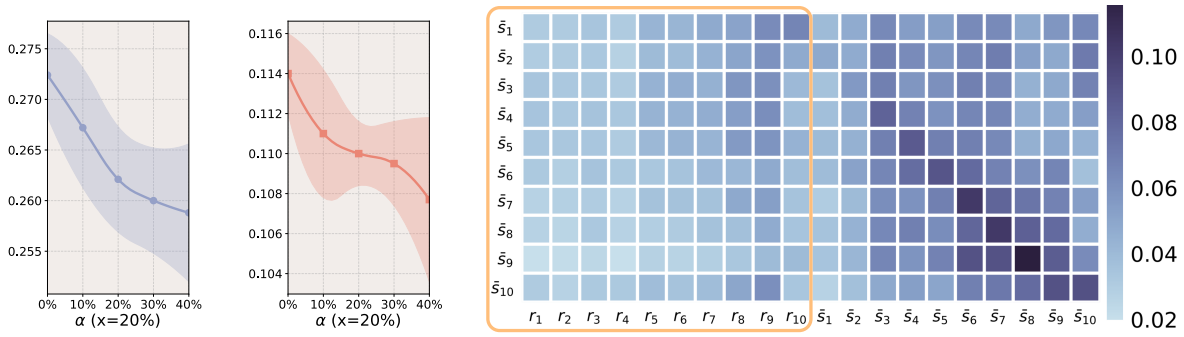
$$\begin{aligned}\tilde{\mathbf{V}}_i &= \left[\tilde{\mathbf{V}}_i^{(1)};\ldots,\tilde{\mathbf{V}}_i^{(m)};\ldots;\tilde{\mathbf{V}}_i^{(M)}\right]\mathbf{W}_O,\\ \text{where}\ \ \tilde{\mathbf{V}}_i^{(m)} &= \sum_{j=1}^{N} f_{\texttt{att}}\left(\mathbf{Q}_i^{(m)}\to\mathbf{K}_j^{(m)}\right)\cdot\mathbf{V}_j^{(m)}.\end{aligned} \tag{2.1}$$

Here $f_{\texttt{att}}$ is an attention function (e.g. scaled dot-product attention [Vaswani et al., 2017a]) to calculate the "attention" weight for any item-to-item pair (i.e. $\mathbf{Q}_i$ to $\mathbf{K}_j$) in the input sequence; $\mathbf{W}_O\in\mathbb{R}^{d\times d}$ is a learnable projection matrix to get $\tilde{\mathbf{V}}_i$ from concatenated vectors. Then, the item embedding $\mathbf{H}_t^{l+1}$ after the next layer $l+1$ is generated by using $\tilde{\mathbf{V}}_i$ from Equation (2.1) with Residual Connections [He et al., 2016], LayerNorm [Ba et al., 2016] and Pointwise Feed-Forward Networks [Vaswani et al., 2017a]. Although the overall calculations involve complex steps, it is worth emphasizing that the key aspect of "global" self-attention is to calculate pairwise "attention" weights between the given query vector $\mathbf{Q}_i$ and any key vectors $\mathbf{K}_j$, $\forall j\in[1,N]$.

2.2.3 Self-Attentive Recommenders Need Local Constraints

However, we argue that the popular global self-attention modules usually do not effectively capture short-term user dynamics in homogeneous sequential recommender systems. To assess the inadequacy of global self-attention in capturing short-term user dynamics, we introduce an analytical task using the widely adopted MovieLens dataset [Harper and Konstan, 2015] (refer to dataset details in Section 2.4).

In this preliminary experiment, training data is derived from MovieLens through two



(a) HR@20 *vs.* α **(b)** N@20 *vs.* α **(c)** Average attention map (len.=20, #sample=200).

Figure 2.3. Motivating experiments to show short-term user dynamics are not sufficiently learned my global self-attention.

steps, as illustrated in Figure 2.2: (1) To emphasize short-term dependencies, user sequences are truncated, retaining only the last $x\%$ of items. (2) To evaluate the efficacy of global attention in capturing short-term dynamics, $\alpha\%$ random (non-meaningful) items are prepended before the aforementioned $x\%$ items. Ideally, if global self-attention can apprehend "flexible order" interactions from the data, the injected random items should receive minimal attention weight and have negligible impact on recommendation performance.

Model Details. We adopt BERT4Rec [Sun et al., 2019], a self-attention-based recommender, for dataset fitting. The hidden size is set to $d = 64$, and optimal hyperparameters are determined through grid search. Similar self-attention modules are utilized in other recommenders [Kang and McAuley, 2018, Chen et al., 2019d, Zhang et al., 2019, Wu et al., 2020].

Evaluation Details. We follow the [Kang and McAuley, 2018, Sun et al., 2019] evaluations, using a *leave-last-out* data split. Specifically, for each sequence, the first N-2 items are used for training, the (N-1)$^{\text{th}}$ item for validation, and the N$^{\text{th}}$ item for testing. Evaluation metrics include truncated Hit Ratio (HR@K) and Normalized Discounted Cumulative Gain (N@K) [Sun et al., 2019, Kang and McAuley, 2018], with K set to 20 for measuring ranking quality. Consistent with recommendations from evaluation reviews [Krichene and Rendle, 2020], we adopt the *all-item ranking* approach.

We evaluate recommendation performance with different random item length ratios α

and have such observations uncovering the limitations of using global self-attention modules:

Performance gap. Firstly, Figures 2.3a and 2.3b demonstrate that, in the absence of inductive bias, global self-attention consistently underperforms the local model (i.e., $\alpha = 0$). This is observed despite the theoretical adequacy of training data in aiding global self-attention to accurately capture short-term patterns.

Behavior sequence length. Secondly, Figures 2.3a and 2.3b demonstrate a consistent performance deterioration with growing α. This decline is likely due to global self-attention incorporating noise from distant tokens as sequences get longer, given its consideration of all tokens.

Attention map. Lastly, Figure 2.3c illustrates the average attention map. Global self-attention prioritizes tokens in close proximity (denoted by "brighter" neighbors), with an average attention score of $\bar{f}_{\texttt{att}} = 0.062$. Nevertheless, in the absence of local constraints, the model incorporates noise from distant and unrelated items, as indicated by the red box with an average attention score of $\bar{f}_{\texttt{att}} = 0.038$.

2.3 Proposed Approach: LOCKER

Motivated by our initial experiment on the limitations of global self-attention modules for homogeneous sequential recommender systems, we introduce a versatile framework, *Locally Constrained Self-attentive Recommender* (LOCKER), designed for self-attentive recommenders in a plug-and-play manner. This framework aims to improve the capacity to capture short-term dynamics while preserving the ability to capture long-term semantics. Subsequently, we explore various local encoders within the context of LOCKER.

2.3.1 Framework Overview

The core idea of LOCKER is to seamlessly and explicitly introduce local constraints into existing self-attention networks, to enhance the inductive bias of locality for recommendation accuracy improvements. Similar to Equation (2.1), the output value vectors from M attention

heads are concatenated. However, LOCKER defines $M_l + M_g = M$ to partition attention heads into M_l local encoders and M_g global encoders. Formally,

$$\tilde{\mathbf{V}}_i = \left[\tilde{\mathbf{V}}_{i,l}^{(1)}; \ldots; \tilde{\mathbf{V}}_{i,l}^{(M_l)}; \tilde{\mathbf{V}}_{i,g}^{(1)}; \ldots; \tilde{\mathbf{V}}_{i,g}^{(M_g)}\right] \mathbf{W}_O, \tag{2.2}$$

where $\tilde{\mathbf{V}}_{i,l}^{(m_l)}$ ($\tilde{\mathbf{V}}_{i,g}^{(m_g)}$) represents an output value vector from the local (global) encoder. Our focus is on the explicit role of local encoders. Consequently, we maintain global encoders identical to the single global attention head in Equation (2.1) and explore various local encoders, including model-based and masking-based encoders.

2.3.2 Model-based Local Encoders

For the model-based local encoders, we generate $\tilde{\mathbf{V}}_{i,l}^{(m_l)}$ using neural-network operators with the inductive bias of locality.

Fixed-Depth RNN (LOCKER+RNN). Recurrent networks excel in short-term sequence modeling [Hidasi et al., 2015]. To enhance the efficiency of our local encoder while preserving its capability to capture short-term dynamics, we incorporate a fixed-depth RNN module as the local encoder:

$$\tilde{\mathbf{V}}_{i,l}^{(m_l)} = g\big(\mathbf{V}_{i,l}^{(m_l)}, \underbrace{g(\mathbf{V}_{i-1,l}^{(m_l)}, \ldots)}_{\text{recurrent depth } s}\big), \tag{2.3}$$

where g is the recurrent neural unit; here we choose Gated Recurrent Units (GRU) [Cho et al., 2014a] as in GRU4Rec [Hidasi et al., 2015]. Here we use a GRU with fixed and small depth s to simplify computation and concentrate on short-term user dynamics.

Convolutional Network (LOCKER+Conv). A convolutional network presents an alternative approach for capturing neighborhood dynamics. We introduce a convolution-based encoder for $\tilde{\mathbf{V}}_{i,l}^{(m_l)}$ as:

$$\tilde{\mathbf{V}}_{i,l}^{(m_l)} = [\mathbf{c}_1; \ldots; \mathbf{c}_{d/M}],\ \mathbf{c}_j = \texttt{act}\left(\mathbf{V}_{[i]_s,l}^{(m_l)} \odot \mathbf{W}^{(j)}\right), \tag{2.4}$$

where $\odot$ denotes an inner product operator like [Tang and Wang, 2018], $\mathbf{V}_{[i]_s,l}^{(m_l)} \in \mathbb{R}^{s\times d/M}$ denotes the local $[i-(s-1)/2,...,i+(s-1)/2]$ rows (size s is odd number) in $\mathbf{V}_l^{(m_l)} \in \mathbb{R}^{N\times d/M}$. $\mathbf{W}^{(j)} \in \mathbb{R}^{s\times d/M}$ denotes the j-th convolutional kernel; `act` is an activation function to introduce non-linearity such as ReLU. Compared to CASER [Tang and Wang, 2018], which used convolutional networks to capture point-level and union-level item similarities, we adopt convolutional networks as a local operator to enhance short-term user dynamics modeling.

2.3.3 Masking-based Local Encoders

For the masking-based local encoders, we reconsider the global attention function $f_{\texttt{att}}$ by incorporating locality-aware masking to better capture short-term dynamics. Specifically, in Equation (2.1), where the detailed definition of $f_{\texttt{att}}$ involves the "relevance" logit w_{ij} with positions i, j, and is fed into the softmax layer for normalization, i.e.:

$$f_{\texttt{att},l}(\mathbf{Q}_i \to \mathbf{K}_j) = \frac{\exp(w_{ij})\cdot\sigma_{ij}}{\sum_{k=1}^{N}\exp(w_{ik})\cdot\sigma_{ik}} \tag{2.5}$$

where the masking score $\sigma_{ij} \equiv 1$ for global self-attention. In the masking-based local encoder, we improve the ability to capture short-term dynamics by modifying the masking score σ_{ij} using various strategies.

Fixed Window (LOCKER+Win). Fixed window simply deactivates all distant tokens, where σ_{ij} is defined as:

$$\sigma_{ij} = \mathbb{I}(|i-j| \le s), \tag{2.6}$$

where $\mathbb{I}$ is an indicator function. Therefore, the attention map is masked by a fixed-length window to deactivate the dependency on distant (distance $> s$) tokens.

Gaussian Initialization (LOCKER+Initial). LOCKER+Win pre-defines "hard" and "static" masking scores for all training data, which may be overly rigid. We aim to introduce "trainable" masking scores with a well-initialized, unbounded adjustable weight $p_{i-j} = \ln\sigma_{ij}$, incorporating a locality prior into the encoder. The masking operation $\exp(w_{ij})\cdot\sigma_{ij}$ in Equa-

tion (2.5) can be expressed as $\exp(w_{ij} + \ln \sigma_{ij})$, allowing us to "learn" the weight p_{i-j}, where $i-j$ signifies mapping different distances to a trainable weight.

We propose weight initialization using a Gaussian-like function (e.g., $p^0_{i-j} = a\exp(-(i-j)^2/b)$), introducing local concentration in the neighborhood. It is important to note that a change in weight initialization does not guarantee the explicit presence of a local "fixed window" post-training. However, the initialization bias encourages the model to capture local patterns from a more advantageous starting point compared to, for instance, uniform initialization and can adapt during training. To further enhance the capture of locality by trainable weights from the data, we eliminate positional embeddings (as in standard self-attentive recommenders [Kang and McAuley, 2018, Sun et al., 2019]) for local encoder vectors, i.e., only incorporating positional embeddings into the global encoder key and query vectors.

Adaptive Predictor (LOCKER+Adapt). LOCKER+Initial adjusts soft scores but lacks the capability to encode additional information, such as a user identifier u. We enhance LOCKER+Initial by introducing a parameterized adaptive predictor `pred` to forecast diverse masking scores, i.e.:

$$p^{(u)}_{i-j} = \texttt{pred}\left(\mathbf{V}^{(m_l)}_{i,l} + \mathbf{V}^{(m_l)}_{j,l} + \mathbf{v}_u + \mathbf{b}_{i-j}\right), \tag{2.7}$$

where we encode user information $\mathbf{v}_u \in \mathbb{R}^{1\times d}$, distance embedding $\mathbf{b}_{i-j} \in \mathbb{R}^{1\times d}$, and current value vectors $\mathbf{V}^{(m_l)}_{i,l}$ and $\mathbf{V}^{(m_l)}_{j,l}$. The user representations v_u are constructed following the methods proposed in FISM [Kabbur et al., 2013] and FISSA [Lin et al., 2020], eliminating the need for additional user embeddings. Similar to LOCKER+Initial, positional embeddings for local encoder vectors are omitted to promote the model's ability to learn locality from the data. The `pred` component is a two-layer MLP model designed to learn more flexible masking scores by incorporating user, distance, and current token information.

Table 2.1. Data statistics for experiments.

	#Interaction	#Item	#Sequence	Average Length	Density
Beauty	353,962	54,542	40,226	8.80	1e-4
Clothing	831,816	162,193	108,489	7.67	1e-5
MovieLens	1,000,000	3,416	6,040	165.56	1e-2

2.4 Experiments

2.4.1 Experimental Setting

Datasets. We consider the following datasets from different domains with various data distributions (see Table 6.2): **Beauty, Clothing** are datasets collected from Amazon in [McAuley et al., 2015]. **MovieLens** [Harper and Konstan, 2015] is a popular benchmark dataset for top-N recommendation. We are using the 1-million version. We follow the data pre-processing and splitting from [Sun et al., 2019] (details in Section 2.2.3).

Baselines. We incorporate the following baselines in our subsequent experiments. **PopRec**: Recommends items based on their occurrences in the dataset. **BPR-MF** [Rendle et al., 2009]: A classic personalized ranking learning algorithm utilizing matrix factorization. **SASRec** [Kang and McAuley, 2018]: A seminal method employing self-attention mechanism for sequential recommendation. **BERT4Rec**[Sun et al., 2019]: A BERT-like[Devlin et al., 2019] model capturing bi-directional contextual item information through a *cloze* task for next-item recommendation. **SSE-PT** [Wu et al., 2020]: A state-of-the-art self-attentive recommender, extending SASRec by introducing explicit user representations.

Evaluation and Implementation. We adopt the evaluation protocols from Section 2.2.3. All models are implemented using `PyTorch`, and baselines are fine-tuned using grid search with consistent granularity as recommended by each baseline. LOCKER employs the identical training and hyperparameter search strategy as our backbone model (BERT4Rec).[1].

[1] LOCKER implementation in https://github.com/AaronHeee/LOCKER

Table 2.2. Model comparison. We tune LOCKER with M={2,4} where M_l={1,...,M-1}, and use similar settings for the backbone. Compared to the best baselines (underline), Avg. (Max.) for average (maximum) relative improvement of five local encoders.

		Beauty		**Clothing**		**MovieLens**	
	Metric	N@20	HR@20	N@20	HR@20	N@20	HR@20
Baselines	PopRec	0.0048	0.0131	0.0021	0.0056	0.0260	0.0686
	BPR-MF	0.0172	0.0425	0.0035	0.0089	0.0498	0.1298
	SASRec	0.0206	0.0496	0.0052	0.0140	0.1625	0.3652
	BERT4Rec	<u>0.0238</u>	<u>0.0541</u>	<u>0.0062</u>	<u>0.0153</u>	<u>0.1783</u>	<u>0.3870</u>
	SSE-PT	0.0232	0.0547	0.0059	0.0149	0.1763	0.3841
LOCKER	+RNN	0.0258	0.0568	0.0070	0.0166	0.1930	0.4012
	+Conv.	0.2970	0.0661	0.0078	0.0184	**0.1980**	**0.4119**
	+Win.	0.0296	0.0641	0.0074	0.0174	0.1831	0.3972
	+Initial	0.0303	0.0652	0.0077	0.0184	0.1863	0.3900
	+Adapt	**0.0311**	**0.0672**	**0.0079**	**0.0187**	0.1893	0.4047
Impr.	Avg.	+23.1%	+16.8%	+21.9%	+17.0%	+06.5%	+03.6%
	Max.	+30.7%	+22.9%	+27.4%	+22.2%	+11.1%	+06.4%

2.4.2 Result Analysis

General Performance. We have such observations from model ranking performance on three datasets in Table 2.2. Firstly, Self-attentive sequential recommenders (SASRec, BERT4Rec, SSE-PT) consistently surpass classic methods by effectively incorporating sequential information through global self-attention networks. BERT4Rec outperforms SASRec through bidirectional training, while SSE-PT achieves superior performance by introducing explicit user representations. Secondly, our LOCKER framework consistently outperforms all baselines. In comparison to the strongest global self-attention-based recommender models (BERT4Rec, SSE-PT), our model achieves an average improvement of approximately 17.19% in N@20 and 12.46% in HR@20, while maintaining a comparable number of parameters. Additionally, LOCKER with the most effective local encoder exhibits about 23.04% improvement in N@20 and 17.67% improvement in HR@20 across three datasets. This underscores the effectiveness of introducing inductive local bias into self-attentive recommenders using different local encoders.

Table 2.3. Local encoder characteristics in LOCKER.

Local Encoder	Skipped Behavior?	No Extra Param.?	Adapt. Size?
RNN	✗	✗	✗
Conv.	✓	✗	✗
Win.	✓	✓	✗
Initial/Adapt	✓	✗	✓

Local Encoder Discussion. For local encoders, we make some observations as well. First, *Model-* and *masking*-based encoders surpass pure global self-attentive sequential recommenders. On MovieLens, where the average user sequences are the longest, model-based encoders outperform all alternatives. Second, presumably, due to its sequential encoding nature, +*RNN* performs suboptimally on datasets requiring flexible "skip" behaviors, in contrast to other encoders. This is evident in the inferior performance of the three datasets. Third, +*Conv.* and +*Win.* are fixed-size encoders capable of capturing flexible item dependencies. The superior performance of +*Conv.* on three datasets may be attributed to the introduction of additional model parameters. Last, +*Initial* and +*Adapt* outperform fixed-window +*Win.* with learnable masking. +*Adapt* excels in encoding additional information such as user and current tokens, leading to enhanced performance, albeit with an extended training time, as illustrated below. The characteristics of the five encoders are summarized in Table 2.3.

Efficiency and Convergence. Figure 6.4 displays NDCG@20 curves for the Beauty validation set, computed on a single Nvidia 2080s GPU. LOCKER *Max.* represents LOCKER+*Adapt* with the slowest training speed, while LOCKER *Avg.* presents the average NDCG@20 scores and training time across five LOCKER models. A comparison with BERT4Rec reveals that our models achieve comparable performance with significantly fewer training epochs (˜200 *vs.* ˜500). Furthermore, our models exhibit comparable convergence with similar training epochs, experiencing minimal computational overhead (5.7s/epoch on average *vs.* 5.1s/epoch).

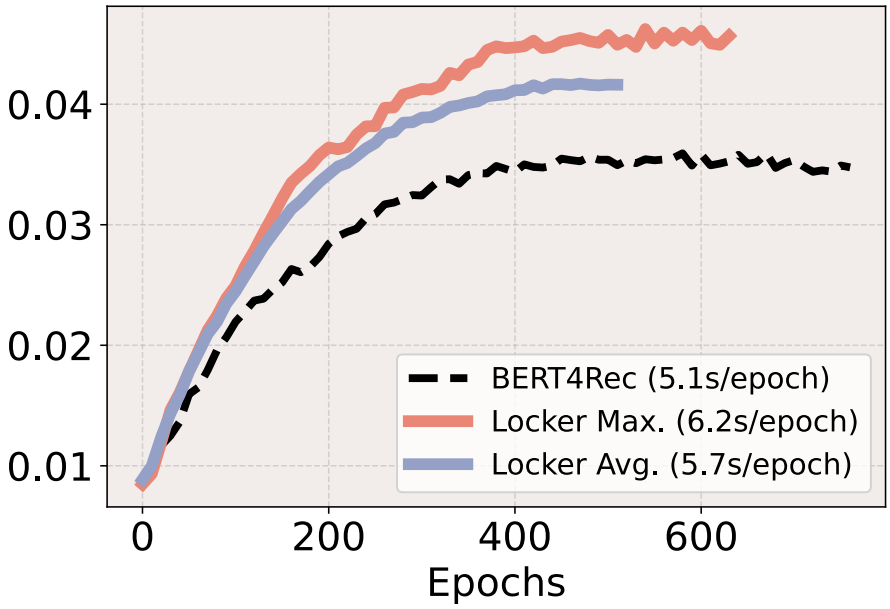


Figure 2.4. LOCKER validation curve on Beauty in terms of NDCG@20.

2.5 Conclusion

Capturing users' long-term preferences and short-term dynamics over time is a core personalization task, which will also be a core personalization component for conversational recommender systems. This sequential recommendation task becomes even more challenging in the homogeneous sequential data setting. Recent self-attentive recommenders exhibit potential in such recommendation tasks, with global self-attention being crucial. However, our research reveals that global self-attention, lacking inductive bias of locality, struggles to effectively capture short-term user dynamics. To address this limitation, we introduce LOCKER, a framework incorporating local inductive bias. Through the integration of five local encoders into existing self-attention networks, we enhance short-term dynamics modeling, demonstrating its efficacy across various datasets. This approach establishes a new state-of-the-art for homogeneous sequential recommender systems and promises to serve as a core recommendation component backbone in many scenarios including conversational recommendations.

Chapter 2, in part, is a reprint of the material as it appears in "Locker: Locally constrained

self-attentive sequential recommendation." by Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley, in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management in 2021* referenced as [He et al., 2021]. The dissertation author was the primary investigator and author of this paper.

Chapter 3

Heterogeneous Sequential Recommender Systems

In this chapter, we take one step further from homogeneous sequential data settings to heterogeneous data settings. The research focus is on considering more user interaction types to form user interaction sequences beyond item clicks. Due to data sparsity and noisy user behaviors, capturing long-term preferences and short-term dynamics is often challenging in homogeneous sequential recommendation, but heterogeneous signals may help. The new challenge for heterogeneous sequential recommender systems is discovering which types of new user signals are worth including and how to incorporate them. In this chapter, we discuss a new type of user signal—textual users' queries—into sequential recommenders, in a proposed heterogeneous sequential recommendation setting, namely *query-aware sequential recommendation*.

The motivations behind this are as follows. First, when users interact with products (like music or movies), their sequential behavior is driven by a combination of explicit queries and subsequent actions. However, most interaction datasets discard queries. Consequently, existing methods often model sequential behavior solely based on items, thereby neglecting the vital context that queries provide about user intent. Second, the query-aware sequential recommendation is a significant and practical step towards conversational recommendations. Here, we explore how to make accurate recommendations in an environment where user clicks and textual queries both exist. Meanwhile, more existing datasets can be accessible (e.g.,

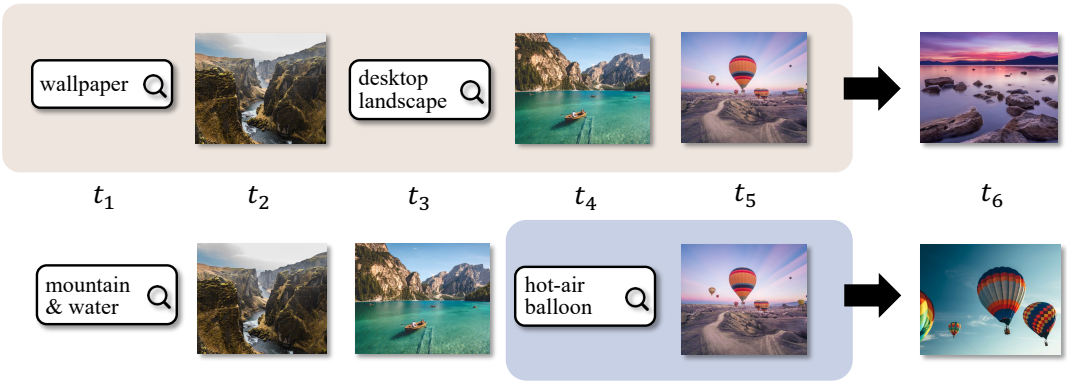


Figure 3.1. Motivating Examples. Given the same *item* sequence with different users' *queries*, the recommendation results are different, and the "boundaries" of useful historical interactions (shaded) also differ.

from searching logs) than in a fully conversational recommendation without popular real-world applications and datasets.

Methodologically, in this chapter, we emphasize the importance of user queries as a powerful contextual signal for sequential recommendation. We introduce a novel query-aware sequential recommendation setting that explicitly integrates user queries to model intent. We propose a new model, QUERYSR, that not only incorporates query information into user behavior sequences but also leverages query-item co-occurrence information to enhance model generalization. Finally, we demonstrate the effectiveness of integrating query features across three datasets, shedding light on combining users' textual signals with traditional homogeneous item interactions.

3.1 Introduction

Sequential recommender systems are crucial in personalized online services, such as e-commerce and streaming media, relying on users' historical action sequences. However, accurately extracting relevant signals in a homogeneous sequential recommendation setting poses challenges, as user intent may evolve gradually or change suddenly, causing a degradation of sequential context among items.

To address issues related to complex and noisy behavior sequences and enhance the capture of users' intent, various methods have been proposed, targeting different aspects, including improving model architectures [Hidasi et al., 2015, Tang and Wang, 2018, Kang and McAuley, 2018] and incorporating heterogeneous information [Liu et al., 2021, Meng et al., 2020, Li et al., 2020a]. Heterogeneous information, such as item *content* [Liu et al., 2021, Zhang et al., 2019, Hidasi et al., 2016] (e.g., reviews, images), user *action* types [Meng et al., 2020] (e.g., clicks, downloads, purchases), and *temporal* information [Li et al., 2020a, Bogina and Kuflik, 2017] (e.g., time intervals), has been leveraged for improved performance.

Despite the success of models utilizing rich heterogeneous information, certain crucial signals remain under-explored. This dissertation focuses on *users' textual queries* within their chronological interaction sequences. In various recommendation scenarios, such as e-commerce, music, and photo-sharing, users engage with the system by posing queries and browsing relevant items alternately. However, these informative signals are often neglected in sequential recommendation datasets, as illustrated in Figure 3.1. This research aims to explore a recommendation environment, with the integration of explicit textual queries and user-item interactions over time, specifically in the context of *query-aware sequential recommendation*.

Textual queries are expressed by users proactively; thus, those queries serve as crucial contextual cues for understanding and predicting users' changing intent. The advantages of leveraging textual queries can be summarized as follows: (1) Textual queries offer insight into intent granularity. As illustrated in Figure 3.1, queries such as "wallpaper" and "hot-air balloon" not only indicate the recommended target but also express a desire for content diversity within a specific context. (2) Textual queries establish connections between interactions, enhancing item representations, particularly for infrequently interacted items. (3) Queries aid in detecting boundaries of user intent. In Figure 3.1, the sequence "wallpaper" followed by "desktop landscape" suggests a refinement of interests, while "mountain & water" followed by "hot-air balloon" implies unrelated intent. These scenarios have distinct semantic implications for understanding relationships among sequential interactions, such as clicks or purchases, and

also pave the way for future fully conversational recommendations where textual queries are presented in a more flexible manner.

In this dissertation, we argue for the value of user textual queries in the sequential recommendation and propose a model tailored for the query-aware sequential recommendation scenario. Our approach involves organizing textual queries and item information into heterogeneous query and item sequences. To enhance the model's generalization capability, we use *graph-based sequence augmentation* based on query-item co-occurrence. Additionally, we address the technical challenge of handling large item embedding tables (e.g., 10 million items) during model training, a scale seldom addressed in existing sequential recommendation literature. The primary contributions are succinctly summarized as follows:

- Introduction of user textual queries for heterogeneous sequential recommendation, wherein explicit queries serve as crucial contextual cues for reflecting and predicting user intent.
- Proposal of a query-aware sequential recommender, denoted as QUERYSR, leveraging heterogeneous user sequences and graph-based sequence augmentation. A self-attentive model is also introduced within this framework.
- Evaluation on two established datasets[1] adapted for the query-aware sequential recommendation context, along with a new industrial dataset featuring millions of items. Experimental results highlight the impact of incorporating explicit user queries and demonstrate the superior performance of our method compared to state-of-the-art baselines.

3.2 Proposed Approach: QUERYSR

3.2.1 Input Sequence Formulation

Homogeneous Item Sequence. In homogeneous sequential recommendation, we are given a user set $\mathscr{U}$, an item set $\mathscr{I}$, and a set of user *item interaction* sequences $\mathscr{S} =$

[1]These datasets collect queries and clicks for diverse applications, with user queries traditionally discarded in conventional sequential recommendation settings.

$\{S_1, \ldots, S_{|\mathscr{U}|}\}$. Each sequence S_u consists of user u's (chronologically ordered) item interactions:

$$S_u = \left[i_1^{(u)}, i_2^{(u)}, \ldots, i_{T_u}^{(u)} \right], \tag{3.1}$$

where $S_u \in \mathscr{S}$, $u \in \mathscr{U}$. $i_t^{(u)} \in \mathscr{I}$ is the item that the user clicked at the timestep t. T_u is the sequence length.

Query-aware Heterogeneous Sequence. We consider user textual queries by introducing an additional query set $\mathscr{Q}$ and word vocabulary $\mathscr{V}$, where a query $q \in \mathscr{Q}$ consists of a list of words $[v_1, \ldots, v_{|q|}]$, $v \in \mathscr{V}$. We enrich the sequence S_u to a heterogeneous sequence $\hat{S}_u$, containing user u's queries and item interactions in chronological order:

$$\hat{S}_u = \left[\hat{s}_1^{(u)}, \hat{s}_2^{(u)}, \ldots, \hat{s}_{\hat{T}_u}^{(u)} \right], \tag{3.2}$$

where $\hat{T}_u$ is the length of this query-aware heterogeneous sequence. $\hat{s}_t^{(u)}$ can be an item interaction or a query action. We use δ to indicate whether $\hat{s}_t^{(u)}$ at t-th step is a query or an item interaction:

$$\hat{s}_t^{(u)} \in \begin{cases} \mathscr{I}, & \text{if } \delta(\hat{s}_t^{(u)}) = 0, \\ \mathscr{Q}, & \text{otherwise.} \end{cases} \tag{3.3}$$

We also examine other ways [Zhang et al., 2019, Liu et al., 2021] to incorporate user queries into item interaction sequences in our empirical studies (see Section 3.3.2).

Recommendation Goal. Given the query-aware heterogeneous sequence $\hat{S}_u$, the model predicts the next item for user u, which is formalized as modeling the probability over all possible items for this user's next item interaction, i.e.:

$$P\left(\hat{s}_{\hat{T}_u+1}^{(u)} = i^* \mid \hat{S}_u,\ \delta(\hat{s}_{\hat{T}_u+1}^{(u)}) = 0 \right). \tag{3.4}$$

$\delta(\hat{s}_{\hat{T}_u+1}^{(u)}) = 0$ assumes the next step is an item interaction. $\hat{s}_{\hat{T}_u+1}^{(u)} = i^*$ denotes that $i^* \in \mathscr{I}$ is the

item the user interacts with at step $\hat{T}_u + 1$. We omit the user identifier u to simplify the notations below.

3.2.2 Graph-Based Sequence Augmentation

In the query-aware sequential recommendation, query-item co-occurrence offers unique information absent in traditional approaches. This information facilitates the construction of a query-item graph, enhancing model generalization through input sequence augmentation. The intuition is to augment a sequence $\hat{S}$ into K sequences $\hat{S}^{(1)}, \ldots, \hat{S}^{(K)}$ by stochastically substituting item i (query q) with similar items (queries). The use of query-item co-occurrence provides valuable cues for semantic similarities.

Graph Construction. We denote the query-item graph as $\mathscr{G} = (\mathscr{A}, \mathscr{E})$, where $\mathscr{A} = \mathscr{Q} \cup \mathscr{I}$ represents item and query nodes. The edge set $\mathscr{E}$ denotes all linkages between queries and items (i.e., (q,i) or $(i,q) \in \mathscr{E}$). We define the neighbors of item i as $\mathscr{N}(i) = \{q \mid (i,q) \in \mathscr{E}\}$, and define $\mathscr{N}(q)$ similarly. The initial edge set $\mathscr{E}_1$ is constructed by connecting item i with its latest query q. To further reduce noise and retain confident linkages, we introduce a threshold α over $\mathscr{E}_1$. We retain the top $\lceil \alpha |\mathscr{N}(i)| \rceil$ linkages for item i and the top $\lceil \alpha |\mathscr{N}(q)| \rceil$ linkages for query q. Therefore, the final edge set is defined as $\mathscr{E} = \mathscr{E}_\alpha$. It is important to note that α trades off the coverage and confidence of query-item linkages, where $0 < \alpha \leq 1$.

Graph-Based Sequence Augmentation. We augment input sequences by adopting a stochastic shared embedding (SSE) idea [Wu et al., 2019] based on our constructed query-item graph. For $\hat{s}_t \in \hat{S}$, we replace $\hat{s}_t$ with probability β following:

$$i \sim \hat{s}_t, j \not\sim \hat{s}_t \rightarrow p(i,\hat{s}_t)/p(j,\hat{s}_t) = \rho, \text{ if } \delta(\hat{s}_t) = 0,\ i,j \in \mathscr{I} \tag{3.5}$$

$$q \sim \hat{s}_t, k \not\sim \hat{s}_t \rightarrow p(q,\hat{s}_t)/p(k,\hat{s}_t) = \rho, \text{ if } \delta(\hat{s}_t) = 1,\ q,k \in \mathscr{Q}. \tag{3.6}$$

Here $p(\cdot,\cdot)$ is the replacement probability, and ρ is a constant greater than 1. We use $\sim$ ($\not\sim$) to denote whether two nodes are similar or not. Given a graph $\mathscr{G}$, we define similar queries

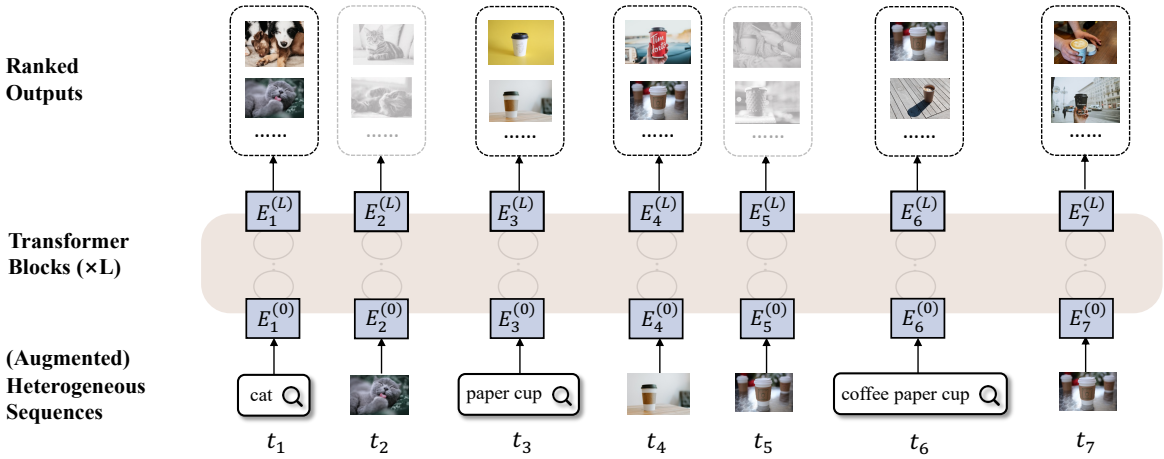


Figure 3.2. Our QUERYSR model with heterogeneous sequences incorporating queries. Here the labels of these deactivated ranked outputs (in gray) are user textual queries; these gray outputs will not be used for loss calculation.

$q \sim k$ when q and k have some common neighbour(s), i.e., $\mathcal{N}(q) \cap \mathcal{N}(k) \neq \varnothing$. This is motivated by the fact that nodes (e.g. "cafe" and "coffee") which connect to many common neighbors are potentially similar, so are more likely to be replaced by each other for data augmentation. We can augment sequences "on-the-fly" for each training epoch rather than generating all augmented sequences in advance.

3.2.3 Transformer-Based Model Backbone

Query-aware sequential recommendation is a new setting that various sequential recommendation backbones can build. In this context, we introduce a Transformer-based model, referred to as QUERYSR [Vaswani et al., 2017a, Kang and McAuley, 2018]. We subsequently present the training of the model, addressing challenges associated with large item pool sizes, such as 10 million items.

Representation. We utilize embeddings $\mathbf{M}^0 \in \mathbb{R}^{|\mathscr{I}| \times d}$, $\mathbf{M}^1 \in \mathbb{R}^{|\mathscr{Q}| \times d}$, $\mathbf{P} \in \mathbb{R}^{\hat{T} \times d}$, and $\mathbf{B} \in \mathbb{R}^{2 \times d}$ to denote d-dimensional representations for *items*, *queries*, *timesteps*, and *interaction types*, respectively. **(1) Item, timestep:** Given item i and timestep t, we perform direct lookups for the corresponding embeddings $\mathbf{M}_i^0$ and $\mathbf{P}_t$. **(2) Query:** For a query $q = [v_1, \ldots, v_{|q|}]$, we

obtain word embeddings and use average pooling to derive the query representation $\mathbf{M}_q^1 \in \mathbb{R}^{1\times d}$. Other methods, such as utilizing a hidden vector from an LSTM [Hochreiter and Schmidhuber, 1997] or a sentence vector from BERT [Devlin et al., 2019], can also be used to obtain $\mathbf{M}_q^1$. The short length and lack of strong sequential patterns in user query words make average pooling an effective way to represent queries in a bag-of-words paradigm. **(3) Interaction type:** We perform a lookup in $\mathbf{B} \in \mathbb{R}^{2\times d}$ to obtain embeddings corresponding to different interaction types, namely, *item* or *query*.

Query-Aware Transformer Layer. Given a heterogeneous sequence $\hat{S}$ described in Section 3.2.1, we retrieve the input embedding matrix from the embedding layer `Emb` as:

$$\mathbf{E}^{(0)} = \texttt{Emb}(\hat{S}) = \begin{bmatrix} \mathbf{M}_{\hat{s}_1}^{\delta(\hat{s}_1)} + \mathbf{B}_{\delta(\hat{s}_1)} + \mathbf{P}_1 \\ \cdots \\ \mathbf{M}_{\hat{s}_{\hat{T}}}^{\delta(\hat{s}_{\hat{T}})} + \mathbf{B}_{\delta(\hat{s}_{\hat{T}})} + \mathbf{P}_{\hat{T}} \end{bmatrix}. \tag{3.7}$$

$\mathbf{E}^{(0)} \in \mathbb{R}^{\hat{T}\times d}$ is the input embedding matrix and $+$ denotes element-wise addition. Here item and query representations (with interaction type embeddings $\mathbf{B}$) are learned in a joint embedding space and are aware of sequential order by positional (timestep) embeddings. We build L Transformer [Vaswani et al., 2017a] blocks on top of the embedding layer `Emb`, which works as sequential encoder to generate $\mathbf{E}^{(L)} \in \mathbb{R}^{\hat{T}\times d}$ as the output embedding matrix. The details of the stacked transformer block construction refer to [Kang and McAuley, 2018, Sun et al., 2019].

Predictor Layer. Given the output $\mathbf{E}_t^{(L)} \in \mathbb{R}^{1\times d}$ at timestep t (i.e., the t-th row in matrix $\mathbf{E}^{(L)} \in \mathbb{R}^{\hat{T}\times d}$), we follow BERT4Rec [Sun et al., 2019] to calculate output probability over a target i as:

$$P\left(\hat{s}_{t+1} = i \mid \hat{S}, \delta(\hat{s}_{t+1}) = 0\right) = \texttt{softmax}_\text{i}\left(\mathbf{E}_t^{(L)}\mathbf{M}^{0^\top}\right), \tag{3.8}$$

where $\texttt{softmax}_\text{i}$ denotes the i-th probability from the softmax layer and the logits are interpreted as inner product similarities between the output $\mathbf{E}_t^{(L)}$ with the original item embeddings from $\mathbf{M}^0$.

3.2.4 Handling Large Item Vocabularies

Loss with Sampled Softmax. Technically, a large item embedding matrix $\mathbf{M}^0 \in \mathbb{R}^{|\mathscr{I}| \times d}$ due to item vocabulary size $|\mathscr{I}|$ (e.g. 10 million items [Ni et al., 2019]) may be prohibitive in terms of GPU memory with the softmax layer in Equation (3.8) in backpropagation (e.g. in the order of 100 GiB). Previous models like BERT4Rec [Sun et al., 2019] did not encounter this problem because the experimental datasets are small (e.g. 30 thousand items). In such cases, we use sampled softmax to reduce the memory cost in backpropagation, and revise Equation (3.8):

$$P_n\left(\hat{s}_{t+1} = i \mid \hat{S}, \delta(\hat{s}_{t+1}) = 0\right) = \texttt{softmax}_1\left(\mathbf{E}_t^{(L)} \mathbf{M}^{(n)\top}\right). \tag{3.9}$$

$\mathbf{M}^{(n)} \in \mathbb{R}^{n \times d}$ denotes sampled item embeddings. P_n is the probability that item i should be the target rather than the other $n-1$ candidates. This cross-entropy loss with sampled softmax also unifies the widely used BPR loss [Rendle et al., 2009] when $n = 2$. We use the same full / sampled softmax for baselines and our models for fair comparison. It also unifies the widely used BPR loss [Rendle et al., 2009] as below:

$$\mathscr{L} = \begin{cases} \text{CE Loss (Full Softmax)} & n = |\mathscr{I}|, \\ \text{CE Loss (Sampled Softmax)} & 2 < n < |\mathscr{I}|, \\ \text{BPR Loss} & n = 2, \end{cases} \tag{3.10}$$

where CE denotes cross entropy. In our experiments, we use the same full / sampled softmax for baselines and our models for a fair comparison.

Multi-GPU Embedding. To accommodate a large item embedding table $\mathbf{M}^0 \in \mathbb{R}^{|\mathscr{I}| \times d}$ in GPU memory, we partition $\mathbf{M}^0$ along the hidden size dimension. This involves loading $\mathbf{M_1}^0 \in \mathbb{R}^{|\mathscr{I}| \times d_1}, \ldots, \mathbf{M_m}^0 \in \mathbb{R}^{|\mathscr{I}| \times d_m}$ onto m GPUs. Subsequently, during training, we retrieve and concatenate the required item embeddings onto a single GPU in the form of mini-batches.

Table 3.1. Data Statistics. *Inter* for item interaction; *S* for sequence; *I* for item; *Q* for query; *A-I* for average number of interactions per item; *A-S* for average sequence length and *A-Q* for average number of query occurrence.

	#Inter	#I	#S	#Q	A-I	A-S	A-Q
Diginetica	52,164	22,587	8,020	5,870	2.31	6.50	1.92
Unsplash	1,623,566	22,517	240,993	56,634	72.10	6.74	9.63
Stock	25,731,635	8,633,462	987,173	1,516,020	2.98	26.07	1.95

3.3 Experiments

3.3.1 Experimental Setting

Evaluation Metrics. We adopt the *leave-last-out* data split strategy [Kang and McAuley, 2018, Sun et al., 2019], wherein each sequence is split into training (first $T-2$ items), validation ($(T-1)$-st item), and testing (T-th item) sets. Truncated Hit Ratio (HR@K) and Normalized Discounted Cumulative Gain (N@K) [Sun et al., 2019, Kang and McAuley, 2018] with $K = \{10, 20\}$ are selected as performance metrics to assess ranking quality. Further details on hyper-parameter tuning and evaluation are provided as follows.

Datasets. To investigate the role of user queries in heterogeneous sequential recommendation, we introduce three datasets (see statistics in Table 3.1).

1. **Diginetica**: *Diginetica*[2] is a dataset introduced in the CIKM 2016 CUP, comprising user search and browsing logs from *diginetica.com*. It is commonly utilized in session-based or sequential recommendation, focusing solely on transaction data and neglecting user queries. However, our experiments incorporate both user clicks (on items) and queries within sessions. It is noteworthy that, due to the inclusion of query information, our processed dataset is significantly smaller than the "item-only" homogenous Diginetica dataset after filtering.

[2]https://competitions.codalab.org/competitions/11161

2. **Unsplash**: *Unsplash*[3] is a dataset derived from the freely-usable[4] photography platform *unsplash.com*, incorporating users' search and download logs. The *lite*-version of the dataset is used.

3. **Stock-Industrial**: *Stock* is the primary dataset used in our experiments, sourced from a commercial Stock Image and Video Search platform Adobe Stock[5] spanning Oct. 16 to Oct. 31, 2020. The dataset is constructed from users' search and click logs for our experimental analysis.

To concentrate on query-aware sequential recommendation, we exclude user sequences lacking queries and sequences with less than three item interactions, in accordance with our *leave-last-out* validation and testing protocol.

Implementation Details. We use the Adam optimizer [Kingma and Ba, 2014] with a learning rate of 1×10^{-3} to train all models. The hidden dimensionality (d) is set to 64. Weight decay is chosen from the set $\{0, 1 \times 10^{-6}, 1 \times 10^{-4}, 1 \times 10^{-2}, 1, 10\}$, and dropout probability from $\{0, 0.2, 0.4, 0.6, 0.8\}$. Sequential augmentation parameters α are searched from $\{0.1, 0.2, ..., 1\}$, and ρ from $\{1, 1.1, ..., 2\}$. For the three datasets, the maximum length of query words is set to 5, and the maximum length of user sequences is 50. The batch size for Diginetica and Unsplash is 128, and evaluation is performed using *all-item-ranking* [Krichene and Rendle, 2020] (positive item with all negatives forming candidate lists). For the Stock dataset, a batch size of 512 is used during training, and *sampled-item-ranking* is used during testing (forming candidate lists using the positive item with 1000 randomly sampled negatives per user). Experiments on Diginetica and Unsplash utilize a single-`2080s`-GPU server (8 GiB GPU, 32 GiB CPU), while Stock experiments are conducted on a four-`V100`-GPU server (16 GiB GPU per card, 128 GiB CPU). The code and models will be released on public datasets upon acceptance.

Baselines. We present two sets of recommendation baselines to demonstrate the efficacy

[3]https://unsplash.com/data
[4]Used for case studies in this research.
[5]https://stock.adobe.com

of user queries in sequential recommendation. The initial set comprises standard homogeneous item-only sequential recommenders that solely factor in users' item interactions:

1. **FPMC [Rendle et al., 2010].** A sequential recommender combining Markov chains with matrix factorization to capture short-term user dynamics and long-term preferences.

2. **GRU4Rec$^+$ [Hidasi and Karatzoglou, 2018].** An RNN-based method to model users' item interaction sequences for session-based recommendation [Hidasi et al., 2015, Hidasi and Karatzoglou, 2018], we use the improved version [Hidasi and Karatzoglou, 2018]. We treat each user's item interaction sequence as a session.

3. **SASRec [Kang and McAuley, 2018].** A self-attentive sequential recommender that uses transformer blocks to assign item-to-item attention weights and predict the next item.

4. **BERT4Rec [Sun et al., 2019].** A BERT-like [Devlin et al., 2019] sequential recommender capturing bi-directional contextual item information via a *cloze* task. It is one of the state-of-the-art models for next-item recommendation.

5. **SSE-PT [Wu et al., 2020].** A state-of-the-art self-attentive recommender, extending SASRec by using explicit user representations with SSE [Wu et al., 2020] regularizations.

The second group comprises widely adopted context-aware baselines that integrate query information without accounting for the sequential order of users' item interactions:

1. **Non-personalized Search (NS).** We project the query representations and item representations into a joint embedding space and measure the relevance with inner product similarities. We only use co-occurrence of queries and items in the data, so this model is non-personalized.

2. **QBPR.** We adopt the VBPR [He and McAuley, 2016b] method to incorporate query (instead of visual) information as part of item representations. We call this QBPR (i.e. Query-Aware BPR).

Table 3.2. Method Comparison. Highest/second highest scores are bolded/underlined. Here Δ_1 represents the relative improvement from SASRec to our QUERYSR, Δ_2 represents the relative improvement from the best baselines to QUERYSR. We use * to note sampled-item-ranking (1k negatives, details in the implementation details.) rather than all-item-ranking metrics (which is infeasible for the industrial-scale dataset). There are three groups of models: (1) Item-Only homogeneous sequential baselines; (2) Query-Aware baselines; (2) Our QUERYSR.

		Diginetica		**Unsplash**		**Stock***	
Group	**Metric**	HR@20	N@20	HR@20	N@20	HR@20	N@20
(1)	**FPMC**	0.2996	0.1953	0.5307	0.2669	0.3832	0.2993
	GRU4Rec+	0.2174	0.1160	0.5874	0.2924	0.4284	0.3412
	SASRec	0.3508	0.1979	0.5881	0.2972	0.4527	0.3404
	BERT4Rec	0.3221	0.1714	<u>0.5912</u>	0.2697	0.4472	0.3445
	SSE-PT	0.3425	0.2315	<u>0.5912</u>	<u>0.2985</u>	<u>0.4549</u>	<u>0.3541</u>
(2)	**NS**	0.2948	0.1760	0.5317	0.2039	0.2215	0.1677
	QBPR	0.1438	0.0986	0.2723	0.1221	0.2153	0.1129
	FM	<u>0.3571</u>	<u>0.2323</u>	0.5199	0.1984	0.1749	0.1319
	NeuFM	0.3359	0.2245	0.5499	0.2109	0.2625	0.1955
(3)	**QUERYSR**	**0.4037**	**0.2361**	**0.6796**	**0.3439**	**0.4831**	**0.3708**
	Δ_1	+15.1%	+19.3%	+15.6%	+15.7%	+06.7%	+08.9%
	Δ_2	+13.0%	+01.6%	+15.0%	+15.2%	+06.2%	+04.7%

3. **FM [Rendle, 2010].** Factorization machines (FM) are a classic context-aware recommendation technique to encode context information. We use the same "bag-of-words" query representations as what we used in Section 3.2.3. We adopt first-order feature interactions among user, item, and query features.

4. **NeuFM [He and Chua, 2017].** A deep architecture for effective feature interaction modeling for context-aware recommendation. We use the same features as FM and adopt MLPs for higher-order feature interactions.

3.3.2 Result Analysis

Model General Performance

Table 3.2 shows recommendation performance of our QUERYSR and other baselines on all datasets. Our conversations are from multiple aspects as follows:

Baseline Performance. The optimal baselines vary across datasets depending on user behavior. For the Diginetica dataset, query-aware baselines, specifically Factorization Machines (FM)[6], demonstrate superior ranking performance compared to homogeneous user sequential pattern baselines (e.g., SSE-PT). This suggests the significance of user historical queries as essential signals for predicting users' subsequent item interactions. On the other two datasets, state-of-the-art sequential recommenders (e.g., SASRec, BERT4Rec, SSE-PT) outperform query-aware baselines, indicating the heightened importance of sequential patterns for predicting the next item on these datasets. Notably, SASRec outperforms BERT4Rec, with both models using the same loss function from Equation (3.10) for fair comparison. In the original implementations, SASRec employs Binary Cross Entropy (BCE) loss with one negative, while BERT4Rec uses Cross Entropy loss with $|\mathscr{I}| - 1$ negatives [Kang and McAuley, 2018, Sun et al., 2019].

General Performance Improvement. Our model surpasses all baselines across all datasets. Specifically, Δ_1 indicates a 12.5% improvement in HR@20 and a 14.6% improvement in N@20 against the SASRec backbone model on average, highlighting the efficacy of integrating query information and sequential augmentation strategies. Further insights are provided in the ablation study below. Δ_2 signifies the relative improvement against the top-performing baselines for each dataset. For instance, our approach achieves an average gain of 11.4% in HR@20 and 7.2% in N@20.

Improving Across Datasets. The impact of incorporating queries varies across datasets. While our model consistently improves on all datasets, the relative enhancements on Diginetica and Unsplash outweigh those on the Stock dataset. For instance, Δ_1 indicates a 15.6% HR@20

[6]NeuFM's comparatively inferior performance against FM on Diginetica may be attributed to overfitting on this small dataset, despite employing regularization techniques such as dropout and weight decay.

Table 3.3. Ablation study for the effectiveness of query information and sequence augmentation. Here *Q* represents incorporating query information as Section 3.2.1; *A* represents our sequence augmentation method as Section 3.2.2; *R* means using uniform random replacement strategy to replace *A*.

		Ablation			
Dataset	**Metric**	QUERYSR	w/ R	w/o A	w/o Q
Diginetica	HR@20	**0.4037**	0.3996	0.3908	0.3508
	N@20	**0.2361**	0.2351	0.2287	0.1979
Unsplash	HR@20	**0.6796**	0.6672	0.6698	0.5881
	N@20	**0.3439**	0.3335	0.3403	0.2972
Stock*	HR@20	**0.4831**	0.4802	0.4758	0.4527
	N@20	**0.3708**	0.3686	0.3653	0.3404

Table 3.4. Different query incorporation methods.

Incorporation	**HR@20**	**N@20**
Heterogeneous	**0.3908**	**0.2287**
Early	0.3719	0.2169
FDSA [Zhang et al., 2019] (Late)	0.3697	0.2081
NOVA [Liu et al., 2021]	0.3594	0.2031

gain for Unsplash compared to a 6.7% HR@20 gain for Stock. This discrepancy is likely attributed to the shorter average sequence lengths of Diginetica and Unsplash, such as 6.74 for Unsplash *vs.* 26.07 for Stock (refer to Table 6.2). Shorter sequences provide insufficient information and introduce more uncertainty regarding user intent. Consequently, incorporating user textual queries proves more beneficial in such scenarios.

Ablation Study

Effectiveness of Each Component. Table 3.3 presents ablation studies on essential components: (1) The effectiveness of incorporating queries is demonstrated by comparing our model with *w/o Q*. The results, along with *w/o A*, highlight that integrating query information significantly enhances recommendation accuracy, as detailed in the case studies (refer to Section 3.3.2). (2) The effectiveness of our sequential augmentation is illustrated by comparing

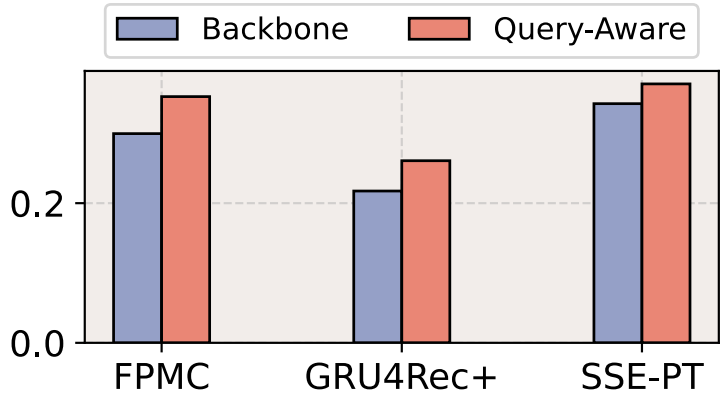
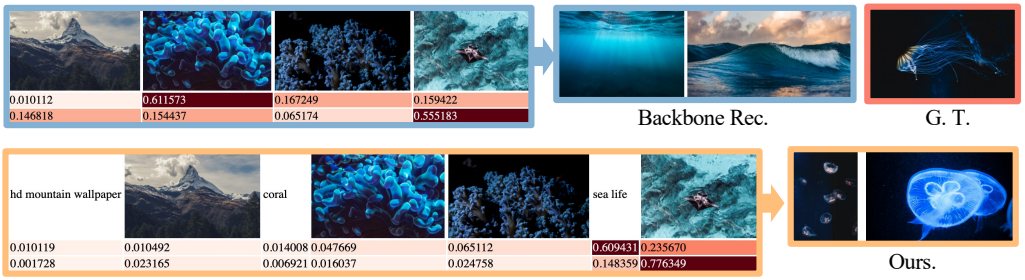


Figure 3.3. Model accuracy improvements for diffrerent QUERYSR backbones, in terms of HR@20 on the testing set of Diginetica.

ours *vs. w/o A*, showing that our augmentation strategy improves query-aware sequential recommenders. Additionally, ours *vs. w/ R* indicates that introducing the query-item graph for augmentation outperforms uniform random replacement, as seen in SSE-PT [Wu et al., 2020].

Different Query Incorporation Methods. Table 3.4 presents the evaluation of various query exploration methods on Diginetica without sequential augmentation. Empirically, Heterogeneous outperforms the other three methods in organizing query- and item-sequences, with a HR@20 of 0.3908 compared to 0.3719 (Early), 0.3697 (FDSA [Zhang et al., 2019]), and 0.3594 (NOVA [Liu et al., 2021]). These methods were introduced in Section 3.2.1.

More Backbone Models. To demonstrate the flexiblity of our query-aware framework, we conducted experiments using alternative sequential recommenders as backbones. Figure 3.3 illustrates that, despite varying model architectures (e.g., FPMC being Markov-Chain-based, GRU4Rec+ being RNN-based, SSE-PT being Transformer-based), integrating user query information within our framework consistently enhances ranking performance significantly. For instance, on Diginetica, the HR@20 of the query-aware FPMC surpasses the corresponding backbone by 17.7% (relative improvement).



CASE #1: Seq. ID: bb0e4fa7-845f-4b76-b727-8f7d33493808

CASE #2: Seq. ID: 27a26997-bc2f-4571-acfb-76cc9deb8eb9

Figure 3.4. Case studies (in Section 3.3.2) to show the influence of user queries to "attention" weights in sequential recommenders. Blue box is the backbone model input sequence and predicted results; yellow box is our query-aware model input and results; red box is the ground truth (G.T.). We visualize "attention" weights from two heads in the first layer (second layer "attention" mostly focuses on the last item, which we omit), where larger weights are darker. Figure 3.5 also follows the same colors.

Visualization and Case Study

We visualize the first-layer attention weights of our two-layer QUERYSR model and demonstrate four representative cases in Figures 3.4 and 3.5. They show the benefits of introducing user textual query information as below:

Guide Attention Weights: CASE#1&2 in Figure 3.4, we present two cases illustrating the impact of incorporating user queries on attention networks for improved relevance. In Case #1, the attention weights of the backbone model (blue box) are overly distributed, emphasizing "coral" images and even an unrelated "mountain" image (e.g., 14.9% weight from attention head #2). Our query-aware model (green box) accurately identifies the boundary of the user behavior sequence and, with knowledge of the query "sea life", provides more relevant recommendations. Case #2 demonstrates that our model not only focuses on the last "query session" but attends to both "pool" and "girl swimming" images, aligning with the ground truth.

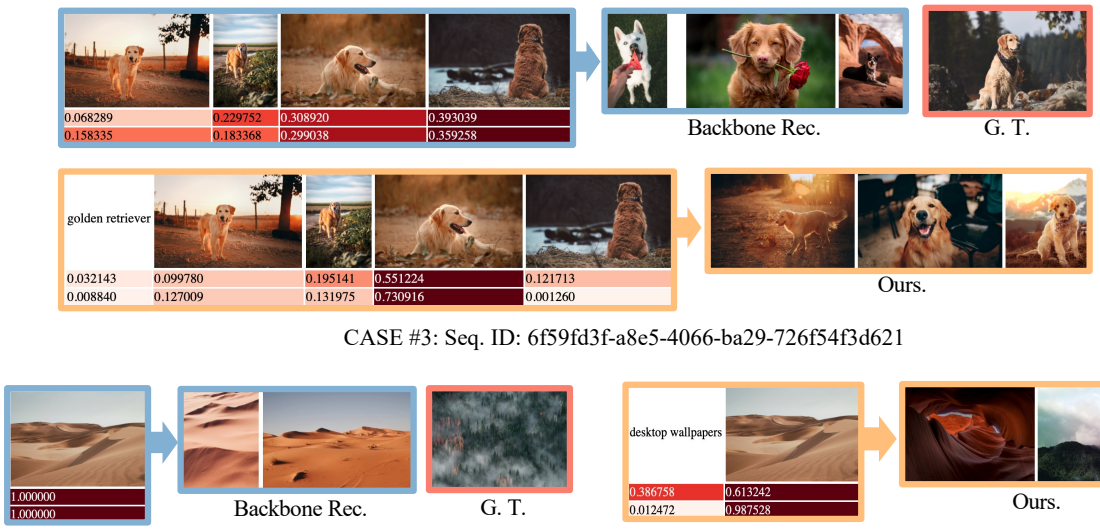


Figure 3.5. Case studies (in Section 3.3.2) to show the influence of user queries on recommendation *granularity*.

Influence on Recommendation Granularity: CASE#3&4 in Figure 3.5 illustrates the impact of user queries on recommendation granularity in CASE#3 and CASE#4. The results from the backbone model in CASE#3 are deemed too "broad" due to the absence of information about "golden retriever", a limitation addressed by query-aware recommenders. Conversely, the backbone model results in CASE#4 are considered too "narrow" as they only recommend "desert" to users, lacking knowledge about the granularity of user intent. Our proposed model, aware of user queries such as "desktop wallpapers", produces more diverse yet relevant outputs consistent with the ground truth.

Limitation and Discussion

We have demonstrated the efficacy of integrating user queries into sequential recommendation within our framework; however, several noteworthy issues remain unexplored. Firstly, our models are trained from scratch. Is it feasible to leverage existing pre-trained models/embeddings to enhance performance with external textual/visual data? Secondly, many platforms use separate models for search and recommendation. In *query-aware* sequential recommendation, can these

models be unified, or can knowledge be shared to mutually reinforce each other? Thirdly, while our datasets are primarily sourced from image search platforms (Unsplash, Stock), originally intended for image retrieval, they may not align perfectly with our query-aware setting. We anticipate that datasets from alternative recommendation scenarios (e.g., e-commerce, music) may unveil more intricate and compelling sequential patterns.

3.4 Conclusion

In this chapter, we are focusing on heterogeneous sequential recommender systems. We posit that *user textual queries* serve as a crucial contextual cue for capturing and forecasting users' evolving intent beyond item clicks only. We introduce a novel query-aware sequential recommendation setting, presenting a comprehensive framework for integrating query information into sequential recommendation. Additionally, we instantiate a self-attentive sequential model specifically designed for query-aware sequential recommendation tasks. Experimental results on three datasets underscore the efficacy of incorporating query information, leading to a substantial improvement in ranking performance compared to existing models. Furthermore, we provide visualizations of representative cases and engage in discussions on intriguing challenges within this new setting. Our proposed QUERYSR model effectively explores incorporation strategies for recommendation settings where user textual inputs and clicks coexist, paving the way towards more complex scenarios like conventional recommender systems.

Chapter 3, in part, is a reprint of the material as it appears in "Query-Aware Sequential Recommendation." by Zhankui He, Handong Zhao, Zhaowen Wang, Zhe Lin, Ajinkya Kale, and Julian McAuley, in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management in 2022*, referenced as [He et al., 2022a]. The dissertation author was the primary investigator and author of this paper.

Part II

Explainability: Enhancing User Understanding and Trust

Chapter 4

Controllable Explanations for Recommender Systems

In this chapter, we begin discussing generating natural language explanations for users to provide more information about the recommended items, which is an important component of conventional recommender systems to answer users' questions about "why those items are recommended to me?". The desired explanations or justifications should be personalized, to match different user styles, accurate, to prevent hallucinations that hurt user trust, and also informative, to provide much useful information for decision making.

In this chapter, we demonstrate a model, UCEPIC, for controllable explanation generations in recommendations. Existing personalized natural language generates often use aspect planning to guide the generation process. However, while showing promise, these aspect-planning methods can struggle to accurately incorporate specific information, hindering the persuasiveness of the generated explanations. We argue that introducing lexical constraints can mitigate these shortcomings. In this paper, we introduce UCEPIC, a model that unifies aspect planning and lexical constraints within an insertion-based generation framework to produce high-quality, personalized explanations for recommendations. This UCEPIC model significantly enhances the diversity and informativeness of generated explanations, as demonstrated on datasets like RateBeer and Yelp, improving the quality of explanations, and can be considered as an important component in conversational recommender systems.

Table 4.1. Comparison of previous explanation generators for recommendation in group (A), general lexically constrained generators in group (B), and our UCEPIC in group (C).

Group	Methods	Personalized generation	Aspect planning	Lexical constraints	Random keyphrases
(A)	**ExpansionNet** [Ni and McAuley, 2018]	✓	✓	✗	✗
	Ref2Seq [Ni et al., 2019]	✓	✓	✗	✗
	PETER [Li et al., 2021b]	✓	✓	✗	✗
(B)	**NMSTG** [Welleck et al., 2019]	✗	✗	✓	✗
	POINTER [Zhang et al., 2020d]	✗	✗	✓	✗
	CBART [He, 2021]	✗	✗	✓	✓
(C)	**Ours**	✓	✓	✓	✓

4.1 Introduction

The recent trend in recommendation systems involves providing justifications or explanations in natural language [Li et al., 2021b, Ni and McAuley, 2018, Lu et al., 2018, Li et al., 2017b, 2020c, 2023b, Ni et al., 2019], which is also an essential component for conversational recommender systems. This approach aims to present product information in a personalized manner and explain how the recommendation aligns with user preferences. Specifically, for a given user-item pair, the system generates explanations like "*nice TV with 4K display and Dolby Atmos!*" to enhance the quality of personalized explanations. Recent studies emphasize aspect planning, which involves incorporating different aspects [Li et al., 2021b, Ni and McAuley, 2018, Li et al., 2023b, Ni et al., 2019] into the generation process. This strategy ensures that the generated explanations cover relevant aspects, making them more pertinent to both products and user interests.

While current methods show promise, they struggle to incorporate precise and accurate information into explanations. This limitation arises because aspects, such as "screen" for a TV, predominantly control the high-level sentiment or semantics of the generated text (e.g., "*good screen and audio*!"). However, many informative product attributes, like "*nice TV with 4K display and Dolby Atmos!*", are too specific to be accurately generated. Despite efforts by

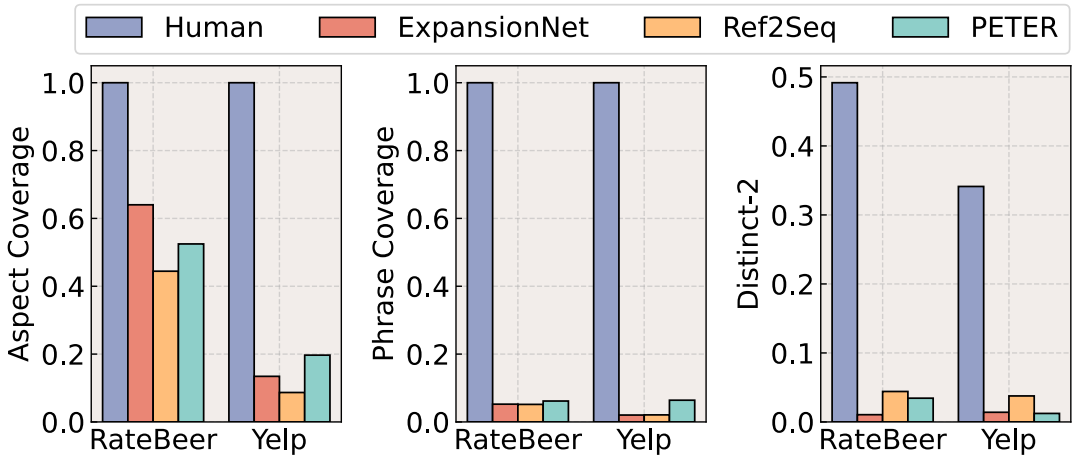


Figure 4.1. Preliminary experiments on the aspect coverage, phrase coverage, and Distinct-2 of generated explanations from previous models ExpansionNet [Ni and McAuley, 2018], Ref2Seq [Ni et al., 2019] and PETER [Li et al., 2021b] on RateBeer and Yelp datasets. Check details in Section 4.3.2

aspect-planning explanation generators to extract expressive and personalized explanations from users' textual reviews [Li et al., 2021b, Ni and McAuley, 2018, Li et al., 2023b, Ni et al., 2019], our preliminary experiments reveal that numerous informative and specific keyphrases from the training corpus (i.e., user reviews) vanish in generated explanations.

As depicted in Figure 4.1, explanations generated by previous methods lack specific keyphrases, resulting in lower Distinct (diversity) scores compared to a human oracle. Consequently, existing methods with aspects only face challenges in generating (1) overly general sentences (e.g., "*good screen!*") that struggle to provide diverse and informative explanations to users, and (2) sentences with inaccurate details (e.g., "*2K screen*" for a 4K TV), which diminish users' trust and relevance to the product.

To address the above problems, we propose incorporating more concrete lexical constraints into recommendation explanations alongside aspects. Our approach involves a model that integrates *lexical constraints* and *aspect planning*. Introducing lexical constraints ensures the utilization of specific keyphrases (e.g., "*Dolby Atmos*"), guaranteeing the inclusion of accurate and detailed information. Similar to the aspect selection in previous explanation generators [Ni

et al., 2019, Li et al., 2021b], these lexical constraints may originate from various sources such as *explanation systems*, which select item attributes using specific strategies; *vendors*, who highlight product features; and *users*, who can modify generated explanations by altering the lexical constraints of interest or guide recommendation-related conversations with different focused topics. Consequently, the informativeness, relevance, and diversity of generated explanations can be significantly enhanced compared to previous methods relying solely on aspect planning. Meanwhile, aspect planning retains its utility when specific information is absent, and multiple aspects need coverage.

To achieve the goal of unifying aspect-planning and lexical constraints for generating explanations in Recommendation, we introduce UCEPIC. Building UCEPIC presents challenges: First, lexical constraints are incompatible with existing explanation generation models (see group (A) in Table 4.1), primarily based on auto-regressive generation frameworks [Li et al., 2019, Ni and McAuley, 2018, Li et al., 2020b, 2021a, Hua and Wang, 2019, Moryossef et al., 2019], which lack guaranteed positions for lexical constraints using a "left-to-right" generation strategy. Second, insertion-based generation models (see group (B) in Table 4.1) can naturally include lexical constraints but struggle to incorporate personalization and aspects within the "encoder-decoder" framework.

To address the first challenge, UCEPIC adopts an insertion-based generation framework and undergoes *robust insertion pre-training* on a bi-directional transformer. This pre-training imparts UCEPIC with the fundamental ability to generate text and handle diverse lexical constraints. Inspired by Masked Language Modeling (MLM) [Devlin et al., 2019], we propose an insertion process that progressively introduces new tokens randomly, enhancing UCEPIC's robustness to random lexical constraints.

To overcome the second challenge, UCEPIC employs *personalized fine-tuning* for personalization and aspect awareness. To address the tendency to "ignore references" in existing insertion-based models, we suggest treating references as inserted tokens during training. This approach ensures that the model learns to insert tokens relevant to references. For aspect planning,

Table 4.2. Notations.

Notation	Description
R^u, R^i	historical review profile of user u and item i.
E^{ui}	generated explanation when item i is recommended to user u.
A^{ui}	aspects controlling explanation generation for item i and user u.
C^{ui}	lexical constraints (e.g., keywords) controlling explanation generation for item i and user u.
S^k, $\hat{S}^k$	text sequence of the k-th stage generation. S^k is training data and $\hat{S}^k$ is model prediction.
$I^{k,k-1}$, $\hat{I}^{k,k-1}$	intermediate sequence between S^{k-1} and S^k. (training data and model prediction)
$J^{k,k-1}$, $\hat{J}^{k,k-1}$	insertion number sequence between S^{k-1} and S^k. (training data and model prediction)
$\mathbf{D}$	a bi-directional transformer for encoding.
$\mathbf{H}_{MI}$	a linear projection layer for insertion numbers.
$\mathbf{H}_{TP}$	a multilayer perceptron with activation function for token prediction.

we introduce aspects as a special insertion stage, generating aspect-related tokens as a starting point for subsequent generation.

UCEPIC is the first explanation generation model that integrates aspect planning and lexical constraints, resulting in substantial enhancements in *relevance*, *coherence*, and *informativeness* when compared to existing methods. The key contributions of this chapter are succinctly outlined below:

- Identification of limitations in the sole reliance on aspect planning in current explanation generation models and introduction of lexical constraints to address these limitations.
- Introduction of UCEPIC, which incorporates robust insertion pre-training and personalized fine-tuning to unify aspect planning, lexical constraints, and references within an insertion-based generation framework.
- Execution of extensive experiments on two datasets, with both objective metrics and human evaluations demonstrating that UCEPIC significantly enhances the diversity, relevance,

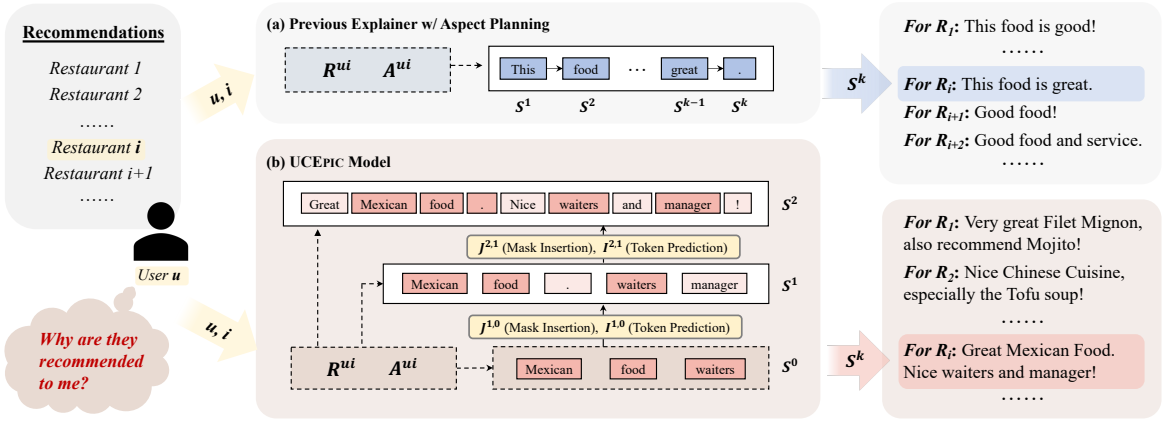


Figure 4.2. Overview of generating explanations for a given user and recommended items using (a) an aspect-planning autoregressive generation model; using (b) our UCEPIC that unifies aspect-planning and lexical constraints.

coherence, and informativeness of generated explanations.

4.2 Proposed Approach: UCEPIC

We present the methodology for aspect planning and incorporation of lexical constraints in explanation generation as follows. Given a user persona R^u and item profile R^i corresponding to user u and item i respectively, the generation model employing aspect planning produces the explanation E^{ui} associated with a designated aspect A^{ui}. This explanation may not necessarily include specific predefined words.

However, for lexical constraints, provided with a set of constraints $\mathrm{C}^{ui} = \{c_1, c_2, \ldots, c_m\}$, representing phrases or keywords, the model generates an explanation $\mathrm{E}^{ui} = (w_1, w_2, \ldots, w_n)$ that must precisely incorporate all given lexical constraints c_i, denoted as $c_i = (w_j, \ldots, w_k)$. These lexical constraints may originate from users, businesses, or item attributes recommended by personalized systems in practical applications. Our approach unifies both constraint types in a single model [1]. Our focus is on the explanation generation method, assuming predefined aspects and lexical constraints. The notations used are summarized in Table 4.2.

[1] UCEPIC operates in two modes: generating under aspect planning or generating under lexical constraints

Table 4.3. Data construction examples.

Data	Example
S^K (sentence)	<s>Good tacos. Love the crispy citrus + tropical fruits flavor. </s>
$I^{K,K-1}$	<s> [MASK] tacos. Love the [MASK] [MASK] + tropical fruits flavor. </s>
$J^{K,K-1}$	[1 0 0 0 2 0 0 0 0 0 0]
S^{K-1}	<s>tacos. Love the + tropical fruits flavor. </s>
...	...
S^0 (lexical constraints)	<s>tropical fruits flavor </s>

4.2.1 Robust Insertion and Data Construction

Our motivation is presented as follows. Previous methods for explanation generation (Ni et al., 2019; Li et al., 2021) typically use auto-regressive approaches conditioned on personalized inputs such as personalized references and aspects. In the auto-regressive process from Figure 4.2 (a), words are generated in a "left-to-right" direction, making it challenging to incorporate lexical constraints into the generation process. Conversely, in the insertion-based generation depicted in Figure 4.2 (b), new tokens are progressively inserted based on existing words, allowing for easy containment of lexical constraints by considering constraints as a starting stage of insertion.

Formulation. We represent the insertion-based generation process as a progressive sequence of K stages, denoted as $S = \{S^0, S^1, \ldots, S^{K-1}, S^K\}$. Here, S^0 represents the initial stage with lexical constraints, and S^K is the final generated text. For each $k \in \{1, \ldots, K\}$, S^{k-1} forms a sub-sequence of S^k. The generation process concludes when no new tokens are inserted into S^K. In the training phase, sentences are prepared as training pairs. Specifically, text sequences are paired at adjacent stages (S^{k-1}, S^k), reflecting the reverse of the insertion-based generation process. Each explanation E^{ui} in the training data is decomposed into a consecutive series of pairs: $(S^0, S^1), (S^1, S^2), \ldots, (S^{K-1}, S^K)$. In the constructed training data, the final stage S^K corresponds to the explanation text E^{ui}.

Data Construction. Given a sequence stage S^k, the previous stage S^{k-1} is obtained through two operations: masking and deletion. Tokens in a sequence are randomly masked with a probability p using Masked Language Modeling (MLM) to produce the intermediate

Algorithm 1. Insertion in the k-th Stage

procedure INSERTION($\hat{S}^{k-1}$)
  $\hat{J}^{k,k-1} \leftarrow$ predict number of masks from $\hat{S}^{k-1}$ via Equation (4.1) ;
  $\hat{I}^{k,k-1} \leftarrow$ build intermediate sequence from $\hat{J}^{k,k-1}$ and $\hat{S}^{k-1}$;
  $\hat{S}^{k} \leftarrow$ predict masked tokens in $\hat{I}^{k,k-1}$ via Equation (4.2);
  return predicted sequence $\hat{S}^{k}$;
end procedure

sequence $I^{k,k-1}$. Subsequently, `[MASK]` tokens are deleted from the intermediate sequence $I^{k,k-1}$ to derive the stage S^{k-1}. The number of deleted `[MASK]` tokens following each token in $I^{k,k-1}$ is recorded as an insertion number sequence $J^{k,k-1}$. Each training instance comprises four sequences: $(S^{k-1}, I^{k,k-1}, J^{k,k-1}, S^{k})$. An illustrative example of the data construction process is presented in Table 4.3. Since $T * \mathrm{p}$ tokens are deleted in sequence S^{k}, where T is the length of S^{k}, the average number of deletions denoted as K is given by $\log_{\frac{1}{1-\mathrm{p}}} T$. Models trained on this data are using knowledge from BERT-like models that conduct a similar pre-training process involving masked word prediction.

Insertion generation (see Algorithm 1) is a reverse process of data construction. In each insertion step prediction from $\hat{S}^{k-1}$ to $\hat{S}^{k}$, the model reconstructs text sequences through two operations: mask insertion and token prediction. Initially, UCEPIC inserts `[MASK]` tokens between any two existing tokens in $\hat{S}^{k-1}$ to obtain $\hat{I}^{k,k-1}$ based on $\hat{J}^{k,k-1}$ predicted by an insertion prediction head. Subsequently, UCEPIC, using a language modeling head, predicts the masked tokens in $\hat{I}^{k,k-1}$ and restores `[MASK]` tokens into words to derive $\hat{S}^{k}$.

Modules. Our model employs a bi-directional Transformer architecture with two distinct prediction heads for mask insertion ($\mathbf{H}_{MI}$) and token prediction ($\mathbf{H}_{TP}$). The model architecture is closely aligned with that of RoBERTa [Liu et al., 2019]. The bi-directional Transformer $\mathbf{D}$ predicts mask insertion numbers and word tokens using the heads $\mathbf{H}_{MI}$ (linear projection layer) and $\mathbf{H}_{TP}$ (multilayer perceptron with GeLU activation) respectively. The final predictions for

mask insertion numbers and word tokens are calculated as follows:

$$y_{MI} = \mathbf{H}_{MI}\left(\mathbf{D}\left(\hat{S}^{k-1}\right)\right),\ \hat{J}^{k,k-1} = \text{argmax}(y_{MI}) \tag{4.1}$$

$$y_{TP} = \mathbf{H}_{TP}\left(\mathbf{D}\left(\hat{I}^{k,k-1}\right)\right),\ \hat{S}^{k} = \text{argmax}(y_{TP}) \tag{4.2}$$

where $y_{MI} \in \mathbb{R}^{l_s \times d_{ins}}$ and $y_{TP} \in \mathbb{R}^{l_I \times d_{vocab}}$, l_s and l_I are the length of $\hat{S}^{k-1}$ and $\hat{I}^{k,k-1}$ respectively, d_{ins} is the maximum number of insertions and d_{vocab} is the size of vocabulary. $\hat{I}^{k,k-1}$ is obtained by inserting `[MASK]` tokens into $\hat{S}^{k-1}$ according to $\hat{J}^{k,k-1}$.

UCEPIC employs a robust insertion method for general text generation without personalization during pre-training, as the random insertion process is more intricate to learn than the traditional autoregressive generation process. This pre-trained model is capable of generating sentences based on randomly provided lexical constraints.

4.2.2 Personalized References and Aspect Planning

To incorporate personalized references and aspects, one approach involves employing a separate text and aspect encoder, coupled with an insertion generation mechanism conditioned on the encoder, akin to the sequence-to-sequence model [Sutskever et al., 2014]. However, our investigation reveals that utilizing a pre-trained insertion model with an additional encoder tends to produce similar sentences with varied personalized references and aspects. This phenomenon arises because the pre-trained insertion model interprets lexical constraints and existing tokens in text sequences as robust signals for determining newly inserted tokens. Even when our encoder imparts personalized features, the model exhibits a propensity to overfit to features derived from existing tokens. In the absence of distinct lexical tokens providing diverse starting points, the generated sentences tend to be homogeneous.

Formulation. To enhance understanding of personalization, we suggest treating references and aspects as distinct tokens during the insertion process. Specifically, we formulate a

training stage S_+^k to incorporate references and aspects as follows:

$$\begin{aligned} S_+^k &= \left[\mathrm{R}^{ui}, \mathrm{A}^{ui}, S^k\right] \\ &= \left[w_0^r, \ldots, w_{|\mathrm{R}^{ui}|}^r, w_0^a, \ldots, w_{|\mathrm{A}^{ui}|}^a, w_0, \ldots, w_{|S^k|}\right] \end{aligned} \tag{4.3}$$

where R^{ui}, A^{ui} denote personalized references and aspects; w^r, w^a and w are tokens or aspect ids in references, aspects and insertion stage tokens respectively. Because insertion-based generation relies on token positions to insert new tokens, we create token position ids in Transformer starting from 0 for R^{ui}, A^{ui} and S^k respectively in order to make it consistent for S^k between pre-training and fine-tuning. Similarly, we obtain the insertion number sequence $J_+^{k,k-1} = [\mathbf{0}_{|\mathrm{R}^{ui}|}, \mathbf{0}_{|\mathrm{A}^{ui}|}, J^{k,k-1}]$ and intermediate training stage $I_+^{k,k-1} = [\mathrm{R}^{ui}, \mathrm{A}^{ui}, I^{k,k-1}]$, where $\mathbf{0}_{|\mathrm{R}^{ui}|}$ and $\mathbf{0}_{|\mathrm{A}^{ui}|}$ are zero vectors which have same length as R^{ui} and A^{ui} respectively, because we do not insertion any tokens into references and aspects.

As for the moduels, we encode $\hat{S}_+^k$ and $\hat{I}_+^{k,k-1}$ with bi-directional Transformer $\mathbf{D}$ to get the insertion numbers y_{MI} and predicted tokens y_{TP} as follows:

$$\left[\mathrm{O}_S^{R^{ui}}, \mathrm{O}_S^{A^{ui}}, \mathrm{O}^{S^k}\right] = \mathbf{D}\left(\hat{S}_+^k\right) \tag{4.4}$$

$$\left[\mathrm{O}_I^{R^{ui}}, \mathrm{O}_I^{A^{ui}}, \mathrm{O}^{I^{k,k-1}}\right] = \mathbf{D}\left(\hat{I}_+^{k,k-1}\right) \tag{4.5}$$

$$y_{MI} = \mathbf{H}_{MI}\left(\mathrm{O}^{S^k}\right) \tag{4.6}$$

$$y_{TP} = \mathbf{H}_{TP}\left(\mathrm{O}^{I^{k,k-1}}\right) \tag{4.7}$$

Similar as Equation (4.1) and Equation (4.2), we can get $\hat{J}^{k,k-1}$ and $\hat{S}^k$ by argmax operation. Due to the distinct recognition of personalized references and aspects as unique tokens, UCEPIC integrates token-level information directly as generation conditions, resulting in the generation of diverse explanations.

Recall that existing text sequences serve as strong signals for token prediction. To enhance aspect-planning generation, we introduce two initial stages S_{+a}^0 and S_{+l}^0 for aspects and

lexical constraints, respectively. Specifically, we anticipate that tokens related to aspects can be generated in the initial stage (i.e., no existing tokens) based on given aspects and personalized references. Thus, the aspect starting stage is defined as $S^0_{+a} = [\mathrm{R}^{ui}, \mathrm{A}^{ui}]$. The lexical constraint starting stage is defined as $S^0_{+l} = [\mathrm{R}^{ui}, \mathrm{A}^{pad}, \mathrm{C}^{ui}]$, where A^{pad} represents a special aspect used for lexical constraints. During training, we sample S^0_{+a} with a probability of p to ensure effective learning of aspect-related generation, a capability absent in pre-training.

4.2.3 Model Training and Inference

Training. The training process of UCEPIC is to learn the inverse process of data generation. Given stage pairs (S^{k-1}_+, S^k_+) and training instance $(S^{k-1}_+, I^{k,k-1}_+, J^{k,k-1}_+, S^k_+)$ from pre-processing [2], we optimize the following objective:

$$
\begin{aligned}
\mathscr{L} &= -\log p(S^k | S^{k-1}) \\
&= -\log \underbrace{p(S^k, J^{k,k-1} | S^{k-1})}_{\text{Unique } J \text{ assumption}} \\
&= -\log p(S^k | J^{k,k-1}, S^{k-1}) p(J^{k,k-1} | S^{k-1}) \\
&= -\log \underbrace{p(S^k | I^{k,k-1})}_{\text{Token prediction}} \underbrace{p(J^{k,k-1} | S^{k-1})}_{\text{Mask insertion}}, \\
&\text{where } I^{k,k-1} = \text{MaskInsert}(J^{k,k-1}, S^{k-1}),
\end{aligned}
\tag{4.8}
$$

where MaskInsert denotes the mask token insertion. We make a reasonable assumption that $J^{k,k-1}_+$ *is unique given* (S^k_+, S^{k-1}_+). This assumption is usually true unless in some corner cases multiple $J^{k,k-1}_+$ could be legal (e.g., masking one "moving" word in "a moving moving moving van"); $I^{k,k-1}_+$ by definition is the intermediate sequence, which is equivalent to the given $(J^{k,k-1}_+, S^{k-1}_+)$. In Equation (4.8), we jointly learn (1) the likelihood of mask insertion number for each token from UCEPIC with $\mathbf{H}_{MI}$, and (2) the likelihood of word tokens for the masked tokens from

[2] For fine-tuning with personalized references and aspects, we train the model with stage pairs (S^{k-1}_+, S^k_+) and training instance $(S^{k-1}_+, I^{k,k-1}_+, J^{k,k-1}_+, S^k_+)$

UCEPIC with $\mathbf{H}_{TP}$.

Similar to BERT training [Devlin et al., 2019], we exclusively optimize the prediction of masked tokens. The selection of tokens to mask involves a 0.1 probability of remaining unchanged and a 0.1 probability of being randomly replaced by another token in the vocabulary. For predicting mask insertion numbers, the majority of values in $J_{+}^{k,k-1}$ are 0 due to the absence of token insertions between existing tokens in most cases. To address this imbalance, we randomly mask the 0 values in $J_{+}^{k,k-1}$ with a probability q. Given the similarity of our mask prediction task to masked language models, pre-trained weights from RoBERTa [Liu et al., 2019] are naturally used for initialization of UCEPIC, providing valuable prior knowledge.

Inference. At inference time, the process initiates from the given aspects A^{ui} or lexical constraint C^{ui} to construct the initial stage S_{+a}^{0} or S_{+l}^{0}, respectively. The model, denoted as `UCEPIC`, predicts the sequence of stages $\{\hat{S}_{+}^{1},\ldots,\hat{S}_{+}^{K}\}$ iteratively until no additional tokens are generated or the maximum stage number is reached. The final generated explanation, denoted as $\hat{S}^{K}$, is derived from $\hat{S}_{+}^{K}$ by removing R^{ui} and A^{ui}.

Without loss of generality, we detail the inference process from the $\hat{S}_{+}^{k-1}$ stage to the $\hat{S}_{+}^{k}$ stage: (1) Given $\hat{S}_{+}^{k-1}$, UCEPIC utilizes $\mathbf{H}_{MI}$ to predict the insertion number sequence $\hat{J}_{+}^{k,k-1}$. We set the predicted insertion number as 0 for the given phrases in S_{+l}^{0} to prevent modifications to these phrases. (2) With $\hat{I}_{+}^{k,k-1}$ obtained from MaskInsert$(\hat{J}_{+}^{k,k-1},\hat{S}_{+}^{k-1})$, UCEPIC employs $\mathbf{H}_{TP}$ to predict $\hat{S}_{+}^{k}$ using a specific decoding strategy, such as greedy search or top-K sampling. (3) Given $\hat{S}_{+}^{k}$, UCEPIC checks if the termination requirements are met. If so, the process concludes; otherwise, it repeats step (1). The termination criterion can be a maximum iteration number, or UCEPIC stops when no new tokens are inserted into $\hat{S}_{+}^{k}$.

Table 4.4. Data Statistics.

Dataset	Train	Dev	Test	#Users	#Items	#Aspects
RateBeer	16,839	1,473	912	4,385	6,183	8
Yelp	252,087	37,662	12,426	235,794	22,412	59

4.3 Experiments

4.3.1 Experimental Setting

Data. For *pre-training*, we utilize English Wikipedia[3] with 11.6 million sentences for robust insertion training. For *fine-tuning*, we leverage Yelp[4] and RateBeer [McAuley and Leskovec, 2013] to evaluate our model (refer to Table 6.2). Reviews exceeding a length of 64 are filtered out. Following the approach of [Ni et al., 2019], we randomly reserve two samples from each user's reviews to construct the development and test sets. As per previous works [Ni et al., 2019, 2017], an unsupervised aspect extraction tool [Li et al., 2022a] is used to obtain phrases and corresponding aspects for lexical constraints and aspect planning, respectively. The tool automatically determines the number of aspects for each dataset, providing coarse-grained semantics for the generated explanations. It is noteworthy that the number of aspects is typically much smaller than the number of lexical constraints, and aspects are more high-level.

Baselines. We evaluate model effectiveness through two baseline groups for automatic evaluation in the context of text generation models for recommendation with *aspect planning*. The first group includes existing models:

1. **ExpansionNet** [Ni and McAuley, 2018] generates reviews conditioned on various aspects derived from a given review title or summary.

2. **Ref2Seq** [Ni et al., 2019] is a Seq2Seq model that incorporates contextual information from reviews and employs fine-grained aspects to control explanation generation.

[3]https://dumps.wikimedia.org/
[4]https://www.yelp.com/dataset

3. **PETER** [Li et al., 2021b] is a Transformer-based model utilizing user- and item-IDs along with given phrases to predict words in target explanation generation, positioning it as a state-of-the-art model for explainable recommendation.

We compare various baselines by evaluating their performance in incorporating lexical constraints, specifically keyphrases, during aspect planning. The evaluation focuses on the models' capacity to integrate and reproduce these keyphrases in the generated text. The second category includes general natural language generation models with *lexical constraints*:

1. **NMSTG** [Welleck et al., 2019]: A tree-based text generation scheme that utilizes lexical constraints in prefix tree form. The model generates words to the left and right based on the given constraints, resulting in a binary tree.

2. **POINTER** [Zhang et al., 2020d]: An insertion-based generation method pre-trained on constructed data using dynamic programming.

3. **CBART** [He, 2021]: Utilizes the pre-trained BART [Lewis et al., 2020] and instructs the decoder to insert and replace tokens based on the encoder's guidance.

The second set of baselines lacks the capability to incorporate personalized information or contextual references. These models operate by training and generating text exclusively based on provided lexical constraints. Excluded from consideration are methods such as NRT [Chen and Wang, 2017], Att2Seq [Dong et al., 2017], ReXPlug [Hada and Shevade, 2021], non-natural-language explainable recommenders like EFM [Zhang et al., 2014] and DEAML [Gao et al., 2019], as well as lexically constrained methods CGMH [Miao et al., 2019] and GBS [Hokamp and Liu, 2017], as PETER and CBART demonstrated superior performance. Furthermore, experiments involving the "encoder-decoder" based UCEPIC, mentioned in Section 4.1, are omitted from the baseline, as this model generates identical sentences for all user-item pairs. Detailed baseline settings can be found in Section 4.3.2.

Evaluation Metrics. We evaluate the quality and diversity of generated sentences through two main criteria: generation quality and diversity. Following the approach in [Ni et al., 2019, Zhang et al., 2020d], we use n-gram metrics, including BLEU (B-1 and B-2) [Papineni et al., 2002], METEOR (M) [Banerjee and Lavie, 2005], and ROUGE-L (R-L) [Lin, 2004], to gauge the similarity between the generated text and human oracle. For evaluating generation diversity, we utilize Distinct (D-1 and D-2) [Li et al., 2015]. Additionally, we introduce BERT-score (BS) [Zhang et al., 2020a] as a semantic metric, beyond traditional n-gram-based evaluations.

Implementation Details. We use the `RoBERTa-base` model [Liu et al., 2019] with approximately 130 million parameters (#params) for our study. During training data construction, we randomly mask 20% of tokens in S^k to generate $I^{k,k-1}$. Zeros in $J^{k,k-1}$ are masked with a probability of 0.9. The tokenizer used is byte-level BPE, following the RoBERTa approach. For *pre-training*, we set the learning rate to 5e-5, the batch size to 512, and use the AdamW optimizer [Loshchilov and Hutter, 2019] for one epoch. For *fine-tuning* on downstream tasks, the learning rate is 3e-5, and the batch size is 128, using the same optimizer as pre-training. The training process spans 10 epochs, and the best model on the development set is selected as the final model, which is then evaluated on the test data. During aspect planning and lexical constraint selection, we randomly sample one aspect and one phrase from the target text, respectively [5].

4.3.2 Result Analysis

Automatic Evaluation

Overall Performance. In Table 4.5, we present evaluation results for various generation methods. For aspect-planning generation, our model (UCEPIC) demonstrates comparable performance to the state-of-the-art model PETER. Although PETER achieves superior B-2 and ROUGE-L scores, UCEPIC exhibits significantly higher diversity in its results. This discrepancy may be attributed to the nature of auto-regressive generation models like PETER,

[5]Further details can be found at https://github.com/JiachengLi1995/UCEpic. Additionally, we have released an extra checkpoint pre-trained on personalized review datasets, including Amazon Reviews [Ni et al., 2019] and Google Local [Yan et al., 2023b].

Table 4.5. Performance comparison of the explanation generation models (ExpansionNet, Ref2Seq, PETER), lexically constrained generation models (NMSTG, POINTER, CBART) and UCEPIC. All values are in percentage (%). We underline the highest scores of aspect-planning generation results and the highest scores of lexically constrained generation are **bold**.

Models	RateBeer							Yelp						
	B-1	B-2	D-1	D-2	M	R	BS	B-1	B-2	D-1	D-2	M	R	BS
Human-Oracle	–	–	8.30	49.16	–	–	–	–	–	3.8	34.1	–	–	–
Aspect-planning generation														
ExpansionNet	8.96	1.79	0.20	1.05	16.30	10.13	75.58	4.92	0.47	0.18	1.40	7.78	5.42	76.27
Ref2Seq	17.15	4.17	0.95	4.41	16.66	15.66	80.76	8.34	0.98	0.46	3.77	7.58	11.19	82.66
PETER	25.25	5.35	0.74	3.44	19.19	20.34	84.03	14.26	2.25	0.26	1.23	12.25	14.75	82.55
UCEPIC	27.42	2.89	4.49	29.23	19.54	15.48	83.53	8.03	0.72	1.89	14.75	8.10	11.58	83.53
Lexically constrained generation														
ExpansionNet	5.41	0.49	0.97	4.91	6.09	5.55	76.14	1.49	0.08	0.40	1.90	2.19	1.93	73.68
Ref2Seq	17.94	4.50	1.09	5.49	17.03	15.17	83.72	6.38	0.77	0.51	3.64	7.02	10.58	82.88
PETER	15.03	2.46	2.04	11.40	9.49	13.27	79.08	7.59	1.32	1.52	8.70	7.64	12.24	80.89
NMSTG	22.82	2.30	6.02	50.39	15.17	15.35	82.31	13.67	0.77	**4.57**	**57.02**	9.64	11.13	80.80
POINTER	6.00	0.31	**11.24**	**56.02**	7.41	11.21	81.80	1.50	0.06	5.49	29.76	3.24	5.23	80.85
CBART	2.49	0.54	8.49	34.74	8.45	13.84	83.30	2.19	0.60	5.32	26.79	9.41	15.00	84.08
UCEPIC	**27.97**	**5.09**	5.24	32.04	**19.90**	**17.05**	**84.03**	**13.77**	**3.06**	2.85	20.39	**14.45**	**16.92**	**84.55**

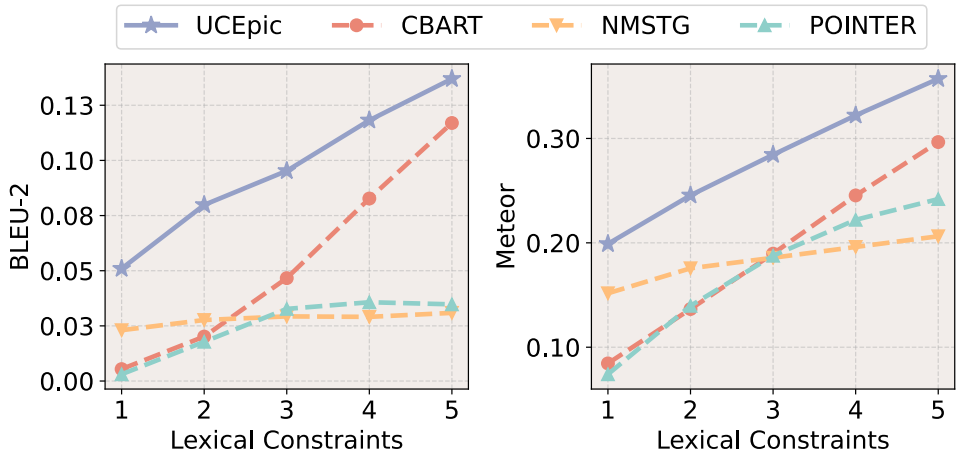


Figure 4.3. Performance (i.e., B-2 and Meteor) of lexically constrained generation models on RateBeer data with different numbers of keyphrases.

which tend to prioritize higher n-gram metric results, while our (insertion-based) UCEPIC considers tokens in both directions during generation. Despite the inherent differences, UCEPIC achieves comparable scores in B-1, Meteor, and BERT metrics when compared to PETER. When subjected to lexical constraints, existing explanation generation models yield lower results than aspect-planning generation, signifying the struggle of current models to incorporate specific information, such as keyphrases, into explanations. While lexically constrained methods produce diverse text, they often insert less-related tokens, resulting in lower coherence (as indicated by low n-gram metric results) compared to UCEPIC. Notably, UCEPIC excels in including keyphrases and learning user-item information from references, outperforming both existing explanation generation models and lexically constrained generation models. In summary, our model (UCEPIC) effectively integrates aspect planning and lexical constraints for explainable recommendations.

Number of Lexical Constraints. Figure 4.3 illustrates the performance of lexically constrained generation models under varying keyphrase numbers. UCEPIC consistently outperforms other models across different quantities of lexical constraints. Specifically, NMSTG and POINTER show limited improvement as the number of keyphrases increases due to their inability to handle random keywords, often breaking given phrases into individual words. The performance

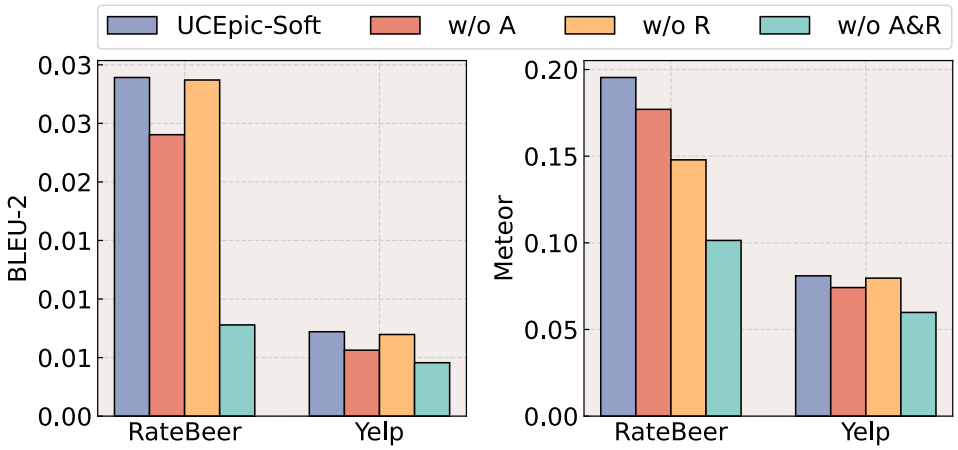


Figure 4.4. Ablation study on aspects and references.

gap between UCEPIC and CBART widens as the number of keyphrases decreases. CBART struggles to generate explanations with only a few keywords, whereas UCEPIC addresses this limitation by incorporating user persona and item profiles from references. These findings suggest that existing lexically constrained generation models are not suitable for explanation generation with lexical constraints.

Ablation Study. To assess the efficacy of our unified method and the indispensability of aspects and references in explanation generation, we conducted an ablation study on two datasets, and the results are illustrated in Figure 4.4. Our model was trained and explanations were generated under three conditions: without aspects (w/o A), without references (w/o R), and without both aspects and references (w/o A&R). The results indicate that the absence of aspects leads to a decrease in BLEU-2 and Meteor scores, emphasizing the guiding role of aspects in shaping the semantics of explanations. Without references, the model tends to generate similar sentences, often containing high-frequency words from the training data. A marked drop in performance is observed when both references and aspects are omitted. Therefore, our unified approach for incorporating references and aspects is deemed effective, providing essential user-item information for explanation generation.

Kind of Constraints. We assess the performance of UCEPIC under various constraints

Table 4.6. UCEPIC with different constraints on Yelp dataset. L denotes lexical constraints.

Constraints	B-1	B-2	D-1	D-2	M	R	BS
Aspect	8.03	0.72	1.89	14.75	8.09	11.58	83.53
L-Extract	13.77	3.06	2.85	20.39	14.45	16.92	84.55
L-Frequent	10.05	0.87	2.02	15.88	9.14	12.23	83.73
L-Random	9.81	0.79	3.00	21.04	8.73	11.61	83.50
Aspect & L	13.12	3.01	2.89	20.34	14.41	16.94	84.56

using the Yelp dataset, presenting the results in Table 4.6. The configurations for Aspect and L-Extract remain consistent, representing UCEPIC under aspect-planning and lexical constraints, respectively, as shown in Table 4.5 [6]. Additionally, we investigate three other constraint types: (1) L-Frequent: Utilizing the most frequent noun phrase of an item as the lexical constraint. (2) L-Random: Randomly sampling the lexical constraint from all noun phrases of an item. (3) Aspect & L: Combining both aspect-planning and lexical constraints, as demonstrated in Table 4.5, and employing both simultaneously. Analysis of the results reveals that: (1) L-Extract and Aspect & L exhibit similar outcomes, suggesting strong constraints from lexical aspects, limiting the controllability of aspect planning on the generation process. (2) Generation with lexical constraints outperforms aspect-planning generation. (3) Selections of lexical constraints (i.e., L-Extract, L-Frequent, L-Random) lead to significant variations in generation performance, indicating potential avenues for further exploration in future research.

Human Evaluation

We conducted a human evaluation of generated explanations using a set of 500 ground-truth explanations extracted from the Yelp dataset. The corresponding generated explanations from PETER-aspect, POINTER, CBART, and UCEPIC were collected for evaluation. Annotators were tasked with selecting the *best* explanation based on different aspects, namely, *relevance*, *coherence*, and *informativeness*, from explanations generated by PETER, POINTER, CBART, and UCEPIC (refer to Section 4.3.2 for detailed information). We define *relevance*, *coherence*

[6] Aspects and phrases are extracted from the generation target, and one aspect and phrase are randomly sampled as model inputs.

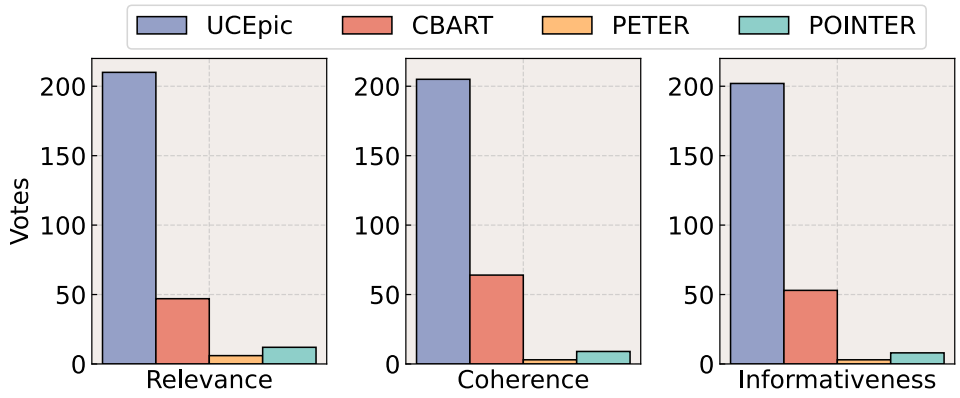


Figure 4.5. Human evaluation on explanation quality.

and *informativeness* as:

- **Relevance:** the details in the generated explanation are consistent and relevant to the ground-truth explanations.
- **Coherence:** the sentences in the generated explanation are logical and fluent.
- **Informativeness:** the generated explanation contains specific information, instead of vague descriptions only.

The voting results are depicted in Figure 4.5. UCEPIC outperforms other methods, particularly in relevance and informativeness. Lexically constrained generation methods (UCEPIC and CBART) enhance explanation quality by incorporating specific product information. However, POINTER, lacking robustness to random keyphrases, does not benefit from improvements through lexical constraints.

Case Studies

We compare explanations generated by existing models (Ref2Seq, PETER), lexically constrained models (POINTER, CBART), and UCEPIC in Table 4.7. Ref2Seq and PETER typically produce non-informative, general sentences due to limitations in traditional auto-regressive generation for specific item information. POINTER and CBART can incorporate

Table 4.7. Generated explanations from Yelp dataset. Lexical constraints (phrases) are highlighted in explanations.

Phrases	**pepper chicken**	**north shore , meat**
Human	Food was great. The pepper chicken is the best. This place is neat and clean. The staff are sweet. I recomend them to anyone!!	Great Italian food on the north shore ! Menu changes daily based on the ingredients they can get locally. Everything is organic and made “clean”. There is no freezer on the property, so you know the meat was caught or prepared that day. The chef is also from Italy! I highly recommend!
Ref2Seq	best restaurant in town ! ! !	what a good place to eat in the middle of the area . the food was good and the service was good .
PETER	This place is great! I love the food and the service is always great. I love the chicken and the chicken fried rice. I love this place.	The food was good, but the service was terrible. The kitchen was not very busy and the kitchen was not busy. The kitchen was very busy and the kitchen was not busy.
POINTER	pepper sauce chicken !	one of the best restaurants in the north as far as i love the south shore . great meat ! !
CBART	Great spicy pepper buffalo wings and chicken wings.	Best pizza on the north shore ever! Meatloaf is to die for, especially with meat lovers.
UCEPIC	Great Chinese restaurant, really great food! The customer service are amazing! Everything is delicious and delicious! I think this local red hot pepper chicken is the best.	I had the best Italian north shore food. The service is great, meat that is fresh and delicious. Highly recommend!

given phrases (e.g., pepper chicken) but may generate inaccuracies (e.g., pepper sauce chicken, chicken wings) as they struggle to learn information from references. In contrast, UCEPIC generates coherent and informative explanations that encompass specific item attributes and maintain high relevance to the recommended item.

More Experiment Implementation Details

Motivating Experiment Details. In this study, we assess the diversity and informativeness of explanations by employing phrase coverage, aspect coverage, and Distinct-2 metrics to evaluate both generated and human-written explanations.

Phrase coverage is determined by extracting noun phrases from explanations using spaCy noun chunks. The comparison of phrases between human-written and generated explanations helps identify covered phrases in the latter. This metric quantifies the extent to which specific information is incorporated into the generated explanations.

Aspect coverage utilizes an aspect extraction tool [Li et al., 2022a] on each dataset to create a table mapping phrases to aspects. The generated explanations are then mapped to aspects by referencing the phrase-aspect table. Aspect coverage is calculated for each sample, indicating how many aspects from the ground-truth explanation are covered in the generated explanations. The average aspect coverage per dataset is reported.

Distinct-2 is computed using the numbers described in Table 4.5.

Baseline Details. For ExpansionNet, we use the default setting which uses hidden size 512 for the RNN encoder and decoder, batch size of 25 and learning rate 2e-4. For aspect planning in ExpansionNet, we use the set of lexical constraints (as concatenated phrases) to replace the *title* or *summary* input as contextual information for training and testing.

For Ref2Seq, we use the default setting with 256 hidden size, 512 batch size and 2e-4 learning rate. For aspect planning, we concatenate our given phrases as references (historical explanations are also incorporated as references following the original implementation) as contextual information in training and testing.

For PETER, we use the original setting with 512 embedding size, 2048 hidden units, 2 self-attention heads with 2 transformer layers, 0.2 dropout. We use the training strategy suggested by the authors. Since original PETER only supports single words as an aspect, we adopt PETER to multiple words with a maximum length of 20 and reproduce the original single-word model on our multi-word model. We input our lexical constraints as the multi-word input for PETER training and testing.

For NMSTG, we use the default settings with an LSTM with 1024 hidden size with the uniform oracle. We convert our lexical constraints into a prefix sub-tree as the input of NMSTG, and then use the best sampling strategy in our testing (i.e., `StochasticSampler`) for NMSTG.

For POINTER, we use the pre-training `BERT-large` [Devlin et al., 2019] (#params $\approx$ 340M.) from WIKI to fine-tune 40 epochs on our downstream datasets. We use all the default settings except batch sizes since POINTER requires 16 GPUs for distributed training that exceeds our computational resources. Instead, we train POINTER with the same configuration on 3 GPUs. For testing, we select the base maximum turn as 3 with the default greedy decoding strategy. We feed lexical constraints as the original implementation.

For CBART, we use the checkpoint pre-trained on `BERT-large` [Devlin et al., 2019] (#params $\approx$ 340M.) with the one-billion-words dataset to fine-tune our downstream datasets. We use the 'tf-idf' training mode and finetune it on one GPU. For testing, we select the greedy decoding strategy. We set other hyper-parameters to default as the code base [7].

Human Evaluation Details. We conduct human evaluation experiments on Yelp datasets to evaluate the generation quality of generated explanations in terms of *relevance*, *coherence* and *informativeness*. We used Amazon Mechanical Turk (MTurk) [8] to gather evaluations, offering a reward of $0.02 per question. The evaluation process involved presenting workers with definitions of *relevance*, *coherence*, and *informativeness*, followed by a randomized order of model-generated explanations to mitigate positional bias. Each question was assigned to three

[7] https://github.com/NLPCode/CBART
[8] https://www.mturk.com

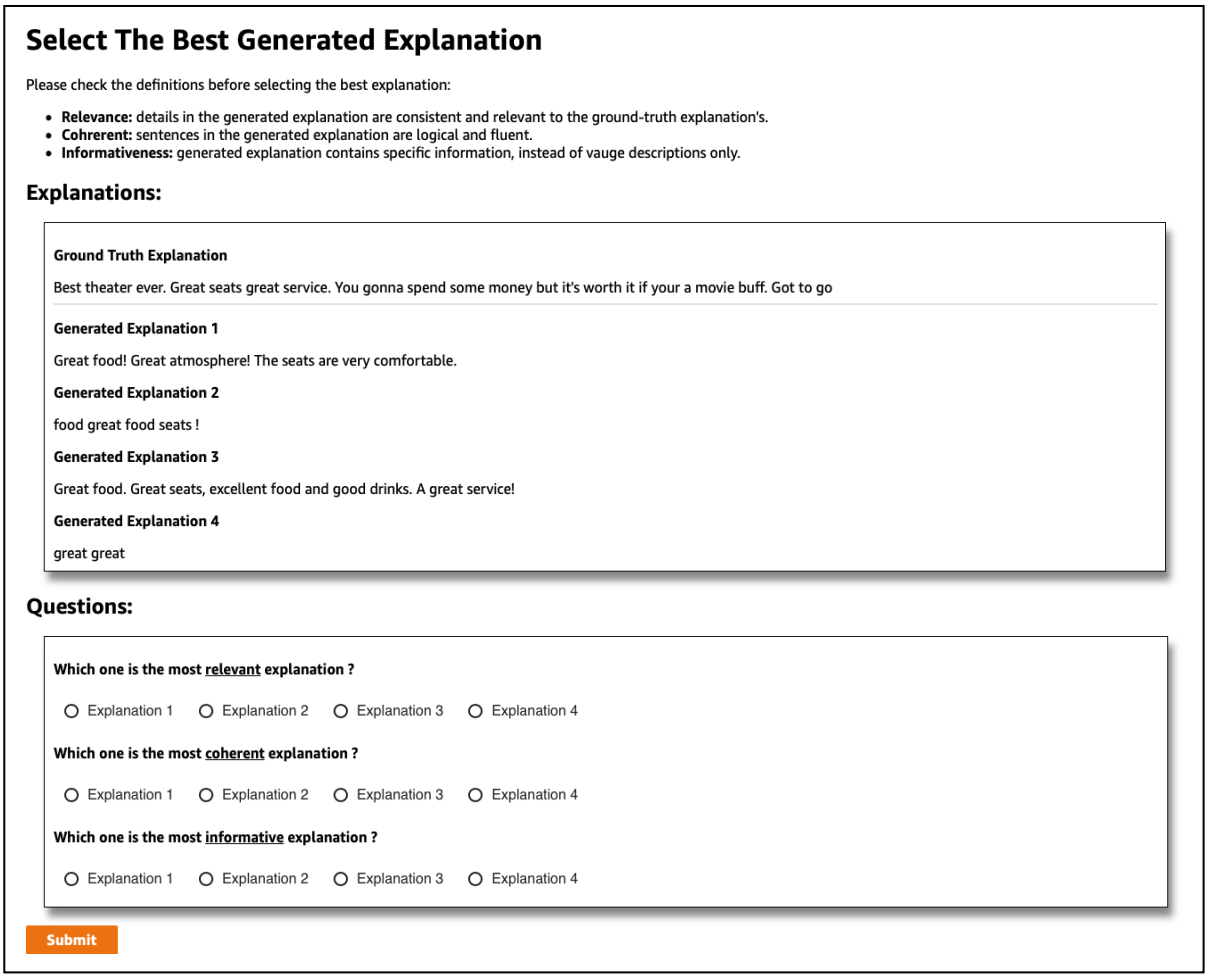


Figure 4.6. Human evaluation example on MTurk.

MTurk workers with a minimum `HIT Approval Rate` of 80% to ensure answer quality. An example of our evaluation template is depicted in Figure 4.6. The collected answers underwent a majority vote process, where a model received a majority vote if it obtained 2 or more out of the 3 worker responses. Questions lacking a majority vote were excluded. In total, we amassed 1,120 valid votes for 370 questions, with 275, 281, and 266 questions achieving majority votes for *relevance*, *coherence*, and *informativeness* respectively.

4.4 Conclusion

In this chapter, we explore the incorporation of lexical constraints in explanation generation to enhance the informativeness and diversity of generated reviews by incorporating specific information. In this way, we are able to build a high-quality explanation generator to provide personalized and useful information for users to make decisions on whether accept the recommended items, and also serving in conversational recommender systems to respond related questions from users.

To achieve this goal, we introduce UCEPIC, an explanation generation model that integrates aspect planning and lexical constraints within an insertion-based generation framework. Comprehensive experiments are conducted using RateBeer and Yelp datasets, demonstrating that UCEPIC surpasses previous explanation generation models and lexically constrained generation models. Human evaluation and a case study affirm that UCEPIC produces coherent and informative explanations closely aligned with the reviewed item.

Chapter 4, in part, is a reprint of the material as it appears in "UCEpic: Unifying Aspect Planning and Lexical Constraints for Generating Explanations in Recommendation." by Jiacheng Li*, Zhankui He*, Jingbo Shang, and Julian McAuley, in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* in 2023, referenced as [Li et al., 2023a]. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Multimodal Explanations for Recommender Systems

In this chapter, we go beyond generating textual explanations for recommendations only. Instead, we explore how to present multimodal (i.e., textual- and visual-) explanations in recommendations to further enrich the relevant information for the user's decision-making. Methodologically, we propose a novel task called *personalized showcases*. This task requires visual and textual elements to provide richer, more compelling explanations for recommendations: First, we select a personalized image set that aligns closely with a user's interests in the context of a recommended item. Second, we generate tailored natural language explanations that are grounded in the selected images, related to the recommended items as well as the user personality.

In this chapter, we also introduce a large-scale dataset collected from *Google Local* (i.e., maps), with a carefully constructed high-quality subset specifically designed for multi-modal explanation generation. Treating this new dataset as a testbed, we propose a personalized multi-modal framework that leverages contrastive learning to produce diverse and visually cohesive explanations. Our experimental results demonstrate the superiority of this framework, as it benefits from multi-modal inputs and consistently outperforms previous methods across various evaluation metrics, opening up new research opportunities for explanation generation for recommender systems.

5.1 Introduction

Personalized explanation generation models enhance recommendation transparency and reliability. Existing approaches focus on generating textual explanations from users' historical reviews [Zang and Wan, 2017], tips [Li et al., 2017b], or justifications [Ni et al., 2019]. However, these methods face challenges in delivering diverse explanations, often producing generic sentences (e.g., "food is very good!") due to a lack of grounding information, such as images, during the generation process. To address this limitation and enhance the diversity and richness of recommendations, we introduce a novel task known as *personalized showcases* (illustrated in Figure 5.1). This task involves explaining recommendations through both **textual** and **visual** information, providing a collection of images relevant to a user's interests, and generating corresponding textual explanations.

For this objective, the primary challenge is constructing an appropriate *dataset*. Conventional review datasets, such as those from Amazon [Ni et al., 2019] and Yelp[1], are predominantly unsuitable for the novel context at hand, as elaborated upon later in this dissertation. In response, we initiate the development of a large-scale multi-modal dataset, denoted as GEST, sourced from Google Local Restaurants. This dataset encompasses both review text and corresponding images, as illustrated in Figure 5.2.

In this novel task, we propose a multi-modal explanation generation framework, namely P-SHOWCASE. Initial steps involve the curation of relevant images sourced from historical photos of the user's preferred business. Subsequently, utilizing the selected images and user profiles, such as historical reviews, as inputs, our model is trained to generate textual explanations employing a multi-modal decoder. Despite advancements, generating text that is both expressive and captivating, while avoiding monotony, remains a significant challenge. This is attributed to the heightened demands on information extraction and multi-modal learning arising from the distinctive alignment requirements between multiple images and generated text.

[1] https://www.yelp.com/dataset

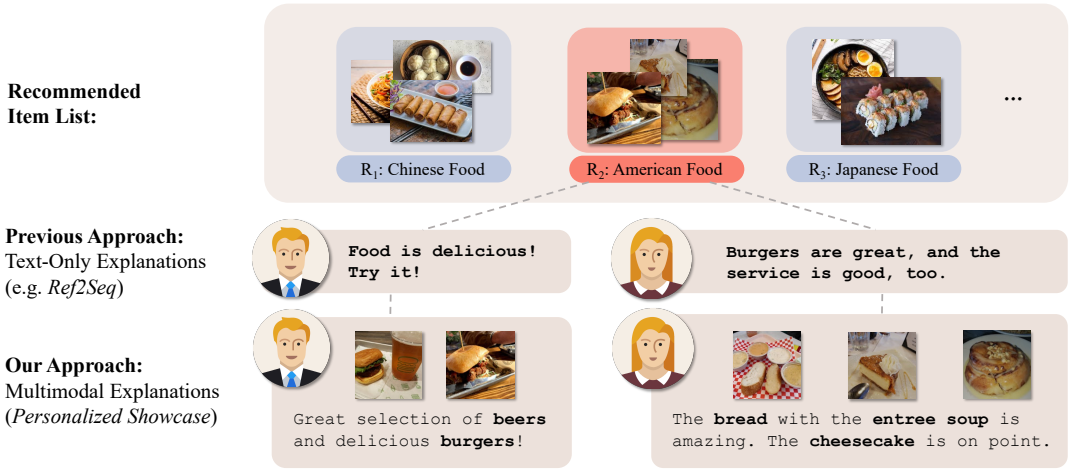


Figure 5.1. Illustration of previous text-only explanation methods and our personalized showcases. Given a recommended item or business: (1) Text-only explanation models only use historical textual reviews to generate textual explanations. (2) We propose a personalized showcases task to enrich the personalized explanations with multi-modal (visual and textual) information, which can largely improve the informativeness and diversity of generated explanations.

Methodologically, conventional encoder-decoder models employing cross-entropy loss often result in repetitive and uninteresting sentences, as observed frequently in the training corpus (e.g., "food is great") as noted by Holtzman et al. (2019). To address the identified challenges, we propose a **P**ersonalized **C**ross-Modal **C**ontrastive **L**earning (PC^{2}L) method, which leverages the contrastive learning paradigm to align input modalities with output sequences. While contrastive learning has gained prominence as a self-supervised representation learning approach [Oord et al., 2018, Chen et al., 2020a], we notice that training solely with randomly selected negative samples in a mini-batch may be suboptimal for certain tasks [Lee et al., 2020], as the embeddings could be easily discriminable in the latent space. To address this limitation, we introduce a cross-modal loss that enforces alignment between images and output explanations. This is achieved by constructing hard negative samples with randomly replaced entities in the output. Additionally, considering the shared interests among users with similar historical reviews, we incorporate a personalized loss that adjusts the weights of negative samples based on historical similarities.

Figure 5.2. Example of business and user reviews in our constructed dataset GEST.

Experimental results, from both automatic and human evaluations, demonstrate that our proposed model outperforms various baselines. Specifically, our model exhibits enhanced capabilities in generating more expressive, diverse, and visually-aligned explanations. Overall, our contributions are as follows:

- Introduction of a new task, *personalized showcases*, aimed at enhancing the informativeness of recommendations by providing multimodal (both textual- and visual-) explanations.
- Collection of a large-scale dataset from *Google Local* (maps) for the multimodal personalized showcases task, with a focus on extracting high-quality samples through preprocessing and filtering.
- Development of a multi-modal framework P-SHOWCASE for the *personalized showcases* tasks, incorporating contrastive learning to improve the diversity and visual alignment of generated textual explanations.

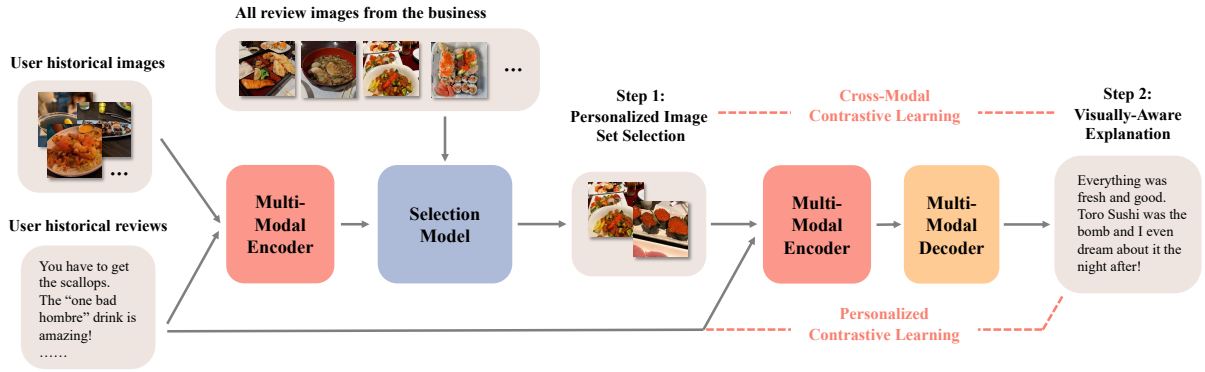


Figure 5.3. Illustration of our P-SHOWCASE framework. We take user historical images and reviews as inputs. We fisrt select an image set that is most relevant to a user's interest, then generate language explanations accordingly. A cross-modal contrastive loss and a personalized contrastive loss are applied between each input modality and the explanations. Last, the generated images and textual explanations will be organized as multi-modal explanations to users.

5.2 Proposed Approach: P-SHOWCASE

5.2.1 Task Formulation

In the *personalized showcases* task, we aim to provide personalized textual and visual explanations for user u, business b, and the image candidate set $I_b = \{i_1^b, i_2^b, \ldots i_{|I_b|}^b\}$ from b. We select a set of images I from I_b based on user u's profile, which includes historical reviews $X_u = \{x_1^u, x_2^u, ..., x_K^u\}$ and images $I_u = \{i_1^u, i_2^u, ..., i_n^u\}$. The selected images I_u serve as visual explanations, and textual explanations S_u are generated using the user's historical reviews X_u. These personalized explanations aim to elucidate why business b is recommended to user u.

To examine the interrelation between modalities and establish benchmarks for future investigations, our paper decomposes the task into two steps, as illustrated in Figure 5.3: (1) Selecting an image set as a visual explanation related to a user's interest; (2) Generating textual explanations based on the selected images and a user's historical reviews. We assess the task with the following criteria:

1. **Accuracy:** Predicting the target images with precision and ensuring that the generated text is relevant to the business.

2. **Diversity:** Ensuring diversity and expressiveness in both visual and textual explanations.

3. **Alignment:** In contrast to prior review generation tasks that involve only the text modality, our visually-aware setting necessitates grounding to the images. Thus, the model should accurately depict the content and encompass key objects (e.g., the names of dishes, and the environment) in the provided images.

5.2.2 Personalized Image Set Selection

To tackle this task, we propose a framework, namely P-SHOWCASE. The initial phase of our framework is choosing a diverse image set that serves as a visually explanatory representation aligned with the user's interests. This selection process is formulated as a diverse recommendation incorporating multi-modal inputs.

Multi-Modal Encoder. CLIP [Radford et al., 2021], a state-of-the-art pre-trained cross-modal retrieval model as both textual- and visual-encoders, encodes raw images as image features, and encodes user textual and visual profiles as user profile features.

Image Selection Model. Determinantal Point Process (DPP) [Kulesza and Taskar, 2012] a technique recently applied in diverse recommendation tasks [Wilhelm et al., 2018, Bai et al., 2019], is used to select image subsets. DPP-based models, unlike other algorithms geared towards *individual* item recommendation, demonstrate suitability for *multiple* image selection scenarios. Given a user u and a business b, our prediction for the image set $\hat{I}_{u,b}$ is expressed as follows:

$$\hat{I}_{u,b} = \text{DPP}(I_b, u), \tag{5.1}$$

where I_b represents the image set associated with business b. In our approach, we determine user-image relevance by utilizing CLIP-based features from the user's profile and image features. Further details about the model can be found in [Wilhelm et al., 2018].

5.2.3 Visually-Aware Explanation Generation

Upon having a set of images, the next phase of our P-SHOWCASE framework is to generate personalized explanations based on both the image set and the user's historical reviews. To achieve this, we use a multi-modal encoder-decoder model utilizing GPT-2 [Radford et al., 2019] as the backbone.

Multi-Modal Encoder. Given a user u's historical reviews $X = \{x_1, x_2, \ldots, x_K\}$, CLIP's text encoder is used to extract review features $R = \{r_1, r_2, \ldots, r_K\}$. Similarly, the CLIP visual encoder is used to extract visual features $V = \{v_1, v_2, \ldots, v_n\}$ from input images $I = \{i_1, i_2, \ldots, i_n\}$. These features are then projected into a latent space using learnable matrices:

$$Z_i^V = W^V v_i, \quad Z_i^R = W_i^R r_i, \tag{5.2}$$

where W^V and W^R are the projection matrices. Subsequently, a multi-modal attention (MMA) module, comprising stacked self-attention layers [Vaswani et al., 2017a], is used to encode the input features:

$$[H^V; H^R] = \text{MMA}([Z^V; Z^R]), \tag{5.3}$$

where each H_i^V and H_i^R aggregate features from the two modalities, and $[;]$ denotes concatenation. This design accommodates varying lengths of each modality and facilitates intermodal interactions through co-attentions.

Multi-Modal Decoder. Inspired by recent advances in pre-trained language models, we utilize GPT-2 as the decoder for generating explanations. To efficiently integrate linguistic knowledge from GPT-2, we incorporate the encoder-decoder attention module into the pre-trained model, following a similar architecture as in [Chen et al., 2021].

With the aim of enhancing generation performance, we further fine-tune GPT-2 using domain-specific data. The resulting multi-modal GPT-2 is trained on 260k samples with causal language modeling, excluding data from users who only wrote one review and are thus not part

of our personalization dataset. The empirical findings indicate that training GPT-2 with domain-specific knowledge significantly improves generation performance. For a target explanation $Y = \{y_1, y_2, ..., y_L\}$, the decoding process at each time step t is formalized as:

$$\hat{y}_t = \text{Decoder}([H^V; H^R], y_1, \ldots, y_{t-1}). \tag{5.4}$$

We maximize the conditional log-likelihood $\log p_\theta(Y|X,I)$ for N training samples $(X^{(i)}, I^{(i)}, Y^{(i)})_{i=1}^N$ by using cross entropy (CE) loss:

$$\mathscr{L}_{CE} = -\sum_{i=1}^{N} \log p_\theta(Y^{(i)}|X^{(i)}, I^{(i)}). \tag{5.5}$$

During training, we use ground-truth images from the user, while images from our image-selection model are used for inference.

5.2.4 Personalized Cross-Modal Contrastive Learning

Our task, distinct from image captioning, involves using multiple images as "visual prompts" for conveying personal feelings and opinions. To help the generation of expressive, diverse, and visually-aligned explanations, we propose PC^2L technique in our framework. This technique initially maps the hidden representations of images, historical reviews, and the target sequence into a latent space:

$$\tilde{H}^V = \phi_V(H^V),\ \tilde{H}^R = \phi_R(H^R),\ \tilde{H}^Y = \phi_Y(H^Y) \tag{5.6}$$

where ϕ_V, ϕ_R, and ϕ_Y consist of two fully connected layers with ReLU activation and average pooling over the hidden states H_V, H_R and H_Y from the last self-attention layers. With the InfoNCE loss [Oord et al., 2018, Chen et al., 2020a], the objective is to maximize the similarity between the source modality and target sequence pair, while simultaneously minimizing the

similarity between negative pairs. This is expressed as follows:

$$\mathscr{L}_{CL} = -\sum_{i=1}^{N} \log \frac{\exp(s_{i,i}^{X,Y})}{\exp(s_{i,i}^{X,Y}) + \sum_{j \in K} \exp(s_{i,j}^{X,Y})}. \tag{5.7}$$

where $s_{i,j}^{X,Y} = sim(\tilde{H}_{(i)}^{X}, \tilde{H}_{(j)}^{Y})/\tau$, *sim* is the cosine similarity between two vectors, τ is the temperature parameter, (i) and (j) are two samples in the mini-batch, K is the set of negative samples for sample (i).

One challenge in this task is describing multiple objects within a set of images. To address the need for visual grounding between various image features and output text, we propose a novel cross-modal contrastive loss. Specifically, for a target explanation $Y = \{y_1, y_2, ..., y_L\}$, we randomly replace entities [2] in the text with other entities from the dataset, creating a hard negative sample $Y^{ent} = \{y'_{ent1}, y_2, ...y'_{ent2}, ...y_L\}$ (e.g., from "I like the sushi" to "I like the burger"). During training, this exposes the model to samples with incorrect entities related to the images, making it non-trivial to distinguish from the original target sequence. Consequently, we introduce the hidden representation of Y^{ent} as an additional negative sample *ent* to formulate the cross-modal contrastive loss:

$$\mathscr{L}_{CCL} = -\sum_{i=1}^{N} \log \frac{\exp(s_{i,i}^{V,Y})}{\exp(s_{i,i}^{V,Y}) + \sum_{j \in K \cup ent} \exp(s_{i,j}^{V,Y})}, \tag{5.8}$$

On the other hand, for improved personalization of explanations, we adjust the weights of negative pairs based on user personalities. The rationale is that individuals with more distinct personalities are prone to generating diverse explanations. Accordingly, we introduce a weighted personalized contrastive loss to address this motivation:

$$\mathscr{L}_{PCL} = -\sum_{i=1}^{N} \log \frac{\exp(s_{i,i}^{R,Y})}{\exp(s_{i,i}^{R,Y}) + f(i,j) \sum_{j \in K} \exp(s_{i,j}^{R,Y})}. \tag{5.9}$$

where negative pairs in a mini-batch are re-weighted based on user personality similarity function

[2] Entities are extracted using spaCy noun chunks (https://spacy.io/).

f. In our framework, user personalities are represented by their historical reviews. Specifically, we define f function as:

$$f(i,j) = \alpha^{(1-sim(\tilde{R}_{(i)},\tilde{R}_{(j)}))}, \tag{5.10}$$

i.e., we reduce the weights of negative pairs with similar histories, and increase those with distinct histories. α ($\alpha > 1$) is a hyperparameter that weighs the negative samples, *sim* is the cosine similarity, $\tilde{R}_{(i)}$ and $\tilde{R}_{(j)}$ are the average features of two users' input historical reviews. Overall, the model is optimized with a mixture of a cross-entropy loss and those two contrastive losses:

$$\mathscr{L}_{loss} = \mathscr{L}_{CE} + \lambda_1 \mathscr{L}_{CCL} + \lambda_2 \mathscr{L}_{PCL}, \tag{5.11}$$

where λ_1 and λ_2 are hyperparameters that weigh the two losses.

5.3 Experiments

5.3.1 Experimental Setting

Baselines. We compare our model with popular baselines from different tasks, including image captioning, report generation and explanation generation: (1) **ST** [Xu et al., 2015] is a classic CNN+LSTM model for image captioning. (2) **R2Gen** [Chen et al., 2020b] is a state-of-the-art memory-driven transformer specialized at generating long text with visual inputs. (3) **Ref2Seq** [Ni et al., 2019] is a popular reference-based seq2seq model for explanation generation in recommendation. (4) **Peter** [Li et al., 2021b] is a recent transformer-based explanation generation model which uses the user and item IDs to predict the words in the target explanation. (5) *img* and *text* refer to image and text features respectively.

Evaluation Metrics. For image selection, we utilize Precision@K, Recall@K, and F1@K to measure ranking quality. Given the nature of our task, we set a small $K = 3$. To assess diversity, we introduce the truncated `div`@K, where $K = 3$. Formally, for a set of K images

Table 5.1. Results on personalized showcases with different models and different input modalities. Results are in percentage (%). *GT* is the ground truth.

Model	Input	N-Gram Metrics			Diversity Metrics		Embedding Metrics	
		BLEU-1	METEOR	NIST	DIST-1	DIST-2	CLIP-S	BERT-S
GT	-	-	-	-	6.06	43.23	28.41	-
ST	*img*	8.24	3.41	28.08	2.74	17.41	24.01	85.20
R2Gen	*img*	6.47	3.10	36.55	3.23	22.45	24.28	85.89
Ref2Seq	*text*	7.09	3.80	30.78	0.92	5.89	22.83	84.71
Peter	*text*	8.89	3.28	34.45	0.38	1.27	23.27	86.94
P-SHOWCASE	*img*	9.92	3.64	37.35	3.37	26.37	**24.68**	88.03
	img+text	**10.40**	**3.83**	**50.64**	**3.58**	**28.58**	24.50	**88.23**

$\{i_1, \ldots, i_K\}$, `div@K` is defined as:

$$div@K = \sum_{1 \leq m < n \leq K} \frac{dis(i_m, i_n)}{K(K-1)/2}. \tag{5.12}$$

Textual explanations are evaluated for relevance using n-gram-based metrics such as BLEU (n=1,4) [Papineni et al., 2002], METEOR [Banerjee and Lavie, 2005], as well as NIST (n=4) [Doddington, 2002]. To assess diversity, we report DISTINCT-1 and DISTINCT-2 proposed in [Li et al., 2015]. Additionally, we use embedding-based metrics using CLIP and BERT. CLIP-SCORE [Hessel et al., 2021] and BERT-SCORE [Zhang et al., 2020a] are recent embedding-based metrics.

Dataset & Implementation Details. We gathered 1.77 million Google Local reviews in the United States and selected 108,000 high-quality reviews about restaurants. We call this dataset GEST, standing for Google Reviews on Restaurants. The data is divided into a 0.8/0.1/0.1 ratio for training, validation, and testing, respectively. Our approach utilizes CLIP with ViT-B/32 for encoding reviews and images, and GPT-2 small as the decoder backbone. Further information on dataset statistics, preprocessing, comparison with other datasets, and implementation details can be found in the supplementary material.

Table 5.2. Ablation study on personalized image selections. Results are reported in percentage (%).

	Accuracy			Diversity
Method	Prec@3	Recall@3	F1@3	Div@3
random	4.87	6.14	5.43	30.24
img	**25.21**	34.05	28.97	17.12
text	15.28	20.58	17.54	**18.68**
img+text	**25.21**	**34.37**	**29.09**	17.07

5.3.2 Result Analysis

Framework Performance

We present the model performance on text evaluation metrics in Table 5.1. The incorporation of image features addresses challenges in generating human-like explanations and avoiding monotonous text. The comparison between text-input and image-input models highlights the clear superiority of visually-aware generation models in terms of diversity and embedding metrics. Meanwhile, our proposed PC^2L exhibits significant improvement across various metrics compared to other transformer-based models. This underscores the effectiveness of a pretrained language model with contrastive learning in generating high-quality explanations. Despite competitive results on n-gram metrics such as BLEU and METEOR, text-based models Ref2Seq and Peter lag behind in diversity and embedding metrics. Additionally, these models demonstrate lower text quality with frequent occurrences of repetitive and non-informative sentences, as confirmed by human evaluations and case studies.

Component Analysis

We perform ablation studies to evaluate the effectiveness of individual components.

Models for image set selection. To assess personalized image set selection performance, we compare our model against random selection and various input modalities. As presented in Table 5.2, despite the text-only model having the highest truncated diversity, its ranking

Figure 5.4. (a) The length distributions of generated texts on the test set. (b) The generation coverage of nouns (Noun), adjectives (ADJ) and adverbs (ADV).

performance is markedly inferior to models incorporating images, indicating the inadequacy of text input alone for user personalization. Historical images, conversely, serve as a crucial visual cue for capturing user preferences. In summary, a multi-modal model combining images and text attains the superior ranking performance for image set selection, showing the significance of our proposed multi-modal approach in achieving personalized showcases.

Effectiveness of Contrastive Learning. We conduct ablation studies on various configurations of our contrastive loss. As presented in Table 5.3, our PC^{2}L outperforms all baseline models across different metrics. CCL enhances visual grounding by compelling the model to distinguish between random and correct entities, resulting in an improvement in CLIP-SCORE compared to the vanilla contrastive framework [Chen et al., 2020a]. On the other hand, PCL enhances diversity by directing the model's focus towards users with dissimilar interests.

To understand the impact of contrastive learning on generation quality, we analyze outputs based on length distributions and keyword coverage. Figure 5.4 (a) compares length distributions on the test set, categorized into 6 groups within the range [0, 60] with a 10-unit interval. The model without PC^{2}L exhibits a sharper distribution, while incorporating PC^{2}L results in a distribution closer to the ground truth, showcasing its efficacy and ability to generalize to unseen images.

Table 5.3. Ablation study on contrastive learning. CL, CCL and PCL are the contrastive losses in Equations (5.7) to (5.9).

	BLEU-1	DISTINCT-2	CLIP-S
Base model w/o CL	7.96	25.90	23.71
img CL + text CL	9.72	27.58	23.93
CCL+ text CL	10.19	28.10	24.36
img CL + PCL	9.96	28.32	24.15
PC^{2}L	**10.40**	**28.58**	**24.50**

Table 5.4. Human evaluation on two models. We present the workers with reference text and images, and ask them to give scores from different aspects. Results are statistically significant via sign test ($p < 0.01$).

Method	Expressiveness	Alignment
Ref2Seq	3.72	3.65
PC^{2}L	**4.25**	**4.10**

Figure 5.4 (b) illustrates keyword coverage in output sentences. A keyword is considered covered if it exists in the corresponding ground truth. Models trained with and without PC^{2}L are compared, revealing that PC^{2}L improves coverage for all types of keywords, indicating that our contrastive learning method enhances the diversity and personalization of generated text. In conclusion, integrating contrastive learning into multi-modal explanation generation yields improved output quality with more diverse and visually-aligned texts.

Human Evaluation

To fully evaluate our model, we conduct human evaluation on Amazon Mechanical Turk.[3] For each model, we randomly sample 500 examples from test set. Each example is scored by three human judges using a 5-point Likert scale to reduce variance. We instruct the annotators to consider expressiveness (semantically correct, diversity, no repetition) and visual alignment (the text describes the context of the images). As is shown in Table 5.4, PC^{2}L significantly outperforms Ref2Seq, which is consistent with the automatic evaluation metrics.

[3] https://www.mturk.com/

5.4 Conclusion

In this chapter, to generate explanations with rich information for recommendations, we explore the multimodal generation tasks for recommender systems. To be specific, we propose a new task, namely *personalized showcases*, and collect a large-scale dataset GEST from *Google Local* for the task. We design a multi-modal explanation framework, *P-Showcase*, with contrastive learning to learn visual and textual explanations from user reviews. Experimental results show that *showcases* provide more informative and diverse explanations compared to previous text-only explanations.

As future work, one promising direction is to develop an end-to-end framework for generating both visual and textual explanations. Besides, we find that visual grounding on multiple images is still challenging for showcases. Hence, effectively leveraging multi-modal information and improving visual alignment are also important works to explore for the new task. At the same time, applying our P-SHOWCASE in real-world conversational recommendations as a multimodal explanation generator is an interesting direction as well. We hope our dataset and framework would benefit the community for future research on multi-modalities and recommendations.

Chapter 5, in part, is a reprint of the material as it appears in "Personalized Showcases: Generating multi-modal explanations for recommendations." by An Yan*, Zhankui He*, Jiacheng Li*, Tianyang Zhang, and Julian McAuley in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval in 2023*, referenced as [Yan et al., 2023a]. The dissertation author was the primary investigator and author of this paper.

Part III

Interactivity: Exploring Complex Recommendation Scenarios

Chapter 6

Item Bundling in Conversational Recommender Systems

In this chapter, we start exploring complex recommendation tasks in the context of conversational recommendations. Many recommendation needs cannot be satisfied by using traditional recommender systems. However, enhancing the interactivity between users and the systems by introducing a conversational recommendation mechanism may help.

Our exploration in this chapter demonstrates how to handle challenging bundle recommendation tasks by introducing conversational mechanisms. Traditional *bundle* recommender systems, which suggest sets of complementary items (e.g., clothing outfits), frequently face challenges due to significant *interaction sparsity* and a *large output space*. We propose extending *multi-round conversational recommendation* (MCR) to address these issues. MCR utilizes a conversational approach to elicit user preferences through questions about tags (e.g., categories or attributes) and incorporates user feedback across multiple rounds. This approach has the potential to acquire valuable user input and refine the recommendation space, but its application within bundle recommendation remains unexplored.

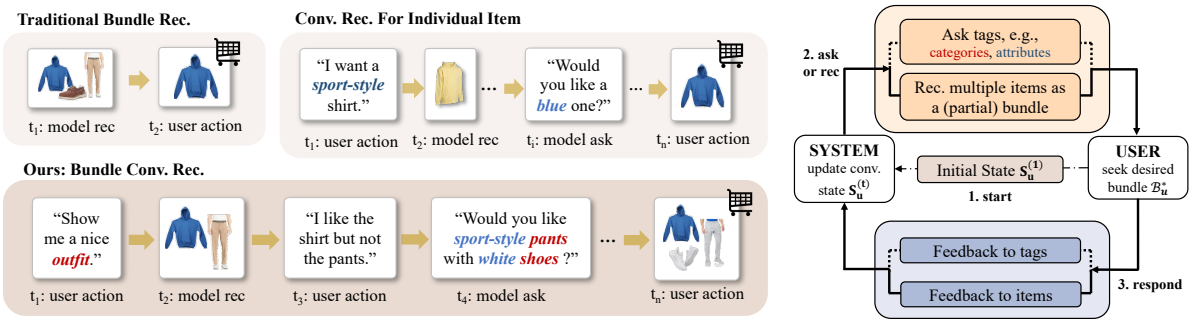
We formulate the bundle recommendation with MCR conversational mechanisms as BUNDLEMCR. Unlike traditional bundle recommendation, which typically involves bundle-aware user modeling and generation, BUNDLEMCR focuses on encoding user feedback as conversation states and strategically formulating questions. Additionally, while existing MCR

systems recommend individual items, BUNDLEMCR must accommodate more complex user feedback involving multiple items and related tags. Furthermore, we introduce a model architecture called Bundle Bert (BUNT) with the following capabilities: (1) item recommendation, (2) question generation, and (3) conversation management based on bundle-aware conversation states. We also devise a two-stage training strategy for BUNT training. Experiments on multiple offline datasets, along with human evaluation, demonstrate the value of extending MCR frameworks to bundle settings and the overall effectiveness of our BUNT design.

6.1 Introduction

Bundle recommendation is suggesting sets of items for simultaneous consumption by users [Pathak et al., 2017, Chen et al., 2019a, Deng et al., 2021a] (e.g., outfits, playlists), enhancing user satisfaction [Deng et al., 2020, Chen et al., 2019c]. However, it faces inherent challenges: (1) **Interaction sparsity**, as user-bundle interactions are sparser than user-item interactions, making accurate modeling of user preferences difficult; (2) **Output space complexity**, as predicting correct bundles from all item combinations is more challenging than traditional individual item recommendations.

Currently, two approaches are proposed in bundle recommendation to address these challenges. The first approach [Pathak et al., 2017, Chang et al., 2020, Chen et al., 2019a] introduces *discriminative* methods, wherein bundles are treated as generalized individual items, avoiding complexity by ranking *existing* bundles. However, these methods often have narrow application scenarios, such as pre-defined bundle sales. The second approach [Bai et al., 2019, Deng et al., 2021a, Hu and He, 2019] employs *generative* methods, capable of generating (potentially new) bundles, offering greater flexibility but suffering from limited accuracy. In these works, bundle recommenders are *one-shot*, recommending a complete bundle with a single attempt. As depicted in the traditional bundle recommendation in Figure 6.1a, the user receives a complete bundle (e.g., shirt, shoes, and pants), reacts to it (e.g., picking a shirt but ignoring



(a) Example of Conversational Bundle Recommendation *vs* Others

(b) Diagram of BUNDLEMCR

Figure 6.1. *Left*: Use case comparisons among traditional bundle recommendation, individual conversational recommendation, and our conversational bundle recommendation. *Right*: Diagram of our proposed Bundle-Aware Multi-Round Conversational Recommendation (i.e., BUNDLEMCR) scenario, which extends traditional individual MCR [Lei et al., 2020a,b, Deng et al., 2021b, Zhang et al., 2022] to bundle settings.

others), and the recommendation process concludes. However, this one-shot setting restricts the model from collecting continuous user feedback to enhance bundle accuracy. Recognizing these limitations, we introduce a new approach, termed *multi-round* and *interactive* bundle recommendation, referred to as BUNDLEMCR. As shown in Table 6.1, this approach allows the user and system to engage in a "conversation" regarding bundle composition over multiple rounds.

The BUNDLEMCR concept extends multi-round conversational recommendation (MCR) mechanisms, specifically designed for individual item recommendation, to address the challenge of bundle recommendation. While existing MCR frameworks, such as Individual MCR, have demonstrated effectiveness in eliciting user preferences for individual items, they are not directly applicable to bundle settings due to factors like neglecting user-bundle interactions, recommending top-K individual items instead of bundles, and handling feedback and questions at the item level without considering the bundle context. This disparity is depicted in Figure 6.1a. Individual MCR focuses on updating user feedback on tags for a single item, while BUNDLEMCR aims to generate multiple items as a bundle or partial bundle, considering user feedback and questions related to different items within the bundle. The proposed BUNDLEMCR framework enhances

conversational recommendation by accommodating bundle contexts, facilitating more accurate bundle recommendations through user feedback on item tags during conversations.

We approach the BUNDLEMCR problem by formulating it as a Markov Decision Process (MDP) with multiple agents. We introduce a novel model architecture, Bundle Bert (BUNT), designed to handle these tasks within a unified self-attentive framework [Vaswani et al., 2017a, Kang and McAuley, 2018, Sun et al., 2019]. To effectively train BUNT, a two-stage training strategy is used. Initially, we pre-train BUNT using multiple *cloze* tasks, enabling the model to acquire foundational knowledge on inferring correct items, tags, and determining when to ask or recommend based on conversation contexts simulated by offline user-bundle interactions. Subsequently, we use a user simulator to create a simulated online environment and fine-tune BUNT agents using reinforcement learning on conversational bundle interactions with users. In summary, the key contributions of this work are as follows:

- Introducing a novel BUNDLEMCR setting, where users and the system collaboratively complete a bundle. This work addresses conversational mechanisms in bundle recommendation, mitigating challenges related to *information sparsity* and *output space complexity*.
- Presenting an MDP framework with multiple agents for BUNDLEMCR. Within this framework, we propose Bundle Bert (BUNT) as a unified self-attentive architecture for conducting multiple BUNDLEMCR functions. A two-stage strategy (pre-training and fine-tuning) is devised for BUNT learning.
- Evaluating conversational bundle recommendations on four offline bundle datasets and conducting a human evaluation to demonstrate the effectiveness of BUNT and the potential of conversational bundle recommendation.

Table 6.1. Functionality requirements in BUNDLEMCR model design and comparisons with individual MCR and bundle recommendation.

Functionalities	Individual MCR	Bundle Rec.	BUNDLEMCR
Bundle-Aware User Modeling	✘	[Deng et al., 2021a, Bai et al., 2019, Pathak et al., 2017, Chang et al., 2020, Chen et al., 2019a,c]	✔
Bundle Generation	✘	[Deng et al., 2021a, Bai et al., 2019, Pathak et al., 2017, Chen et al., 2019c]	✔
Bundle-Aware Feedback Handing	✘	✘	✔
Bundle-Aware Question Asking	✘	✘	✔
Conversation Management	[Lei et al., 2020a,b, Zhang et al., 2020b, Deng et al., 2021b]	✘	✔

6.2 Proposed Scenario: BUNDLEMCR

6.2.1 BUNDLEMCR Scenario Formulation

We extend multi-round conversational recommendation (MCR) [Lei et al., 2020a,b, Deng et al., 2021b] to a bundle setting, termed BUNDLEMCR. In contrast to individual MCR, we introduce the concept of a *slot* in BUNDLEMCR[1], which serves as a placeholder for a consulted item. For instance, an outfit (1: shoes, 2: pants, 3: shirt) comprises three *slots* $\mathscr{X} = \{1,2,3\}$. The objective of bundle MCR is to (1) determine the number of slots and (2) fill target items in these slots during conversations.

The formulation of BUNDLEMCR involves the set of users $\mathscr{U}$ and items $\mathscr{I}$, where we collect tags corresponding to items, such as the set of attributes $\mathscr{P}$ (e.g., "dark color") and categories $\mathscr{Q}$ (e.g., "shoes"). As depicted in Figure 6.1b, for a user $u \in \mathscr{U}$:

1. Conversation starts from a state $\mathbf{S}_u^{(1)}$, which encodes user historical bundle interactions $\{\mathscr{B}_1, \mathscr{B}_2, \dots\}$, where $\mathscr{B}_*$ represents a bundle of multiple items. Let us set conversational

[1]This *slot* concept differs from that in dialog systems.

round $t = 1$, the system creates multiple slots as $\mathscr{X}^{(t)}$.

2. Then, the system decides to recommend or ask, i.e., (i) recommending $|\mathscr{X}^{(t)}|$ items as a (partial) bundle to fill these proposed slots, denoting as $\mathscr{B}_u^{(t)} = \{\hat{i}_x \mid x \in \mathscr{X}^{(t)}\}$; or (ii) asking for user preference per slot on attributes $\mathscr{A}_u^{(t)} = \{\hat{a}_x \mid x \in \mathscr{X}^{(t)}\}$ and categories $\mathscr{C}_u^{(t)} = \{\hat{c}_x \mid x \in \mathscr{X}^{(t)}\}$. Here in each slot x, $\hat{i}_x \in \mathscr{I}$, $\hat{a}_x \in \mathscr{P}$ and $\hat{c}_x \in \mathscr{Q}$.

3. Next, user u is required to provide feedback (i.e., accept, ignore, reject) to the proposed partial bundle $\mathscr{B}_u^{(t)}$ or attributes $\mathscr{A}_u^{(t)}$ and categories $\mathscr{C}_u^{(t)}$ per slot $x \in \mathscr{X}^{(t)}$.

4. After that, the system updates user feedback into new state $\mathbf{S}_u^{(t+1)}$, records all the accepted items into a set, denoting as $\check{\mathscr{B}}_u$[2], and updates the slots of interest as $\mathscr{X}^{(t+1)}$ by creating new slots and removing the slots x in which user has accepted the recommended item $\hat{i}_x$.

After multiple rounds of step (2)-(4), the system collects rich contextual information and create bundle $\check{\mathscr{B}}_u$ for user. The conversation terminates when u is satisfied with the current bundle (i.e., $\check{\mathscr{B}}_u$ equals the target bundle $\mathscr{B}_u^*$) or this conversation reaches the maximum number of rounds T.

In BUNDLEMCR, we address several key questions: (1) how to encode user feedback into the bundle-aware state $\mathbf{S}_u^{(t+1)}$; (2) how to accurately predict bundle-aware items or tags; (3) how to effectively train models in BUNDLEMCR; and (4) how to determine the size of slots $\mathscr{X}^{(t)}$ per round. In this study, we concentrate on addressing questions (1)-(3), while adopting a straightforward strategy for (4), specifically, maintaining a fixed slot size K. Despite the fixed slot size per round, the final bundle sizes exhibit diversity due to varying user feedback and conversation rounds. More flexible slot strategies are deferred to future investigations.

It is important to note that we use attribute set $\mathscr{P}$ and category set $\mathscr{Q}$, along with associated models, in all baseline and proposed methods. However, for clarity in the subsequent methodology sections, we exclusively refer to the attribute set $\mathscr{P}$ as an example of tags.

[2]We use the `\check` mark for the meaning of "being accepted by user"; similarly, we use `\hat` for the meaning of "being proposed to user".

6.2.2 General MDP Framework for BUNDLEMCR

We model BUNDLEMCR as a two-step Markov Decision Process (MDPs) involving multiple agents. This is due to the system's two-step decision-making process: first, deciding whether to recommend or ask (conversation management), and second, determining what to recommend or ask. Additionally, multiple agents are assigned distinct responsibilities: an agent (using π_M) handles conversation management, a bundle agent (using π_I) decides on recommending items to form a bundle, and an attribute agent (using π_A) determines which attributes to inquire about. The overarching objective of our framework is to maximize the expected cumulative rewards by learning distinct policy networks $\pi_M^*, \pi_I^*, \pi_A^*$. We segment a conversation round into user modeling, consultation, and feedback handling stages, following a structure similar to [Zhang et al., 2022]. Subsequently, we elaborate on the design of *state*, *policy*, *action*, and *transition* within this framework across the relevant stages.

States: Bundle-Aware User Modeling

We first introduce the shared conversation state $\mathbf{S}_u^{(t)}$ for all agents. $\mathbf{S}_u^{(t)}$ is encoded (specific encoder is introduced in Section 6.3) from the conversational information $\mathbb{S}_u^{(t)}$ at conversational round t, which is defined as:

$$\mathbb{S}_u^{(t)} = \big(\underbrace{\{\mathscr{B}_1, \mathscr{B}_2, \dots\},}_{\text{long-term preference}} \underbrace{\{(\check{i}_x^{(t)}, \check{\mathscr{A}}_x^{(t)}) \mid x \in \mathscr{X}^{(\leq t)}\},}_{\text{short-term contexts}} \underbrace{\{(\mathscr{I}_x^{(t)}, \mathscr{P}_x^{(t)}) \mid x \in \mathscr{X}^{(\leq t)}\}}_{\text{candidate pools}} \big). \tag{6.1}$$

- **Long-term preference** is represented by the set of user u's historical bundle interactions $\{\mathscr{B}_1, \mathscr{B}_2, \dots\}$.
- **Short-term contexts** collect accepted items and attributes in conversations before conversational round t. $\mathscr{X}^{(\leq t)}$ is the set of slots till rounds t, i.e., $\mathscr{X}^{(\leq t)} = \bigcup_{t'=1}^{t} \mathscr{X}^{(t')}$. In slot x at round t, we record the tuple $(\check{i}_x^{(t)}, \check{\mathscr{A}}_x^{(t)})$, where $\check{i}_x^{(t)}$ denotes the item id accepted by the user. If no item accepted in slot x, $\check{i}_x^{(t)}$ is set as a mask token `[MASK]`; $\check{\mathscr{A}}_x^{(t)}$ is the set of

accepted attributes in slot x. For example, the initial short-term context is a K-sized set of ([MASK], $\emptyset$) tuples, meaning we know nothing about accepted items or attributes.[3]

- **Candidate pools** contain item and attribute candidates per slot at round t (it is not space costly by black lists). They are initialed as completed pools $\mathscr{I}$ and $\mathscr{P}$, and updated with user u's feedback, described in Section 6.2.2.

Second, we introduce an additional conversation information denoted as $\bar{\mathbb{S}}_u^{(t)}$, encoded as an additional state $\bar{\mathbf{S}}_u^{(t)}$ (refer to Section 6.3.1) for the conversation management agent. $\bar{\mathbb{S}}_u^{(t)}$ maintains a record of the result IDs from the previous t-1 rounds in the form of a list, e.g., [rec_fail, ask_fail, ...]. This state representation is commonly used for conversation management agents in existing works such as [Lei et al., 2020a,b, Deng et al., 2021b]. The result ID settings align with [Lei et al., 2020a], with the addition of a "bundle_suc" ID to capture the outcome of successfully recommending the entire bundle, alongside the existing "rec_suc" ID for successfully recommending a single item.

Policies and Actions: Bundle-Aware Consultation

The system transitions to the consultation stage following the acquisition of conversation states during the user modeling stage. At this point, a two-step decision-making process occurs: (1) determining whether to recommend or inquire, guided by policy π_M; (2) selecting what to recommend through policy π_I or deciding what to ask through policy π_A. We define these policies as:

- π_M **– conversation management:** use $\bar{\mathbf{S}}_u^{(t)}$ and $\mathbf{S}_u^{(t)}$ to predict a binary action (recommending or asking).

- π_I **– (partial) bundle generation:** if recommending, the agent uses $\mathbf{S}_u^{(t)}$ as input to generate $|\mathscr{X}^{(t)}|$ (i.e., K) items as $\mathscr{B}_u^{(t)} = \{\hat{i}_x \mid x \in \mathscr{X}^{(t)}\}$, where $\hat{i}_x$ is the action corresponding to slot x and the actions space is $\mathscr{I}_x^{(t)}$.

[3]We can record rejected items or attributes as well, but we omit them since they are currently not effective empirically in our experiments.

- π_A **– attributes consultation:** if asking, the agent uses $\mathbf{S}_u^{(t)}$ as input to generate $|\mathscr{X}^{(t)}|$ (i.e., K) attributes as $\mathscr{A}_u^{(t)} = \{\hat{a}_x \mid x \in \mathscr{X}^{(t)}\}$, where $\hat{a}_x$ is the action corresponding to slot x and the actions space is $\mathscr{P}_x^{(t)}$.

Transitions: Bundle-Aware Feedback Handling

The system handles user feedback in a transition step. The user u will react to the proposed K items or attributes with acceptance, rejection or ignoring. Generally, in our transition step, "acceptance" is mainly used to update short-term contexts, "rejection" is used to update candidate pools and we change nothing when getting "ignoring".

- **Update** $\mathbb{S}_u^{(t+1)}$**:** long-term preference is fixed, we update the short-term contexts and candidate pools as follows: **(1) Feedback to items:** for each consulted item $\hat{i}_x$, (i) all item candidates pools $\mathscr{I}_{x'}^{(t+1)}$ where $x' \in \mathscr{X}^{(t)}$ delete $\hat{i}_x$ because it has been recommended; (ii) if $\hat{i}_x$ is accepted, short-term contexts in slot x will assign $\hat{i}_x$ to $\check{i}_x^{(t)}$. **(2) Feedback to attributes:** for each consulted attribute $\hat{a}_x$, (i) different from consulted items, only attribute pool $\mathscr{P}_x^{(t+1)}$ removes $\hat{a}_x$ because the user has a different preference on attributes in different slots (e.g., *white* shirt but *black* pants); (ii) if $\hat{a}_x$ is accepted, $\check{\mathscr{A}}_x^{(t+1)}$ in short-term context is updated by $\check{\mathscr{A}}_x^{(t)} \cup \{\hat{a}_x\}$; (iii) if $\hat{a}_x$ is explicitly rejected, it only happens when user strongly dislikes this attribute. So $\hat{a}_x$ will be removed from all attributes candidate pools, and items associated with $\hat{a}_x$ will be removed from all item candidate pools as well.

- **Update** $\bar{\mathbb{S}}_u^{(t+1)}$**:** it is updated by appending a new result id for round t, resulting in a t-sized list.

- **Update slots** $\mathscr{X}^{(t+1)}$**:** as Section 6.2 described, if items accepted, we remove the corresponded slots from $\mathscr{X}^{(t)}$, and create new slots to keep the size as K. For a new slot x', the short-term contexts are $(\texttt{[MASK]}, \emptyset)$, and candidate pools are the union sets of previous candidate pools to excluded items or attributes that the user strongly dislikes.

Rewards: Two-Level Reward Definitions

We define two-level rewards for multiple agents. **(1) Low-level rewards** pertain to π_I and π_A, focusing on enhancing the accuracy of item recommendations and question posting online. At round t, the reward $r_x^I = 1$ is assigned if π_I successfully recommends the target item, otherwise 0. Similarly, reward r_x^A for π_A follows the same pattern. **(2) High-level rewards** are designated for the conversation management agent π_M to reflect the overall conversation quality. The reward r^M remains 0 unless the conversation concludes, at which point r^M is computed using a final bundle metric such as F1 score or accuracy.

6.3 Proposed Model: BUNT

In this framework, we introduce a unified model called Bundle BERT (BUNT). The architecture of BUNT is outlined in this section, followed by a discussion on the training methodology involving offline pre-training and online fine-tuning.

BUNT is an encoder-decoder framework designed for user modeling, consultation, and feedback handling, supporting multiple input and output types. Utilizing a self-attentive architecture, we justify this choice for three reasons: (1) Self-attentive models have demonstrated effectiveness in encoding representations and generating accurate recommendations [Kang and McAuley, 2018, Sun et al., 2019, Chen et al., 2019c, He et al., 2021, Li et al., 2021b]; (2) Unlike RNN-based models that require ordered inputs, self-attentive models discard unnecessary order information, aligning with the unordered nature of bundles; (3) Self-attentive models are well-suited for *cloze* tasks, such as those found in BERT [Devlin et al., 2019], making them suitable for predicting unknown items or attributes in slots.

6.3.1 BUNT for Bundle-Aware User Modeling

Long-Term Preference Representation. We encode user historical interactions, which are denoted as $\{\mathscr{B}_1, \mathscr{B}_2, \dots\}$, as user long-term preferences $\mathbf{E}_u$ using hierarchical transformer

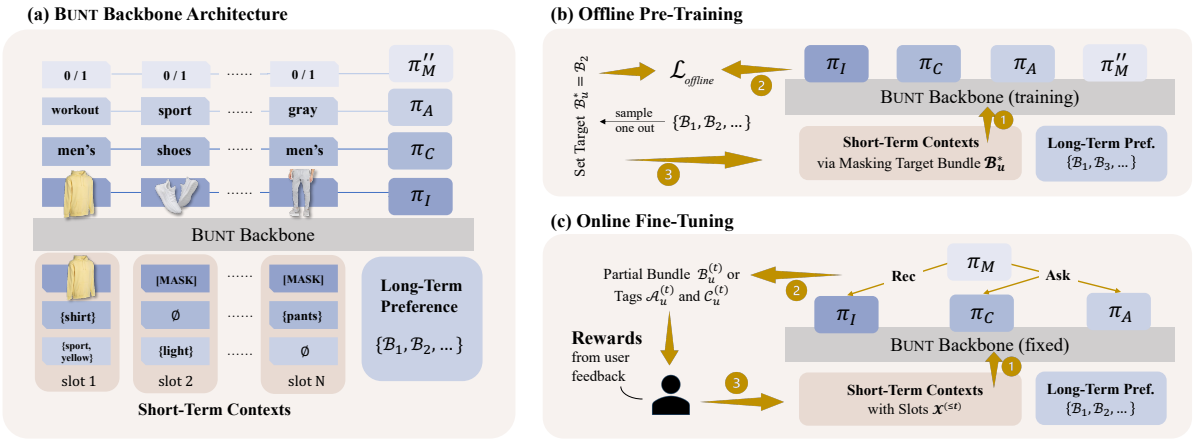


Figure 6.2. (a) BUNT architecture illustration. Bunt is a Bert-like model that encodes long-term preference and short-term contexts to infer masked items, categories and attributes per slot $x \in \mathscr{X}^{(t)}$. In this example, $\mathscr{X}^{(t)} = \{2, N\}$ because the related items are still unknown (i.e.,with `[MASK]`), and $\mathscr{X}^{(\leq t)} = \{1, \ldots, N\}$. We define long-term preference and short-term contexts in Section 6.2.2. (b) BUNT offline pre-training diagram, where ① denotes user modeling, ② mimics the consultation step, ③ mimics the feedback handling step, but instead of updating the conversation state at the next round, offline training simply re-masks the target bundle to generate the next masked bundle as BUNT inputs. (c) BUNT online training diagram, where ① is user modeling, ② is the consultation step to generate partial bundle $\mathscr{B}_u^{(t)}$ or attributes $\mathscr{A}_u^{(t)}$ and categories $\mathscr{C}_u^{(t)}$, ③ is the feedback handling step to update short-term contexts. We describe steps ①-③ in Sections 6.2.2 and 6.3, where we keep π_A and omit the similar policy π_C, for ease of description.

(`TRM`) [Kang and McAuley, 2018] encoders:

$$\mathbf{E}_u = \texttt{TRM}_{\texttt{bundle}}(\{\mathbf{B}_1, \mathbf{B}_2, \ldots\}), \quad \text{where } \mathbf{B}_n = \texttt{AVG}(\texttt{TRM}_{\texttt{item}}(\mathscr{B}_n)), \; n = 1, 2, \ldots. \tag{6.2}$$

$\texttt{TRM}_{\texttt{bundle}}$ is a transformer encoder over the set of bundle-level representations $\{\mathbf{B}_1, \mathbf{B}_2, \ldots\}$, the output $\mathbf{E}_u \in \mathbb{R}^{N_u \times d}$ represents user long-term preferences, N_u is the number of historical bundles, and d is the hidden size of the $\texttt{TRM}_{\texttt{bundle}}$ model. The bundle representation $\mathbf{B}_n \in \mathbb{R}^{1 \times d}$ is also extracted by a transformer encoder, namely $\texttt{TRM}_{\texttt{item}}$, over the set of item embeddings in this bundle, then the set of output embeddings from $\texttt{TRM}_{\texttt{item}}$ is aggregated by average pooling (`AVG`) as $\mathbf{B}_n$. Our two-level transformers contain no positional embeddings since the input representations are unordered.

Short-Term Contexts Representation. We delineate the representation of short-term

contexts $\{(\check{i}_x^{(t)}, \check{\mathscr{A}}_x^{(t)}) | x \in \mathscr{X}^{(\leq t)}\}$, which are subsequently inputted into a dedicated embedding layer EMB. Subsequently, we derive two sets of embeddings for items and attributes.

$$\mathbf{E}_{I,u}^{(t)},\ \mathbf{E}_{A,u}^{(t)} = \texttt{EMB}(\{(\check{i}_x^{(t)}, \check{\mathscr{A}}_x^{(t)}) | x \in \mathscr{X}^{(\leq t)}\}), \tag{6.3}$$

where $\mathbf{E}_{*,u}^{(t)} \in \mathbb{R}^{|\mathscr{X}^{(\leq t)}| \times d}$ represents embeddings for items (I) and attributes (A). For items, the embeddings are retrieved for accepted item ids (or [MASK] id). Regarding attributes, embeddings corresponding to accepted attribute ids (or [PAD] id) in $\check{\mathscr{A}}_x^{(t)}$ for $x \in \mathscr{X}^{(t)}$ are retrieved, followed by applying average pooling (AVG) on embeddings to obtain $\mathbf{E}_{A,u}^{(t)} \in \mathbb{R}^{|\mathscr{X}^{(\leq t)}| \times d}$.

Long- and Short-Term Representation Fusion. We input user long-term preferences $\mathbf{E}_u$ and short-term contexts $\mathbf{E}_{*,u}^{(t)}$ into an L-layer transformer. For simplicity in notation[4], we represent $\mathbf{E}_{I,u}^{(t)}$ as $\mathbf{O}^0$, obtaining the fused representation:

$$\mathbf{O}^l = \texttt{TRM}_l(\widetilde{\mathbf{O}}^{l-1}, \mathbf{E}_u),\ \widetilde{\mathbf{O}}^{l-1} = \texttt{LN}\left(\mathbf{O}^{l-1} \oplus \mathbf{E}_{A,u}^{(t)} \mathbf{W}^{l-1}\right),\ \text{where } l = 1, \ldots, L, \tag{6.4}$$

where $\texttt{TRM}_l$ is the l^{th} transformer layer with cross attention [Vaswani et al., 2017a], $\mathbf{W}^{l-1} \in \mathbb{R}^{d \times d}$ is a learnable projection matrix at layer l-1 for attribute representation. $\oplus$ is element-wise addition and LN denotes LayerNorm [Vaswani et al., 2017a] for training stabilization. We incorporate the attribute feature $\mathbf{E}_{A,u}^{(t)}$ before each transformer layer in order to incorporate multi-resolution levels, which is effective in transformer-based recommender models [Li et al., 2020d]. Thus for the output representation $\mathbf{O}^L \in \mathbb{R}^{|\mathscr{X}^{(\leq t)}| \times d}$, each row $\mathbf{O}_x^L$ ($x \in \mathscr{X}^{(\leq t)}$) contains contextual information from slots in conversation contexts. We treat $\mathbf{O}^L$ and candidate pools $\mathscr{I}_x^{(t)}, \mathscr{P}_x^{(t)}$ for all slots $x \in \mathscr{X}^{(\leq t)}$ as the encoded state $\mathbf{S}_u^{(t)}$. Moreover, for the additional conversation records $\bar{\mathbb{S}}_u^{(t)}$ introduced in Section 6.2.2, we encode it as a vector $\bar{\mathbf{S}}_u^{(t)}$ by using result id embeddings and average pooling.

[4]To be precisely represented as $\mathbf{O}_{I,u}^{(t,0)}$; some notations are omitted for brevity in the decoder description below.

Algorithm 2. BUNT Offline Pre-Training

Input: historical user bundles $\mathscr{D}$, masking ratio ρ, BUNT (including π''_M, π_I, π_A) parameters Θ, slot size K;
Output: BUNT parameters Θ after pre-training;

1: **while** not meet training termination criterion **do**
2: Sample a user $u \in \mathscr{U}$, get historical bundles $\{\mathscr{B}_1, \ldots, \mathscr{B}_{N_u}\}$ from $\mathscr{D}$; sample a historical bundle as target bundle, e.g., $\mathscr{B}_n$;
3: Get $\mathbf{E}_u \leftarrow$ Equation (6.2) with input $\{\mathscr{B}_1, \ldots, \mathscr{B}_{N_u}\} \setminus \{\mathscr{B}_n\}$;
4: Sample l items in $\mathscr{B}_n$ as $\mathscr{B}'_n$; ▷ Mimic partial bundle. W.l.o.g., assume $|B_n| > K$
5: Sample $k \in [1, K]$, then mask k items in B'_n, set the masked positions as slots $\mathscr{X}$;
6: Retrieve attributes for items in $\mathscr{B}'_n$ and mask attributes with probability ρ; ▷ Mimic short-term contexts
7: Predict the distributions of masked items, attributes, and conversation management in slot $x \in \mathscr{X}$ via Equation (6.5);
8: Compute loss $\mathscr{L}_{offline}$ with Equations (6.6) to (6.8); update Θ using gradient-related optimizer (e.g., [Kingma and Ba, 2014]).
9: **end while**

6.3.2 BUNT for Bundle-Aware Consultation

For the consultation step, we feed the encoded state into multiple policy networks to get outputs for each slot $x \in \mathscr{X}^{(t)}$:

$$\begin{cases} P_M(a \mid \bar{\mathbf{S}}_u^{(t)}, \mathbf{O}_x^L) = \beta \cdot \pi'_M(a \mid \bar{\mathbf{S}}_u^{(t)}) + (1-\beta) \cdot \pi''_M(a \mid \mathbf{O}_x^L), \quad \text{where } a \in \{0,1\}, & \text{Conv. Management} \\ P_I(a \mid \mathbf{O}_x^L) = \pi_I(a \mid \mathbf{O}_x^L), \quad \text{where } a \in \mathscr{I}_x^{(t)}, & \text{Bundle Generation} \\ P_A(a \mid \mathbf{O}_x^L) = \pi_A(a \mid \mathbf{O}_x^L), \quad \text{where } a \in \mathscr{P}_x^{(t)}. & \text{Attribute Consultation} \end{cases} \tag{6.5}$$

P_* denotes probability. The policy network π_M is a linear combination of two sub-models, π'_M and π''_M, corresponding to states $\bar{\mathbf{S}}_u^{(t)}$ and $\mathbf{O}_x^L$ respectively. The gating weight β is predicted by an MLP model with a sigmoid function, taking concatenated $\bar{S}_u^{(t)}, \mathbf{O}_x^L$ as input. The models π'_M, π''_M, π_I, and π_A are MLP models with ReLU activation and softmax layer. We utilize π_I or π_A for inferring masked items or attributes in slot x. During the inference stage, actions with the highest probability determine whether to recommend or ask, constructing the consulted (partial) bundle $\mathscr{B}_u^{(t)}$ or questions on attributes $\mathscr{A}_u^{(t)}$. In contrast to other individual-item MCR models, the contextual information stored in different slots plays a crucial role in bundle recommendation. Hence, a unified self-attentive architecture is used to share the state encoded from various slots for both recommendation and question predictions.

Algorithm 3. Online BUNT Fine-Tuning

Input: trainable BUNT parameters Θ_I, Θ_A and Θ_M for three networks π_I, π_A and π_M, empty buffer $\mathbf{M}_M$, $\mathbf{M}_I$ and $\mathbf{M}_A$;
Output: BUNT policy networks parameters Θ_I, Θ_A and Θ_M;

for episode $e = 1, 2, \ldots$ **do**
  Sample a user u, get target bundle $\mathscr{B}_u^*$; initialize $\check{\mathscr{B}}_u \leftarrow \emptyset$ for recording all the accepted items;
  for conversation round $t = 1, 2, \ldots, T$ **do**
    Get conversation states $\mathbf{S}_u^{(t)}$ and $\bar{\mathbf{S}}_u^{(t)}$ via Section 6.3.1; get slots $\mathscr{X}^{(t)}$ via Section 6.2.2; ▷ 1. user modeling
    Sample action a_M from $\{0, 1\}$ using π_M via Section 6.3.2 ; ▷ 2. consultation
    if $a_M == 1$ **then**
      Use $\mathbf{O}^L$ from $\mathbf{S}_u^{(t)}$ to generate a partial bundle $\mathscr{B}_u^{(t)}$ using π_I via Section 6.3.2; ▷ 2.1 recommending
      Update conversation states $\mathbf{S}_u^{(t+1)}$ and $\bar{\mathbf{S}}_u^{(t+1)}$ via Sections 6.2.2 and 6.3.1; get $\tilde{\mathbf{O}}^L$ from $\mathbf{S}_u^{(t+1)}$; ▷ 3. feedback handling
      Add $\{(\mathbf{O}_x^L, \tilde{\mathbf{O}}_x^L, \hat{i}_x, r_x^I) \mid x \in \mathscr{X}^{(t)}\}$ to $\mathbf{M}_I$, calculating r_x^I via Section 6.2.2; ▷ i.e., (state, next_state, action, reward)
      Add accepted items into $\check{\mathscr{B}}_u$;
    else if $a_M == 0$ **then**
      Use $\mathbf{O}^L$ from $\mathbf{S}_u^{(t)}$ to generate questions on attributes $\mathscr{A}_u^{(t)}$ using π_A via Section 6.3.2; ▷ 2.1 asking
      Update conversation states $\mathbf{S}_u^{(t+1)}$ and $\bar{\mathbf{S}}_u^{(t+1)}$ via Sections 6.2.2 and 6.3.1; get $\tilde{\mathbf{O}}^L$ from $\mathbf{S}_u^{(t+1)}$; ▷ 3. feedback handling
      Add $\{(\mathbf{O}_x^L, \tilde{\mathbf{O}}_x^L, \hat{a}_x, r_x^A) \mid x \in \mathscr{X}^{(t)}\}$ to $\mathbf{M}_A$, calculating r_x^A via Section 6.2.2; ▷ i.e., (state, next_state, action, reward)
    end if
    Add $((\mathbf{O}^L, \bar{\mathbf{S}}_u^{(t)}), (\tilde{\mathbf{O}}^L, \bar{\mathbf{S}}_u^{(t+1)}), a_M, r^M)$ to $\mathbf{M}_M$, calculating r^M via Section 6.2.2; ▷ i.e., (state, next_state, action, reward)
    if $\check{\mathscr{B}}_u = \mathscr{B}_u^*$ or $t = T$ **then**
      Current conversation terminates;
    end if
  end for
  if $\mathbf{M}_k$ ($k = \{M, I, A\}$) meets pre-defined buffer training criterion (e.g., buffer size) **then**
    Update Θ_k using $\mathbf{M}_k$ with RL methods (e.g., DQN [Mnih et al., 2015], PPO [Schulman et al., 2017]); Then reset $\mathbf{M}_k$; ▷ policy learning
  end if
end for

6.3.3 Offline Pre-Training

Given the extensive action spaces of items and attributes, training agents directly from scratch poses challenges. Consequently, we initiate the pre-training of the BUNT model using offline user-bundle interactions. The essence of pre-training involves emulating model inputs and outputs within the context of BUNDLEMCR. This can be conceptualized as multiple *cloze* (i.e., "fill the slot") tasks, wherein a few accepted items and attributes are provided to infer the masked items and attributes.

Multi-Task Loss. BUNT offline training is based on a multi-task loss for recommendation and question asking simultaneously, i.e., $\mathscr{L}_{offline} = \mathscr{L}_{rec} + \lambda \mathscr{L}_{ask}$, where λ is a trade-off hyper-parameter to balance the importance of these two losses in offline pre-training. We treat item prediction as a multi-class classification task for masked slots $\mathscr{X}^{(t)}$:

$$\mathscr{L}_{rec} = -\sum_{x \in \mathscr{X}^{(t)}} \sum_{i \in \mathscr{I}_x^{(t)}} y_i \log P_I(i \mid \mathbf{O}_x^L), \tag{6.6}$$

where y_i is the binary label (0 or 1) for item i. Meanwhile, attribute predictions are formulated as multi-label classification tasks. We use a weighted cross-entropy loss function considering the imbalance of labels to prevent the model from only predicting popular attributes. The loss function of attribute predictions is:

$$\mathscr{L}_{ask} = -\sum_{x \in \mathscr{X}^{(t)}} \sum_{a \in \mathscr{P}_x^{(t)}} w_a \cdot y_a \log P_A(a \mid \mathbf{O}_x^L), \tag{6.7}$$

where w_a is a balance weight of attribute a following [King and Zeng, 2001], and note that multiple y_a can be 1 for multi-label classification. Furthermore, we pre-train part of the conversational manager, i.e., π_M'', to decide whether to recommend or ask:

$$\mathscr{L}_{conv} = -\sum_{x \in \mathscr{X}^{(t)}} \mathbb{I}(l_x \neq -1) \cdot \log \pi_M''(l_x \mid \mathbf{O}_x^L). \tag{6.8}$$

For slot x, as long as item agent π_I hits the target item, l_x is set as 1; otherwise, if the attribute agent hits the target, l_x is 0. l_x is set as -1 when no agents make successful predictions. We denote $\mathscr{L}_{ask} = \mathscr{L}_{cate} + \mathscr{L}_{attr} + \mathscr{L}_{conv}$.

Training Details. Figure 6.2b illustrates BUNT offline training. We pre-train BUNT on offline user-bundle interactions, to obtain the basic knowledge to predict the following items or attributes given historical bundle interactions and conversational information. The training details are in Algorithm 2.

6.3.4 Online Fine-Tuning

Figure 6.2c shows the online-training diagram, where we fine-tune BUNT agents during interactions with (real or simulated) users. Our core idea is fixing BUNT backbone parameters, fine-tune agents π_I, π_A and π_M in a BUNDLEMCR environment to update related parameters and improve the accuracy after interacting with users. The online fine-tuning details are in Algorithm 3. We omit the details of RL value networks like [Lei et al., 2020b].

6.4 Experiments

6.4.1 Experimental Setting

Evaluation Protocol and Metrics. Following [Deng et al., 2021a, Kang and McAuley, 2018, Lei et al., 2020b], we use a *leave-one-out* data split. Specifically, for each user, N-1 bundles are randomly selected for offline training. In contrast, the remaining bundle is used for online training, validation, and testing, with a ratio of 6:2:2. We evaluate the quality of the generated bundle using multi-label precision, recall, F1, and accuracy as defined in [Zhang and Zhou, 2014].

Data. We augment four datasets for BUNDLEMCR, as outlined in Table 6.2. Due to the absence of item attributes or category information, alternative bundle datasets such as *Youshu* and *NetEase* are not utilized. **(1) Steam**: Utilizing user interactions with game bundles from [Pathak et al., 2017] on the Steam platform[5], we use item *tags* as *attributes* in BUNDLEMCR, and item *genres* as *categories*. Users with fewer than two bundles are excluded according to our evaluation protocol. **(2) MovieLens**: Employing the ML-10M benchmark dataset [Harper and Konstan, 2015] for collaborative filtering tasks, we consider movies rated with the same timestamps as a bundle. *Genres* are treated as *categories*, and *tags* as *attributes* in BUNDLEMCR. **(3) Clothing**: Derived from [McAuley et al., 2015] on the Amazon e-commerce platform[6], this dataset focuses

[5]https://store.steampowered.com
[6]https://www.amazon.com

Table 6.2. Data Statistics, where # denotes quantity number, *U* denotes user, *I* denotes item, *B* denotes bundle, *C* denotes category and *A* denotes attributes. *B/U* represents the number of bundles per user, *B size* represents the average number of items per bundle.

Dataset	#U	#I	#B	#C	#A	#Inter	B/U	B Size
Steam	13,260	2,819	229	21	327	261,241	2.95	5.76
MovieLens	46,322	5,899	851,361	19	190	3,997,583	27.81	3.11
Clothing	19,065	25,408	79,610	668	4,027	285,391	5.03	3.17
iFashion	340,762	68,921	5,593,387	61	4,264	21,552,716	16.41	3.79

on the *clothing* subcategory. Co-purchased items form a bundle by timestamp. Item *categories* in the metadata serve as *categories* in BUNDLEMCR, and *style* in item reviews (e.g., "format" is "hardcover," we use "hardcover") serve as *attributes*. To enhance data quality, users and items appearing no more than three times are filtered out. **(4) iFashion**: An outfit dataset with user interactions [Chen et al., 2019c]. Following [Wang et al., 2021], iFashion is pre-processed as a 10-core dataset for improved data quality. *Categories* features from iFashion metadata and tokenized *title* are utilized as *categories* and *attributes* in BUNDLEMCR, respectively.

Baselines. We evaluate BUNDLEMCR and our proposed BUNT (referred to as BUNT-Learn in Section 6.3) against three groups of recommendation baselines.

1. **Traditional bundle recommenders. Freq** predicts the most frequent bundle without personalization. **BBPR [Pathak et al., 2017]** ranks existing bundles due to the impractical time cost of cold bundle generation in BBPR. **BGN [Bai et al., 2019]** employs an encoder-decoder [Sutskever et al., 2014] architecture to encode user interactions and generate a sequence of items as a bundle. We use the top-1 bundle in BGN's generated bundle list as the result. **PoG [Chen et al., 2019c]** is a transformer-based [Vaswani et al., 2017a] encoder-decoder model for personalized outfits. We use it for general bundle recommendations. **BYOB [Deng et al., 2021a]** is the latest bundle generator using reinforcement learning methods.

2. **Adopted individual recommenders for BUNDLEMCR. FM-All** is an FM [Rendle, 2010] variant used in MCR frameworks [Lei et al., 2020a,b], where "All" indicates

recommending top-K items per round without questions. **FM-Learn** follows the item predictions in FM-All but utilizes other pre-trained agents in BUNT for conversation management and question posting. **EAR [Lei et al., 2020a]** and **SCPR [Lei et al., 2020b]** are popular Individual MCR frameworks based on FM. We maintain the core ideas of estimation-action-reflection in our EAR, and path reasoning in our SCPR, renaming them to EAR* and SCPR* to adapt to BUNDLEMCR. We exclude recent UNICORN [Deng et al., 2021b] and zhang2022multiple [Zhang et al., 2022] due to incompatible action spaces and question settings with BUNDLEMCR.

3. **Simple bundle recommenders for BUNDLEMCR. BUNT-One-Shot** uses BUNT in traditional bundle recommendation following the inference of PoG [Chen et al., 2019c]. **{BYOB, BGN, BUNT}-All** are straightforward bundle recommender implementations in BUNDLEMCR, recommending only top-K items per round without posing questions.

Training Details. Our training process consists of two stages[7]: (1) In the offline pre-training phase, we use Algorithm 2 to implement and train the BUNT model using `PyTorch`. The number of transformer layers and heads is selected from the set $\{1,2,4\}$, with $d = 32$, $K = 2$, $\lambda = 0.1$, and a masking ratio $\rho = 0.5$. We utilize the Adam optimizer [Kingma and Ba, 2014] with an initial learning rate of 1e-3 for all datasets, employing a batch size of 32. The maximum bundle size is specified as 20. (2) For the online fine-tuning phase, we implement Algorithm 3 using the `OpenAI Stable-Baselines` RL training code. We utilize Proximal Policy Optimization [Schulman et al., 2017] (PPO) in `Stable-Baselines`[8] to jointly train four agents (π_M, π_I, π_C, π_A). Here, π_C represents the category policy, similar to π_A. The training is conducted using the Adam optimizer with lr=1e-3. Remaining hyper-parameters follow default settings in `Stable-Baselines`. We repeat all experiments three times with different random seeds and report the average performance along with related standard errors.

[7]For detailed metric definitions, data processing, BUNT implementation, and human evaluation setup, refer to https://github.com/AaronHeee/Bundle-MCR.

[8]https://stable-baselines3.readthedocs.io

User Simulator Setup Due to the challenges and expenses associated with interacting with real users, our framework evaluations primarily utilize user simulators, as seen in previous studies [Lei et al., 2020a,b, Deng et al., 2021b, Zhang et al., 2022]. We emulate user behavior by simulating a user with a target bundle $\mathscr{B}^*$ sampled from our online dataset. The user simulator *accepts* system-provided items that align with the target bundle $\mathscr{B}^*$ and *accepts* categories and attributes corresponding to potential target items in the current slot.[9] The user simulator explicitly *rejects* categories or attributes not associated with any items in $\mathscr{B}^*$. In other scenarios, the user simulator *ignores* items, categories, and attributes provided by the system. The user simulator can terminate conversations when all items in $\mathscr{B}^*$ have been recommended; otherwise, the system concludes conversations after $t = T$ rounds. We set the maximum conversation rounds T to 10 in our experiments.

6.4.2 Result Analysis

Main Performance of BUNDLEMCR and BUNT-Learn

Table 6.3, Figures 6.3 and 6.4 illustrate the primary performance metrics of our proposed framework and model architecture in comparison to various conversational recommendation baselines. Several key observations can be derived from the results:

BUNT Backbone Performance. While we propose BUNT for the BUNDLEMCR task, we demonstrate its competitiveness in traditional *one-shot* bundle recommendation. Figure 6.3 illustrates that BUNT significantly outperforms classic bundle recommenders (BBPR), and performs comparably or superiorly to recent bundle generators (BGN, PoG). This suggests that the BUNT backbone effectively acquires fundamental "bundle recommendation" knowledge similar to other models.

Effectiveness of BUNDLEMCR. We demonstrate the efficacy of BUNDLEMCR by comparing models (e) and (i) in Table 6.3. Notably, the accuracy on MovieLens data significantly

[9] Initially, all items in $\mathscr{B}^*$ are potential items for a given slot x, but some items are removed with the acceptance of items, categories, and attributes in slot x.

Table 6.3. BUNT and other individual conversational recommendation methods that are adopted for bundle settings. Here *P* denotes Precision, *R* is Recall, *Acc* is accuracy. The best is **bold**.

		Steam				MovieLens			
Group	**Method**	P	R	F1	Acc	P	R	F1	Acc
Individual Rec. Model	(a) FM-All	.149	.611	.239	.138	.019	.087	.031	.017
	(b) FM-Learn	.269	.664	.382	.239	.038	.096	.055	.031
	(c) EAR*	.186	.592	.282	.166	.036	.099	.053	.029
	(d) SCPR*	.173	.544	.262	.151	.044	.110	.063	.032
Bundle Rec. Model	(e) BUNT-One-Shot	.456	.452	.454	.450	.075	.093	.083	.061
	(f) BYOB-All	.328	.799	.463	.323	.020	.113	.034	.018
	(g) BGN-All	.568	.919	.702	.567	.073	.216	.109	.070
	(h) BUNT-All	.633	.927	.752	.632	.100	.289	.149	.096
	(i) BUNT-Learn	**.737**	**.928**	**.822**	**.727**	**.251**	**.302**	**.275**	**.181**

		Clothing				iFashion			
Group	**Method**	P	R	F1	Acc	P	R	F1	Acc
Individual Rec. Model	(a) FM-All	.003	.013	.005	.003	.006	.026	.010	.005
	(b) FM-Learn	.006	.010	.008	.004	.008	028	.012	.006
	(c) EAR*	.011	.022	.014	.008	.017	.026	.020	.010
	(d) SCPR*	.013	.028	.018	.009	.014	.032	.019	.010
Bundle Rec. Model	(e) BUNT-One-Shot	.006	.005	.005	.005	.008	.007	.007	.005
	(f) BYOB-All	.002	.010	.003	.002	.005	.023	.008	.004
	(g) BGN-All	.009	.023	.013	.008	.011	.032	.016	.010
	(h) BUNT-All	.008	.023	.012	.008	.014	**.043**	.021	.014
	(i) BUNT-Learn	**.019**	**.026**	**.021**	**.015**	**.020**	.035	**.025**	**.017**

improves from 0.061 to 0.181 when introducing the conversational mechanism. This underscores that, even with the same backbone model, the incorporation of BUNDLEMCR enables the collection of immediate feedback, leading to enhanced recommendation performance. Furthermore, we observe a higher relative improvement on three datasets compared to Steam. Specifically, the accuracy increases by 61.56% on Steam, while it rises by 196.72% on MovieLens. This highlights that challenging datasets, characterized by sparser and larger item spaces, derive more significant benefits from BUNDLEMCR, as it facilitates user feedback during interactions.

Effectiveness of BUNT-Learn. We incorporated various Individual MCR recom-

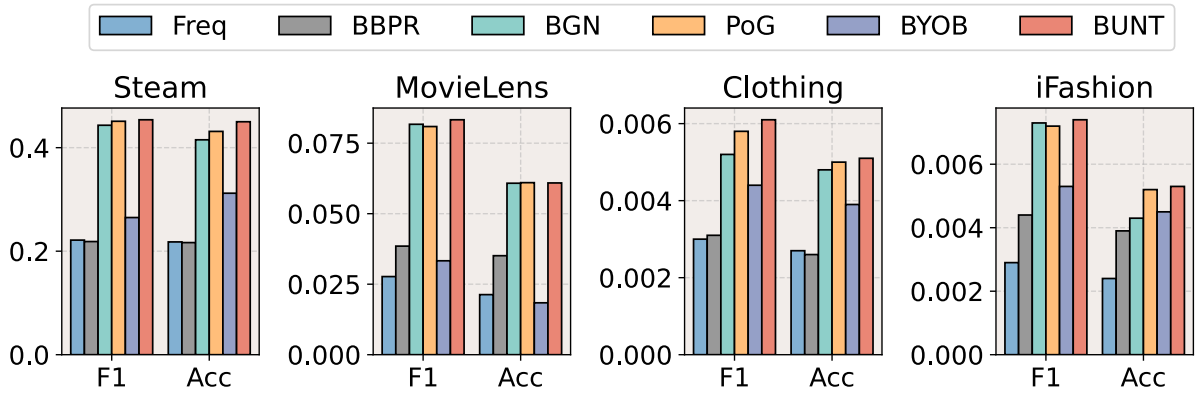


Figure 6.3. BUNT performance compared with other bundle recommenders in *one-shot* bundle recommendation.

menders (a)-(d) from Table 6.3 into bundle settings, where the FM backbone model [Rendle, 2010] suggests top-K items without considering bundle contexts. BUNT-*Learn* outperforms these individual MCR recommenders, achieving the highest performance. For instance, compared to model (b) where we replace the BUNT backbone with FM, our approach improves accuracy from 0.239 to 0.727 on Steam. This highlights that directly applying existing individual MCR recommenders in BUNDLEMCR is suboptimal and underscores the advantages of our BUNT design. Furthermore, in comparison to bundle recommenders exclusively suggesting items (models (f)-(h)), our approach incorporates *question asking*, consistently enhancing recommendation performance, except for the recall score in iFashion. This discrepancy arises as we replace recommending with asking, potentially leading to a decrease in recall due to fewer recommendation rounds (although F1-Score still improves).

Accuracy Curve with Conversation Rounds. The cumulative accuracy curves in Figure 6.4 show BUNT-All achieves the best results in beginning conversation rounds, then is outperformed by BUNT-Learn. This is because BUNT-Learn requires several rounds to ask questions and elicit preferences. Thus, BUNT-Learn in late rounds can recommend more accurately and surpass baselines. For example, on MovieLens, BUNT-Learn outperforms the baselines after $t = 6$.

BUNT-Learn Component Analysis. Compared with (a), (b)-(d) show the effectiveness

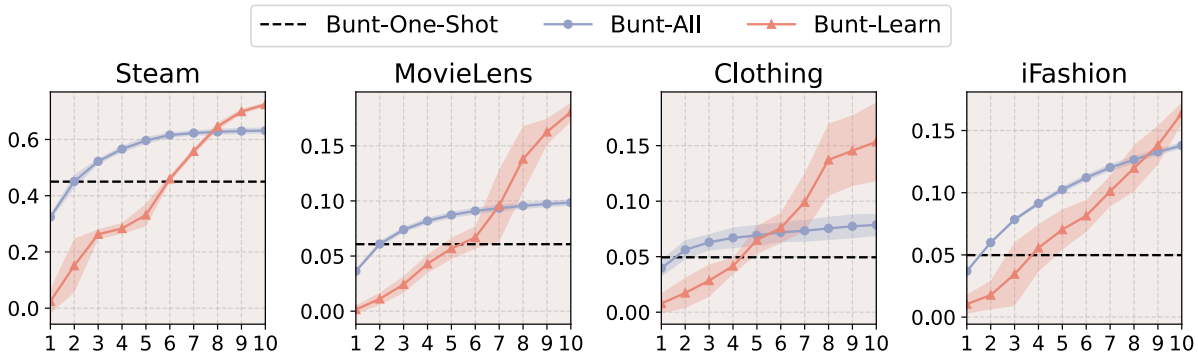


Figure 6.4. Cumulative accuracy curve, i.e., accuracy after t rounds, $t \in [1, 10]$.

of long-term preference and short-term context encoding; (e) indicates the importance of using bundle-aware models; (f)-(i) show the benefit of online fine-tuning, which helps π_M most because conversation management is hard to mimic in offline datasets, and π_M with only a binary action space is easier for online learning than other policies; (j)-(m) show pre-training is necessary, especially for item policy, because bundle generation is challenging to directly learn from online interactions with RL. This also indicates the proposed multiple cloze pre-training tasks are suitable for training BUNDLEMCR effectively.

Human Evaluation on Conversation Trajectories

Considering the expenses associated with deploying interactive Bundle MCRs, as exemplified in previous studies ([Xian et al., 2021, Jannach and Manzoor, 2020]), we conducted human evaluations where real users assessed generated conversation trajectories from Section 6.4.2. Sampling 1000 pairs of conversation trajectories from the Steam and MovieLens datasets, derived from either (BUNT-Learn, SCPR*) or (BUNT-Learn, FM-Learn) (with SCPR* and FM-Learn representing the best baselines utilizing individual item recommenders), we posted each pair on MTurk[10] to collect 5 responses. MTurk workers, spending more than 30 seconds and possessing a `LifeTimeAcceptanceRate` of 100%, were instructed to assess subjective quality by selecting the superior model within the given pair. Majority votes were tallied for each pair, resulting

[10] https://requester.mturk.com/

Table 6.4. Ablation Studies (F1-score) to evaluate model architecture, fine-tuning (FT) and pre-training (PT).

Ablation	Steam	MovieLens
(a) Bunt-Learn	**.822**	**.275**
(b) w/o Long-term Pref.	.701	.148
(c) w/o Short-term Tags	.787	.165
(d) w/o Short-term Items	.330	.084
(e) replace BUNT π_I with FM	.382	.032
(f) w/o FT π_M	.765	.210
(g) w/o FT π_I	.817	.268
(h) w/o FT $\pi_{\{A,C\}}$	.811	.257
(i) w/o FT All	.753	.206
(j) w/o PT π_M	.807	.258
(k) w/o PT π_I	.056	.008
(l) w/o PT $\pi_{\{A,C\}}$	.815	.171
(m) w/o PT All	.003	.001

in 388 valid responses: (BUNT-Learn, SCPR*) received 121 votes to 88, and ¡BUNT-Learn, FM-Learn¿ received 110 votes to 69. This outcome demonstrates the superiority of BUNT-Learn. Notably, the performance gap in human evaluation is not as pronounced as observed in simulator results (e.g., on Steam, BUNT-Learn accuracy is 3 times that of FM-Learn).

6.5 Conclusion and Future Work

In this chapter, we extend the Multi-Round Conversational Recommendation (MCR) framework to encompass bundle recommendation scenarios, formulating it as a Multi-Agent Markov Decision Process (MDP). We introduce the BUNT model architecture to handle bundle contexts in conversations. To enable BUNT to acquire bundle knowledge from both offline datasets and an online environment, we propose a two-stage training strategy involving *cloze* tasks and multi-agent reinforcement learning. The effectiveness of our model and training strategy is demonstrated through experiments on four offline bundle datasets and human evaluation. As the first work addressing conversation mechanisms in bundle recommendation, potential

future research directions are highlighted. The incorporation of *free text* in bundle conversational recommendation remains an open question, given that our question spaces in BUNT pertain to categories and attributes. Explicitly integrating item relationships (e.g., substitutes, complements) into conversational bundle recommendation poses an interesting and challenging task. Additionally, unifying existing individual conversational recommenders into conversational bundle recommendations, thereby enhancing conversational agents' capabilities at no extra cost, presents an intriguing avenue for exploration.

We investigate the challenging task of bundle recommendation within the context of conversational recommendations. Bundle recommendation involves suggesting multiple dependent items (e.g., outfits, playlists) to users rather than individual items. This scenario offers additional benefits to users and platforms but poses exceptional challenges to traditional recommendation methods due to its complexity and limited user signals. We reexamine this task in a conversational setting, referred to as BUNDLEMCR, and demonstrate that the conversational mechanism breaks down the bundling process, easing tasks across multiple real-world datasets, and opening up new opportunities for research in this field.

Chapter 6, in part, is a reprint of the material as it appears in "Bundle MCR: Towards Conversational Bundle Recommendation." by Zhankui He, Handong Zhao, Tong Yu, Sungchul Kim, Fan Du, and Julian McAuley, in *Proceedings of the 16th ACM Conference on Recommender Systems in 2022*, referenced as [He et al., 2022b]. The dissertation author was the primary investigator and author of this paper.

Chapter 7

Large Language Models in Conversational Recommender Systems

In this chapter, we delve into another intricate recommendation scenario – natural-language interaction scenarios. In these scenarios, users express their needs and provide feedback to the recommender systems using natural languages. This is typically considered a fully free-text conversational recommendation scenario. Our contribution involves presenting empirical studies in this scenario using representative large language models in a zero-shot setting.

In particular, we make three primary contributions. (1) Data: To gain insights into model behavior in "in-the-wild" natural-language conversational recommendation scenarios, we construct a new dataset of recommendation-related conversations by scraping a popular discussion website. This is the largest public real-world conversational recommendation dataset to date. (2) Evaluation: On the new dataset and two existing conversational recommendation datasets, we observe that even without fine-tuning, large language models can outperform existing fine-tuned conversational recommendation models. (3) Analysis: We propose various probing tasks to investigate the mechanisms behind the remarkable performance of large language models in conversational recommendation. We analyze both the large language models' behaviors and the characteristics of the datasets, providing a holistic understanding of the models' effectiveness, and limitations and suggesting directions for the design of future conversational recommenders.

7.1 Introduction

Conversational recommender systems (CRS) aim to offer personalized recommendations through interactive conversations, considering users' natural-language inputs alongside historical actions. Unlike traditional recommenders, CRS can provide human-like responses, serving purposes like preference refinement, knowledgeable discussion, or recommendation justification. A typical CRS consists of a *generator* for natural-language responses and a *recommender* for ranking items [Li et al., 2018b, Chen et al., 2019b, Zhou et al., 2020b, Wang et al., 2022].

Recent advancements in large language models (LLMs)[1], such as ChatGPT [John Schulman, 2022], have sparked interest in leveraging LLMs for recommendation and personalization tasks [Kang et al., 2023, Bao et al., 2023, Hou et al., 2023, Salemi et al., 2023, Liu et al., 2023]. However, existing efforts primarily focus on evaluating LLMs in traditional recommendation settings, relying on users' past actions like clicks [Kang et al., 2023, Bao et al., 2023, Hou et al., 2023, Liu et al., 2023]. The conversational recommendation scenario, which involves more natural language interactions, is still nascent [Friedman et al., 2023, Wang et al., 2023].

This work proposes the use of *large language models as zero-shot conversational recommenders* and empirically investigates their recommendation abilities [John Schulman, 2022, OpenAI, 2023, Chiang et al., 2023, Xu et al., 2023]. Our contributions encompass three key aspects related to *data*, *evaluation*, and *analysis*.

Data. We introduce REDDIT-MOVIE, a substantial conversational recommendation dataset comprising over 634k recommendation-seeking dialogs from users on Reddit[2]. Unlike existing crowd-sourced conversational recommendation datasets such as ReDIAL [Li et al., 2018b] and INSPIRED [Hayati et al., 2020], where workers simulate user-recommender interactions, REDDIT-MOVIE captures real-world conversations where users naturally seek and provide item recommendations. This dataset is the largest public conversational recommendation dataset

[1] LLMs refer to large-sized pre-trained language models with exceptional zero-shot abilities as defined in [Zhao et al., 2023].

[2] https://www.reddit.com/

to date, containing 50 times more dialogs than ReDIAL.

Evaluation. In assessing the recommendation performance of Language Models (LLMs) across multiple Conversational Recommendation System (CRS) datasets, we identify a *repeated item shortcut* in current CRS evaluation protocols. This involves the presence of "repeated items'' in testing samples, serving as ground-truth items and enabling the creation of a simplistic baseline (e.g., copying mentioned items from the conversation history) that outperforms many existing models. We address this issue by eliminating "repeated items" from training and testing data and re-evaluate various representative conversational recommendation models [Li et al., 2018b, Chen et al., 2019b, Zhou et al., 2020b, Wang et al., 2022] on ReDIAL, INSPIRED, and our Reddit dataset. In this refined experimental setup, we empirically demonstrate that LLMs can outperform existing fine-tuned conversational recommendation models even without additional fine-tuning.

Analysis. Given the notable zero-shot performance of LLMs in Conversational Recommendation Systems, a fundamental question emerges: *What contributes to their outstanding performance?* Similar to [Penha and Hauff, 2020], we propose that LLMs leverage both *content/context knowledge* (e.g., "genre," "actors," and "mood") and *collaborative knowledge* (e.g., "users who like A typically also like B") for making conversational recommendations. We design probing tasks to unveil the model's inner workings and the characteristics of CRS data. Furthermore, empirical findings are presented, highlighting specific limitations of LLMs as zero-shot CRS, despite their effectiveness.

The main findings of this study are summarized as follows: (1) Re-evaluating CRS recommendation abilities requires avoiding using repeated items as ground truth. (2) Zero-shot conversational recommenders, specifically LLMs, exhibit enhanced performance on both established and new datasets compared to fine-tuned CRS models. (3) LLMs primarily leverage their superior content/context knowledge for recommendations, rather than relying on collaborative knowledge. (4) CRS datasets inherently contain a substantial amount of content/context information, making them well-suited for LLMs in comparison to traditional recommendation tasks.

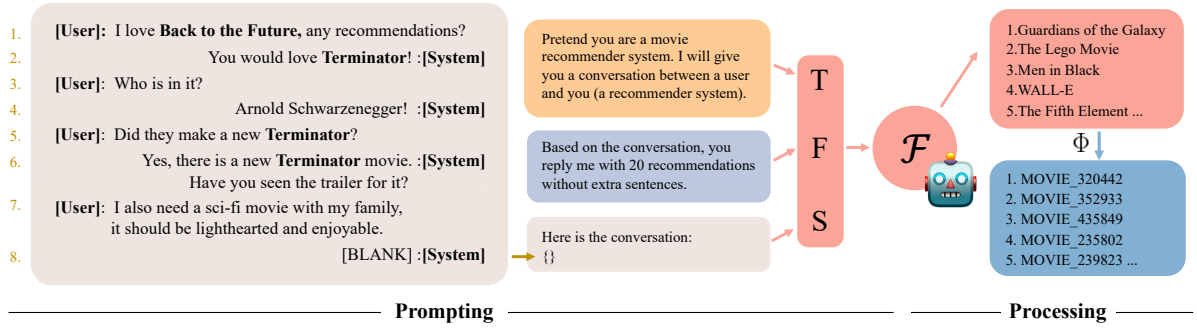


Figure 7.1. Large Language Models (LLMs) as Zero-Shot Conversational Recommenders (CRS).We introduce a simple prompting strategy to define the task description T, format requirement F and conversation context S for a LLM, denoted as $\mathscr{F}$, we then post-process the generative results into ranked item lists with processor Φ.

(5) Despite their effectiveness, LLMs face limitations such as popularity bias and sensitivity to geographical regions.

These findings highlight the unique significance of superior content/context knowledge in LLMs for CRS tasks, highlighting the potential of LLMs as a valuable approach in CRS. However, it is crucial for future CRS designs with LLMs to acknowledge challenges in evaluation, datasets, and potential issues like debiasing.

7.2 Proposed Approach: LLM4CRS

7.2.1 Task Formation

Given user set $\mathscr{U}$, item set $\mathscr{I}$, and vocabulary $\mathscr{V}$, a conversation is represented as $C = (u_t, s_t, \mathscr{I}_t)_{t=1}^{T}$. In the t^{th} turn, a speaker $u_t \in \mathscr{U}$ generates an utterance $s_t = (w_i)_{i=1}^{m}$, where $w_i \in \mathscr{V}$. The utterance s_t includes a set of mentioned items $\mathscr{I}_t \subset \mathscr{I}$ ($\mathscr{I}_t$ can be empty if no items are mentioned). Typically, two users in the conversation play the roles of *seeker* and *recommender*. For example, in the 2^{nd} turn of the conversation shown in Figure 7.1, $t = 2$, u_t is `[System]`, s_t is "You would love Terminator !", and $\mathscr{I}_2$ is a set containing the movie `Terminator`. Following the approach in many Conversational Recommender Systems (CRS) papers [Li et al., 2018b, Chen et al., 2019b, Zhou et al., 2020b, Wang et al., 2022], the *recommender* component aims to

optimize the following objective during the k^{th} turn: with u_k as the recommender, it takes the conversational context $(u_t, s_t, \mathscr{I}_t)_{t=1}^{k-1}$ as input and generates a ranked list of items $\hat{\mathscr{I}}_k$ that best matches the ground-truth items in $\mathscr{I}_k$.

7.2.2 Framework

Prompting. Our objective is to use Large Language Models (LLMs) as zero-shot conversational recommenders. Specifically, without the necessity for fine-tuning, we aim to elicit responses from an LLM, denoted as $\mathscr{F}$, using a task description template T, format requirement F, and conversational context S before the k^{th} turn. This procedure can be formally expressed as:

$$\hat{\mathscr{I}}_k = \Phi\left(\mathscr{F}(T, F, S)\right). \tag{7.1}$$

To better understand this zero-shot recommender, we present an example in Figure 7.1 with the prompt setup in our experiments.[3]

Models. We consider several popular LLMs $\mathscr{F}$ that exhibit zero-shot prompting abilities in two groups. To try to ensure deterministic results, we set the decoding temperature to 0 for all models. (1) `GPT-3.5-turbo` [John Schulman, 2022][4] and `GPT-4` [OpenAI, 2023] from OPENAI with abilities of solving many complex tasks in zero-shot setting [OpenAI, 2023, Bubeck et al., 2023] but are closed-sourced. (2) **BAIZE** [Xu et al., 2023][5] and **Vicuna** [Chiang et al., 2023], which are representative open-sourced LLMs fine-tuned based on `LLAMA-13B` [Touvron et al., 2023].

Processing. We do not assess model weights or output logits from LLMs. Therefore, we apply a post-processor Φ (e.g., fuzzy matching) to convert a recommendation list in natural language to a ranked list $\hat{\mathscr{I}}_k$. The approach of generating item titles instead of ranking item IDs is referred to as a *generative retrieval* [Cao et al., 2021, Tay et al., 2022] paradigm.

[3] We leave more prompting techniques such as CoT [Wei et al., 2022] in future work.
[4] Referred as `GPT-3.5-t` hereafter
[5] We use **BAIZE-V2** in https://huggingface.co/project-baize/baize-v2-13b

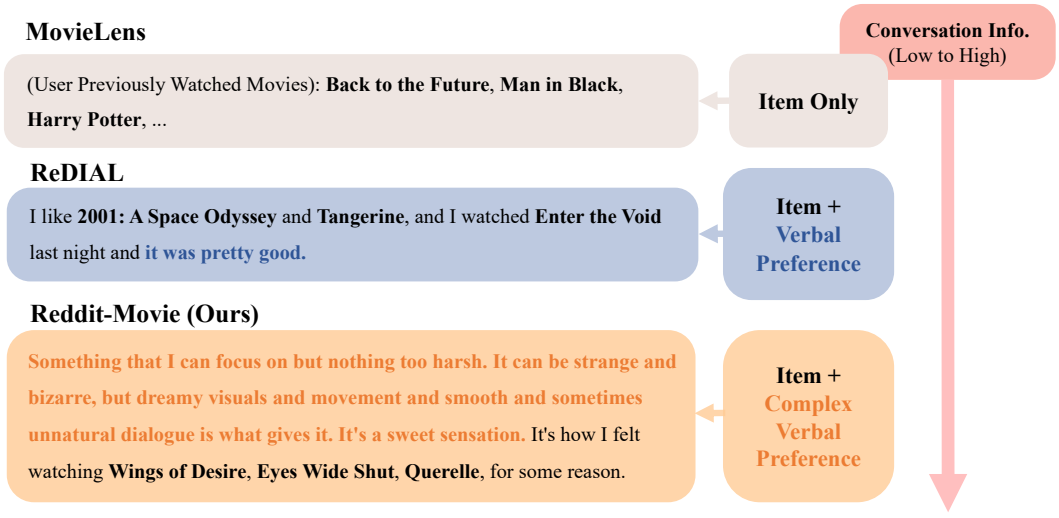


Figure 7.2. Typical model inputs from a traditional recommendation dataset (MovieLens [Harper and Konstan, 2015]), an existing CRS dataset (ReDIAL [Li et al., 2018b]), and our REDDIT-MOVIE dataset. The REDDIT-MOVIE dataset contains more information in its textual content compared to existing datasets where users often explicitly specify their preference. See Section 7.5.2 for quantitative analysis.

Table 7.1. Dataset Statistics. We denote a subset of REDDIT-MOVIE in 2022 as `base`, and the entire ten-year dataset as `large`.

Dataset	#Conv.	#Turns	#Users	#Items
INSPIRED [Hayati et al., 2020]	999	35,686	999	1,967
ReDIAL [Li et al., 2018b]	11,348	139,557	764	6,281
REDDIT-MOVIE$^{\texttt{base}}$	85,052	133,005	10,946	24,326
REDDIT-MOVIE$^{\texttt{large}}$	634,392	1,669,720	36,247	51,203

7.3 Constructed Dataset: REDDIT-MOVIE

Ideally, to assess the conversational recommendation models accurately, a large-scale dataset featuring diverse interactions and real-world conversations is crucial. Existing conversational recommendation datasets, typically obtained through crowdsourcing [Li et al., 2018b, Hayati et al., 2020, Kang et al., 2019, Zhou et al., 2020c], only partially capture realistic conversation dynamics. For instance, responses like *"Whatever Whatever I'm open to any suggestion."* in the ReDIAL dataset reflect crowd workers' general lack of specific preferences during task completion. This stands in contrast to real users who may have distinct and specific needs, as

illustrated in Figure 7.2.

To construct CRS datasets, we introduce the REDDIT-MOVIE dataset. This dataset, the largest-scale conversational movie recommendation dataset to date, comprises naturally occurring movie recommendation conversations from Reddit. It provides richer perspectives for training and evaluating CRS models when combined with existing crowdsourced datasets such as ReDIAL [Li et al., 2018b] and INSPIRED [Hayati et al., 2020]. Our model evaluation and analysis encompass these three datasets, demonstrating qualitative examples from the Reddit dataset (see Figure 7.2) and quantitative analyses (refer to Section 7.5.2).

7.3.1 Dataset Construction

To construct the CRS dataset from Reddit, we utilize *pushshift.io*[6], processing posts spanning from January 2012 to December 2022. Focusing on movie recommendation contexts, we extract pertinent posts from five relevant subreddits: *r/movies*, *r/bestofnetflix*, *r/moviesuggestions*, *r/netflixbestof*, and *r/truefilm*. Our methodology involves a pipeline comprising *conversational recommendation identification*, *movie mention recognition*, and *movie entity linking*. Additional details, including evaluation data, scripts, and results, are available in our GitHub repository[7]. For evaluation, the most recent 9k conversations in REDDIT-MOVIE$^{\texttt{base}}$ from December 2022 constitute the test set, with the remaining 76k conversations serving as the training and validation set for other compared models.

7.3.2 Discussion

Analyzing the statistics presented in Table 7.1, we observe that REDDIT-MOVIE is the most extensive conversational recommendation dataset, featuring 634,392 conversations encompassing 51,203 movies. In comparison to ReDIAL [Li et al., 2018b] and INSPIRED [Hayati et al., 2020], REDDIT-MOVIE exhibits a lower number of multi-turn conversations, attributed to the inherent characteristics of Reddit posts. Representative instances depicted in Figure 7.2 illustrate

[6]https://pushshift.io/

[7]https://github.com/AaronHeee/LLMs-as-Zero-Shot-Conversational-RecSys

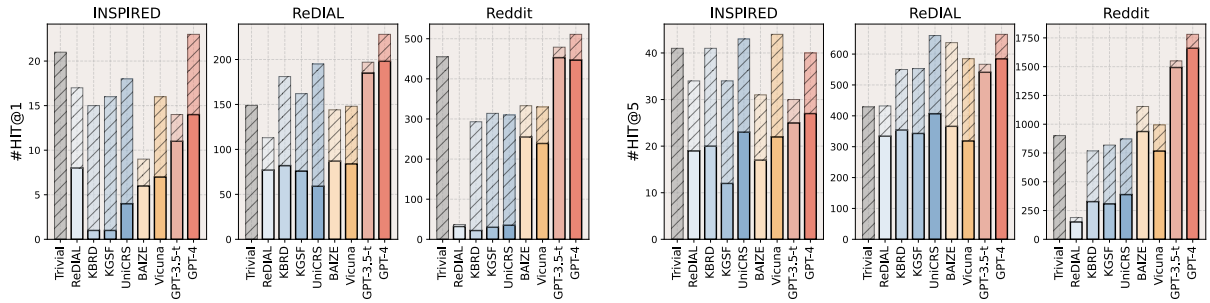


Figure 7.3. To show the *repeated item shortcut*, we count CRS recommendation hits using the Top-K ranked list $K = \{1, 5\}$. We group the ground-truth hits by repeated items (shaded bars) and new items (not shaded bars). The trivial baseline copies existing items from the current conversation history in chronological order, from the most recent and does not recommend new items.

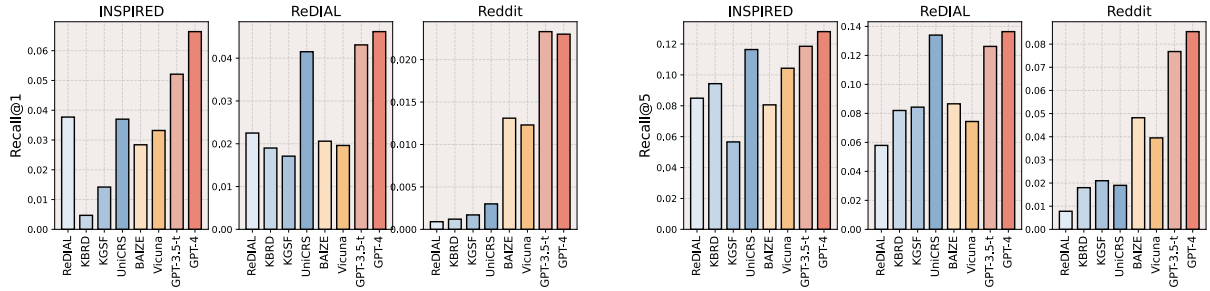


Figure 7.4. CRS recommendation performance on New Items in terms of Recall@K, with $K = \{1, 5\}$. To exclude the influence of repeated items in CRS evaluation, we remove all repeated items in training and testing datasets and re-train all baselines.

that REDDIT-MOVIE conversations manifest more intricate and detailed user preferences, thereby contributing to the diversity of discussions within conversational recommendation datasets.

7.4 General Experiments

In this section, we evaluate the proposed LLMs-based framework on ReDIAL [Li et al., 2018b], INSPIRED [Hayati et al., 2020], and our Reddit datasets. We first explain the evaluation setup and a *repeated item shortcut* of the previous evaluation in Sections 7.4.1 and 7.4.2. Then, we re-train models and discuss LLM performance in Section 7.4.2.

7.4.1 Experimental Setting

Repeated *vs.* New Items. Given a conversation $C = (u_t, s_t, \mathscr{I}_t)_{t=1}^{T}$, identifying ground-truth recommended items, $\mathscr{I}_k$, at the k^{th} turn is challenging. In the common evaluation setup, if u_k is the recommender, all items $i \in \mathscr{I}_k$ are assumed to be ground-truth recommended items. We introduce a finer categorization of items $i \in \mathscr{I}_k$ into *repeated items* and *new items*. Repeated items are those that appeared in previous turns, i.e., $\{i \mid \exists t \in [1,k), i \in \mathscr{I}_t\}$, while new items are not mentioned in earlier turns. Details of this categorization are provided in Section 7.4.2.

Evaluation Protocol. On the three datasets, we assess the recommendation capabilities of various CRS models and LLMs. Utilizing the authors' training code, we establish baselines and measure prediction performance through Recall@K [Li et al., 2018b, Chen et al., 2019b, Zhou et al., 2020b, Wang et al., 2022] (i.e., HIT@K). We present the mean and standard errors[8] of the metric for $K = \{1,5\}$.

Compared CRS Models. We utilize representative CRS models, employing entity linking results from the ReDIAL and INSPIRED datasets by UniCRS [Wang et al., 2022] as baselines that rely on structured knowledge. Omitted are additional works [Li et al., 2022b, Ren et al., 2022, Ma et al., 2021] as UniCRS [Wang et al., 2022] is considered representative with comparable results.

- **ReDIAL** [Li et al., 2018b]: This model is released along with the ReDIAL dataset with an auto-encoder [Sedhain et al., 2015]-based recommender.
- **KBRD** [Chen et al., 2019b]: This model proposes to use the DBPedia [Auer et al., 2007] to enhance the semantic knowledge of items or entities.
- **KGSF** [Zhou et al., 2020b]: This model incorporates two knowledge graphs to enhance the representations of words and entities, and uses the Mutual Information Maximization method to align the semantic spaces of those two knowledge graphs.

[8] Standard errors are depicted as error bars in figures and gray numbers in tables.

- **UniCRS** [Wang et al., 2022]: UniCRS uses DialoGPT [Zhang et al., 2020c], a pre-trained language model, with prompt tuning to conduct recommendation and conversation generation tasks respectively.

7.4.2 Result Analysis

Repeated Items Can Be Shortcuts

Current evaluation methodologies for conversational recommendation systems do not distinguish between repeated and new items in a conversation. This evaluation approach tends to favor systems optimizing for repeated item mentions. As illustrated in Figure 7.3, a basic baseline that consistently copies seen items from the conversation history outperforms most previous models under the standard evaluation criteria.

This observation raises concerns about shortcut learning [Geirhos et al., 2020], where a decision rule excels against specific benchmarks and evaluations but fails to capture the true intent of the system designer. Specifically, the #HIT@1 metric for the tested models decreases by over 60% on average when focusing solely on new item recommendations, a nuance not apparent in the overall recommendation performance. Upon manual inspection, a recurring pattern of repeated items is identified, as exemplified in the conversation snippet presented in Figure 7.1. In this instance, the movie `Terminator` at the 6^{th} turn serves as the ground-truth item. The system repeats this item because it was quoted for content-based discussion, rather than making a recommendation. Given the context of recommendation conversations, it is more likely that repeated items are intended for discussion rather than serving as recommendations.

We argue that considering the substantial occurrence of repeated items (e.g., more than 15% of ground-truth items are repeated items in INSPIRED), it is advantageous to eliminate repeated items and reevaluate conversational recommendation system (CRS) models to gain a better understanding of their true recommendation capabilities. It is noteworthy that repetition patterns have also been explored in the evaluation of other recommender systems, such as *next-basket* recommendation [Li et al., 2023c].

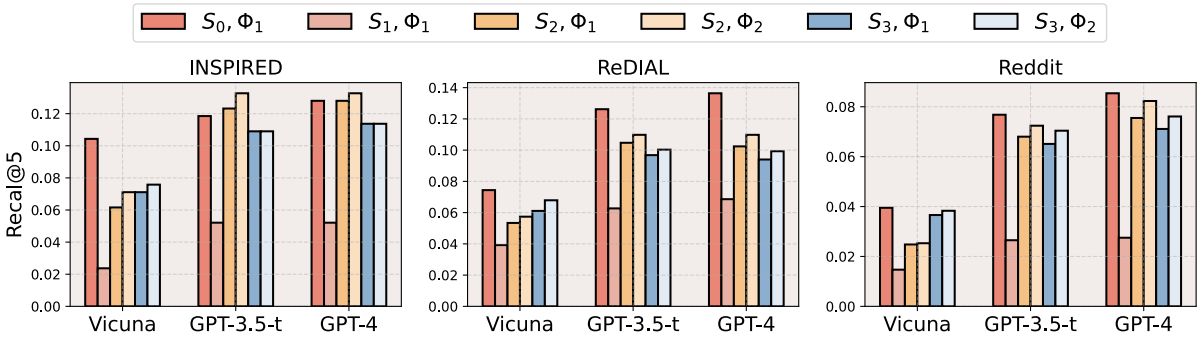


Figure 7.5. Ablation studies for the research question about the primary knowledge used by LLMs for CRS. Here Φ_1 is the post-processor which only considers in-dataset item titles; Φ_2 is the post-processor based on Φ_1 and further excludes all seen items in conversational context from generated recommendation lists. For inputs like *Original* (S_0) and *ItemOnly* (S_1), LLMs show similar performance with Φ_1 or Φ_2, so we only keep Φ_1 here. We consider Φ_2 because *ItemRemoved* (S_2) and *ItemRandom* (S_3) have no information about already mentioned items, which may cause under-estimated accuracy using Φ_1 compared to *Original*.

Table 7.2. Recall@1 results of considering all generated item titles (Φ_0) and only considering in-dataset item titles (Φ_1).

	INSPIRED		**ReDIAL**		**Reddit**	
Model	Φ_0	Φ_1	Φ_0	Φ_1	Φ_0	Φ_1
BAIZE	.019	**.028**	**.021**	**.021**	.012	**.013**
Vicuna	.028	**.033**	**.020**	**.020**	**.012**	**.012**
`GPT-3.5-t`	.047	**.052**	.041	**.043**	.022	**.023**
`GPT-4`	.062	**.066**	.043	**.046**	.022	**.023**

LLM4CRS Performance

Finding 1 - LLMs outperform fine-tuned CRS models in a zero-shot setting. For a comparison between models' abilities to recommend new items to the user in conversation, we re-train existing CRS models on all datasets for new item recommendation only. The evaluation results are as shown in Figure 7.4. Large language models, although not fine-tuned, have the best performance on all datasets. Meanwhile, the performance of all models is uniformly lower on Reddit compared to the other datasets, potentially due to the large number of items and fewer conversation turns, making recommendations more challenging.

Finding 2 - GPT-based models achieve superior performance than open-sourced

Table 7.3. Fraction of Top-K ($K = 20$ in our prompt setup) recommendations (#rec) that can be string matched in the IMDB movie database (%imdb) for the different models, which shows a lower bound of non-hallucinated movie titles.

BAIZE		**Vicuna**		`GPT-3.5-t`		`GPT-4`	
#rec	%imdb	#rec	%imdb	#rec	%imdb	#rec	%imdb
259,333	81.56%	258,984	86.98%	321,048	95.51%	322,323	94.86%

LLMs. As shown in Figure 7.4, large language models consistently outperform other models across all three datasets, while GPT-4 is generally better than GPT-3.5-t. We hypothesize this is due to GPT-4's larger parameter size enables it to retain more correlation information between movie names and user preferences that naturally occurs in the language models' pre-training data. Vicuna and BAIZE, while having comparable performance to prior models on most datasets, have significantly lower performance than its teacher, GPT-3.5-t. This is consistent with previous works' finding that smaller distilled models via imitation learning cannot fully inherit larger models' ability on downstream tasks [Gudibande et al., 2023].

Finding 3 - LLMs may generate out-of-dataset item titles, but few hallucinated recommendations. We note that language models trained on open-domain data naturally produce items out of the allowed item set during generation. In practice, removing these items improves the models' recommendation performance. Large language models outperform other models (with GPT-4 being the best) consistently regardless of whether these unknown items are removed or not, as shown in Table 7.2. Meanwhile, Table 7.3 shows that around 95% generated recommendations from GPT-based models (around 81% from BAIZE and 87% from Vicuna) can be found in IMDB [9] by string matching. Those lower bounds of these matching rates indicate that there are only a few hallucinated item titles in the LLM recommendations in the movie domain.

[9] Movie titles in https://datasets.imdbws.com/.

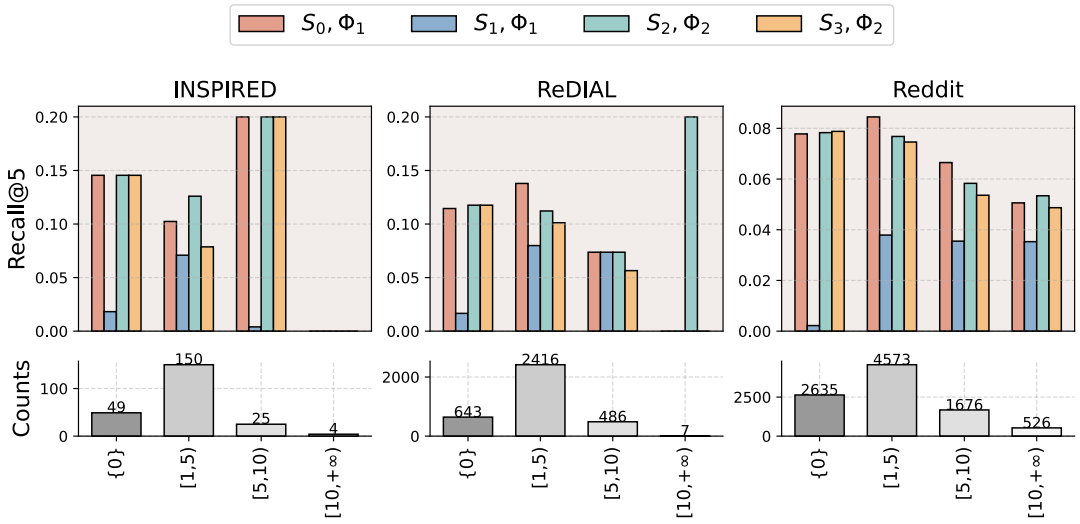


Figure 7.6. `GPT-3.5-t` Recall@5 results grouped by the occurrences of items in conversation context, and count the conversations per dataset.

7.5 Detailed Analysis

Observing LLMs' remarkable conversational recommendation performance for the zero-shot recommendation, we are interested in *what accounts for their effectiveness* and *what their limitations are*. We aim to answer these questions from both a model and data perspective.

7.5.1 Knowledge in LLMs

Experiment Setup. Motivated by the probing work of [Penha and Hauff, 2020], we posit that two types of knowledge in LLMs can be used in CRS:

- **Collaborative knowledge**, which requires the model to match items with similar ones, according to community interactions like "users who like A typically also like B". In our experiments, we define the collaborative knowledge in LLMs as *the ability to make accurate recommendations using item mentions in conversational contexts*.
- **Content/context knowledge**, which requires the model to match recommended items with their content or context information. In our experiments, we define the content/context knowledge in LLMs as *the ability to make accurate recommendations based on all other*

Table 7.4. To understand the content/context knowledge in LLMs and existing CRS models, we re-train the existing CRS models using the same perturbed conversation context *ItemRemoved* (S_2). We include the results of the representative CRS model UniCRS (denoted as CRS*) as well as a representative text-encoder `BERT-small` [Devlin et al., 2019] (denoted as TextEnc*).

	INSPIRED		**ReDIAL**		**Reddit**	
Model	R@1	R@5	R@1	R@5	R@1	R@5
Vicuna	.024	.062	.014	.053	.008	.025
`GPT-3.5-t`	.057	.123	.030	**.105**	.018	.068
`GPT-4`	**.062**	**.128**	**.032**	.102	**.019**	**.075**
CRS*	.039	.087	.015	.058	.001	.008
TextEnc*	.038	.090	.013	.053	.002	.009

conversation inputs rather than item mentions, such as contextual descriptions, mentioned genres, and director names.

To understand how LLMs use these two types of knowledge, given the original conversation context S (Example in Figure 7.1), we perturb S with three different strategies as follows and subsequently re-query the LLMs. We denote the original as S_0:

- **S_0 (Original):** we use the original conversation context.
- **S_1 (ItemOnly):** we keep mentioned items and remove all natural-language descriptions in the conversation context.
- **S_2 (ItemRemoved):** we remove the mentioned items and keep other content in the conversation context.
- **S_3 (ItemRandom):** we replace the mentioned items in the conversation context with items that are uniformly sampled from the item set $\mathscr{I}$ of this dataset, to eliminate the potential influence of S_2 on the sentence grammar structure.

Finding 4 - LLMs mainly rely on content/context knowledge to make recommendations. Figure 7.5 shows a drop in performance for most models across various datasets when

replacing the original conversation text *Original* (S_0) with other texts, indicating that LLMs leverage both content/context knowledge and collaborative knowledge in recommendation tasks. However, the importance of these knowledge types differs. Our analysis reveals that content/context knowledge is the primary knowledge utilized by LLMs in CRS. When using *ItemOnly* (S_1) as a replacement for *Original*, there is an average performance drop of more than 60% in terms of Recall@5. On the other hand, GPT-based models experience only a minor performance drop of less than 10% on average when using *ItemRemoved* (S_2) or *ItemRandom* (S_3) instead of *Original*. Although the smaller-sized model Vicuna shows a higher performance drop, it is still considerably milder compared to using *ItemOnly*. To accurately reflect the recommendation abilities of LLMs with *ItemRemoved* and *ItemRandom*, we introduce a new post-processor denoted as Φ_2 (describe in the caption of Figure 7.5). By employing Φ_2, the performance gaps between *Original* and *ItemRemoved* (or *ItemRandom*) are further reduced. Furthermore, Figure 7.6 demonstrates the consistent and close performance gap between *Original* and *ItemRemoved* (or *ItemRandom*) across different testing samples, which vary in size and the number of item mentions in *Original*.

The findings indicate that, in the context of a conversation, Language Models (LLMs) primarily rely on content/context knowledge rather than collaborative knowledge for generating recommendations. This behavior contrasts with conventional recommenders such as collaborative filtering [Koren et al., 2009, Rendle, 2010, Liang et al., 2018, He and Chua, 2017, He et al., 2018, Sedhain et al., 2015] or sequential recommenders [Kang and McAuley, 2018, Sun et al., 2019, Zhou et al., 2020a, He et al., 2021], where user-interacted items play a crucial role.

Finding 5 - GPT-based LLMs possess better content/context knowledge than existing CRS. From Table 7.4, we observe the superior recommendation performance of GPT-based LLMs against representative conversational recommendation or text-only models on all datasets, showing the remarkable zero-shot abilities in understanding user preference with the textual inputs and generating correct item titles. We conclude that GPT-based LLMs can provide more accurate recommendations than existing trained CRS models in an *ItemRemoved* (S_2) setting, demonstrating better content/context knowledge.

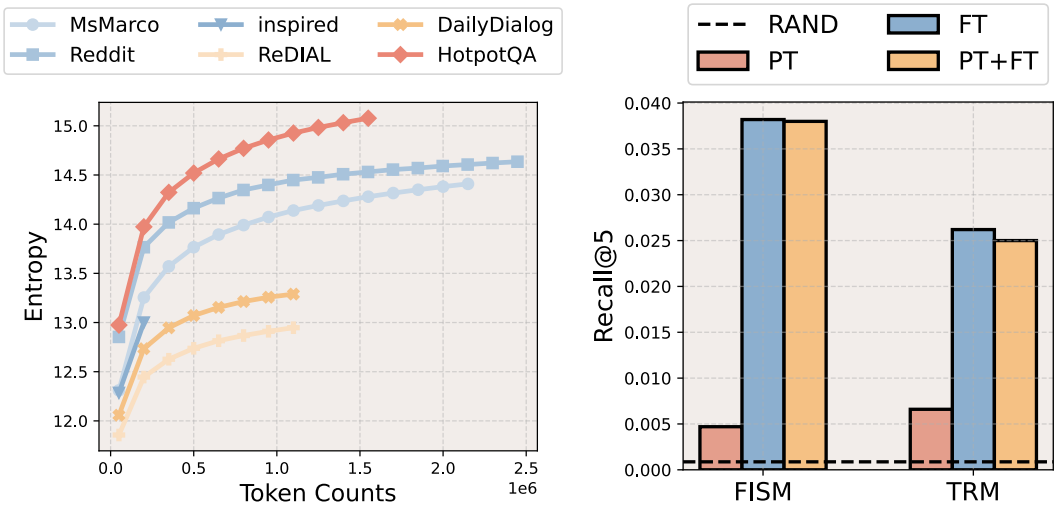
Table 7.5. To understand the collaborative knowledge in LLMs and existing CRS models, we re-train the existing CRS models using the same perturbed conversation context *ItemOnly* (S_1). We include the results of the representative CRS model UniCRS (denoted as CRS*) as well as a representative item-based collaborative model FISM [Kabbur et al., 2013] (denoted as ItemCF*).

	INSPIRED		**ReDIAL**		**Reddit**	
Model	R@1	R@5	R@1	R@5	R@1	R@5
Vicuna	.005	.024	.011	.039	.005	.015
`GPT-3.5-t`	.024	.052	.021	.063	.007	.026
`GPT-4`	.014	.052	.025	.069	**.007**	**.028**
CRS*	.038	.085	.025	.072	.003	.015
ItemCF*	**.042**	**.087**	**.029**	**.088**	.004	.018

Finding 6 - LLMs generally possess weaker collaborative knowledge than existing CRS. In Table 7.5, the results from INSPIRED and ReDIAL indicate that LLMs underperform existing representative CRS or ItemCF models by 30% when using only the item-based conversation context *ItemOnly* (S_1). It indicates that LLMs, trained on a general corpus, typically lack the collaborative knowledge exhibited by representative models trained on the target dataset. There are several possible reasons for this weak collaborative knowledge in LLMs. First, the training corpus may not contain sufficient information for LLMs to learn the underlying item similarities. Second, although LLMs may possess some collaborative knowledge, they might not align with the interactions in the target datasets, possibly because the underlying item similarities can be highly dataset- or platform-dependent.

However, in the case of the Reddit dataset, LLMs outperform baselines in both Recall@1 and Recall@5, as shown in Table 7.5. This outcome could be attributed to the dataset's large number of rarely interacted items, resulting in limited collaborative information. The Reddit dataset contains 12,982 items with no more than 3 mentions as responses. This poses a challenge in correctly ranking these items within the Top-5 or even Top-1 positions. LLMs, which possess at least some understanding of the semantics in item titles, have the chance to outperform baselines trained on datasets containing a large number of cold-start items.

Recent research on LLMs in traditional recommendation systems [Liu et al., 2023, Hou



(a) Entropy v.s. Token Counts **(b)** Pre-training Effectiveness

Figure 7.7. The left subfigure shows the entropy of the frequency distribution of 1,2,3-grams with respect to the number of words drawn from each dataset (item names excluded) to measure the content/context information across datasets. The right subfigure shows the results of the processed Reddit collaborative dataset aligned to ML-25M [Harper and Konstan, 2015]. RAND denotes random baseline, FT denotes fine tuning on Reddit, PT denotes pre-training on ML-25M, PT+FT means FT after PT.

et al., 2023, Kang et al., 2023] also observes the challenge of effectively leveraging collaborative information without knowing the target interaction data distribution. Additionally, another study [Bao et al., 2023] on traditional recommendation systems suggests that LLMs are beneficial in a setting with many cold-start items. Our experimental results support these findings within the context of conversational recommendations.

7.5.2 Information from CRS Data

Experimental Setup for Finding 7. To understand LLMs in CRS tasks from the data perspective, we first measure the *content/context information* in CRS datasets. Content/context information refers to the amount of information contained in conversations, excluding the item titles, which reasonably challenges existing CRS and favors LLMs according to the findings in Section 7.5.1. Specifically, we conduct an entropy-based evaluation for each CRS dataset and compare the conversational datasets with several popular conversation and question-answering

datasets, namely DailyDialog (chit chat) [Li et al., 2017c], MsMarco (conversational search) [Bajaj et al., 2018], and HotpotQA (question answering). We use *ItemRemoved* (S_2) conversation texts like Section 7.5.1, and adopt the geometric mean of the entropy distribution of 1,2,3-grams as a surrogate for the amount of information contained in the datasets, following previous work on evaluating information content in text [Jhamtani et al., 2018]. However, entropy naturally grows with the size of a corpus, and each CRS dataset has a different distribution of words per sentence, sentences per dialog, and corpus size. Thus, it would be unfair to compare entropy between corpus on a per-dialog, per-turn, or per-dataset basis. To ensure a fair comparison, we repeatedly draw increasingly large subsets of texts from each of the datasets, compute the entropy of these subsets, and report the trend of entropy growth with respect to the size of the subsampled text for each CRS dataset.

Finding 7 - Reddit provides more content/context information than the other two CRS datasets. Based on the results in Figure 7.7a, we observe that the Reddit dataset has the most content/context information among the three conversational recommendation datasets. Those observations are also aligned with the results in Figure 7.5 and table 7.4, where LLMs – which possess better content/context knowledge than baselines – can achieve higher relative improvements compared to the other two datasets. Meanwhile, the content/context information in Reddit is close to question answering and conversational search, which is higher than existing conversational recommendation and chit-chat datasets.

Finding 8 - Collaborative information is insufficient for satisfactory recommendations, given the current models. Quantifying the collaborative information in datasets is challenging. Instead of proposing methods to measure collaborative information, we aim to make new observations based on general performance results presented in Figure 7.4 and recommendation results using only collaborative information in Table 7.5. Comparing the performance of the best models in Table 7.5 under an *ItemOnly* (S_1) setting with the performance of the best models in Figure 7.4 under an *Original* (S_0) setting reveals a significant disparity. For instance, on ReDIAL, the Recall@1 performance is 0.029 for ItemCF* compared to 0.046 for GPT-4,

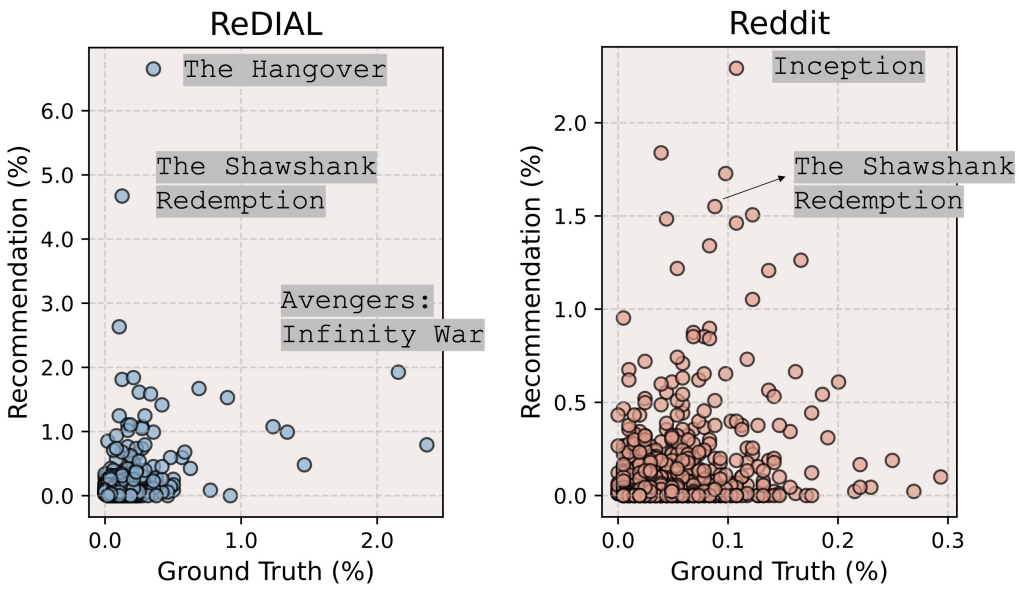


Figure 7.8. Scatter plots of the frequency of LLMs (GPT-4) generated recommendations and ground-truth items.

representing a 39.96% decrease. Similarly, for Reddit, the Recall@1 performance is 0.007 compared to 0.023 for GPT-4, which is 69.57% lower. We also experimented with other recommender systems, such as transformer-based models [Kang and McAuley, 2018, Sun et al., 2019] to encode the item-only inputs and found similar results. Based on the current performance gap, we find that using the existing models, relying solely on collaborative information is insufficient to provide satisfactory recommendations. We speculate that either (1) more advanced models or training methods are required to better comprehend the collaborative information in CRS datasets, or (2) the collaborative information in CRS datasets is too limited to support satisfactory recommendations.

Experimental Setup for Finding 9. To understand whether the collaborative information from CRS datasets is aligned with pure interaction datasets, we conduct an experiment on the Reddit dataset. In this experiment, we first process the dataset to link the items to a popular interaction dataset ML-25M [Harper and Konstan, 2015] [10]. We then experiment with two

[10]We only use items that can be linked to ML-25M in this experiment. Here 63.32% items are linked using the `links.csv` file from ML-25M.

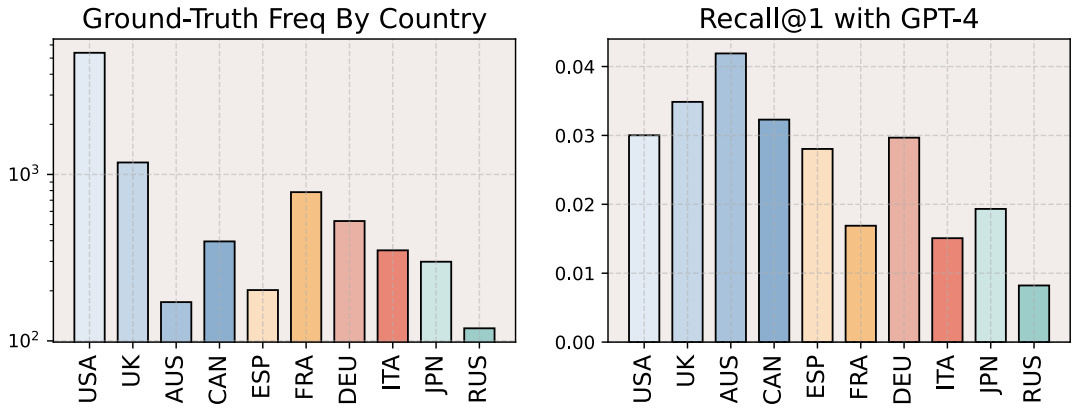


Figure 7.9. Ground-truth item counts in Reddit by country (in log scale) and the corresponding Recall@1 by country.

representative encoders for item-based collaborative filtering based on FISM [Kabbur et al., 2013] and Transformer [Sun et al., 2019] (TRM), respectively. We report the testing results on Reddit, with fine-tuning on Reddit (FT), pre-training on ML-25M (PT), and pre-training on ML-25M then fine-tuning Reddit (PT+FT). Note that since it is a linked dataset with additional processing, the results are not comparable with the aforementioned results on Reddit.

Finding 9 - Collaborative information can be dataset- or platform-dependent. From Figure 7.7b shows that the models solely pre-trained on ML-25M (PT) outperform a random baseline, indicating that the data in CRS may share item similarities with pure interaction data from another platform to some extent. However, Figure 7.7b also shows a notable performance gap between PT and fine-tuning on Reddit (FT). Additionally, we do not observe further performance improvement when pre-training on ML-25M then fine-tuning on Reddit (PT+FT). These observations indicate that the collaborative information and underlying item similarities, even when utilizing the same items, can be largely influenced by the specific dataset or platform. The finding also may partially explain the inferior zero-shot recommendation performance of LLMs in Table 7.5 and suggest the necessity of further checking the alignment of collaborative knowledge in LLMs with the target datasets.

7.5.3 Limitations of Our LLM4CRS

Finding 10 - LLM recommendations suffer from popularity bias in CRS. Popularity bias is the tendency for popular items to be recommended more frequently than their inherent popularity [Chen et al., 2023]. In fig. 7.8, the presence of popularity bias in language model (LLM) recommendations is evident, although it may not align with the popularity distribution in the target datasets. For instance, on ReDIAL, highly popular movies like `Avengers: Infinity War` appear approximately 2% of the time across all ground-truth items; on Reddit, movies like `Everything Everywhere All at Once` make up less than 0.3% of ground-truth items. However, in the *generated* recommendations from GPT-4 (similar trends observed in other LLMs), highly popular items like `The Shawshank Redemption` are present around 5% of the time on ReDIAL and approximately 1.5% of the time on Reddit. The LLMs' recommendations exhibit a concentration on popular items compared to the target datasets, potentially leading to issues such as the bias amplification loop [Chen et al., 2023]. Notably, the consistently recommended popular items across different datasets may reflect the prevalence of these items in the LLMs' pre-training corpus.

Finding 11 - Recommendation performance of LLMs is sensitive to geographical regions. Despite the effectiveness in general, it is unclear whether LLMs can be good recommenders across various cultures and regions. Specifically, pre-trained language models' strong open-domain ability can be attributed to pre-training from massive data [Brown et al., 2020]. But it also leads to LLMs' sensitivity to data distribution. To investigate LLMs recommendation abilities for various regions, we take test instances from the Reddit dataset and obtain the production region of 7,476 movies from a publicly available movie dataset [11] by exact title matching, then report the Recall@1 for the linked movies grouped by region. We only report regions with more than 300 data points available to ensure enough data to support the result. As shown in Figure 7.9 the current best model, GPT-4's performance on recommendation is higher for movies

[11] https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset

produced in English-speaking regions. This could be due to bias in the training data - the left of Figure 7.9 show items on Reddit forums are dominated by movies from English-speaking regions. Such a result highlights large language model's recommendation performance varies by region and culture and demonstrates the importance of cross-regional analysis and evaluation for language model-based conversational recommendation models.

7.6 Conclusion and Discussion

Our empirical study addresses a repetition shortcut in standard CRS evaluations, highlighting its potential to yield unreliable conclusions on model design. Furthermore, we showcase the superior performance of LLMs as zero-shot CRS, outperforming fine-tuned existing models. Inspired by this efficacy, a comprehensive analysis from both model and data perspectives is conducted to elucidate the working mechanisms of LLMs, characteristics of CRS tasks, and limitations of direct LLM utilization in CRS. Experimental evaluations utilize publicly available datasets and a newly created, extensive movie recommendation dataset sourced from a popular discussion website, establishing the largest public CRS dataset for more diverse and realistic conversations. Future directions based on our findings are also discussed in this section.

On LLMs. Given the remarkable performance even without fine-tuning, LLMs exhibit potential as an effective CRS approach, leveraging superior content and contextual knowledge. The success of open-sourced LLMs [Xu et al., 2023, Chiang et al., 2023] opens avenues for improving CRS through efficient tuning [Hu et al., 2021, Bao et al., 2023] and collaborative filtering [Koren et al., 2009] ensembling. Concurrently, tasks like debiasing [Chen et al., 2023] and ensuring trustworthiness [Ge et al., 2022] in the context of LLMs merit reconsideration.

On CRS. Our findings advocate systematic re-benchmarking of various CRS models to comprehend their recommendation abilities and task characteristics comprehensively. A deeper understanding of CRS tasks necessitates diverse datasets from sources such as crowd-sourcing platforms [Li et al., 2018b, Hayati et al., 2020], discussion forums, and realistic CRS applications

spanning domains, languages, and cultures. Additionally, our analysis of information types underscores the unique importance of superior content and contextual knowledge in LLMs for CRS tasks. This distinction sets CRS tasks apart from traditional recommendation settings, prompting exploration of interconnections between CRS tasks and traditional *recommendation* [Harper and Konstan, 2015] or *conversational search* [Bajaj et al., 2018] tasks.

Chapter 7, in part, is a reprint of the material as it appears in "Large language models as zero-shot conversational recommenders." by Zhankui He*, Zhouhang Xie*, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley in *Proceedings of the 32nd ACM International Conference on Information & knowledge management in 2023*, referenced as [He et al., 2023]. The dissertation author was the primary investigator and author of this paper.

Chapter 8

Related Work

8.1 Related Work for LOCKER and QUERYSR

To accurately model users' sequential interactions, we focus on capturing complex item-to-item patterns and user dynamics. To this end, various model architectures have been investigated for sequential recommendation. Traditional models integrate factorized user models with Markov Chains (MCs) to capture sequential dynamics [Rendle et al., 2010, He and McAuley, 2016a]. For example, FPMC [Rendle et al., 2010] used first-order MCs to capture item-to-item transitions and Matrix Factorization (MF) models [Koren et al., 2009] to model static user preferences; FOSSIL [He and McAuley, 2016a] adopts higher-order MCs with the item-based factorized model FISM [Kabbur et al., 2013].

Recently, the effectiveness of deep neural networks has been widely demonstrated for sequential modeling (e.g. language models [Cho et al., 2014b, Vaswani et al., 2017a]). Therefore, RNN-based models have been proposed [Hidasi et al., 2015, Hidasi and Karatzoglou, 2018, Donkers et al., 2017] for sequential recommendation with several variants [Huang et al., 2018b, Ma et al., 2019]. CNN-based models have been proposed [Tang and Wang, 2018, Ma et al., 2019] to capture more complex user behavior patterns (e.g. "skip behaviors" and "union-level" dependencies). Attention-based models [Kang and McAuley, 2018, Sun et al., 2019] and related variants [Wu et al., 2020, Lin et al., 2020] have achieved state-of-the-art performance with the help of Transformer [Vaswani et al., 2017a] architectures to identify relevant' items in sequences

to predict the next action(s). We can leverage any model architecture above as a backbone for *query-aware* sequential recommendation. In this paper, we build our model on SASRec [Kang and McAuley, 2018], which is a popular transformer-based "backbone" for state-of-the-art sequential recommendation variants [Wu et al., 2020, Lin et al., 2020].

Conventional user behavior sequences take the form of *(user, action, item, time)* tuples in chronological order. Side information can be introduced among these four aspects to better understand user behavior. For *user profiles*, ARNN [Song and Lee, 2018] considers user-side information (e.g. age, gender) to model user preferences with a product-based network. For *user actions*, MKM-SR [Meng et al., 2020] incorporates *user* micro-behavior and *item* knowledge to learn better representations. For *items*, to enrich item representations, Parallel-RNN [Hidasi et al., 2016], FDSA [Zhang et al., 2019] and NOVA-BERT [Liu et al., 2021] propose different frameworks on the basis of sequential recommenders [Hidasi et al., 2015, Kang and McAuley, 2018, Sun et al., 2019] to incorporate heterogeneous item features such as keywords, genres and textual reviews. For *temporal information*, TiSASRec [Li et al., 2020a] proposes a time interval-aware self-attention module for sequential recommendation; dwell time is also considered by a few works [Bogina and Kuflik, 2017, Dallmann et al., 2017].

8.2 Related Work for UCEPIC and P-SHOWCASE

Generating explanations for recommended items has been extensively explored in various formats [Zhang et al., 2020e, 2014, Gao et al., 2019] such as item aspects, attributes, and similar users. Recently, there has been a growing interest in natural-language-based explanation generation [Li et al., 2021b, Ni and McAuley, 2018, Lu et al., 2018, Li et al., 2017b, 2020c, 2023b, Ni et al., 2019] for producing post-hoc explanations in a personalized style. For instance, Li et al. [2017b] used an RNN-based model to generate explanations based on predicted ratings. To enhance control over the explanation generation process, Ni et al. [2019] extracted aspects and managed the semantics of generated explanations conditioned on different aspects. Addi-

tionally, Li et al. [2021b] proposed a personalized transformer model that generates explanations based on given item features. The area of review generation is closely related, as explanation generation methods often derive expressive and informative explanations from user reviews. Several controllable review generators [Dong et al., 2017, Ni and McAuley, 2018] serve as baseline models in early experiments for explanation generation. Although previous works have increased generation controllability, they are predominantly based on auto-regressive generation frameworks [Li et al., 2019, Ni and McAuley, 2018, Li et al., 2020b, 2021a, Hua and Wang, 2019, Moryossef et al., 2019], focusing solely on aspect planning.

In particular, for UCEPIC, lexically constrained generation involves ensuring that generated text adheres to specific lexical constraints, such as keywords. Early approaches typically used specialized decoding methods. Hokamp et al. [Hokamp and Liu, 2017] introduced a lexical-constrained grid beam search decoding algorithm, integrating constraints into the decoding process. Post et al. [Post and Vilar, 2018] proposed an algorithm for lexically constrained decoding with reduced complexity in constraint handling. Hu et al. [Hu et al., 2019] enhanced decoding through a vectorized dynamic beam allocation. Miao et al. [Miao et al., 2019] introduced a sampling-based conditional decoding method, placing constraints in a template and decoding words using Metropolis-Hastings sampling. While special decoding methods often incur high time complexity, Zhang et al. [Zhang et al., 2020d] achieved hard-constrained generation with $\mathcal{O}(\log n)$ time complexity. This was accomplished through language model pre-training and insertion-based generation [Stern et al., 2019, Gu et al., 2019b, Chan et al., 2019, Gu et al., 2019a] commonly used in machine translation. CBART [He, 2021] leverages the pre-trained BART model [Lewis et al., 2020], utilizing the encoder for insertion instruction and the decoder for predicting masks.

In terms of multimodal generative methods for P-SHOWCASE, recent successes in deep learning for multi-modal learning and pretraining [Tan and Bansal, 2019, Radford et al., 2021, Chen et al., 2021] leverage Transformer [Vaswani et al., 2017a] structures to encode visual and textual features for pretraining, benefiting downstream multimodal tasks. CLIP [Radford

et al., 2021] is a powerful model trained on massive image-caption pairs, exhibiting strong zero-shot capabilities across vision and language tasks. Approaches like CLIPScore [Hessel et al., 2021] and Imagine [Zhu et al., 2021] use CLIP embeddings for modality similarity metrics in text generation tasks. Particularly, Contrastive learning [Oord et al., 2018] aims to learn representations by contrasting positive and negative pairs, applied in various machine learning fields. In computer vision [Chen et al., 2020a, He et al., 2020], natural language processing [Huang et al., 2018a], and recommender systems [Wei et al., 2021], promising results emerge. Recent works demonstrate the effectiveness of contrastive learning in conditional text generation, involving adversarial example generation [Lee et al., 2020] and hard negative identification using pre-trained language models [Yan et al., 2021].

8.3 Related Work for BUNDLEMCR and LLM4CRS

We discuss interactivity in the context of Conversational recommender systems (CRS). CRS aims to understand user preferences and provide personalized recommendations through conversations. Existing CRS interactivities are categorized based on dialogue setups:

(1) About Free Text: Generates human-like responses in natural language [Kang et al., 2019, Li et al., 2018b, Chen et al., 2019b]. For instance, [Li et al., 2018b] introduces the *ReDial* dataset and employs a hierarchical RNN framework. KBRD [Chen et al., 2019b] integrates knowledge-grounded information.

(2) About Items: Utilizes absolute (e.g., "Do you want item *A*?") or relative-question templates (e.g., "Choose between item *A* or *B*?")[Christakopoulou et al., 2016, Xie et al., 2021]. Strategies include *Greedy*, *UCB*[Auer, 2002], or *Thompson Sampling* [Chapelle and Li, 2011].

(3) About Tags: Poses questions on user preferences for different tags associated with items [Sun and Zhang, 2018, Zhang et al., 2018]. CRM [Sun and Zhang, 2018] integrates conversation and recommender systems using a unified deep reinforcement learning framework. SAUR [Zhang et al., 2018] employs a *System Ask-User Respond* paradigm. Multi-round Conver-

sational Recommendation (MCR) [Lei et al., 2020a,b, Deng et al., 2021b, Zhang et al., 2022] falls under this setting.

In particular, MCR is arguably the most practical setting available [Lei et al., 2020a,b, Deng et al., 2021b, Zhang et al., 2022] before the appearance of LLMs to support free-text conversation interactions and has been widely adopted in recent conversational recommender systems. For instance, EAR [Lei et al., 2020a] introduces an *Estimation-Action-Reflection* framework that solicits attributes and models users' online feedback. Additionally, SCPR [Lei et al., 2020b] integrates an item-attribute graph to offer explainable conversational recommendations. UNICORN [Deng et al., 2021b] proposes a unified reinforcement learning framework based on a dynamic weighted graph for MCR. To enhance the realism of individual MCR, zhang2022multiple [Zhang et al., 2022] permits users to select multiple choices for questions and models user preferences using multi-interest encoders. However, existing MCR frameworks are primarily designed for individual item recommendation (Individual MCR). Consequently, the overall model architecture (e.g., FM [Rendle, 2010]) and question strategy design are incompatible with bundle contexts. In this dissertation, our BUNDLEMCR adopts a similar conversation management approach as existing individual MCRs. Nevertheless, we specifically design model architectures for bundle-aware user modeling, question asking, feedback handling, and bundle generation.

For free-text conversational recommendations, as natural language processing has advanced, the community developed "deep" CRS [Li et al., 2018b, Chen et al., 2019b, Wang et al., 2022] that support interactions in natural language. Aside from collaborative filtering signals, prior work shows that CRS models benefit from various additional information. Examples include knowledge-enhanced models [Chen et al., 2019b, Zhou et al., 2020b] that make use of external knowledge bases [Auer et al., 2007, Liu and Singh, 2004], review-aware models [Lu et al., 2021], and session/sequence-based models [Zou et al., 2022, Li et al., 2022b]. Presently, UniCRS [Wang et al., 2022], a model built on DialoGPT [Zhang et al., 2020c] with prompt tuning [Brown et al., 2020], stands as the state-of-the-art approach on CRS datasets such as

ReDIAL [Li et al., 2018b] and INSPIRED [Hayati et al., 2020]. Currently, by leveraging LLMs, [Friedman et al., 2023] proposes a new CRS pipeline but does not provide quantitative results, and [Wang et al., 2023] proposes better user simulators to improve evaluation strategies in LLMs. In this dissertation, our LLM4CRS uncover a *repeated item shortcut* in the previous evaluation protocol and propose a framework where LLMs serve as zero-shot CRS with detailed analyses to support our findings from both model and data perspectives.

Chapter 9

Conclusion and Future Outlook

9.1 Summary of Contributions

In this dissertation, we present three research pillars towards the core components of conversational recommender systems, including *accuracy*, *explainability*, and *interactivity*.

In terms of *accuracy*, we study how to accurately model user sequential and dynamic behaviors with both homogeneous and heterogeneous sequential data. One contribution from LOCKER is to uncover and address the intrinsic limitations of the vanilla self-attention architectures in capturing users' interests in sequential action sequences. Another contribution is from QUERYSR, which shows that the heterogenous user behavior sequences punctuated by textual queries provide strong signals to uncover user dynamics. Those two models are evaluated in sequential recommendation datasets as testbeds, yet they have the potential to serve as backbones of pure conversational recommendation applications where user sequential behaviors are mixed with text inputs and combined by long-term preferences and short-term interests.

In terms of *explainability*, we focus on the controllability and multi-modality of explanation generation for recommended items. One contribution from UCEPIC shows the effectiveness of including accurate and detailed keyphrases in explanation generation to offer richer information to users. Another contribution from P-SHOWCASE is that we are able to go beyond text-only explanations to involve more modalities (e.g., visual) in explanation generation to provide more intuitive and useful explanations for users. Those models are evaluated by the

explanation sub-tasks and human evaluations and can serve as an important part of conversational recommender systems as well.

In terms of *interactivity*, we enable multi-round interactions between systems and users by investigating two complex recommendation scenarios. First, our contribution in BUNDLEMCR demonstrates that complex recommendation tasks, which traditional recommender systems are struggling with, become much easier when conversational mechanisms are introduced. Second, another contribution in LLM4CRS provides insights for using LLMs in natural-language-based conversational recommendations, where LLMs are able to understand item content and complex contextual descriptions from users, yet more efforts should be made to enhance LLMs' collaborative knowledge to make recommendations.

In addition to those technical contributions, our works also contribute two important research resources for the community. The first is the dataset GEST from Google Local reviews, serving as a large-scale recommendation dataset with unique multi-modal explanations along with user-item interactions, and other interesting features such as locations; The second is the dataset REDDIT-MOVIE from Reddit platforms, serving as the largest-scale conversational recommendation dataset to date, from real discussion platforms, to facilitate the further research in the conversational recommendation field.

9.2 Future Directions

In addition to the directions explored in this dissertation, there are several potential research avenues or challenges that merit further investigation to work toward an ideal conversational recommender system:

1. *Fairness and Debiasing.* Fairness and debiasing have been a focus in traditional recommender systems, but remain underexplored in conversational contexts. An open question is whether systems proactively asking questions or collecting explicit textual feedback (e.g., critiques) from users can significantly alleviate biased issues or improve fairness.

2. *Evaluation with User Simulators.* With the advancement of large language models (LLMs), an open question is whether we can develop user simulators to serve as satisfactory surrogates for real users in conversational interactions. If so, these simulators could significantly change the future of conversational recommendation research by providing inexpensive and scalable evaluation.

3. *Resources and Benchmarking.* Although some contributions have provided resources for conversational recommendations, this field is still in its nascent stage. More datasets, benchmarks, and software libraries could greatly facilitate research by providing common platforms. Developing such research-oriented resources also poses interesting system-design challenges.

4. *Real-world Applications.* Conversational recommender systems in the real world face challenges like open-domain settings, distribution shifts, and adversarial attacks. Formulating such practical issues as research questions and contributing solutions is critical for advancing conversational recommendation research.

Bibliography

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *J. Mach. Learn. Res.*, 3:397–422, 2002.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The Semantic Web: 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007+ ASWC 2007, Busan, Korea, November 11-15, 2007. Proceedings*, pages 722–735. Springer, 2007.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Jinze Bai, Chang Zhou, Junshuai Song, Xiaoru Qu, Weiting An, Zhao Li, and Jun Gao. Personalized bundle list recommendation. *The World Wide Web Conference*, 2019.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. Ms marco: A human generated machine reading comprehension dataset, 2018.

Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. *arXiv preprint arXiv:2305.00447*, 2023.

Veronika Bogina and Tsvi Kuflik. Incorporating dwell time in session-based recommendations with recurrent neural networks. In *RecTemp@ RecSys*, pages 57–59, 2017.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901,

2020.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. Autoregressive entity retrieval. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=5k8F6UU39V.

William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. Kermit: Generative insertion-based modeling for sequences. *ArXiv*, abs/1906.01604, 2019.

Jianxin Chang, Chen Gao, Xiangnan He, Depeng Jin, and Yong Li. Bundle recommendation with graph convolutional networks. In *Proceedings of the 43rd international ACM SIGIR conference on Research and development in Information Retrieval*, pages 1673–1676, 2020.

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *NIPS*, 2011.

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

Jun Chen, Han Guo, Kai Yi, Boyang Li, and Mohamed Elhoseiny. Visualgpt: Data-efficient adaptation of pretrained language models for image captioning. 2021.

Li Chen and Feng Wang. Explaining recommendations based on feature sentiments in product reviews. pages 17–28, 2017.

Liang Chen, Yang Liu, Xiangnan He, Lianli Gao, and Zibin Zheng. Matching user with item set: Collaborative bundle recommendation with deep attention network. In *IJCAI*, 2019a.

Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. Towards knowledge-based recommender dialog system. 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)*, pages 1803–1813, 2019b.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020a.

Wen Chen, Pipei Huang, Jiaming Xu, Xin Ze Guo, Cheng Guo, Fei Sun, Chao Li, Andreas Pfadler, Huan Zhao, and Binqiang Zhao. Pog: Personalized outfit generation for fashion

recommendation at alibaba ifashion. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019c.

Xusong Chen, Dong Liu, Chenyi Lei, Rui Li, Zheng-Jun Zha, and Zhiwei Xiong. Bert4sessrec: Content-based video relevance prediction with bidirectional encoder representations from transformer. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2597–2601, 2019d.

Zhihong Chen, Yan Song, Tsung-Hui Chang, and Xiang Wan. Generating radiology reports via memory-driven transformer. *arXiv preprint arXiv:2010.16056*, 2020b.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric 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/.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014a.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014b.

Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. Towards conversational recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 815–824, 2016.

Alexander Dallmann, Alexander Grimm, Christian Pölitz, Daniel Zoller, and Andreas Hotho. Improving session recommendation with recurrent neural networks by exploiting dwell time. *arXiv preprint arXiv:1706.10231*, 2017.

Qilin Deng, Kai Wang, M. Zhao, Zhene Zou, Runze Wu, Jianrong Tao, Changjie Fan, and Liang Chen. Personalized bundle recommendation in online games. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020.

Qilin Deng, Kai Wang, Minghao Zhao, Runze Wu, Yu Ding, Zhene Zou, Yue Shang, Jianrong Tao, and Changjie Fan. Build your own bundle - a neural combinatorial optimization method. *Proceedings of the 29th ACM International Conference on Multimedia*, 2021a.

Yang Deng, Yaliang Li, Fei Sun, Bolin Ding, and Wai Lam. Unified conversational recommendation policy learning via graph-based reinforcement learning. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*,

2021b.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

Robin Devooght and Hugues Bersini. Long and short-term recommendations with recurrent neural networks. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, pages 13–21, 2017.

George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145, 2002.

Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. Learning to generate product reviews from attributes. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 623–632, 2017.

Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. Sequential user-based recurrent neural network recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 152–160, 2017.

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

Jingyue Gao, Xiting Wang, Yasha Wang, and Xing Xie. Explainable recommendation through attentive multi-view learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3622–3629, 2019.

Yingqiang Ge, Shuchang Liu, Zuohui Fu, Juntao Tan, Zelong Li, Shuyuan Xu, Yunqi Li, Yikun Xian, and Yongfeng Zhang. A survey on trustworthy recommender systems. *arXiv preprint arXiv:2207.12515*, 2022.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard S. Zemel, Wieland Brendel, Matthias Bethge, and Felix Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2:665 – 673, 2020.

Jiatao Gu, Qi Liu, and Kyunghyun Cho. Insertion-based decoding with automatically inferred generation order. *Transactions of the Association for Computational Linguistics*, 7:661–676, 2019a.

Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. volume 32, 2019b.

Arnav Gudibande, Eric Wallace, Charlie Snell, Xinyang Geng, Hao Liu, Pieter Abbeel, Sergey Levine, and Dawn Song. The false promise of imitating proprietary llms, 2023.

Maosheng Guo, Yu Zhang, and Ting Liu. Gaussian transformer: a lightweight approach for natural language inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6489–6496, 2019.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Xiangyang Xue, and Zheng Zhang. Multi-scale self-attention for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7847–7854, 2020.

Deepesh V Hada and Shirish K Shevade. Rexplug: Explainable recommendation using plug-and-play language model. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 81–91, 2021.

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

Shirley Anugrah Hayati, Dongyeop Kang, Qingxiaoyang Zhu, Weiyan Shi, and Zhou Yu. Inspired: Toward sociable recommendation dialog systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8142–8152, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

Ruining He and Julian McAuley. Fusing similarity models with markov chains for sparse sequential recommendation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 191–200. IEEE, 2016a.

Ruining He and Julian McAuley. Vbpr: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016b.

Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 355–364, 2017.

Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. Adversarial personalized ranking for recommendation. In *The 41st International ACM SIGIR conference on research & development in information retrieval*, pages 355–364, 2018.

Xingwei He. Parallel refinements for lexically constrained text generation with bart. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8653–8666, 2021.

Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley. Locker: Locally constrained self-attentive sequential recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3088–3092, 2021.

Zhankui He, Handong Zhao, Zhaowen Wang, Zhe Lin, Ajinkya Kale, and Julian Mcauley. Query-aware sequential recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4019–4023, 2022a.

Zhankui He, Handong Zhao, Tong Yu, Sungchul Kim, Fan Du, and Julian McAuley. Bundle mcr: Towards conversational bundle recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 288–298, 2022b.

Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 720–730, 2023.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021.

Balázs Hidasi and Alexandros Karatzoglou. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 843–852, 2018.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 241–248, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Chris Hokamp and Qun Liu. Lexically constrained decoding for sequence generation using grid

beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, 2017.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. Large language models are zero-shot rankers for recommender systems. *arXiv preprint arXiv:2305.08845*, 2023.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Haoji Hu and Xiangnan He. Sets2sets: Learning from sequential sets with neural networks. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.

J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850, 2019.

X Hua and L Wang. Sentence-level content planning and style specification for neural text generation. In *Conference on Empirical Methods in Natural Language Processing*, 2019.

Jiaji Huang, Yi Li, Wei Ping, and Liang Huang. Large margin neural language model. *arXiv preprint arXiv:1808.08987*, 2018a.

Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. Improving sequential recommendation with knowledge-enhanced memory networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 505–514, 2018b.

D. Jannach and Ahtsham Manzoor. End-to-end learning for conversational recommendation: A long way to go? In *IntRS@RecSys*, 2020.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, Graham Neubig, and Taylor Berg-Kirkpatrick. Learning to generate move-by-move commentary for chess games from large-scale social forum data. In *The 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Melbourne, Australia, July 2018.

C Kim Jacob Hilton Jacob Menick Jiayi Weng Juan Felipe Ceron Uribe Liam Fedus Luke Metz Michael Pokorny Rapha Gontijo Lopes Sengjia Zhao John Schulman, Barret Zoph. Chatgpt: Optimizing language models for dialogue. *OpenAI*, 2022.

Santosh Kabbur, Xia Ning, and George Karypis. Fism: factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 659–667, 2013.

Dongyeop Kang, Anusha Balakrishnan, Pararth Shah, Paul A Crook, Y-Lan Boureau, and Jason Weston. Recommendation as a communication game: Self-supervised bot-play for goal-oriented dialogue. 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)*, pages 1951–1961, 2019.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206. IEEE, 2018.

Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. Do llms understand user preferences? evaluating llms on user rating prediction. *arXiv preprint arXiv:2305.06474*, 2023.

Gary King and Langche Zeng. Logistic regression in rare events data. *Political Analysis*, 9:137 – 163, 2001.

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

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

Walid Krichene and Steffen Rendle. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1748–1757, 2020.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Found. Trends Mach. Learn.*, 5:123–286, 2012.

Seanie Lee, Dong Bok Lee, and Sung Ju Hwang. Contrastive learning with adversarial perturbations for conditional text generation. *arXiv preprint arXiv:2012.07280*, 2020.

Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 304–312, 2020a.

Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. Interactive path reasoning on graph for conversational recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery &*

data mining, pages 2073–2083, 2020b.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020.

Jiacheng Li, Yujie Wang, and Julian McAuley. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 322–330, 2020a.

Jiacheng Li, Jingbo Shang, and Julian McAuley. UCTopic: Unsupervised contrastive learning for phrase representations and topic mining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6159–6169, Dublin, Ireland, May 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.426. URL https://aclanthology.org/2022.acl-long.426.

Jiacheng Li, Zhankui He, Jingbo Shang, and Julian McAuley. Ucepic: Unifying aspect planning and lexical constraints for generating explanations in recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1248–1257, 2023a.

Jian Li, Zhaopeng Tu, Baosong Yang, Michael R Lyu, and Tong Zhang. Multi-head attention with disagreement regularization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2897–2903, 2018a.

Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1419–1428, 2017a.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*, 2015.

Junyi Li, Wayne Xin Zhao, Ji-Rong Wen, and Yang Song. Generating long and informative reviews with aspect-aware coarse-to-fine decoding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1969–1979, 2019.

Junyi Li, Siqing Li, Wayne Xin Zhao, Gaole He, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. Knowledge-enhanced personalized review generation with capsule graph neural network. pages 735–744, 2020b.

Junyi Li, Wayne Xin Zhao, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. Knowledge-based review generation by coherence enhanced text planning. pages 183–192, 2021a.

Lei Li, Yongfeng Zhang, and Li Chen. Generate neural template explanations for recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 755–764, 2020c.

Lei Li, Yongfeng Zhang, and Li Chen. Personalized transformer for explainable recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4947–4957, 2021b.

Lei Li, Yongfeng Zhang, and Li Chen. Personalized prompt learning for explainable recommendation. *ACM Transactions on Information Systems*, 41(4):1–26, 2023b.

Ming Li, Sami Jullien, Mozhdeh Ariannezhad, and Maarten de Rijke. A next basket recommendation reality check. *ACM Transactions on Information Systems*, 41(4):1–29, 2023c.

Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 345–354, 2017b.

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. Towards deep conversational recommendations. *Advances in neural information processing systems*, 31, 2018b.

Shihao Li, Dekun Yang, and Bufeng Zhang. Mrif: Multi-resolution interest fusion for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1765–1768, 2020d.

Shuokai Li, Ruobing Xie, Yongchun Zhu, Xiang Ao, Fuzhen Zhuang, and Qing He. User-centric conversational recommendation with multi-aspect user modeling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 223–233, 2022b.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan, November 2017c. Asian Federation of Natural Language Processing. URL https://aclanthology.org/I17-1099.

Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 world wide web conference*, pages 689–698, 2018.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization*

branches out, pages 74–81, 2004.

Jing Lin, Weike Pan, and Zhong Ming. Fissa: fusing item similarity models with self-attention networks for sequential recommendation. In *Fourteenth ACM Conference on Recommender Systems*, pages 130–139, 2020.

Chang Liu, Xiaoguang Li, Guohao Cai, Zhenhua Dong, Hong Zhu, and Lifeng Shang. Noninvasive self-attention for side information fusion in sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4249–4256, 2021.

Hugo Liu and Push Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. Is chatgpt a good recommender? a preliminary study, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Yichao Lu, Ruihai Dong, and Barry Smyth. Why i like it: multi-task learning for recommendation and explanation. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 4–12, 2018.

Yu Lu, Junwei Bao, Yan Song, Zichen Ma, Shuguang Cui, Youzheng Wu, and Xiaodong He. Revcore: Review-augmented conversational recommendation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1161–1173, 2021.

Chen Ma, Peng Kang, and Xue Liu. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 825–833, 2019.

Wenchang Ma, Ryuichi Takanobu, and Minlie Huang. CR-walker: Tree-structured graph reasoning and dialog acts for conversational recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, November 2021. URL https://aclanthology.org/2021.emnlp-main.139.

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

Julian John McAuley and Jure Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. pages 897–908, 2013.

Wenjing Meng, Deqing Yang, and Yanghua Xiao. Incorporating user micro-behaviors and item knowledge into multi-task learning for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1091–1100, 2020.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842, 2019.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. Step-by-step: Separating planning from realization in neural data-to-text generation. pages 2267–2277, 2019.

Jianmo Ni and Julian McAuley. Personalized review generation by expanding phrases and attending on aspect-aware representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 706–711, 2018.

Jianmo Ni, Zachary C Lipton, Sharad Vikram, and Julian McAuley. Estimating reactions and recommending products with generative models of reviews. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 783–791, 2017.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. 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)*, pages 188–197, 2019.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

OpenAI. Gpt-4 technical report, 2023.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing 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.

Apurva Pathak, Kshitiz Gupta, and Julian McAuley. Generating and personalizing bundle recommendations on steam. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017.

Gustavo Penha and Claudia Hauff. What does bert know about books, movies and music? probing bert for conversational recommendation. In *Proceedings of the 14th ACM Conference on Recommender Systems*, pages 388–397, 2020.

Matt Post and David Vilar. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

Zhaochun Ren, Zhi Tian, Dongdong Li, Pengjie Ren, Liu Yang, Xin Xin, Huasheng Liang, Maarten de Rijke, and Zhumin Chen. Variational reasoning about user preferences for conversational recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 165–175, 2022.

Steffen Rendle. Factorization machines. In *2010 IEEE International conference on data mining*, pages 995–1000. IEEE, 2010.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461, 2009.

Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 811–820, 2010.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. Lamp: When large language models meet personalization. *arXiv preprint arXiv:2304.11406*, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th international conference on World*

Wide Web, pages 111–112, 2015.

Younghun Song and Jae-Gil Lee. Augmenting recurrent neural networks with high-order user-contextual preference for session-based recommendation. *arXiv preprint arXiv:1805.02983*, 2018.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pages 5976–5985. PMLR, 2019.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1441–1450, 2019.

Yueming Sun and Yi Zhang. Conversational recommender system. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. volume 27, 2014.

Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.

Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 565–573, 2018.

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35:21831–21843, 2022.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017a. URL https://arxiv.org/pdf/1706.03762.pdf.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017b.

Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. Learning intents behind interactions with knowledge graph for recommendation. *Proceedings of the Web Conference 2021*, 2021.

Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1929–1937, 2022.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua. Contrastive learning for cold-start recommendation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 5382–5390, 2021.

Sean Welleck, Kianté Brantley, Hal Daumé Iii, and Kyunghyun Cho. Non-monotonic sequential text generation. In *International Conference on Machine Learning*, pages 6716–6726. PMLR, 2019.

Mark Wilhelm, Ajith Ramanathan, Alexander Bonomo, Sagar Jain, Ed H. Chi, and Jennifer Gillenwater. Practical diversified recommendations on youtube with determinantal point processes. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018.

Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. Stochastic shared embeddings: data-driven regularization of embedding layers. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 24–34, 2019.

Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. Sse-pt: Sequential recommendation via personalized transformer. In *Fourteenth ACM Conference on Recommender Systems*, RecSys '20, page 328–337, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450375832. doi: 10.1145/3383313.3412258. URL https://doi.org/10.1145/3383313.3412258.

Yikun Xian, Handong Zhao, Tak Yeon Lee, Sungchul Kim, Ryan A. Rossi, Zuohui Fu, Gerard de Melo, and S. Muthukrishnan. Exacta: Explainable column annotation. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021.

Zhihui Xie, Tong Yu, Canzhe Zhao, and Shuai Li. Comparison-based conversational recommender system with relative bandit feedback. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *arXiv preprint arXiv:2304.01196*, 2023.

Ke Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

An Yan, Zexue He, Xing Lu, Jiang Du, Eric Chang, Amilcare Gentili, Julian McAuley, and Chun-Nan Hsu. Weakly supervised contrastive learning for chest x-ray report generation. *arXiv preprint arXiv:2109.12242*, 2021.

An Yan, Zhankui He, Jiacheng Li, Tianyang Zhang, and Julian McAuley. Personalized showcases: Generating multi-modal explanations for recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2251–2255, 2023a.

An Yan, Zhankui He, Jiacheng Li, Tianyang Zhang, and Julian McAuley. Personalized showcases: Generating multi-modal explanations for recommendations. In *SIGIR*, 2023b.

Baosong Yang, Zhaopeng Tu, Derek F Wong, Fandong Meng, Lidia S Chao, and Tong Zhang. Modeling localness for self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4449–4458, 2018.

Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. A simple convolutional generative network for next item recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 582–590, 2019.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ByxRM0Ntvr.

Hongyu Zang and Xiaojun Wan. Towards automatic generation of product reviews from aspect-sentiment scores. In *INLG*, 2017.

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26:1819–1837, 2014.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore:

Evaluating text generation with bert. *ICLR*, 2020a.

Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. Feature-level deeper self-attention network for sequential recommendation. In *IJCAI*, pages 4320–4326, 2019.

Xiaoying Zhang, Hong Xie, Hang Li, and John C.S. Lui. Conversational contextual bandit: Algorithm and application. *Proceedings of The Web Conference 2020*, 2020b.

Yiming Zhang, Lingfei Wu, Qi Shen, Yitong Pang, Zhihua Wei, Fangli Xu, Bo Long, and Jian Pei. Multiple choice questions based multi-interest policy learning for conversational recommendation. In *Proceedings of the ACM Web Conference 2022*, pages 2153–2162, 2022.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. Dialogpt: Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, 2020c.

Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and William B Dolan. Pointer: Constrained progressive text generation via insertion-based generative pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8649–8670, 2020d.

Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 83–92, 2014.

Yongfeng Zhang, X. Chen, Qingyao Ai, Liu Yang, and W. Croft. Towards conversational search and recommendation: System ask, user respond. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018.

Yongfeng Zhang, Xu Chen, et al. Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval*, 14(1):1–101, 2020e.

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

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1893–1902, 2020a.

Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1006–1014, 2020b.

Kun Zhou, Yuanhang Zhou, Wayne Xin Zhao, Xiaoke Wang, and Ji-Rong Wen. Towards topic-guided conversational recommender system. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4128–4139, 2020c.

Wanrong Zhu, Xin Eric Wang, An Yan, Miguel Eckstein, and William Yang Wang. Imagine: An imagination-based automatic evaluation metric for natural language generation. *arXiv preprint arXiv:2106.05970*, 2021.

Jie Zou, Evangelos Kanoulas, Pengjie Ren, Zhaochun Ren, Aixin Sun, and Cheng Long. Improving conversational recommender systems via transformer-based sequential modelling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2319–2324, 2022.