Repetition and Exploration in Recommendation

Ming Li

Repetition and Exploration in Recommendation

Academisch Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in
de Agnietenkapel
op 21 februari 2024, te 11:00 uur

door

Ming Li

geboren te Shandong

Promotiecommissie

Promotor:	prof.dr. Maarten de Rijke	Universiteit van Amsterdam
Co-promotor:	dr. Andrew Yates	Universiteit van Amsterdam
Overige leden:	dr. Mohammad Aliannejadi	Universiteit van Amsterdam
	prof.dr. Paul Groth	Universiteit van Amsterdam
	dr. Claudia Hauff	Spotify
	prof.dr. Evangelos Kanoulas	Universiteit van Amsterdam
	prof.dr. Min Zhang	Tsinghua University

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

The research was supported by the China Scholarship Council.

Cover by Kexin Liang
Printed by Off Page, Amsterdam

ISBN: 978-94-93278-73-8

Acknowledgements

As I conclude my journey as a student, completing a Ph.D. emerges as a significant and fortunate chapter in this adventure. Over the last 3 years and 2 months, it has been a remarkable and enjoyable experience.

Maarten, I am profoundly fortunate to have you as my supervisor. Thank you for bringing me to Amsterdam from Delft to restart my Ph.D. during the COVID pandemic. I learned everything from scratch; thank you for your patience and guidance during my studies. I am also impressed by your passion and attitude toward conducting research. Special thanks for consistently reminding me to "pay attention to details!"

Andrew, thank you for being my co-promotor and my friend. Your expertise and insights have been instrumental in shaping my Ph.D. research. I am deeply thankful for your guidance and support in navigating my career!

Claudia, Evangelos, Min, Mohammad, and Paul, it is my honor to have you as my PhD committee. Thank you for the precious time you spent time to examine my thesis!

IRLab, I feel fortunate to have conducted research in such an exceptional environment. My sincere thanks to all the brilliant IRLabers who enriched my experience during my time there: Ali, Amin, Ana, Andrew, Antonis, Arezoo, Arian, Barrie, Chang, Chuan, Clara, Clemencia, Dan, David, Gabriel, Gabrielle, Georgios, Evangelos, Harrie, Hongyi, Ilias, Ilya, Ivana, Jiahuan, Jie, Jin, Jingfen, Jingwei, Julien, Justine, Kidist, Maarten M, Maartje, Maria, Mariya, Maurits, Mohammad, Mounia, Negin, Olivier, Panagiotis, Pablo, Petra, Philipp, Pooya, Romain, Roxana, Ruben, Ruqing, Saedeh, Sam, Sami, Sebastian, Shaojie, Shashank, Shubha, Simon, Songgaojun, Svitlana, Thilina, Thong, Vaishali, Vera, Weijia, Xinyi, Yang, Yangjun, Yibin, Yifei, Yongkang, Yuanna, Yuanxing, Yuyue, Zahra, Zihan, Ziyi, and Ziming. Thank you all!

Ali, Jin, Mozhdeh, Sami, Songgaojun, Yibin, and Yuanna, thank you for the inspiring discussions and fruitful collaborations!

Ali, Jin, Ruqing, and Yuyue, thank you for being my office mates in Lab42! I enjoyed the chit-chats and laughter that went beyond research!

Kexin, you have added immense meaning to this journey. Your support has been my source of joy and strength, and I couldn't have reached this point without you. Together, we have explored the world and planned for countless adventures. Thank you, my love!

最后, 感谢我的祖国，感谢父母，爷爷奶奶，姥爷姥姥一如既往的鼓励与支持！

Ming Li
Amsterdam
January 2024

Contents

1 Introduction **1**
- 1.1 Research Outline and Questions . . . 2
- 1.2 Main Contributions . . . 5
- 1.3 Thesis Overview . . . 7
- 1.4 Origins . . . 8

2 Repetition and Exploration in Next Basket Recommendation **11**
- 2.1 Introduction . . . 11
  - 2.1.1 Types of recommendation methods . . . 12
  - 2.1.2 A new analysis perspective . . . 12
  - 2.1.3 Main findings . . . 13
- 2.2 Related Work . . . 14
  - 2.2.1 Reproducibility in information retrieval . . . 14
  - 2.2.2 Next basket recommendation . . . 14
- 2.3 Experimental Setup . . . 15
  - 2.3.1 Datasets . . . 15
  - 2.3.2 Baseline methods and reproducible methods . . . 16
  - 2.3.3 Implementation details . . . 19
- 2.4 Performance Comparison Using Conventional Metrics . . . 19
  - 2.4.1 Conventional next basket recommendation metrics . . . 19
  - 2.4.2 Results with conventional next basket recommendation metrics . . . 20
  - 2.4.3 Upshot . . . 22
- 2.5 Performance w.r.t. Repetition and Exploration . . . 22
  - 2.5.1 Metrics for repetition vs. exploration . . . 23
  - 2.5.2 The components of a recommended basket . . . 24
  - 2.5.3 Performance w.r.t. repetition and exploration . . . 27
  - 2.5.4 The relative contribution of repetition and exploration . . . 29
  - 2.5.5 Treatment effect for users with different repetition ratios . . . 31
  - 2.5.6 Looking beyond the average performance . . . 33
  - 2.5.7 Treatment effect for items with different frequencies . . . 34
  - 2.5.8 Upshot . . . 36
- 2.6 Conclusion . . . 38
  - 2.6.1 Main findings . . . 38
  - 2.6.2 Insights for NBR model evaluation . . . 38
  - 2.6.3 Insights for NBR model design . . . 39
  - 2.6.4 Limitations . . . 39
  - 2.6.5 Future work . . . 40
- 2.7 Appendix . . . 40
  - 2.7.1 Additional plots . . . 40
  - 2.7.2 Reproducibility . . . 40

3 Next Novel Basket Recommendation **45**
- 3.1 Introduction . . . 45

3.1.1 Next novel basket recommendation . . . 45
3.1.2 From NBR to NNBR . . . 47
3.1.3 BTBR: Bi-directional transformer basket recommendation 47
3.1.4 Masking and training . . . 47
3.1.5 Main contributions . . . 48
3.2 Related Work . . . 48
3.2.1 Sequential recommendation . . . 49
3.2.2 Next basket recommendation . . . 49
3.3 Problem Formulation . . . 50
3.4 Our Method . . . 50
3.4.1 Bi-directional transformer basket recommendation model 50
3.4.2 Masking strategy . . . 52
3.4.3 Swapping strategy . . . 55
3.5 Experimental Setup . . . 56
3.5.1 Research questions . . . 56
3.5.2 Datasets. . . . 56
3.5.3 Baselines . . . 57
3.5.4 Configurations . . . 57
3.5.5 Metrics . . . 58
3.6 Experimental Results . . . 58
3.6.1 Train-all and Train-explore . . . 58
3.6.2 Effectiveness of BTBR . . . 60
3.6.3 Effectiveness of the item swapping strategy . . . 62
3.6.4 Effect of mask ratio and training dynamics . . . 63
3.6.5 Effectiveness of joint masking . . . 65
3.7 Conclusion . . . 66
3.7.1 Main findings . . . 66
3.7.2 Implications . . . 67
3.7.3 Limitations . . . 67

4 Reverse Next-Period Recommendation 69
4.1 Introduction . . . 69
4.1.1 Item-centered recommendations . . . 70
4.1.2 User-centered methods for item-centered tasks . . . 71
4.1.3 Repetition vs. exploration . . . 71
4.1.4 Efficiency . . . 72
4.1.5 Main contributions . . . 72
4.2 Related Work . . . 73
4.2.1 User-centered recommendation . . . 73
4.2.2 Item-centered recommendations . . . 74
4.3 Problem Formulation . . . 74
4.3.1 Reverse next-period recommendation . . . 76
4.3.2 Candidate filtering . . . 77
4.4 Repetition Analysis . . . 77
4.5 Reverse Next-Period Recommendation . . . 78
4.5.1 Habit-interest fusion model for Expl-RNPR . . . 79

4.5.2 Time-aware frequency model for Rep-RNPR 83
4.5.3 The REUR algorithm for Mixed-RNPR 84
4.6 Candidate Filtering 85
4.6.1 Repetition-rule based candidate filtering 86
4.6.2 Model-based candidate filtering 86
4.7 Experimental Setup 88
4.7.1 Research questions 88
4.7.2 Datasets 89
4.7.3 Baselines 90
4.7.4 Metrics 91
4.7.5 Configurations 92
4.8 Experimental Results 92
4.8.1 Performance comparison of "user-centered" methods . . . 92
4.8.2 Performance of our proposed methods 94
4.8.3 Ablation study of HIF 98
4.8.4 Trade-off analysis for Mixed-RNPR 99
4.8.5 Candidate filtering 100
4.9 Conclusion 105
4.9.1 Main findings 105
4.9.2 Limitations 106
4.9.3 Future work 106

5 Sequential Item Recommendation 107
5.1 Introduction 107
5.1.1 Repetition and exploration 108
5.1.2 Sequential recommendation: A user-centered perspective . 108
5.1.3 Sequential recommendation: An item-centered perspective 108
5.1.4 Repetition "shortcuts" and inherent repetitive bias 109
5.1.5 Main contributions 110
5.2 Related Work 110
5.2.1 Sequential recommendation 110
5.2.2 Accuracy 111
5.2.3 Beyond accuracy 111
5.3 Problem Formulation and Definitions 112
5.3.1 Sequential recommendation task 112
5.3.2 Repeat vs. explore 112
5.3.3 Explore exposure vs. repeat exposure 112
5.4 Experimental Setup 113
5.4.1 Research questions 113
5.4.2 Datasets 114
5.4.3 Methods 115
5.4.4 Configurations 115
5.4.5 Metrics 116
5.5 Repetition Accuracy and Exploration Accuracy 116
5.6 Explore Exposure and Repeat Exposure 119
5.6.1 Importance of item explore exposure 119

5.6.2 Less/zero explore exposure issue 120
5.7 Repetitive Bias 123
5.8 Conclusion 127
5.8.1 Main findings 128
5.8.2 Implications 128
5.8.3 Limitations and future work 128

6 Conclusions **131**
6.1 Main Findings 131
6.2 Future Directions 133
6.2.1 Other recommendation scenarios 134
6.2.2 Evaluation 134

Bibliography **137**

Summary **145**

Samenvatting **147**

1 Introduction

Recommender systems have become an essential component of many online platforms [131]. Recommender systems can help users find relevant items that meet their preferences. They can also help items to get exposed to potential users. When users interact with a recommender system, they leave a trace of their behavior on the platform, which can be collected and used by the recommender system to infer the users' potential preferences [37, 85, 110]. In this thesis, we focus on recommendation scenarios, where the users' preferences may change over time.

Over the years, advances in deep learning have led to the development of numerous recommendation models that employ deep learning techniques [49, 58, 62, 65, 73, 88, 89, 92, 96, 101, 117, 125, 127]. These approaches have often focused on the goal of increasing overall accuracy. However, they neglect an essential aspect of user behavior: *repetition* and *exploration*. People often have regular habits and display repetition behavior when they interact with the platform [3, 12, 19]. That is, users may repurchase the same item, re-listen to the same song, etc. Apart from repetition behavior, people also explore and discover new and diverse items on a recommendation platform: the discovery of something new can be a great source of joy and user satisfaction [47, 57]. Understanding the dynamics of repetition and exploration is crucial for a more rigid evaluation and developing more accurate and effective recommender systems. Thus, in this thesis, we study the repetition and exploration through four types of recommendation problems in Figure 1.1, and focus on two aspects: *evaluation* and *optimization*.

To start with, we consider the scenario of next basket recommendation (NBR) in grocery shopping, where repetition behavior is prevalent. We provide a novel angle on the evaluation of NBR methods, centered on the distinction between repetition and exploration: the next basket for a user is typically composed of previously consumed items (i.e., repeat items) and items that are new to the user (i.e., explore items).

Next, always recommending previously purchased items has the risk of reducing user interest [3], thus, an important goal of recommendation is to help users discover new items they have not purchased before. We isolate the exploration task from conventional NBR and define the next novel basket recommen-

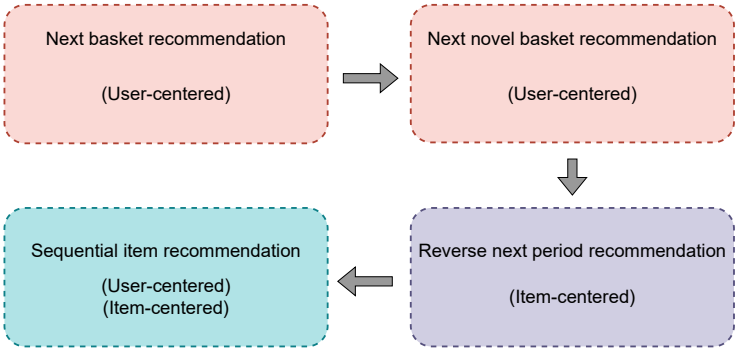


Figure 1.1: Four recommendation problems studied in this thesis.

dation (NNBR) task. Besides, it is worth thinking about the potential impact of the repetition signals on the models' exploration recommendation performance. Thus, we investigate methods for enhancing exploration recommendation in the presence of repetition signals.

Then, expanding our scope of research, we look into an item-centered recommendation scenario, aiming to assist items in finding their potential users. We introduce the reverse next-period recommendation (RNPR) task, analyze repetition behaviors at both the item and category levels, and propose item-centered models and training strategies using the insights obtained from the repetition analysis.

Finally, we shift our focus to the broader task of sequential recommendation (SR) and item-side exposure evaluation. Similar to the NBR scenario, most previous research neglects the difference between the repetition task and the exploration task, and focuses on increasing the overall accuracy. The notion of item exposure is used to measure how items get exposed to the users, which has become an important factor that models need to consider, as items and producers are important participants within a recommender system. However, existing research w.r.t. item exposure is mostly focused on individual or group fairness and does not consider different types of users. Therefore, beyond offering analyses of repetition and exploration behavior, we propose novel evaluation metrics, namely, item explore exposure and item repeat exposure, to analyze the allocation of exposure.

In this thesis, we gain insights into the role of repetition and exploration in recommender systems. Our findings emphasize the importance of evaluating and optimizing recommendation models from repetition and exploration perspectives.

1.1 Research Outline and Questions

Throughout this thesis, we attempt to understand and utilize *repetition and exploration in recommendation* from several aspects. We mainly focus on two re-

search themes: (i) *evaluating* recommendation performance, and (ii) *optimizing* and *designing* recommendation methods. We scope our research by answering the following four main research questions across four chapters.

We start by investigating the recommendation in the grocery shopping scenario, where people have regular habits and repetition behavior. Specifically, we focus on the next basket recommendation task, which recommends a set of items (the next basket) based on the user's historical baskets. Previous methods for next basket recommendation only focus on designing models to achieve better overall performance but fail to realize that the next basket is typically composed of previously consumed items (i.e., repeat items) and new items (i.e., explore items). So, we formulate our first research question as follows:

RQ1 *How to evaluate the next basket recommendation performance from the perspective of repetition and exploration?*

To answer this question, we first perform an analysis of the overall performance of state-of-the-art NBR methods on datasets with different levels of repetition behavior. We find that no NBR method is able to consistently achieve the best performance in terms of average accuracy across different datasets. Next, we dive deeper into analyzing the components of the recommended basket, and propose a set of evaluation metrics that measure the basket components and the performance in terms of a repetition recommendation task (i.e., recommending repeat items) and an exploration recommendation task (i.e., recommending explore items), respectively. Using the proposed new metrics, we provide a second analysis of state-of-the-art NBR methods. We identify a substantial imbalance in difficulty between the repetition recommendation task and the exploration recommendation task. Being biased towards the easier repetition task is an important strategy that helps to boost the overall NBR performance. Furthermore, this analysis also examines the NBR performance from various aspects (e.g., limitations of overall performance, treatment effect, etc.) and points out several important guidelines that researchers working on NBR should follow when designing or evaluating an NBR model.

Naturally, people might simply get tired of repurchasing the same set of items and one important goal of a recommender system is to help users discover new items that meet their preferences. Therefore, we formulate our second research question as follows:

RQ2 *How to design basket recommendation models targeted at the exploration task, and how to optimize the model to explore items in a scenario with many repetition signals?*

To answer this question, we first isolate the exploration task from the conventional NBR task and formulate the next novel basket recommendation (NNBR) task, which focuses on recommending a novel basket, i.e., a set of items that are new to a given user. Next, we investigate the performance of several representative NBR methods w.r.t. the NNBR task and find that training specifically for the exploration task (i.e., removing repeat items in the training labels) does not

always lead to better performance. We propose a simple bi-directional transformer basket recommendation (BTBR) model that learns item-to-item correlations across baskets. A straightforward approach to training the BTBR model involves randomly masking the items in the basket sequence and then attempting to predict these masked items. We identify one potential issue w.r.t. the random masking strategy, i.e., masked items (prediction target) might still exist in the non-masked positions, so the model might mainly predict the masked item via its repetition information rather than inferring new items based on item-to-item correlations. To this end, we propose and investigate several types of masking strategy with different methods for dealing with the repetition signal. We also consider flexible orders of items across baskets in a grocery shopping scenario and propose item-swapping strategies to enrich item-to-item correlations. Our experimental results validate the effectiveness of the proposed BTBR model and the proposed strategies.

Now, the above two research questions focus on the user-centered recommendation tasks, that help users find their preferred items. Another potential application of the recommender system is to turn this around and help an item find its potential users. Our next research question turns to this type of item-centered recommendation.

RQ3 *How to help a given item find its potential users in an item-centered setting, and how do repetition and exploration influence the design and optimization of the recommendation model?*

To answer this question, we first formulate a novel "item-centered" recommendation task in a sequential setting, namely reverse next-period recommendation (RNPR). Considering different types of target users, we define three subtasks for RNPR, i.e., Expl-RNPR, Rep-RNPR and Mixed-RNPR. Before designing models, we first analyze the users' repetition behaviors on both item- and category-levels, and we find that category-level repetition behavior is more stable than item-level. Benefiting from this finding, we propose several item-centered models, i.e., the habit-interest fusion (HIF) model for the Expl-RNPR task and the repetition-exploration ranking user (REUR) algorithm to investigate the trade-off within the Mixed-RNPR task. We propose two candidate filtering strategies, i.e., repetition rule-based and using a candidate filtering model. Besides, we propose two ways of constructing training samples for the HIF model. Our experimental results demonstrate the effectiveness of the proposed methods and validate the importance of considering the repetition and exploration in an item-centered recommendation setting.

Until now, we have mainly focused on recommendations in the grocery shopping domain. Next, we switch to a more general recommendation task, i.e., sequential recommendation, and focus on the following research question:

RQ4 *How do sequential recommendation models perform, and how should we evaluate item exposure from the perspective of repetition and exploration?*

To answer this question, we first analyze the accuracy of sequential recommendation (SR) models through the lens of repeat and explore items. We confirm

that the imbalance in performance and difficulty between the repetition task and exploration task known from previous NBR tasks also exists in the SR scenario. We point out evaluation issues of only using the overall average performance (in terms of accuracy) with a significance test. Next, we propose *item explore exposure* and *item repeat exposure* as metrics to analyze the exposure allocation at a more fine-grained level and demonstrate the importance of considering item explore exposure. We find that several state-of-the-art SR models suffer from the problem of zero/few item explore exposure, i.e., an item might be only exposed to users who have purchased it before. Surprisingly, we find that even in a pure exploration scenario, SR models still recommend repeat items to the user. We identify this issue as inherent repetitive bias. We find that the widely used shared embeddings in the prediction layer will increase repetitive bias and propose a *remove repeat items rule* (3R strategy) to address this issue in a pure exploration scenario.

Overall, by taking a deep dive into repetition and exploration from different aspects across different tasks, our work in this thesis sheds light on the evaluation of recommendations and yields useful insights for the design of recommendation models.

1.2 Main Contributions

In this section, we summarize the main contributions of this thesis in three groups:

Theoretical contributions.

- A methodology to evaluate the next basket recommendation performance from the perspective of repetition and exploration (Chapter 2).
- A new task, *next novel basket recommendation* (NNBR), which focuses on the exploration recommendation task in the next basket recommendation scenario (Chapter 3).
- A new task, *reverse next period recommendation* (RNPR), which focuses on the item-side recommendation, together with a taxonomy of RNPR tasks according to different types of target users from the perspective of repetition and exploration (Chapter 4).
- A methodology to evaluate the item exposure in the setting of sequential recommendation from the perspective of repetition and exploration (Chapter 5).

Algorithmic contributions.

- A bi-directional transformer basket recommendation (BTBR) model for the NNBR task, which directly models item-to-item correlations across baskets (Chapter 3).

- A group of training and masking methods to effectively train the BTBR model for the NNBR task, which considers different tasks and different training signals (Chapter 3).
- An item swapping method to train the BTBR model for the NNBR task, which considers flexible orders in the grocery shopping scenario (Chapter 3).
- A habit-interest fusion (HIF) model for the Expl-RNPR task, which leverages high-level (category) repetition behavior to help low-level (item) exploration tasks (Chapter 4).
- Two training sample construction strategies for the habit-interest fusion model, which consist of a positive sample augmentation strategy and a negative sample adjustment strategy (Chapter 4).
- A repetition-exploration user ranking algorithm for the Mixed-RNPR task, which decouples the repetition and exploration tasks and investigates their trade-off in an item-centered setting (Chapter 4).
- Two repetition rule-based candidate filtering methods (i.e., RRBF-item and RRBF), which help to reduce the computational costs of the RNPR task (Chapter 4).
- A model-based candidate filtering method to further reduce the computational costs of RNPR task, which predicts whether a user likes to repurchase a category or not (Chapter 4).
- A method, *remove repeat items rule* (3R strategy), which can mitigate the repetitive bias issue in a pure exploration scenario (Chapter 5).

Empirical contributions.

- An empirical evaluation of state-of-the-art NBR methods from the perspective of repetition and exploration, which covers the following seven aspects: (i) overall performance on different scenarios with different levels of repetition behavior; (ii) components of the recommended basket; (iii) repetition and exploration performance; (iv) the contribution of repetition and exploration to the overall performance; (v) the treatment effect for different user groups; (vi) the potential limitations of the use of averaged metrics; and (vii) the treatment effect for different items (Chapter 2).
- An empirical comparison of training existing NBR method for NNBR tasks using different training strategies, i.e., Train-all and Train-explore (Chapter 3).
- An empirical comparison of training the BTBR model using five masking strategies, i.e., item-level random masking, item-level select masking, basket-level all masking, basket-level explore masking, and joint masking (Chapter 3).

- An empirical evaluation of training the BTBR model using an item-swapping strategy (Chapter 3).
- An empirical analysis of users' repetition behavior on different levels from both a user and item perspective (Chapter 4).
- An empirical evaluation of item-centered NBR methods on the RNPR task (Chapter 4).
- An empirical comparison of the HIF model and REUR algorithm against baselines on the RNPR task (Chapter 4).
- An empirical analysis of the trade-off between repetition and exploration in RNPR task (Chapter 4).
- An empirical evaluation of the rule-based candidate filtering methods and model-based candidate filtering methods (Chapter 4).
- An empirical evaluation of representative SR methods from repetition and exploration perspective, which covers both user-centered perspective and item-centered perspective (Chapter 5).
- An empirical analysis of repetitive bias from two aspects: (i) repetition "shortcuts", and (ii) shared embeddings and independent embeddings used in the prediction layer (Chapter 5).
- An empirical comparison of SR models with the 3R strategy against without the 3R strategy in a pure exploration scenario (Chapter 5).

1.3 Thesis Overview

In this thesis, we focus on *understanding* recommendation evaluation from a repetition and an exploration perspective, and *optimizing* the recommendation algorithms according to our findings.

In Chapter 2, we focus on the next basket recommendation task in the context of grocery shopping. We propose several evaluation metrics and systematically analyze existing next basket recommendation models through the lenses of repetition and exploration.

In Chapter 3, we focus on the exploration recommendation task in a scenario with lots of repetition signals. We formulate the next novel basket recommendation task, and propose a transformer-based basket recommendation model for this specific task. We also design and investigate various training strategies to enhance the model's exploration ability.

In Chapter 4, we focus on an item-centered recommendation scenario and formulate the reverse next-period recommendation task, which aims to assist items in identifying potential users who are likely to purchase them in the near future. Again, considering repetition and exploration, we analyze users' behavior on both the category level and the item level, from both the item

side and the user side. Based on the findings derived from the analysis, we propose several sub-tasks and design corresponding recommendation models and training strategies for different sub-tasks.

In Chapter 5, we focus on a more general and extensively studied recommendation task, i.e., sequential recommendation. We perform a systematic analysis of several representative sequential recommendation models. Apart from the accuracy analysis, we also analyze item exposure from the perspective of repetition and exploration, and identify two important issues, i.e., zero/less explore exposure and repetitive bias.

In Chapter 6, we give a summary of the thesis, discuss limitations, and provide insights and future directions around this research.

1.4 Origins

The materials in this thesis come from the following publications:

Chapter 2 is based on the following paper:

- M. Li, S. Jullien, M. Ariannezhad, and M. de Rijke. A next basket recommendation reality check. *ACM Transactions on Information Systems*, 41(4):Article 116, 2023.

ML designed the evaluation metrics, conducted the experiments, and analyzed the results. All authors contributed to the conception and writing.

Chapter 3 is based on the following paper:

- M. Li, M. Ariannezhad, A. Yates, and M. de Rijke. Masked and swapped sequence modeling for next novel basket recommendation in grocery shopping. In *RecSys 2023: 17th ACM Conference on Recommender Systems*, pages 35–46. ACM, September 2023.

ML designed the algorithms, conducted the experiments, and analyzed the results. All authors contributed to the conception and writing.

Chapter 4 is based on the following paper:

- M. Li, M. Ariannezhad, A. Yates, and M. de Rijke. Who will purchase this item next? Reverse next period recommendation in grocery shopping. *ACM Transactions on Recommender Systems*, 1(2):Article 10, 2023.

ML designed the algorithms, conducted the experiments, and analyzed the results. All authors contributed to the conception and writing.

Chapter 5 is based on the following paper:

- M. Li, A. Vardasbi, A. Yates, and M. de Rijke. Repetition and exploration in sequential recommendation. In *SIGIR 2023: 46th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2532–2541. ACM, July 2023.

ML designed the evaluation metrics, conducted the experiments, and analyzed the results. All authors contributed to the conception and writing.

The thesis also benefited from work on the following publications:

- S. Deng, O. Sprangers, M. Li, S. Schelter, and M. de Rijke. Domain generalization in time series forecasting. *ACM Transactions on Knowledge Discovery from Data*, To appear.
- Y. Liu, M. Li, M. Ariannezhad, M. Mansoury, M. Aliannejadi, and M. de Rijke. Measuring item fairness in next basket recommendation: A reproducibility study. In *ECIR 2024: 46th European Conference on Information Retrieval*. Springer, April 2024.
- M. Li, J. Huang, and M. de Rijke. Repetition and exploration in offline reinforcement learning-based recommendations. In *DRL4IR workshop@CIKM 2023*, October 2023.
- M. Ariannezhad, M. Li, S. Jullien, and M. de Rijke. Tutorial: Complex item set recommendation. In *SIGIR 2023: 46th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3444–3447. ACM, July 2023.
- M. Ariannezhad, M. Li, S. Schelter, and M. de Rijke. A personalized neighborhood-based model for within-basket recommendation in grocery shopping. In *WSDM 2023: The Sixteenth International Conference on Web Search and Data Mining*, pages 87–95. ACM, February 2023.
- M. Ariannezhad, S. Jullien, M. Li, M. Fang, S. Schelter, and M. de Rijke. ReCANet: A repeat consumption-aware neural network for next basket recommendation in grocery shopping. In *SIGIR 2022: 45th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1240–1250. ACM, July 2022.

2

Repetition and Exploration in Next Basket Recommendation

In this chapter, we focus on the repetition and exploration in the next basket recommendation (NBR) task. The goal of a next basket recommendation system is to recommend items for the next basket for a user, based on the sequence of their prior baskets. We examine whether the performance gains of the NBR methods reported in the literature hold up under a fair and comprehensive comparison. To clarify the mixed picture that emerges from our comparison, we provide a novel angle on the evaluation of next basket recommendation (NBR) methods, centered on the distinction between repetition and exploration: the next basket is typically composed of previously consumed items (i.e., repeat items) and new items (i.e., explore items). In this manner, we address the thesis-level research question **RQ1**:

> *How to evaluate the next basket recommendation performance from the perspective of repetition and exploration?*

2.1 Introduction

Over the years, NBR has received a considerable amount of interest from the research community [9, 88, 125]. Baskets, or sets of items that are purchased or consumed together, are pervasive in many real-world services, with e-commerce and grocery shopping as prominent examples [43, 86]. Given a sequence of baskets that a user has purchased or consumed in the past, the goal of a NBR system is to generate the basket of items that the user would like to purchase or consume next. Within a basket, items have no temporal order and are equally important. A key difference between NBR and session-based or sequential item recommendations is that NBR systems need to deal with multiple items in one set. Therefore, models designed for item-based recommendation are not fit for basket-based recommendation, and dedicated NBR methods have been

This chapter was published as: M. Li, S. Jullien, M. Ariannezhad, and M. de Rijke. A next basket recommendation reality check. *ACM Transactions on Information Systems*, 41 (4):Article 116, 2023.

proposed [36, 51, 52, 62, 84, 93, 112, 125].

2.1.1 Types of recommendation methods

Over the years, we have seen the development of a wide range of recommendation methods. *Frequency-based* methods continue to play an important role as they are able to capture global statistics concerning popularity of items; this holds true for item-based recommendation scenarios as well as for NBR scenarios. Similarly, *nearest neighbor-based* methods have long been used for both item-based and basket-based recommendation scenarios. More recently, deep learning techniques have been developed to address sequential item recommendation problems, building on the capacity of deep learning-based methods to capture hidden relations and automatically learn representations [11]. Recent years have also witnessed proposals to address different aspects of the NBR task with deep learning-based methods, e.g., item-to-item relations [62], cross-basket relations [127], and noise within a basket [84].

Recent analyses indicate that deep learning-based approaches may not be the best performing for all recommendation tasks and under all conditions [55]. For the task of generating a personalized ranked list of items, linear models and nearest neighbor-based approaches outperform deep learning-based methods [29]. For sequential recommendation problems, deep learning-based methods may be outperformed by simple nearest neighbor or graph-based baselines [27]. What about the task of next basket recommendation? Here, the unit of retrieval — a basket – is more complex than in the recommendation scenarios considered in [27, 29, 55], with complex dependencies between items and baskets, across time, thus creating a potential for sophisticated representation learning-based approaches to NBR to yield performance gains. In this chapter, we take a closer look at the field to see if this is actually true.

2.1.2 A new analysis perspective

We find important gaps and flaws in the literature on NBR. These include weak or missing baselines, the use of different datasets in different papers, and of non-standard metrics. We evaluate the performance of three families of state-of-the-art NBR models (frequency-based, nearest neighbor-based, and deep learning-based), on three benchmark datasets, and find that no NBR method consistently outperforms all other methods across all settings.

Given these outcomes, we propose a more thorough analysis of the successes and failures of NBR methods. As we show in Figure 2.1, baskets recommended in a NBR scenario consist of *repeat items* (items that the user has consumed before, in previous baskets) and *explore items* (items that are new to the user). The novelty of recommended items has been studied before, and related metrics have also been proposed [100], but novelty-oriented metrics are not NBR specific and only focus on one aspect, i.e., evaluating the novelty of the list of recommendations. In order to improve our understanding of the relative performance of NBR models, especially regarding repeat items and explore items,

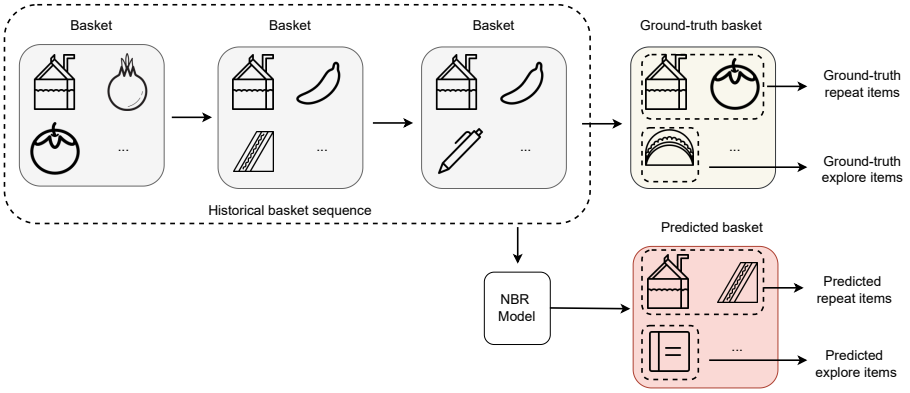


Figure 2.1: Four types of items in next basket recommendation.

we introduce a set of task-specific metrics for NBR. Our newly proposed metrics help us understand which types of items are present in the recommended basket and assess the performance of NBR models when proposing new items vs. already-purchased items.

2.1.3 Main findings

Equipped with our newly proposed metrics for NBR, we repeat our large-scale comparison of NBR models and arrive at the following important findings:

- No NBR method consistently outperforms all other methods across different datasets.
- All published methods are heavily skewed towards either repetition or exploration compared to the ground-truth, which might harm long-term engagement.
- There is a large NBR performance gap between repetition and exploration; repeat item recommendation is much easier.
- In many settings, deep learning-based NBR methods are outperformed by frequency-based baselines that fill a basket with the most frequent items in a user's history, possibly complemented with items that are most frequent across all users.
- A bias towards repeat items accounts for most of the performance gains of recently published methods, even though many complex modules or strategies specifically target explore items.
- We propose a new protocol for evaluating NBR methods, with a new frequency-based NBR baseline as well as new metrics to assess the potential performance gains of NBR methods.

- Existing NBR methods have different treatment effects on user performance and item exposure for users with different repetition ratios and items with different frequencies, respectively.

Overall, our work sheds light on the state-of-the-art of NBR, provides suggestions to improve our evaluation methodology for NBR, helps us understand the reasons underlying performance differences, and provides insights to inform the design of future NBR models.

2.2 Related Work

2.2.1 Reproducibility in information retrieval

Reproducibility is a topic that has been at the center of information retrieval (IR) research for many years. The *mechanics* of reproducibility have been a constant factor since the early days of community benchmarking [104], resulting in a large number of datasets and metrics. Artifact badging is a matter of ongoing and active interest [38] as are ways to objectively quantify to what extent a system-oriented IR experiment has been replicated or reproduced [16].

Asking *which lessons hold up* under closer scrutiny is not new either in IR. Papers of this type have been written for query performance prediction [44], ranking [7], learning to rank [97], search result diversification [1], online learning to rank [80], question answering [26], and neural rankers [71, 122]. We are particularly interested in this "which lessons hold up" aspect of reproducibility in the context of recommender systems. Dacrema et al. [27, 29] and Jannach et al. [55] have recently examined the relative strength of deep learning-based methods for item recommendation, both in a traditional static setting and in a sequential setting. In contrast, we consider the task of *next basket recommendation* (NBR) and assess the relative merits of deep learning-based methods for this task.

Some NBR methods [9, 62, 84, 112, 125] only compare with previous (deep) learning-based methods and avoid comparing with frequency-based baselines that recommend the k most frequent items in a users' historical records as the next basket. In several cases, recent publications on NBR omit comparisons to other recent methods [36, 52, 84, 127]. In [84], sampled metrics [58] are used to evaluate the performance even though this is not encouraged [61]. As our systematic comparisons below show, not all previously published lessons on deep learning-based NBR methods hold up.

2.2.2 Next basket recommendation

The NBR problem has been studied for many years. As we explain in Section 2.3 below, we analyze the performance of three families of NBR methods. First, we consider frequency-based methods; in different configurations they have been considered as baselines in most prior work on NBR that we are aware of. Second are nearest neighbor-based methods. TIFUKNN [52] and UP-CF@r [36] model temporal patterns over frequency information and then combine with neighbor

information or user-wise collaborative filtering. Third are deep learning-based methods. Such methods often have a strong focus on learning representations of baskets. Early precursors include the factorizing personalized Markov chains (FPMC) [88], which leverage matrix factorization (MF) and Markov chains to model users' general interest and basket transition relations, and hierarchical representation models (HRMs) [108], which apply aggregation operations to learn a hierarchical representation of baskets; these two MC-based methods only capture local short-term relations between adjacent baskets. In contrast, RNNs have been used for the NBR task to learn long-term trends by modeling the whole basket sequence. For instance, DREAM [125] uses max/avg pooling to encode baskets. Sets2Sets [51] adapts an attention mechanism and adds frequency information to improve performance. Some methods [62, 112] consider item relations to obtain a better representation. Yu et al. [127] argue that item-item relations between baskets are important and leverage GNNs to capture these relations; the authors also use a self-attention mechanism to learn temporal dependencies between baskets. Some methods [9, 21, 63, 93, 109] exploit auxiliary information, including product categories, amounts, prices, and explicit time stamps; for the sake of a fair comparison, we omit these from our reproducibility study.

What we add on top of prior work is not yet another NBR method but a systematic comparison under the same experimental conditions across multiple datasets as well as an analysis of the relative performance of state-of-the-art NBR methods in terms of repetition and exploration, which helps to explain the observed performance differences.

2.3 Experimental Setup

2.3.1 Datasets

In order to ensure the reproducibility of our study, we conduct our experiments on three publicly available real-world datasets:

- **TaFeng** – contains four months of shopping transactions collected from a Chinese grocery store. All products purchased on the same day by the same user are treated as a basket.[1]
- **Dunnhumby** – covers two years of household-level transactions at a retailer. All products bought by the same user in the same transaction are treated as a basket. We use the first two months of the data in our experiments.[2]
- **Instacart** – contains over three million grocery orders of Instacart users. We treat all items purchased by the same user in the same order as a basket.[3]

[1] `https://www.kaggle.com/chiranjivdas09/ta-feng-grocery-dataset`
[2] `https://www.dunnhumby.com/source-files/`
[3] `https://www.kaggle.com/c/instacart-market-basket-analysis/data`

Table 2.1: Statistics of the processed datasets.

Dataset	#Items	#Users	Avg. basket size	Avg. #baskets per user	Repeat ratio	Explore ratio
TaFeng	11,997	13,858	6.27	6.58	0.188	0.812
Dunnhumby	3,920	22,530	7.45	9.53	0.409	0.591
Instacart	13,897	19,435	9.61	13.21	0.597	0.403

Following previous work [51, 52, 62, 125, 127], we also employ a sampling strategy instead of using the whole dataset. In each dataset, users with a basket size between 3 and 50 are sampled to conduct experiments. We also remove rare and unpopular items and the remainder covers more than 95% of the interactions. A *ground truth basket* is a basket that we aim to predict or recommend, and the last basket of a sequence or purchased baskets is regarded as the ground truth basket. The *repetition ratio* and *exploration ratio* are calculated based on the ground truth basket as the proportion of *repeat items* and *explore items* in the ground truth baskets, respectively. The statistics of the processed datasets are summarized in Table 2.1.

For our experiments, we split every dataset across users in the same way that previous works have done [51, 52, 127], i.e., 72% users for training, 8% users for validation, and 20% users for the test set. The training users, validation users, and test users are totally independent from each other. We repeat the dataset split five times for independent experiments. Note that we did not use an absolute timestamp splitting strategy (i.e., splitting the dataset into several sub-datasets according to real-time ranges) for the following reasons:

1. As a reality check chapter, we decided to follow the widely used data splitting strategy in the existing NBR methods.

2. We checked three datasets and found that users' baskets are spread out across diverse time ranges and the average basket lengths are limited, so splitting using an absolute timestamp would be likely to break a large number of basket sequences into very short sequences, which cannot be used for effectively training or evaluating NBR methods.

3. As the chapter focuses on the analysis from a repetition and exploration perspective, we acknowledge that global trends are likely to influence the repetition ratio of the users, therefore we use Section 2.5.5 to analyze at a fine-grained level, i.e., the treatment on users with different repetition ratios.

2.3.2 Baseline methods and reproducible methods

We follow the same strategy as [27] to collect relevant and reproducible NBR papers. Specifically, we include papers in our analysis that have been pub-

lished during the last five years (i.e., from 2016 to 2021) in one of the following conferences: KDD [51, 93, 127], SIGIR [9, 52, 125], IJCAI [62], AAAI [112], RecSys [21] and UMAP [36]. Papers targeting NBR [9, 36, 52, 62, 84, 112, 125] and sequential set prediction [51, 93, 127] are considered to be relevant papers. For a fair comparison, methods [9, 21, 93] using auxiliary information other than item-basket sequences are not included in this chapter. Like [27], we consider a paper to be reproducible if they meet the following criteria:

1. A working version of the *source code* is publicly available[4] or the code has to be modified in minimal ways to work correctly.[5]

2. At least one *dataset* used in the original paper is available.

Through this selection process, we end up with eight relevant representative papers [36, 51, 52, 62, 84, 112, 125, 127], and seven of them are considered to be reproducible methods in our analysis [36, 51, 52, 62, 84, 125, 127].[6] Of the seven reproducible methods, four methods [36, 52, 84, 127] have been published during the last two years, and have not been compared with each other.

As simple, yet effective baselines we include two widely known frequency-based methods, i.e., *global top-frequency* (G-TopFreq) and *personal top-frequency* (P-TopFreq), which are often shown as simple baselines in recommendation tasks. Surprisingly, we find that 3 of the 5 deep-learning based methods that we consider only compare with the global top-frequency baseline G-TopFreq [62, 84, 125], but do not compare to the personal top-frequency baseline P-TopFreq, which is known to have higher performance in general [36, 51, 52, 127].

There is an important limitation of the *personal top-frequency* method (P-TopFreq) w.r.t. the basket size in a NBR setting that is ignored in previous work. P-TopFreq can only recommend items from the past transactions of a user, which means that it might not be able to fully make use of the available basket slots like other methods, and this may lead to an unfair comparison. We analyze the percentage of basket slots used for P-TopFreq, and Table 2.2 shows the results. To address this limitation, we propose a simple combination of G-TopFreq and P-TopFreq as an additional baseline, called *GP-TopFreq*: GP-TopFeq first uses P-TopFreq to fill a basket, and then uses G-TopFeq to fill any remaining slots.

Frequency-based baselines

- **G-TopFreq** – uses the k most popular items in the dataset to form the recommended next basket. It is widely used in recommendation systems due to its effectiveness and simplicity.

[4] We first check whether the paper provides a link to their code; if not, we search GitHub using the title of the paper.

[5] We re-implement the Dream algorithm [125].

[6] We do not include FPMC [88] in our chapter for the following two reasons: first, it will break the selection criteria [27] we employ in this chapter; second, FPMC is among the worst performing method in all NBR related papers [52, 62, 84, 125] recently.

Table 2.2: Percentage of the basket slots used for P-TopFreq.

	Dataset		
Basket size	TaFeng	Dunnhumby	Instacart
10	92.39%	95.66%	96.70%
20	79.71%	87.98%	90.48%

- **P-TopFreq** – a personalized top frequency method that treats the most frequent k items in the users' historical records as the next basket. This method only has repeat items in the prediction.
- **GP-TopFreq** – a combination of P-TopFreq and G-TopFreq to make full use of the available basket slots.

Nearest neighbor-based methods

- **TIFUKNN** – models the temporal dynamics of frequency information of users' past baskets to introduce personalized frequency information (PIF), then uses a nearest neighbor-based method on PIF [52].
- **UP-CF@r** – a combination of recency-aware, user-wise popularity and user-wise collaborative filtering. The recency of shopping behavior is considered in this method [36].

Deep learning-based methods

- **Dream** – the first deep learning-based method that models users' global sequential basket history for NBR. It uses a pooling strategy to generate basket representations, which are then fed into an RNN to learn user representations and predict the corresponding next set of items [125].
- **Sets2Sets** – uses a pooling operation to get basket embeddings and an attention mechanism to learn a user's representation from their past interactions. Furthermore, item frequency information is adopted to improve performance [51].
- **DNNTSP** – leverages a GNN and self-attention techniques. It encodes item-item relations via a graph and employs a self-attention mechanism to capture temporal dependencies of users' basket sequence [127].
- **Beacon** – an RNN-based method that encodes the basket considering the incorporating information on pairwise correlations among items [62].
- **CLEA** – an RNN-based method that uses a contrastive learning model to automatically extract items relevant to the target items and generates the representation via a GRU-based encoder [84].

Table 2.3: Notation used in this chapter.

Symbol	Description
U	Set of all users
U_g	Set of users in group g
I	Set of all items
u	A single user in U
i	A single item in I
S_j	Sequence of historical baskets for u_j
B_j^t	t-th basket in S_j, a set of items $i \in I$
T_{u_j}	Target/ground truth basket for u_j that we aim to predict
$T_{u_j}^{rep}$	Set of *repeat items* in the ground truth basket T_{u_j} for u_j
$T_{u_j}^{expl}$	Set of *explore items* in the ground truth basket T_{u_j} for u_j
P_{u_j}	Predicted basket for u_j
$P_{u_j}^{rep}$	Set of *repeat items* in the predicted basket P_{u_j} for u_j
$P_{u_j}^{expl}$	Set of *explore items* in the predicted basket P_{u_j} for u_j
$I_{j,t}^{rep}$	Repeat items for u_j at timestamp t; set of items that have appeared in u_j's baskets up to timestamp t
$I_{j,t}^{expl}$	Explore items for u_j at timestamp t; set of items that have not appeared in u_j's baskets up to timestamp t

2.3.3 Implementation details

For deep learning-based methods [51, 62, 84, 125, 127], we strictly follow the hyper-parameter setting and tuning strategy in their respective paper or related GitHub repository. Following the same strategy as [27], we use the suggested best parameters in TIFUKNN [52] to achieve its best performance. For UP-CF@r, the recency window is tuned on {1, 5, 10, 50}, the locality is tuned on {1, 20, 50, 100}, the asymmetry is tuned on {0, 0.25, 0.5, 0.75, 1.0}. We perform a grid search on the validation dataset to tune hyper-parameters and select the best model for testing. For all methods, we rely as much as possible on the original source code and construct a pipeline to perform experiments. We share the code and data used in our experiments online.

2.4 Performance Comparison Using Conventional Metrics

2.4.1 Conventional next basket recommendation metrics

To analyze the performance of NBR methods, we first consider three conventional metrics: recall, normalized discounted cumulative gain (NDCG), and personalized hit ratio (PHR), all of which are commonly used in previous NBR studies [51, 52, 127]. We do not consider the F1 and Precision metrics in this chapter, since we focus on the basket recommendation with a fixed basket size K, which means the Precision@K and F1@K are proportional to Recall@K for

each user. F1@K and Precision@K are more suitable for NBR with a dynamic basket size for each user.

Recall measures the ability to find all relevant items and is calculated as follows:

$$Recall@K(u_j) = \frac{\left|P_{u_j} \cap T_{u_j}\right|}{\left|T_{u_j}\right|}, \tag{2.1}$$

where P_{u_j} is the predicted basket with K recommended items and T_{u_j} is the ground truth basket for user u_j. The average recall score of all users is adopted as the recall performance.

NDCG is a ranking quality measurement metric, which takes item order into consideration and it is calculated as follows, for a user $u \in U$ and its ground truth basket T_u:

$$\text{NDCG@}K(u_j) = \frac{\sum_{k=1}^{K} p_k / \log_2(k+1)}{\sum_{k=1}^{\min(K,|T_{u_j}|)} 1/\log_2(k+1)}, \tag{2.2}$$

where p_k equals 1 if $P_{u_j}^k \in T_u$, otherwise $p_k = 0$. P_u^k denotes the k-th item in the predicted basket P_u. The average score across all users is the NDCG performance of the algorithm.

PHR focuses on user level performance and calculates the ratio of predictions that capture at least one item in the ground truth basket as follows:

$$\text{PHR@}K = \frac{\sum_{j=1}^{N} \varphi(P_{u_j}, T_{u_j})}{N}, \tag{2.3}$$

where N is the number of test users, and $\varphi(P_{u_j}, T_{u_j})$ returns 1 when $P_{u_j} \cap T_{u_j} \neq \emptyset$, otherwise returns 0.

2.4.2 Results with conventional next basket recommendation metrics

Performance results for the conventional NBR metrics are shown in Table 2.4. The performance of different methods varies across datasets; there is no method that consistently outperforms all other methods, independent of dataset and basket size. This calls for a further analysis of the factors impacting performance, which we conduct in the next section.

Among the frequency-based baselines, P-TopFreq outperforms G-TopFreq in all scenarios, which indicates that personalization improves the NBR performance. P-TopFreq can only recommend items that have appeared in a user's previous baskets. As pointed out in Section 2.3.2, the number of repeat items of a user may be smaller than the basket size, which means there might be empty slots in a basket recommended by P-TopFreq. Despite this limitation, P-TopFreq is a competitive NBR baseline. GP-TopFreq makes full use of the available basket slots by filling any slots with top ranked items suggested by G-TopFreq. GP-TopFreq outperforms P-TopFreq with no surprise, and, as expected, the difference shrinks as the repetition ratio of the dataset increases. For future fair comparisons, we believe that GP-TopFreq should be the baseline

Table 2.4: Overall performance comparison of frequency-based, nearest neighbor-based, and deep learning-based NBR methods. Highlights indicate the highest score per basket size and metric. We write * to indicate that the highest score for a given basket size and metric is significantly better than the second highest score (paired t-test, p-value < 0.05).

Size			10			20		
Dataset	Methods	Recall	NDCG	PHR	Recall	NDCG	PHR
TaFeng	G-TopFreq	0.0831 (0.0018)	0.0864 (0.0017)	0.2498 (0.0024)	0.1114 (0.0029)	0.0961 (0.0018)	0.3311 (0.0006)
	P-TopFreq	0.1069 (0.0023)	0.0955 (0.0019)	0.3473 (0.0033)	0.1395 (0.0026)	0.1096 (0.0019)	0.4329 (0.0038)
	GP-TopFreq	0.1211 (0.0031)	0.1015 (0.0023)	0.3691 (0.0043)	0.1693 (0.0031)	0.1208 (0.0022)	0.4834 (0.0040)
	UP-CF@r	0.1249 (0.0027)	0.1104 (0.0019)	0.3983 (0.0035)	0.1694 (0.0034)	0.1280 (0.0021)	0.4877 (0.0048)
	TIFUKNN	0.1251 (0.0033)	0.1016 (0.0014)	0.3852 (0.0029)	0.1817 (0.0037)	0.1232 (0.0016)	0.5043 (0.0035)
	Dream	0.1134 (0.0023)	0.1022 (0.0018)	0.3035 (0.0024)	0.1463 (0.0034)	0.1149 (0.0021)	0.3905 (0.0039)
	Beacon	0.1139 (0.0032)	0.1033 (0.0023)	0.3055 (0.0044)	0.1475 (0.0026)	0.1154 (0.0020)	0.4002 (0.0024)
	CLEA	0.1184 (0.0038)	0.1046 (0.0030)	0.3076 (0.0044)	0.1477 (0.0035)	0.1165 (0.0028)	0.3957 (0.0037)
	Sets2Sets	0.1349 (0.0034)	0.1124 (0.0025)	0.4080 (0.0058)	0.1904 (0.0021)	0.1342 (0.0020)	0.5261 (0.0041)
	DNNTSP	0.1526 (0.0034) *	0.1314 (0.0023) *	0.4460 (0.0044) *	0.2077 (0.0044) *	0.1532 (0.0025) *	0.5496 (0.0038) *
Dunnhumby	G-TopFreq	0.0982 (0.0009)	0.1050 (0.0010)	0.4621 (0.0033)	0.1264 (0.0006)	0.1192 (0.0008)	0.5314 (0.0030)
	P-TopFreq	0.2319 (0.0013)	0.2340 (0.0012)	0.6559 (0.0017)	0.3030 (0.0012)	0.2695 (0.0011)	0.7173 (0.0017)
	GP-TopFreq	0.2356 (0.0013)	0.2358 (0.0013)	0.6649 (0.0019)	0.3141 (0.0012)	0.2743 (0.0011)	0.7372 (0.0026)
	UP-CF@r	0.2428 (0.0006)	0.2468 (0.0012)	0.6747 (0.0024)	0.3185 (0.0009)	0.2848 (0.0013)	0.7339 (0.0014)
	TIFUKNN	0.2396 (0.0008)	0.2408 (0.0011)	0.6761 (0.0022)	0.3191 (0.0015)	0.2799 (0.0012)	0.7407 (0.0030)
	Dream	0.0950 (0.0008)	0.1036 (0.0008)	0.4586 (0.0040)	0.1300 (0.0013)	0.1205 (0.0011)	0.5395 (0.0033)
	Beacon	0.0995 (0.0009)	0.1066 (0.0011)	0.4700 (0.0037)	0.1354 (0.0009)	0.1241 (0.0010)	0.5506 (0.0025)
	CLEA	0.1552 (0.0010)	0.1732 (0.0010)	0.5541 (0.0038)	0.1866 (0.0012)	0.1864 (0.0007)	0.6273 (0.0029)
	Sets2Sets	0.1691 (0.0023)	0.1473 (0.0015)	0.5802 (0.0053)	0.2552 (0.0018)	0.1880 (0.0015)	0.6893 (0.0046)
	DNNTSP	0.2404 (0.0007)	0.2430 (0.0010)	0.6767 (0.0022)	0.3242 (0.0004) *	0.2839 (0.0007)	0.7427 (0.0016)
Instacart	G-TopFreq	0.0710 (0.0003)	0.0811 (0.0001)	0.4542 (0.0010)	0.0990 (0.0001)	0.0962 (0.0002)	0.5248 (0.0017)
	P-TopFreq	0.3260 (0.0008)	0.3378 (0.0007)	0.8449 (0.0018)	0.4307 (0.0008)	0.3939 (0.0002)	0.8957 (0.0019)
	GP-TopFreq	0.3269 (0.0008)	0.3383 (0.0007)	0.8463 (0.0018)	0.4354 (0.0007)	0.3961 (0.0003)	0.9011 (0.0017)
	UP-CF@r	0.3506 (0.0007)	0.3631 (0.0007)	0.8652 (0.0020)	0.4591 (0.0008)	0.4222 (0.0005)	0.9079 (0.0012)
	TIFUKNN	0.3601 (0.0015) *	0.3721 (0.0008) *	0.8642 (0.0005)	0.4709 (0.0015) *	0.4323 (0.0003) *	0.9097 (0.0011)
	Dream	0.0712 (0.0004)	0.0805 (0.0005)	0.4551 (0.0008)	0.0997 (0.0006)	0.0957 (0.0007)	0.5304 (0.0027)
	Beacon	0.0734 (0.0003)	0.0838 (0.0003)	0.4628 (0.0006)	0.1050 (0.0006)	0.1009 (0.0004)	0.5462 (0.0022)
	CLEA	0.1221 (0.0014)	0.1449 (0.0016)	0.5603 (0.0053)	0.1514 (0.0045)	0.1592 (0.0019)	0.6347 (0.0094)
	Sets2Sets	0.2125 (0.0013)	0.1923 (0.0019)	0.7185 (0.0040)	0.3077 (0.0040)	0.2402 (0.0020)	0.8284 (0.0040)
	DNNTSP	0.3330 (0.0003)	0.3412 (0.0004)	0.8525 (0.0011)	0.4423 (0.0005)	0.4000 (0.0005)	0.9042 (0.0003)

for every NBR method to compare with, especially in high exploration scenarios, to be able to determine what value is added beyond simple frequency-based recommendations.

As to the nearest neighbor-based methods, we see that TIFUKNN and UP-

CF@r have a similar performance across different scenarios and outperform all frequency-based baselines. The two methods are similar in the sense that both model temporal information, combined with a user-based nearest neighbor method. Their performance in a high exploration scenario is lower than several deep learning-based methods (i.e., the TaFeng dataset), but on the Dunnhumby and Instacart datasets, which have a relatively low exploration ratio, they are among the best performing methods.

Most of the deep learning-based methods outperform G-TopFreq, which is the only frequency-based baseline considered in many papers. Surprisingly, P-TopFreq and GP-TopFreq achieve a highly competitive performance and outperform four deep learning-based methods (i.e., Dream, Beacon, CLEA and Sets2sets), by a large margin in the Dunnhumby and Instacart datasets, where the improvements in terms of *Recall*@10 range from 35.8% to 141.9% and from 53.6% to 353.3%, respectively. Moreover, the proposed GP-Topfreq baseline outperforms the deep learning-based Beacon, Dream and CLEA algorithm on the TaFeng dataset, the scenario with a high exploration ratio. Of the deep learning-based methods, DNNTSP is the only one to have a consistently high performance in all scenarios.

2.4.3 Upshot

Based on the above experiments and analysis, we conclude that the choice of dataset plays an important role in evaluating the performance of NBR methods, and no state-of-the-art NBR method is able to consistently achieve the best performance across datasets.

Several deep learning-based NBR methods [62, 84, 125] aim to learn better performing representations by capturing long-term temporal dependencies, denoising, etc. They do indeed outperform the G-TopFreq baseline, but many are inferior to the P-TopFreq baseline, especially in datasets with a relatively high repetition ratio. The proposed GP-TopFreq baseline in some sense "recalibrates" the improvements reported for recently introduced complex, deep learning-based NBR methods; compared to the simple GP-TopFreq baseline, their improvements are modest or even absent.[7]

So far we have used conventional metrics to examine the performance of NBR methods. To account for the findings reported in this section and provide insights for future NBR method development, we will now consider additional metrics.

2.5 Performance w.r.t. Repetition and Exploration

In order to understand which factors contribute to the overall performance of a NBR method, we dive deeper into the basket components from a repetition and exploration perspective.

[7] According to our analyses, performance differences that were reported in previous work still stand. However, the set of baselines used for comparison in previous work is too limited.

2.5.1 Metrics for repetition vs. exploration

We propose several metrics that are meant to capture repetition and exploration aspects of a basket. First, we adopt widely used definitions of repetition and exploration in the recommender systems literature [3, 18, 19, 100] to define what constitutes a repeat item and an explore item in the context of NBR. Specifically, an item i^r is considered to be a *repeat item* for a user u_j if it appears in the sequence of the user's historical baskets S_j, that is, if $i^r \in I^{rep}_{j,t} = B^1_j \cup B^2_j \cup \cdots \cup B^t_j$. Otherwise, the item is an *explore item*, denoted as $i^e \in I^{expl}_{j,t} = I - I^{rep}_{j,t}$. We write $P_{u_j} = P^{rep}_{u_j} \cup P^{expl}_{u_j}$ for the predicted next basket B^{t+1}_j, which is the union of *repeat items* $P^{rep}_{u_j}$ and *explore items* $P^{expl}_{u_j}$. As an edge case, a basket may consist of repeat or explore items only.

Novelty of recommendation is a concept that is similar to, but different from, the notion of exploration that we use in this chapter. Several novelty related metrics have been proposed, i.e., EPC and EPD [100]. It is important to note that these metrics are not suitable for our analysis in this chapter. First, they only focus on measuring the novelty of a ranked list, while we want to not only understand the components within the predicted basket, but also analyze a model's ability to fulfill a user's needs w.r.t. repetition and exploration. Second, these metrics are not NBR specific and only focus on one aspect, i.e., novelty, while we want to make a comparison between repetition and exploration to assess the NBR performance.

To analyze the composition of a predicted basket, we propose the *repetition ratio*, *RepR*, and the *exploration ratio*, *ExplR*. *RepR* and *ExplR* represent the proportion of *repeat items* and *explore items* in a recommended basket, respectively. The overall *RepR* and *ExplR* scores are calculated over all test users as:[8]

$$RepR = \frac{1}{N}\sum_{j=1}^{N}\frac{\left|P^{rep}_{u_j}\right|}{K}, \qquad ExplR = \frac{1}{N}\sum_{j=1}^{N}\frac{\left|P^{expl}_{u_j}\right|}{K}, \tag{2.4}$$

where N denotes the number of test users, K is the size of the model's predicted basket for user u_j, $P^{rep}_{u_j}$ and $P^{expl}_{u_j}$ are the sets of repeat items in P_{u_j} and of explore items in P_{u_j}, respectively.

Next, we pay attention to a basket's ability to fulfill a user's need for repetition and exploration, and propose the following metrics. $Recall_{rep}$ and PHR_{rep} are used to evaluate the *Recall* and *PHR* w.r.t. the repetition performance; similarly, we use $Recall_{expl}$ and PHR_{expl} to assess the exploration performance. More precisely:

$$Recall_{rep} = \frac{1}{N_r}\sum_{j=1}^{N_r}\frac{\left|P_{u_j} \cap T^{rep}_{u_j}\right|}{\left|T^{rep}_{u_j}\right|}, \qquad PHR_{rep} = \frac{\sum_{j=1}^{N_r}\varphi\left(P_{u_j}, T^{rep}_{u_j}\right)}{N_r} \tag{2.5}$$

[8] To assess this performance on a dataset, we use the average performance across users; we also show the corresponding user level *RepR* distribution in our analysis.

and

$$Recall_{expl} = \frac{1}{N_e} \sum_{i=1}^{N_e} \frac{\left| P_{u_j} \cap T_{u_j}^{expl} \right|}{\left| T_{u_j}^{expl} \right|}, \qquad PHR_{expl} = \frac{\sum_{j=1}^{N_e} \varphi\left(P_{u_j}, T_{u_j}^{expl}\right)}{N_e}, \quad (2.6)$$

where N_r and N_e denote the number of users whose ground truth basket contains *repeat items* and *explore items* respectively; $\varphi(P, T)$ returns 1 when $P \cap T \neq \emptyset$, otherwise it returns 0.

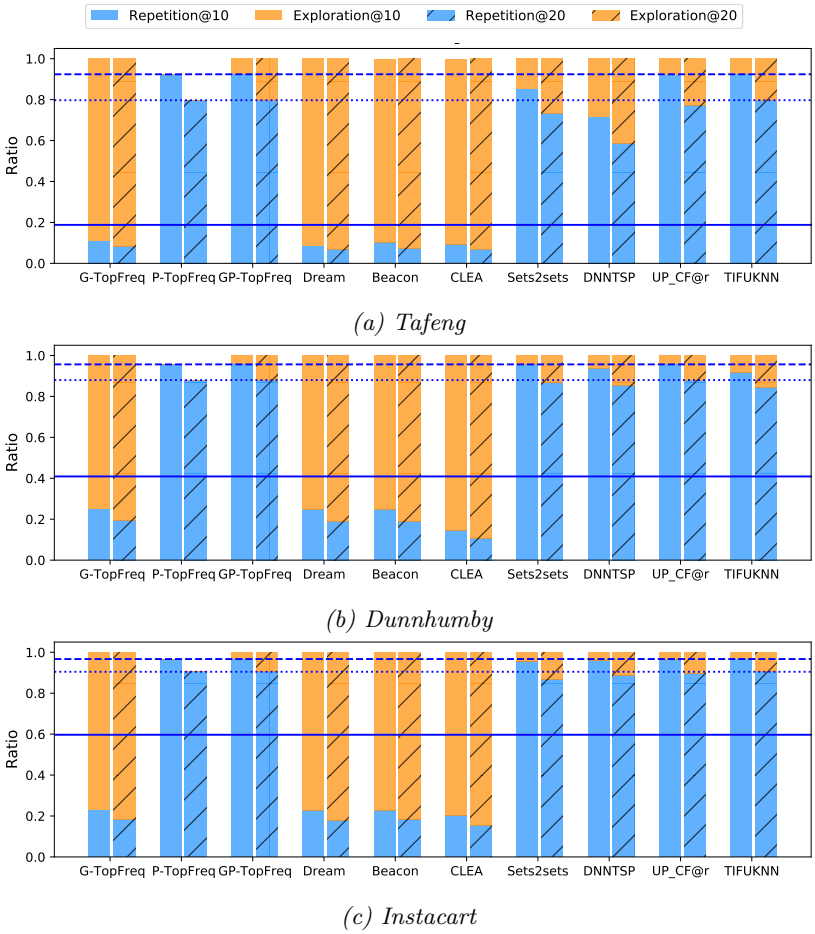
Next, we first use the repetition ratio and exploration ratio to examine the recommended baskets; we then use our repetition and exploration metrics to re-assess the NBR methods that we consider, examine how repetition and exploration contribute to the overall recommendation performance, and how users with different degrees of repeat behavior benefit from different NBR methods.

2.5.2 The components of a recommended basket

We analyze the components of the recommended basket for each NBR method to understand what makes up the recommendation. The results are shown in Figure 2.2. First, we see that, averaged over all users, all recommended baskets are heavily skewed towards either item repetition or exploration, relative to the ground-truth baskets that are much more balanced between already seen and new items. We can divide the methods that we compare into repeat-biased methods (i.e., P-TopFreq, GP-TopFreq, Sets2sets, DNNTSP, UP-CF@r, and TIFUKNN) and explore-biased methods (i.e., G-TopFreq, Dream, Beacon, and CLEA). Importantly, a large performance gap exists between the two types. None of the published NBR methods can properly balance the repeat items and explore items of users' future interests. Figure 2.3 shows the repetition ratio *RepR* distribution for the ground truth basket and the recommended basket derived by a repeat-biased method or an explore-biased method. We show the *RepR* distribution of a representive explore-biased method, i.e., CLEA, and a representive repeat-biased method, i.e., DNNTSP, in Figure 2.3. The *RepR* distribution of the other eight NBR methods are provided in our appendix (Figure 2.7).

Among the explore-biased methods, G-TopFreq is not a personalized method; it always provides the most popular items. Dream, Beacon, and CLEA treat all items without any discrimination, which means the explore items are more likely to be in the predicted basket and their basket components are similar to G-TopFreq. Looking at the performance in Table 2.4, we see that repeat-biased methods generally perform much better than explore-biased methods on conventional metrics across the datasets, especially when the dataset has a relatively high repetition ratio.

The repetition ratios of P-TopFreq and GP-TopFreq serve as the upper bound repetition ratio for the recommended basket. Most baskets recommended by repeat-biased methods are close to or reach this upper bound, even when the datasets have a low ratio of repeat behavior in the ground truth, except for two cases (Sets2sets and DNNTSP on the TaFeng dataset).

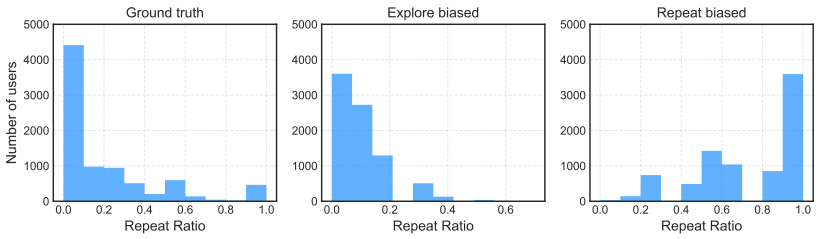


(a) Tafeng

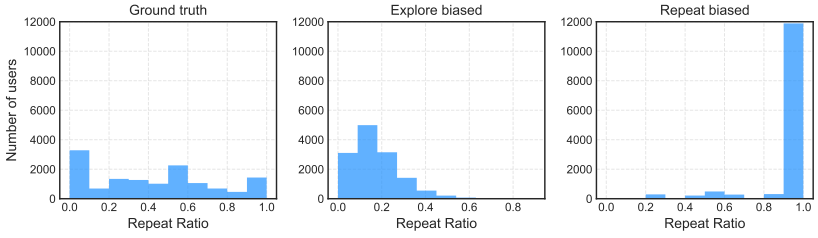
(b) Dunnhumby

(c) Instacart

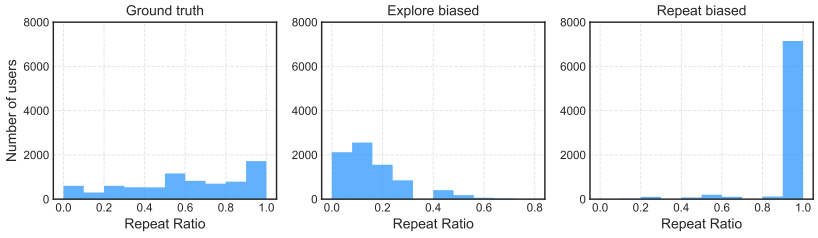
Figure 2.2: The repetition ratio *RepR* and exploration ratio *ExplR* of recommended baskets averaged over all users; solid lines indicate the repetition ratio of the ground truth; dashed lines indicate the upper bound of the repetition ratio for basket size 10 and dotted lines for basket size 20.



(a) Tafeng



(b) Dunnhumby



(c) Instacart

Figure 2.3: Distribution of the repetition ratio *RepR* of recommended baskets on different datasets for an explore-biased method (CLEA) and a repeat-biased method (DNNTSP).

Finally, the exploration ratio of repeat-biased methods increases from basket size 10 to 20; we believe that this is because there are simply no extra repeat items available: it does not mean that the methods actively increase the exploration ratio in a larger basket setting.

2.5.3 Performance w.r.t. repetition and exploration

The results in terms of repetition and exploration performance are shown in Table 2.5. First of all, using our proposed metrics, we observe that the repetition performance $Recall_{rep}$ is always higher than the exploration performance, even when the explore items form almost 90% of the recommended basket. This shows that the repetition task (recommending repeat items) and the exploration task (recommending explore items) have different levels of difficulty and that capturing users' repeat behavior is much easier than capturing their explore behavior.

Three deep learning-based methods perform worst w.r.t. repeat item recommendation *and* best w.r.t. explore item recommendation at the same time, as they are heavily skewed towards explore items. We also see that there are improvements in the exploration performance compared to G-TopFreq with the same level of exploration ratio, which indicates that the representation learned by these methods does capture the hidden sequential transition relationship between items. Repeat-biased methods perform better w.r.t. repetition in all settings, since the baskets they predict contain more repeat items. Similarly, we can see that DNNTSP, UP-CF@r, and TIFUKNN perform better than P-TopFreq w.r.t. repeat performance with the same or a lower level of repetition ratio.

Third, explore-biased methods spend more resources on the more difficult and uncertain task of explore item prediction, which is not an optimal choice when considering the overall NBR performance. Being biased towards the easier task of repeat item prediction leads to gains in the overall performance, which is positively correlated with the repetition ratio of the dataset.

To understand the potential reasons for a method being repeat-biased or explore-biased, we provide an in-depth analysis of the methods' architectures. P-TopFreq and GP-TopFreq are repeat-biased methods as they both mainly rely on the frequency of historical items to recommend the next basket. Two nearest neighbor-based methods, i.e., TIFUKNN and UP-CF@r, have a module to model both the frequency and the recency of historical items; besides, they both have a parameter to emphasize the frequency and recency information. Similarly, Sets2sets is also repeat-biased as it adds the historical items' frequency information to the prediction layer. DNNTSP does not consider frequency information, however, it has an indicator vector to indicate whether an item has appeared in the historical basket sequence or not, which can be regarded as a repeat item indicator. G-TopFreq is explore-biased since it is not a personalized method and can only recommend top-k popular items within the dataset. The remaining three explore-biased methods (Dream, Beacon, and CLEA) do not consider the frequency of historical items or the indicator of items' appearance,

Table 2.5: Repetition and exploration performance comparison of frequency-based, nearest neighbor-based, and deep learning-based NBR methods. Highlights indicate the highest score per basket size, for the exploration and repetition metrics. As in Table 2.4, we write * to indicate that the highest score for a given basket size and metric is significantly better than the second highest score (paired t-test, p-value < 0.05).

Size	Dataset	TaFeng				Dunnhumby				Instacart			
	Methods	Recall -rep	Recall -expl	PHR -rep	PHR -expl	Recall -rep	Recall -expl	PHR -rep	PHR -expl	Recall -rep	Recall -expl	PHR -rep	PHR -expl
10	G-TopFreq	0.1268	0.0573	0.1947	0.1738	0.1882	0.0393	0.4954	0.1590	0.1002	0.0382	0.4138	0.1575
	P-TopFreq	0.5234	0.0000	0.6766	0.0000	0.5612	0.0000	0.8553	0.000	0.5388	0.0000	0.9085	0.0000
	GP-TopFreq	0.5234	0.0157	0.6766	0.0266	0.5612	0.0049	0.8553	0.0148	0.5388	0.0014	0.9085	0.0046
	UP-CF@r	0.6046*	0.0086	0.7515*	0.0153	0.5913*	0.0018	0.8752*	0.0064	0.5805	0.0011	0.9295	0.0038
	TIFUKNN	0.5616	0.0176	0.7037	0.0284	0.5726	0.0082	0.8646	0.0257	0.5931*	0.0018	0.9287	0.0060
	Dream	0.1312	0.0921	0.1874	0.2378	0.1843	0.0412	0.4906	0.1668	0.1007	0.0367	0.4181	0.1503
	Beacon	0.1456	0.0822	0.2126	0.2223	0.1893	0.0404	0.4963	0.1605	0.1010	0.0365	0.4193	0.1492
	CLEA	0.1442	0.0935	0.2073	0.2398	0.2492	0.0569*	0.5702	0.1803*	0.1715	0.0337	0.5435	0.1297
	Sets2Sets	0.5647	0.0271	0.7222	0.0495	0.4260	0.0026	0.7583	0.0064	0.3515	0.0005	0.7713	0.0013
	DNNTSP	0.5291	0.0556	0.6912	0.1294	0.5861	0.0073	0.8684	0.0224	0.5477	0.0018	0.9110	0.0054
20	G-TopFreq	0.1637	0.0789	0.2530	0.2385	0.2279	0.0609	0.5565	0.2353	0.1335	0.0602	0.4767	0.2251
	P-TopFreq	0.7251	0.0000	0.8439	0.0000	0.7399	0.0000	0.9353	0.0000	0.7193	0.0000	0.9631	0.0000
	GP-TopFreq	0.7251	0.0339	0.8439	0.0724	0.7399	0.0153	0.9353	0.0470	0.7193	0.0084	0.9631	0.0285
	UP-CF@r	0.7785*	0.0259	0.8742	0.0560	0.7718	0.0078	0.9430*	0.0261	0.7612	0.0083	0.9735	0.0215
	TIFUKNN	0.7664	0.0410	0.8707	0.0806	0.7474	0.0159	0.9344	0.0677	0.7747*	0.0108	0.9728	0.0318
	Dream	0.1723	0.1234	0.2572	0.3264	0.2316	0.0687	0.5612	0.2596	0.1348	0.0586	0.4849	0.2222
	Beacon	0.1748	0.1230	0.2621	0.3262	0.2333	0.0672	0.5628	0.2568	0.1338	0.0599	0.4822	0.2252
	CLEA	0.1799	0.1217	0.2675	0.3189	0.2808	0.0874*	0.6168	0.2648*	0.2081	0.0578	0.6051	0.2044
	Sets2Sets	0.7478	0.0558	0.8703	0.1166	0.6317	0.0086	0.8881	0.0273	0.5095	0.0032	0.8873	0.0073
	DNNTSP	0.6820	0.0899	0.8130	0.2103	0.7758	0.0183	0.9391	0.0609	0.7284	0.0095	0.9657	0.0320

so they fail to identify the benefits of recommending repeat items.

2.5.4 The relative contribution of repetition and exploration

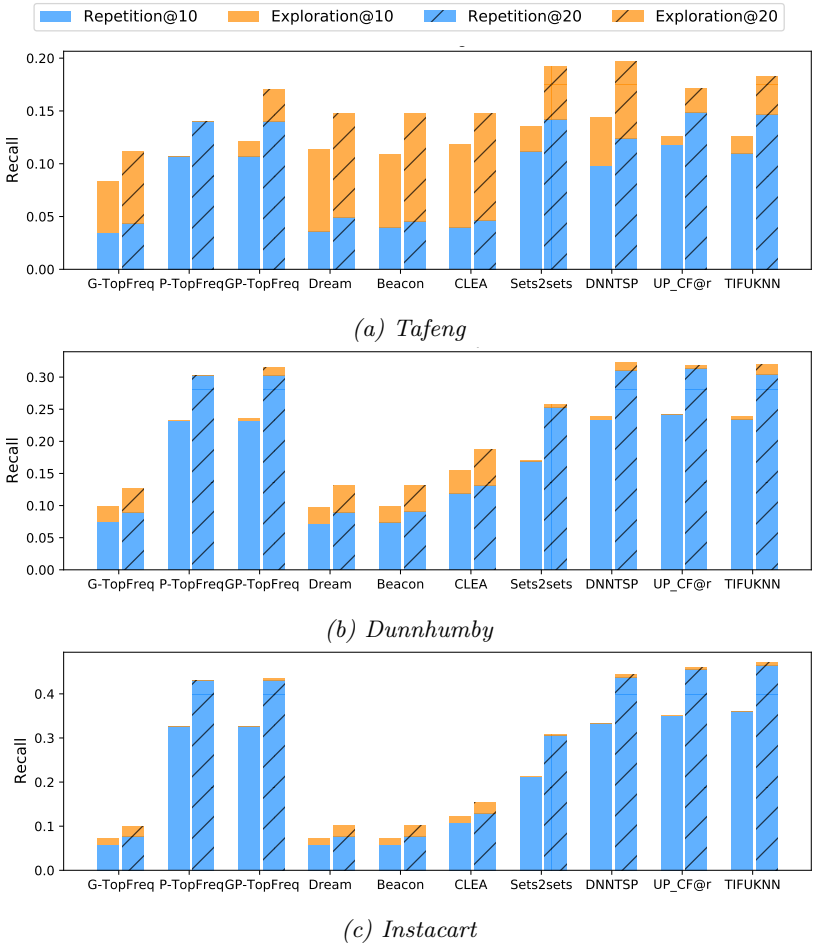
Even though a clear improvement w.r.t. either repeat or explore performance can be observed in the previous section, this does not mean that this improvement is the reason for the better overall performance, since repeat and explore items account for different ground truth proportions in different datasets. To better understand where the performance gains of the well-performing methods in Table 2.4 come from, we remove *explore items* and keep *repeat items* in the predicted basket to compute the contribution of repetition, similarly, we remove *repeat items* and keep *explore items* to compute the performance, which can be regarded as the contribution of exploration.

Experimental results on three datasets are shown in Figure 2.4. We consider G-TopFreq, P-TopFreq, and GP-TopFreq as simple baselines to compare with. From Figure 2.4, we conclude that Dream and Beacon perform better than G-TopFreq on the TaFeng dataset, as the main performance gain is from improvements in the exploration prediction. As a consequence, in the Dunnhumby and Instacart datasets, Dream, Beacon, and G-TopFreq achieve similar performance, and the repeat prediction contributes the most to the overall performance, even when their recommended items are heavily skewed towards explore items. Also, we observe that CLEA outperforms other explore-biased methods due to its improvements in the repetition performance without sacrificing the exploration performance.

At the same time, it is clear that TIFUKNN, UP-CF@r, Sets2Sets, and DNNTSP outperform explore-biased methods because of the improvements in the repetition performance, even at the detriment of exploration. The repeat items make up the majority of their correct recommendations. Specifically, repeat recommendations contribute to over 97% of their overall performance on the Dunnhumby and Instacart datasets.

An interesting comparison is between Sets2Sets and P-TopFreq. The strong performance gain of Sets2sets on the TaFeng dataset is mainly due to the exploration part, whereas P-TopFreq outperforms it by a large margin on the other two datasets at the same level of repetition ratio, even though the personal frequency information is considered in the Sets2sets model. We believe this indicates that the loss on repeat items seems to be suppressed by the loss on explore items during the training process, which weakens the influence of the frequency information.

Recall that the number of repetition candidates for a user may be smaller than the basket size, which means that there might be empty slots in the basket recommended by P-TopFreq. From Figure 2.2 and Table 2.2, we observe that the empty slots account for a significant proportion of exploration slots in many settings. However, existing studies omit this fact when making the comparison with P-TopFreq, leading to an unfair comparison and overestimation of the improvement, as their predictions leverage more slots. For example, Dream, Beacon, and CLEA can beat P-TopFreq, but they are inferior to GP-TopFreq.



(a) Tafeng

(b) Dunnhumby

(c) Instacart

Figure 2.4: Performance contribution from repeat and explore recommendations on the Tafeng, Dunnhumby, and Instacart datasets.

TIFUKNN and UP-CF@r model the temporal order of the frequency information, leading to a higher repetition performance than P-TopFreq in general. Even though the contribution of the repetition performance improvement is obvious on the Instacart dataset, it is less meaningful on the other two datasets, where the performance gain is mainly from the exploration part by filling the empty slots. When compared with the proposed GP-TopFreq baseline on the Tafeng and Dunnhumby datasets, the improvement is around a modest 3%.

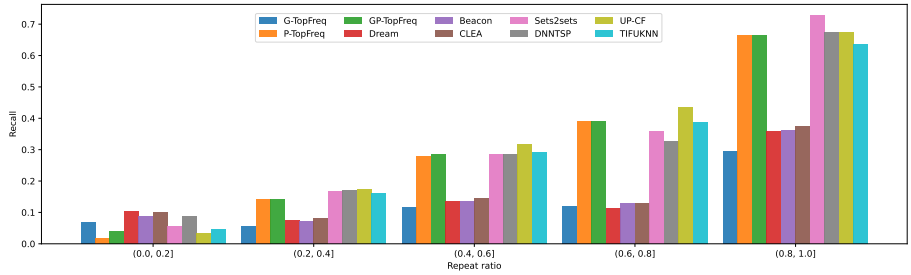
DNNTSP is always among the best-performing methods across the three datasets and is able to model exploration more effectively than other repeat-biased methods. Moreover, it also actively recommends explore items, rather than being totally biased towards the repeat recommendation in high exploration scenarios. However, the improvement is limited due to the relatively high repetition ratios and the huge difficulty gap between repetition and exploration tasks. Compared with GP-TopFreq, the improvement of DNNTSP w.r.t. *Recall*@10 on the Dunnhumby and Instacart datasets is merely 1.3% and 1.9% respectively, which is modest considering the complexity and training time added by DNNTSP.

Obviously, even though many advanced NBR algorithms learn rich user and/or item representations, the main performance gains stem from the prediction of repeat behavior. Yet, limited progress w.r.t. overall performance has so far been made compared to the simple P-TopFreq and GP-TopFreq baseline methods.

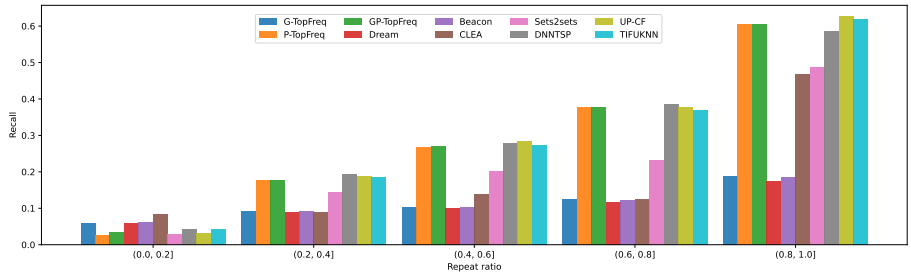
2.5.5 Treatment effect for users with different repetition ratios

As the average repetition ratio in a dataset has a significant influence on a model's performance (see Section 2.4), existing NBR methods are skewed to repetition or exploration (see Section 2.5.2) and global trend might influence the users repetition patterns, it is of interest to investigate the treatment effect for users with different repetition ratios. We examine the performance of NBR methods w.r.t. different groups of users with different repetition ratios. We divide the users into 5 groups according to their repetition ratio: $[0, 0.2]$, $(0.2, 0.4]$, $(0.4, 0.6]$, $(0.6, 0.8]$, $(0.8, 1.0]$, and calculate the average performance within each group. Note that the repetition ratio indicates the user's preference w.r.t. *repeat items* and *explore items*, e.g., users with a low repetition ratio prefer to purchase new items in their next basket. The results are shown in Figure 2.5.

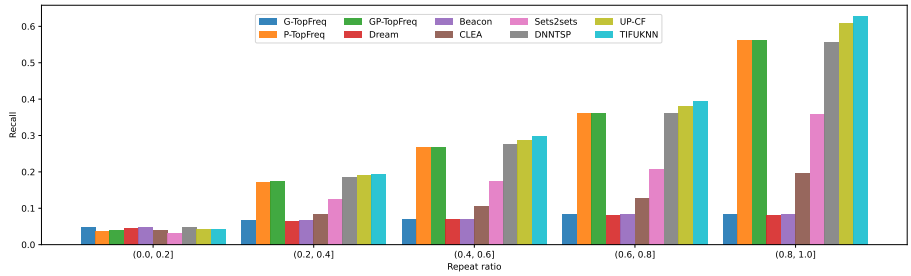
First, we can see that the methods' performance within different user groups is different from the performance computed over all users (Table 2.4). For example, several explore-biased methods (G-TopFreq, Dream, Beacon, CLEA) can outperform recent repeat-biased methods (TIFUKNN, UP-CF@r, Sets2Sets, DNNTSP) in the user group with a low repetition ratio, $[0, 0.2]$, but these explore-biased methods are inferior to the repeat-biased methods when computing the performance over all users. Second, the performance of repeat-biased NBR models increases, as the repetition ratio increases. Interestingly, we observe an analogous trend w.r.t. the performance of explore-biased NBR meth-



(a) Tafeng



(b) Dunnhumby



(c) Instacart

Figure 2.5: Treatment effect for users on the Tafeng, Dunnhumby, and Instacart datasets, for ten NBR methods for users with different repetition ratios (binned in five groups).

ods as the repetition ratio increases, but the rate of the increase is smaller. We believe that this is because the NBR task gets easier for users with a higher repetition ratio, and the repeat-biased methods benefit more from an increase in repetition ratio.

From the perspective of user group fairness, explore-biased methods seem to be fairer than repeat-biased methods across different user groups, as they have a very similar performance across groups. Explore-biased methods have lower variation in performance than repeat-biased methods. However, we should be aware of intrinsic difficulty gaps between different user groups, e.g., it is easier for NBR methods to find correct items for users who like to repeat purchase. Taking this into consideration, we take G-TopFreq and GP-TopFreq as two anchor baselines to evaluate whether recent NBR methods put a specific user group at a disadvantage or not. On the Tafeng and Dunnhumby datasets, repeat-biased methods (Sets2Sets, UP-CF, TIFUKNN, DNNTSP) fail to achieve the performance of G-TopFreq within users whose repetition ratio is in $[0, 0.2]$, which means they do not cater to users of this group. At the same time, recent explore-biased methods (Dream, Beacon, CLEA) fail to achieve the performance derived by the very simple baseline, i.e., GP-TopFreq, on four user groups on Tafeng, Dunnhumby, and Instacart dataset. This analysis indicates that both repeat-biased and explore-biased NBR methods do not treat all user groups fairly.

2.5.6 Looking beyond the average performance

In the recommender systems literature it is customary to compute the average performance over all test users to represent the performance of a recommendation method. Given the diverse treatment effect across different user groups, we want to drill down and see how much the different user groups contribute towards the overall average performance. As before, we use five groups as defined in Section 2.5.5 in terms of the repetition ratio. Specifically, for each individual group g_j, we analyze its *proportion of all users* (PAU) and its *contribution to the average performance* (CAP) as follows:

$$PAU_j = \frac{|U_{g_j}|}{\sum_{j=1}^{q} |U_{g_j}|} \tag{2.7}$$

$$CAP_j = \frac{\sum_{u \in U_{g_j}} Perf_u}{\sum_{j=1}^{q} \sum_{u \in U_{g_j}} Perf_u}, \tag{2.8}$$

where U_{g_j} denotes the set of users in group g_j, q denotes the number of user groups, $Perf_u$ represents the method's performance w.r.t. user u. Note that the performance metric we analyze in this section is *Recall*@10, but similar phenomena can be observed for other metrics.

The results in terms of PAU and CAP are shown in Table 2.6. Under the ideal circumstances, the contribution to the average performance CAP of each user group should be equal to its proportion of all users PAU; this would allow us to use the average performance of a method as its overall performance and leave no user group behind. However, we can see that $CAP_{(0.8,1]}$ is much

higher than $PAU_{(0.8,1]}$ and $CAP_{[0,0.2]}$ is much lower than $PAU_{[0,0.2]}$ for every NBR method (both repeat-biased methods and explore-biased methods) on all datasets. On the Tafeng dataset, only 5.5% of the users belong to group $(0.8, 1]$. However, their contribution to the average performance ranges from 18.8% to 36.8%. On the Dunnhumby dataset, 31.4% of the users belong to group $[0, 0.2]$, while the $CAP_{[0,0.2]}$ for repeat-biased methods (i.e., P-TopFreq, GP-TopFreq, Sets2Sets, DNNTSP, TIFUKNN and UP-CF@r) only ranges from 3.1% to 5.0%. The results reflect that there might be a long-tail distribution w.r.t. the user's contribution to the average performance (i.e., few users contribute a large proportion to the performance), since the NBR task for different users might have different difficulty levels.

Given the previous observations, we construct a simple example to demonstrate the potential limitations of average performance. Assume we have two user groups, i.e., group g_a with 10 users and group g_b with only 1 user, where the NBR task for g_b is easier than g_a. Assume, also, that we have a baseline method M_b whose performance *Perf* can achieve 0.02 in group g_a and 0.4 in group g_b. We have another two optimized methods M_α and M_β. Compared to baseline M_b, M_α can achieve 100% improvement in group g_a, M_β can also achieve 100% improvement in group g_b at the cost of 50% reduction in group g_a. In this case, the cumulative improvement of M_α is $0.02 \times 10 = 0.2$, while the cumulative improvement of M_β is $0.4 \times 1 - 0.01 \times 10 = 0.3$. M_β is considered to be better than M_α, since M_β's average performance is higher. However, we notice that M_α can improve the performance of 10 users, while M_β can only improve the performance of 1 user and at the detriment of the other 10 users. To sum up, the average performance has limitations to represent the performance of methods on the NBR task and it might put users in a specific group at disadvantage.[9] We should calculate the performance of each user group in order to have a comprehensive understanding of the NBR method.

2.5.7 Treatment effect for items with different frequencies

The NBR scenario can be thought of in terms of a two-sided market with items and users [14, 81, 107]. So far, we have analyzed the user-side performance from several aspects. In this section, we analyze treatment effects of NBR methods from the item side. Specifically, we investigate the relation between an item's exposure and its frequency in training labels (the ground-truth items of the training users) or test inputs (the historical items of the test users). As the item exposure in recommended baskets and the item frequency have different scales, we use the exposure and frequency of all items, respectively, to normalize them. In order to visualize the relation between an item's exposure and its frequency, we rank items according to their frequency and select the top-500 items. The frequency and the exposure distributions for different methods on the Tafeng

[9] In this chapter, to remain focused we only analyze the repetition-exploration issue. However, there might be other factors (e.g., basket size, historical basket length) that also influence the difficulty level of NBR problem.

Table 2.6: Group proportion of all users (PAU) and contribution to the average performance (CAP).

Dataset	Method	User group				
		[0.0, 0.2]	(0.2, 0.4]	(0.4, 0.6]	(0.6, 0.8]	(0.8, 1.0]
TaFeng	*PAU*	68.8%	14.9%	8.5%	2.4%	5.5%
	G-TopFreq	53.4%	10.2%	10.4%	3.3%	22.7%
	P-TopFreq	12.3%	20.9%	21.9%	8.2%	36.8%
	GP-TopFreq	21.9%	18.8%	19.5%	7.3%	32.5%
	Dream	60.3%	9.7%	9.0%	2.2%	18.8%
	Beacon	59.0%	9.3%	9.4%	2.5%	19.9%
	CLEA	60.4%	9.3%	9.2%	2.3%	18.9%
	Sets2Sets	27.9%	18.2%	18.3%	6.4%	29.1%
	DNNTSP	37.4%	16.3%	16.0%	5.5%	24.7%
	TIFUKNN	24.2%	19.8%	19.8%	7.1%	29.0%
	UP-CF@r	18.4%	21.2%	21.4%	8.4%	30.8%
Dunnhumby	*PAU*	31.4%	19.6%	21.8%	14.3%	12.9%
	G-TopFreq	16.5%	18.6%	25.2%	18.1%	21.6%
	P-TopFreq	3.1%	14.8%	25.7%	22.6%	33.8%
	GP-TopFreq	4.3%	14.7%	25.5%	22.3%	33.3%
	Dream	16.8%	19.2%	24.9%	17.9%	21.2%
	Beacon	16.8%	18.7%	24.8%	17.9%	21.8%
	CLEA	15.4%	12.2%	20.0%	12.2%	40.1%
	Sets2Sets	4.2%	16.1%	25.1%	19.7%	34.9%
	DNNTSP	5.0%	15.7%	25.6%	22.5%	31.2%
	TIFUKNN	4.9%	15.1%	25.1%	22.0%	32.9%
	UP-CF@r	3.7%	15.3%	25.2%	22.3%	33.5%
Instacart	*PAU*	13.2%	14.9%	20.0%	21.9%	30.0%
	G-TopFreq	8.9%	12.3%	19.7%	24.0%	35.0%
	P-TopFreq	1.6%	7.8%	16.4%	23.7%	50.4%
	GP-TopFreq	1.8%	7.9%	16.4%	23.6%	50.3%
	Dream	9.2%	12.3%	19.8%	23.8%	34.8%
	Beacon	8.9%	12.4%	19.5%	23.5%	35.6%
	CLEA	7.0%	8.6%	14.1%	17.6%	52.6%
	Sets2Sets	2.0%	8.7%	15.8%	21.3%	52.2%
	DNNTSP	2.0%	8.1%	16.5%	23.3%	50.1%
	TIFUKNN	1.8%	8.0%	16.3%	23.6%	50.3%
	UP-CF@r	1.8%	8.0%	16.5%	23.5%	50.2%

dataset are shown in Figure 2.6.[10]

First, we observe the long-tail distribution w.r.t. the item exposure in all NBR methods; a small number of items get a large proportion of the total exposure. Surprisingly, a large proportion of items do not get any exposure in the baskets recommended by Dream, Beacon and CLEA on the TaFeng dataset, which we consider to be sub-optimal from the item perspective. Second, the item exposure distributions of P-TopFreq, TIFUKNN, and UP-CF@r are more related to the frequency distribution of the test input than to the frequency distribution of the training labels. We believe that the repeat-biased nature of those algorithms, as well as the absence of training, results in recommendations that are strongly dependent on the items' frequency in historical baskets, i.e., on the test inputs. Third, in deep learning-based methods (Dream, Beacon, CLEA, Sets2Sets, and DNNTSP), we can see that the distribution of items with high exposure shifts to the left, from Figure 2.6b to Figure 2.6a. This result reflects the fact that an item's high exposure is more closely related to its high frequency in the training labels. To sum up, the item frequency distributions in the training labels and test inputs have a different impact on the item exposure of different NBR methods.

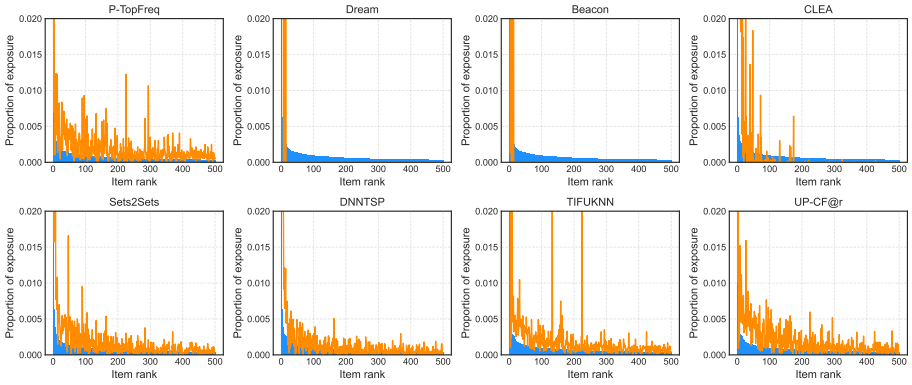
2.5.8 Upshot

Based on our second round of analyses of state-of-the-art NBR methods that we conducted with purpose-built metrics, we observe that there is a clear difficulty gap and trade-off between the repetition task and the exploration task. As a rule of thumb, being biased towards the easier repetition task is an important strategy that helps to boost the overall NBR performance. Deep learning-based methods do not effectively exploit the repetition behavior. Indeed, they achieve a relatively good exploration performance, but they are not able to outperform the simple frequency baseline GP-TopFreq in several cases. Some recent state-of-the-art NBR methods are skewed towards the repetition task and outperform GP-TopFreq. However, the improvements they achieve are limited, especially considering the complexity and computational costs, e.g., for the training process [127] and for hyper-parameters search [36, 52].

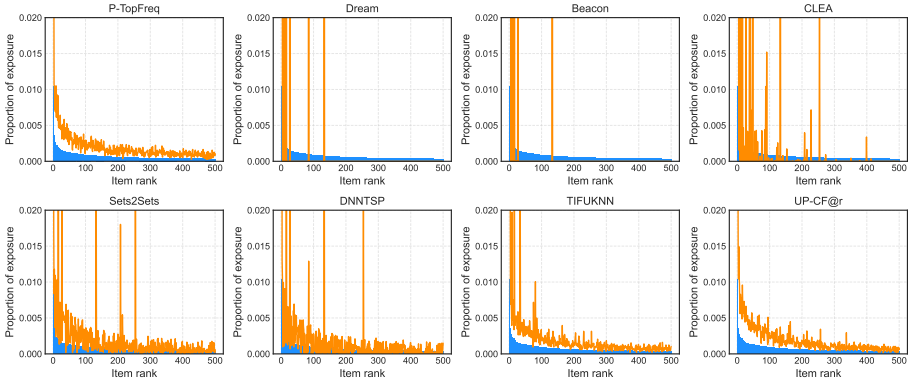
Moreover, current NBR methods usually focus on improving the overall performance, but they often fail to provide, or exploit, deeper insights into the components of their recommended baskets (skewed towards repetition or exploration).

Furthermore, different NBR methods have different treatment effects across different user groups, and the widely-used average performance can not fully evaluate the models' performance, e.g., methods might achieve high overall performance at the detriment of a specific user group, which accounts for a large proportion of all users. From the item-side perspective, few items account for a large proportion of the total exposure in all NBR methods, and some NBR

[10] Experimental results on the Dunnhumby and Instacart datasets are provided in the appendix, and qualitatively similar patterns can be observed. We do not include G-TopFreq in this analysis, since it always recommends top-K items in the historical dataset.



(a) Items ranked according to their frequency in the training labels.



(b) Items ranked according to their frequency in the test inputs (i.e., historical baskets of test users).

Figure 2.6: Treatment effect for items on the Tafeng dataset, for eight NBR methods for items with different frequencies in training labels and testing inputs. The blue bar shows the frequency distribution, and the orange line denotes the exposure distribution.

methods might only recommend a small set of items to users.

2.6 Conclusion

In this chapter, we have re-examined the NBR methods performance to answer the main research question **RQ1**:

> *How to evaluate the next basket recommendation performance from the perspective of repetition and exploration?*

Specifically, we analyzed state-of-the-art NBR methods on the following seven aspects: (i) the overall performance on different scenarios; (ii) the basket components; (iii) the repeat and explore performance; (iv) the contribution of repetition and exploration to the overall performance; (v) the treatment effect for different user groups; (vi) the potential limitations of the average metrics; and (vii) the treatment effect for different items.

2.6.1 Main findings

We arrive at several important findings: (i) No state-of-the-art NBR method, deep learning-based or otherwise, consistently shows the best performance across datasets. Compared to a simple frequency-based baseline, the improvements are modest or even absent. (ii) There is a clear difficulty gap and trade-off between the repeat task and the explore task. As a rule of thumb, being biased towards the easier repeat task is an important strategy that helps to boost the overall NBR performance. (iii) Some NBR methods might achieve better average overall performance at the detriment of a user group with a large proportion of users. (iv) Deep learning-based methods do not effectively exploit repeat behavior. They indeed achieve relatively good explore performance, but are not able to outperform the simple frequency-based baseline GP-TopFreq in terms of the relatively easy repetition task. Some state-of-the-art NBR methods are skewed towards the repeat task and because of this they are able to outperform GP-TopFreq; however, their improvements are limited, especially considering their added complexity and computational costs.

2.6.2 Insights for NBR model evaluation

Our work highlights the following important guidelines that practitioners and researchers working on NBR should follow when evaluating or designing an NBR model: (i) Use a diverse set of datasets for evaluation, with different ratios of repeat items and explore items; (ii) Use GP-TopFreq as a baseline when evaluating NBR methods; (iii) Apart from the conventional accuracy-based metrics, consider the newly introduced repeat and explore metrics, $Recall_{rep}$, PHR_{rep}, $Recall_{expl}$ and PHR_{expl}, as a set of fundamental metrics to understand the performance of NBR methods; (iv) The *RepR* and *ExplR* statistics should be included to understand what kind of items shape the recommended baskets;

and (v) Calculate the performance of each user group to get a comprehensive understanding of the NBR methods.

2.6.3 Insights for NBR model design

From the analysis of this chapter, apart from the difficulty imbalance between the repetition and exploration task, we should also be aware that the repetition recommendation task and exploration recommendation task have different characteristics. For instance, the repetition recommendation task focuses on predicting whether historical items will be repurchased or not, where the frequency and recency of historical items are very important, and the exploration recommendation task focuses on inferring *explore items* from a much bigger search space via modeling item-to-item correlations, which deep-learning methods might be good at. Therefore, just blindly designing complex NBR models without considering the difference between repetition and exploration might be sub-optimal.

We think that it is better to separate the repetition recommendation and exploration recommendation in the NBR task (e.g., using frequency and recency to address the repetition task, and using NN-based models to model item-to-item correlations), which not only allows us to address repetition and exploration according to their respective characteristics but also offers the flexibility of controlling repetition and exploration in the recommended basket. Besides, we also think that future NBR methods should be able to combine repetition and exploration based on users' preferences.

2.6.4 Limitations

One of the limitations of this study is that we did not consider the training and inference execution time in this chapter, which is important for the real-world value of methods used for NBR [6]. We use the original implementations of NBR methods to check their reproducibility and avoid potential mistakes that may come with re-implementations, however, the original implementations are based on different frameworks, which leads to an inability to make a fair execution time comparison. A second limitation is that we only follow the widely used binary definition of repeat items and explore items but do not consider a more fine-grained formalization based on the frequency of historical items, which would allow for a more flexible definition of repetition and analysis. A further limitation is that we only considered the short-term utility of NBR methods: will users be satisfied with their next basket? Limited by our experimental setup, where we replay users' past behavior, we have ignored any potential long-term effects of having a strong focus on short-term utility by emphasizing repeat items as opposed to, for instance, long-term engagement which, likely, benefits from a certain degree of exploration so as to enable surprise and discovery.

2.6.5 Future work

Obvious avenues for future work include addressing the limitations that we have summarized above. Another important line of future work concerns the use of domain-specific knowledge, either concerning complementarity or substitutability of items or concerning hierarchical relations between items, both of which would allow one to consider more semantically informed notions of repeat consumption behavior [4] for next basket recommendation purposes. In addition, our focus in this chapter has been on users – in the sense that we compared methods that produce a basket for a given user –, it would be interesting to consider repetition and exploration aspects of the reverse scenario [67] – given an item, who are the users to whose baskets this item can best be added? Finally, even though we focused on next basket recommendation, it would be interesting to contrast our outcomes with an analysis of repeat and explore behavior in traditional sequential recommendation scenarios.

In the next chapter, we will focus on the exploration task in the basket recommendation scenario.

2.7 Appendix

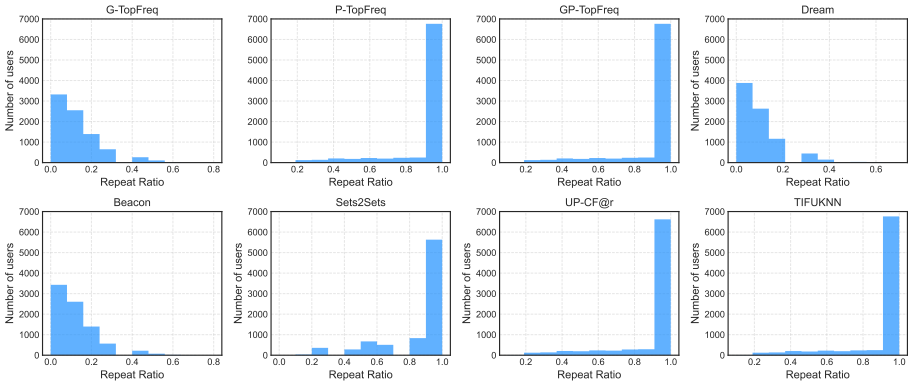
2.7.1 Additional plots

We include additional plots to supplement Figure 2.3 (Figure 2.7 below) and 2.6 (Figures 2.8 and 2.9), respectively.

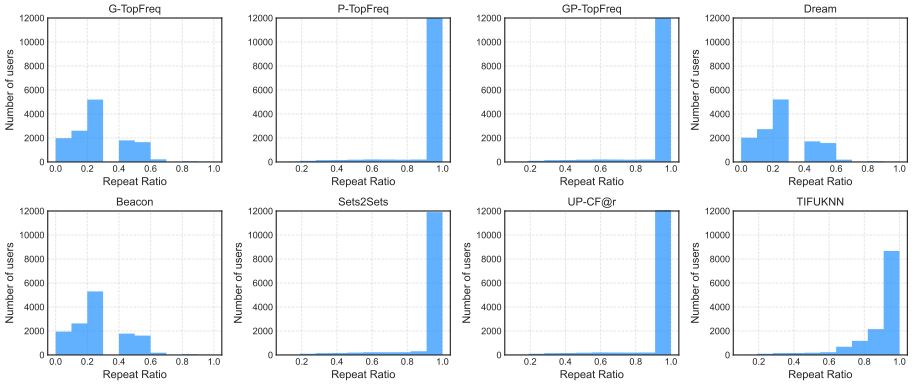
2.7.2 Reproducibility

To facilitate the reproducibility of the results, we release our online repository,[11] which contains the following resources: (i) source code and datasets; (ii) descriptions of different dataset format; and (iii) pipelines about how to run and get results.

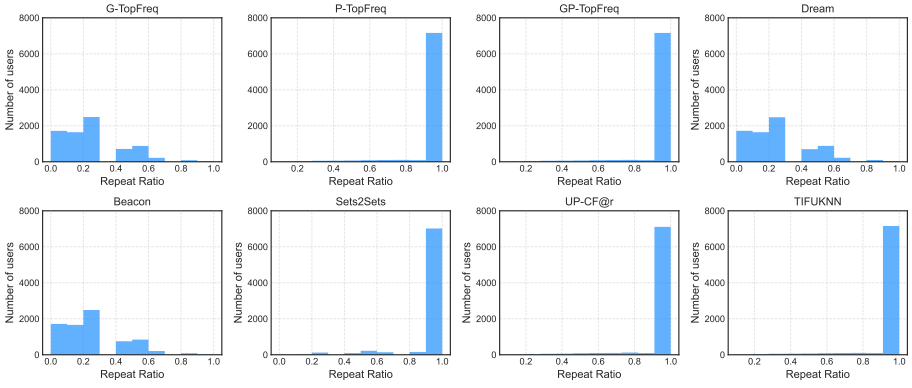
[11] `https://github.com/liming-7/A-Next-Basket-Recommendation-Reality-Check`



(a) TaFeng

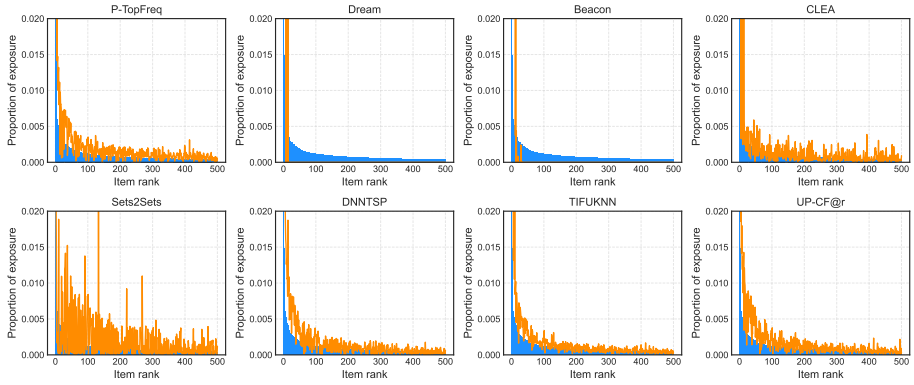


(b) Dunnhumby

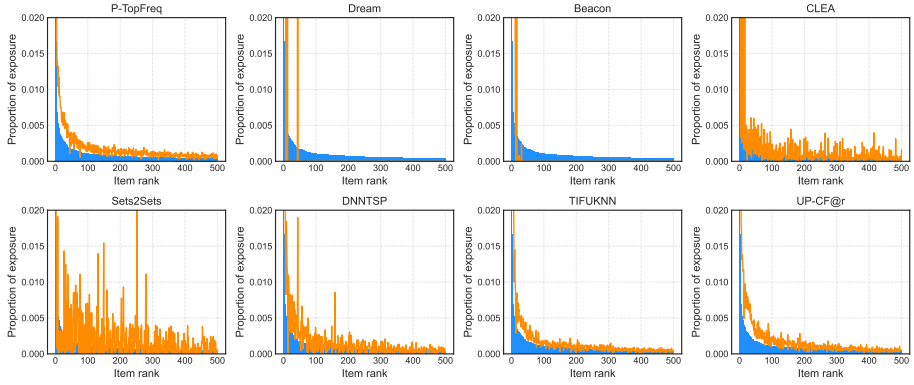


(c) Instacart

Figure 2.7: Distribution of the repetition ratio *RepR* of recommended baskets on TaFeng dataset for eight NBR methods (G-TopFreq, P-TopFreq, GP-TopFreq, Dream, Beacon, Sets2Sets, Up-CF@r, TIFUKNN).

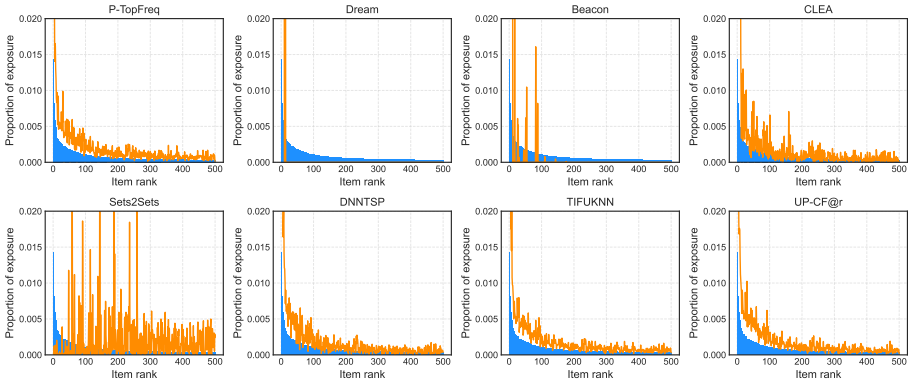


(a) Items ranked according to their frequencies in the training labels.

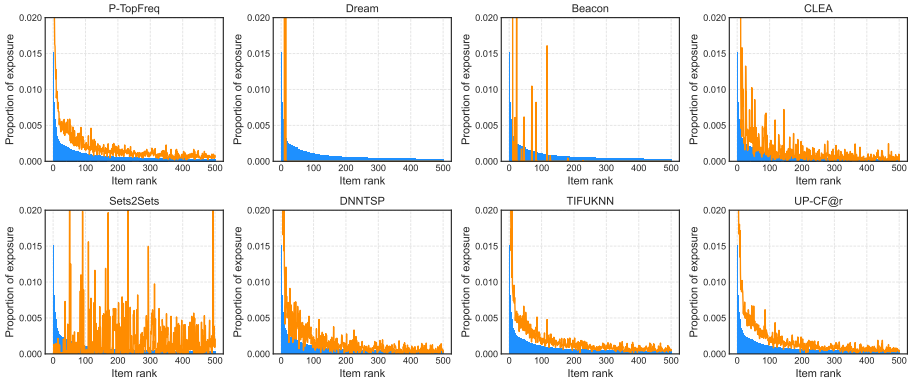


(b) Items ranked according to their frequencies in the testing inputs (i.e., historical baskets of testing users).

Figure 2.8: Treatment effect for items on the Dunnhumby dataset, for eight NBR methods for items with different frequencies in training labels and testing inputs. The blue bar shows the frequency distribution, and the orange line denotes the exposure distribution.



(a) Items ranked according to their frequencies in the training labels.



(b) Items ranked according to their frequencies in the testing inputs (i.e., historical baskets of testing users).

Figure 2.9: Treatment effect for items on the Instacart dataset, for eight NBR methods for items with different frequencies in training labels and testing inputs. The blue bar shows the frequency distribution, and the orange line denotes the exposure distribution.

3 Next Novel Basket Recommendation

From the previous chapter, we conclude that it is useful to distinguish between repeat items, i.e., items that a user has consumed before, and explore items, i.e., items that a user has not consumed before. However, most NBR work either ignores this distinction or focuses on repeat items. In this chapter, we formulate the *next novel basket recommendation* (NNBR) task, i.e., the task of recommending a basket that only consists of *novel items*, which is valuable for both real-world application and NBR evaluation.

In particular, we address the thesis-level research question **RQ2**:

> *How to design basket recommendation models targeted at the exploration task, and how to optimize the model to explore items in a scenario with many repetition signals?*

3.1 Introduction

Next basket recommendation is a type of sequential recommendation that aims to recommend the next basket, i.e., set of items, to users given their historical basket sequences. Recommendation in a grocery shopping scenario is one of the main use cases of the NBR task, where users usually purchase a set of items instead of a single item to satisfy their diverse needs. Many methods, based on diverse underlying techniques (i.e., RNNs [9, 51, 62, 84, 125], self-attention [22, 93, 127], and denoising via contrastive learning [84]), have been proposed for, and achieve good performance on, the NBR task.

3.1.1 Next novel basket recommendation

The previous chapter published as [69] offers a new evaluation perspective on the next basket recommendation (NBR) task by distinguishing between *repetition* (i.e., recommending items that users have purchased before) and *exploration*

This chapter was published as: M. Li, M. Ariannezhad, A. Yates, and M. de Rijke. Masked and swapped sequence modeling for next novel basket recommendation in grocery shopping. In *RecSys 2023: 17th ACM Conference on Recommender Systems*, pages 35–46. ACM, September 2023.

Table 3.1: Three types of basket recommendation.

Task	Target items	Recommended basket
NBR	Repeat items & novel items	Repeat items & novel items
NBRR	Repeat items	Only repeat items
NNBR	Novel items	Only novel items

(i.e., recommending items that are new to the user) tasks in NBR and points out the imbalance in difficulty between the two tasks. According to the analysis of existing methods in [69], the performance of many existing NBR methods mainly comes from being biased towards (i.e., giving more resource to) the repetition task and sacrificing the ability of exploration. Building on these insights, the recent literature on NBR has seen a specific focus on the pure repetition task as well the introduction of specific methods for the repetition task [4, 59].

Novelty and serendipity are two important objectives when evaluating recommendation performance [47, 57]. People might simply get tired of repurchasing the same set of items. Even when they engage in a considerable amount of repetition behavior, there is still a large proportion of users who would like to try something new when shopping for grocery [69]. This phenomenon is especially noticeable for users with fewer transactions in their purchase history [4]. Therefore, one of the key roles of recommender systems is to assist users in discovering potential novel items that align with their interests. However, in contrast to the pure repetition task, the pure exploration task in NBR remains under-explored. Besides, due to the difference in difficulty between the two tasks, many online e-commerce and grocery shopping platforms have started to design "buy it again" service to isolate repeat items from the general recommendation.[1,2]

Motivated by the research gaps and real-world demands, we formulate the *next novel basket recommendation* (NNBR) task, which focuses on recommending a novel basket, i.e., a set of items that are new to the user, given the user's historical basket sequence. Different from the repetition task which predicts the probability of repurchase from a relatively small set of items, the NNBR task needs to predict possible items from many thousands of candidates by modeling item-item correlations, which is more complex and difficult [69]. NNBR is especially relevant to the "Try Something New" concept in the grocery shopping scenario. Table 3.1 compares three types of basket recommendation.

[1]After login, users may see a "buy it again" page on e-commerce platforms (see, e.g., Amazon, `https://amazon.com`) and grocery shopping platforms (see, e.g., Picnic, `https://picnic.app`), where the platform collects repeat items. Similarly, in the grocery shopping scenario, "Try Something New" services also exist, where only novel items are recommended to the user.

[2]See, e.g., `http://community.apg.org.uk/fileUploads/2007/Sainsburys.pdf` for an example of the "Try Something New" concept in offline retail, and the Weekly New Recipe service at `https://ah.nl/allerhande/wat-eten-we-vandaag/weekmenu` for an example in online retail.

3.1.2 From NBR to NNBR

The NNBR task can be seen as a sub-task of the conventional NBR task, in which NBR methods are designed to find all possible items (both *repeat items* and *novel items*) in the next basket. Therefore, it is possible to generate a novel basket by only selecting top-k novel items predicted by NBR methods. To modify NBR methods for the NNBR task, an intuitive solution is to remove the repeat items from the ground-truth labels and train models only depending on the novel items in the ground-truth labels. Given this obvious strategy and given that many methods have already been proposed for NBR, an important question is:

> *If we already have an NBR model, do we need to train another model specifically for the NNBR task?*

Surprisingly, we find that training specifically for exploration does not always lead to better performance in the NNBR task, and might even reduce performance in some cases.

3.1.3 BTBR: Bi-directional transformer basket recommendation

In NNBR, item-to-item correlations are especially important, since we need to infer the utility of new items based on previously purchased items. Besides, a single basket is likely to address diverse needs of a user [105]; e.g., what a user would like to drink is more likely to depend on what he or she drank before rather than on the tooth paste they previously purchased. However, most existing NBR approaches [51, 62, 84, 125, 127] are two-stage methods, which first generate a basket-level representation [99], and then learn a temporal model based on basket-level representations, which will lead to information loss w.r.t. item-to-item correlations [62, 93, 127]. Some methods [62, 93, 127] learn partial item-to-item correlations based on the co-occurrence within the same or adjacent basket as auxiliary information beyond basket-level correlation learning. Instead of learning or exploiting complex basket representations, we learn item-to-item correlations from direct interactions among different items across different baskets. To do so, we propose a bi-directional transformer basket recommendation model (BTBR) that adopts a bi-directional transformer [101] and uses the shared basket position embedding to indicate items' temporal information.

3.1.4 Masking and training

To properly train BTBR, we propose and investigate several masking strategies and training objectives at different levels and tasks, as follows: (i) item-level random masking: a cloze-task loss [33, 98], in which we randomly mask the historical sequence at the item level; (ii) item-level select masking: a cloze-task loss designed for exploration, in which we first select the items we need to mask and then mask all the occurrences of the selected item; (iii) basket-level all

masking: a general basket recommendation task loss, in which we mask and predict the complete last basket at the end of the historical sequence; (iv) basket-level explore masking: an explore-specific basket recommendation task loss, in which we remove the repeat items and only mask the novel items in the last basket of the historical sequence; and (v) joint masking: a loss that follows the pre-train-then-fine-tune paradigm, in which we first adopt item-level masking for the cloze task, then fine-tune the model using basket-level masking.

In addition, conventional sequential item recommendation usually assumes that the items in a sequence are strictly ordered and sequentially dependent. However, recent work [e.g., 24, 83, 110, 120] argues that the items may occur in any order, i.e., the order is flexible, and ignoring flexible orders might lead to information loss. Similarly, it is unclear whether the items that are being purchased across baskets have a strict order in the grocery shopping scenario. Thus, we propose an item swapping strategy that allows us to randomly move an item to another basket according to a certain ratio, which can enrich item interactions within the same basket.

We conduct extensive experiments on three publicly available grocery datasets to understand the effectiveness of the BTBR model and the proposed strategies on datasets with various repeat ratios and characteristics.

3.1.5 Main contributions

The main contributions of this chapter are:

- To the best of our knowledge, we are the first to formulate and investigate the next novel basket recommendation (NNBR) task, which aims to recommend a set of novel items that meet users' preferences in the next basket.
- We investigate the performance of several representative NBR methods w.r.t. the NNBR task and find (i) that training specifically for the exploration task does not always lead to better performance, and (ii) that limited progress has been made w.r.t. the NNBR task.
- We propose a simple bi-directional transformer basket recommendation (BTBR) model that learns item-to-item correlations across baskets.
- We propose several types of masking and item swapping strategies for optimizing BTBR for the NNBR task. Extensive experiments are done on three open grocery shopping datasets to assess the effectiveness of the proposed strategies. BTBR with a proper masking and swapping strategy is the new state-of-the-art method w.r.t. the NNBR task.

3.2 Related Work

In this section, we describe two lines of research in the recommender systems literature that are related to our work: sequential recommendation and next

basket recommendation.

3.2.1 Sequential recommendation

Sequential item recommendation has been widely studied for many years, and models [48, 49, 58, 65, 73, 92, 96, 117] with various deep learning techniques, e.g., RNN [48, 49], CNN [96], GNN [89, 117], contrastive learning [120], attention [65, 73] and self-attention [58, 92, 101] mechanism are proposed. Self-attention (Transformer) model [101] with multi-head attention is first proposed and shows strong performance in natural language processing, and SASRec [58] is the first sequential recommendation model that employs the self-attention mechanism. Later, BERT4Rec [92] upgrades the left-to-right training scheme in SASRec and uses a bi-direction transformer with a Cloze task [98], which is the closest sequential recommendation method to this chapter. Motivated by the success of BERT4Rec, some follow-up work has applied masked-item-prediction training to more specific scenarios [126].

However, BERT4Rec and follow-up work only focus on the item sequential recommendation with only random masking during training [126]. We extend BERT4Rec to the basket sequence setting and propose several types of masking strategies and training objectives that are specifically designed for the NNBR task. Furthermore, in this work we study the next novel basket recommendation task, where both historical interactions and the predicted target are baskets (sets of items). None of the sequential recommendation models listed above are designed to handle a sequence of baskets.

3.2.2 Next basket recommendation

Next basket recommendation is another sequential recommendation task that addresses the sequence of baskets in the grocery shopping scenario. Existing methods can be classified into three types: frequency neighbor-based methods [36, 52], Markov chain (MC)-based methods [88], and deep learning-based methods [4, 9, 22, 51, 59, 62, 63, 84, 93, 105, 109, 112, 125, 127]. Recently, Li et al. [69] have evaluated and assessed NBR performance from a new repetition and exploration perspective; they find that the repetition task, i.e., recommending repeat items, is much easier than the exploration task, i.e., recommending explore items (a.k.a. novel items in this chapter), besides the improvements of many recent methods come from the performance of the repetition task rather than better capturing correlations among items. Inspired by this finding, an NBR method [4] that only models the repetition behavior has been proposed, and an NBRR task [59] that only focuses on recommending repeat items has been formulated.

In this chapter, we propose and formulate the next novel basket recommendation task that focuses on recommending novel items to the user, whereas all of the NBR methods mentioned above focus on the conventional NBR, and their performance when generalized to the NNBR task remains unknown.

3.3 Problem Formulation

In this section, we describe and formalize the next novel basket recommendation task which is the focus of this chapter.

Formally, given a set of users $U = \{u_1, u_2, \ldots, u_n\}$ and a set of items $I = \{i_1, i_2, \ldots, i_m\}$, $S_u = [B_u^1, B_u^2, \ldots, B_u^t]$ represents the historical interaction sequence for user u, where B_u^t represents a set of items $i \in I$ that user u purchased at time step t. For a user u, the *repeat item* i_u^{rep} is the item that user u has purchased before, which is defined as $i_u^{rep} \in I_u^{rep} = B_u^1 \cup B_u^2 \cup \ldots \cup B_u^t$, and the novel item i_u^{novel} is the item that user u have not purchased before, i.e., $i_u^{novel} \in I_u^{novel} = I - I_u^{rep}$.

The goal of the *next novel basket recommendation* task is to predict the following novel basket which only consists of novel items i_u^{novel} that the user would probably like, based on the user's past interactions S_u, that is,

$$P_u = \hat{B}_u^{t+1} = f(S_u) \tag{3.1}$$

where P_u denotes a recommended item list that *only consists of the novel items* i_u^{novel} *of user* u.

3.4 Our Method

In this section, we first describe the base bi-directional transformer basket recommendation model (BTBR) we use, then introduce several types of masking strategies for the NNBR task, and finally describe the item swapping strategy.

3.4.1 Bi-directional transformer basket recommendation model

Learning basket representations [99] and modeling temporal dependencies across baskets are two key components in almost all neural-based NBR methods. Many NBR methods introduce complex architectures to learn representations for baskets in grocery shopping [22, 62, 84, 93, 127]. Instead of proposing more complex architectures to learn better basket representations and temporal dependencies, we want to simplify the model and only focus on the item-level correlations across different baskets, which helps us to infer novel items from users' historical items.

As a widely used method to model temporal dependencies, a recurrent neural network (RNN) [25, 41] requires passing information sequentially according to the temporal order, whereas there is no temporal order for items within the same basket, and basket-level representations at each timestamp are required [51, 62, 84, 125]. Another alternative method is the self-attention mechanism (a.k.a. transformer) [101], which is capable of learning the representations of every position via exchanging the information across all positions. Therefore, we adopt the bi-directional transformer [33, 101] as the backbone of our BTBR model, which not only allows us to learn item-to-item correlations from the direct interactions among items across different baskets but also is able to handle

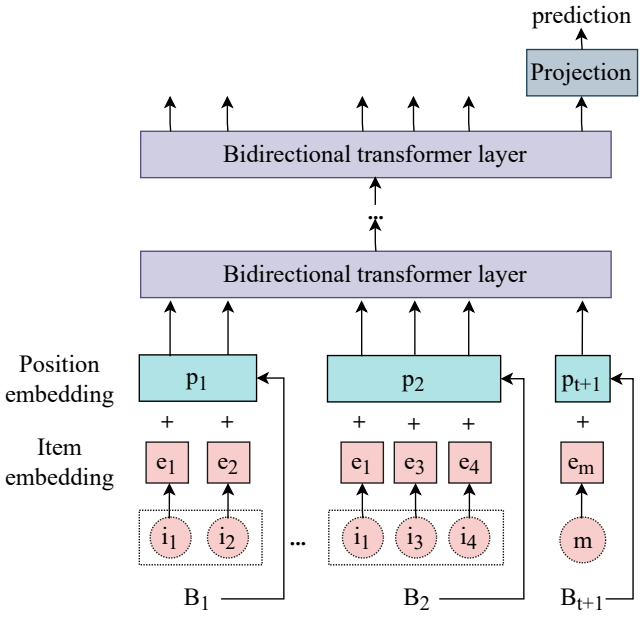


Figure 3.1: The overall architecture of the BTBR model.

basket sequence information in grocery shopping. The overall architecture of BTBR is shown in Figure 3.1.

Embedding layer. In order to use transformers [101] for NNBR, we first transfer the basket sequence to the item sequence via a "flatten" operation, e.g., $[\{i_1, i_2\}, \{i_1, i_3, i_4\}] \rightarrow [i_1, i_2, i_1, i_3, i_4]$. It has been shown that the positions of items are informative in the sequential recommendation scenario [58, 92]. Different from the solutions in conventional item sequential recommendation, where each item is combined with its unique position embedding w.r.t. its position in the item sequence, we use learnable position embedding for every basket, and items within the same basket will share the same position embedding. For example, given a basket sequence $S = [\{i_1, i_2\}, \{i_1, i_3, i_4\}, \{i_4, i_5\}]$, we first flatten S and get a sequence of item embedding $E_i = [e^i_1, e^i_2, e^i_1, e^i_3, e^i_4, e^i_4, e^i_5]$, and get a position embedding sequence $E_p = [e^p_1, e^p_2, e^p_3]$, finally the input sequence of transformer layer will be $E_{i,p} = [e^i_1 + e^p_1, e^i_2 + e^p_1, e^i_1 + e^p_2, e^i_3 + e^p_2, e^i_4 + e^p_2, e^i_4 + e^p_3, e^i_5 + e^p_3]$. Note that, the padding and truncating operation is also employed to handle the sequences of various lengths.

Bi-directional transformer layer. The transformer architecture contains two sub-layers:

1. *Multi-head attention layer*, which adopts the popular attention mechanism [101] and aggregates all items' embeddings across different baskets with adaptive weights.

2. *Point-wise feed-forward layer*, which aims to endow nonlinearity and interactions between different latent dimensions.

We use stacked transformer layers to learn more complex item-to-item correlations, that is:

$$H^1 = \mathrm{Trm}(E_{i,p}), \ldots, H^L = \mathrm{Trm}(H^{L-1}), \tag{3.2}$$

where Trm denotes the bi-directional transformer layer, $H^L = [h_1^L, h_2^L, \ldots, h_d^L]$ denotes a representation sequence derived from the last transformer layer, and d denotes the maximum sequence length of input sequence $E_{i,p}$. Besides, residual connections [45], dropout [90], layer normalization [8], and GELU activation [46] are adopted to enhance the ability of representation learning. For more details about the bi-directional transformer layer, we refer to [58, 92, 101].

Prediction layer. After hierarchically exchanging information of all items across baskets using the transformer, we get $H^L \in \mathbb{R}^{m \times d}$, which contains the corresponding representations h^L for all items in the input sequence. Following [58, 92], we use the same item embedding $E_I \in \mathbb{R}^{m \times d}$ as the input layer to reduce the model size and alleviate the overfitting problem. For a masked position (item), we get its learned representation $h \in \mathbb{R}^d$ and compute the interaction probability distribution p of candidate items by:

$$p = \mathrm{Softmax}(hE^{\mathrm{T}} + b), \tag{3.3}$$

where E is the embedding matrix for candidate items and b denotes a bias term.

3.4.2 Masking strategy

Since there are repetition signals in the basket sequence, it is unclear whether these signals are merely noise/shortcuts or contain valuable information for the task of recommending novel items. After constructing the base model (BTBR), the challenging problem that needs to be addressed is how to properly train the model to improve its ability of finding novel items that meet users' interests. In this section, we propose four types of alternative masking strategies for the next novel basket recommendation task considering different tasks and levels, as well as the repetition-exploration signals. Figure 3.2 shows examples of four types of masking strategies and Table 3.2 shows the characteristics of different training strategies.

Cloze task. The first type of training objective is a cloze task [98], i.e., "masked language model" in [33]. Specifically, we mask a proportion of items in the input sequence, i.e., replace each of them with a "mask token", and then try to predict the original items based on their contexts. We call this masking "item-level". Two main advantages of this item-level masking & training strategy are (i) it allows us to generate more item-level training samples by breaking the definition of "basket", and (ii) it learns both sides' information via the bi-directional transformer, which might allow the model to better capture item-to-item correlations. We first introduce two item-level masking strategies as follows:

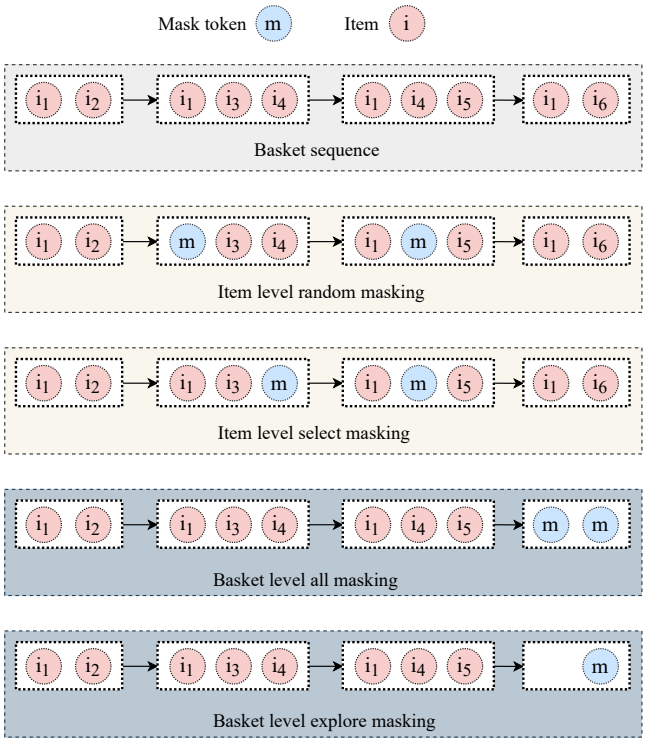


Figure 3.2: The original basket sequence (at the top) and four types of masking strategies.

1. *Random*: This is a conventional masking strategy, which has been adopted in BERT4Rec [92]. Specifically, given a flattened item sequence, we randomly select several positions of the sequence and mask the corresponding items of the selected position according to mask ratio α as input.

2. *Select*: One potential issue w.r.t. the above *Random* masking is that the masked items (prediction target) might still exist in the non-masked positions, so the model might mainly predict the masked item via its repetition information rather than inferring new items based on item-to-item correlations. Therefore, we propose the select masking strategy, which is specifically targeted at the exploration demand of the NNBR task. Specifically, given a flattened item sequence, we first derive the item set I in this sequence, then randomly select several items $i_m \in I$ according to mask ratio α, and finally mask all the occurrences of i_m in the sequence. Since there is no repetition information available, the model can only infer the targeted items, i.e., novel items, via learning the item-to-item correlations.

Table 3.2: Comparison of four types of masking strategies from three aspects, i.e., temporal orders, explore specific and amount of training signals.

Strategy	Strict temporal order	Explore specific	Number of training signals[3]
Item-Random	×	×	***
Item-Explore	×	✓	***
Basket-All	✓	×	**
Basket-Explore	✓	✓	*

Basket recommendation task. Using the cloze task as the learning objective has limitations: (i) it is not able to fully respect the temporal dependencies of a sequence, since we can only use the historical information (left-side context) when we make the recommendation; and (ii) it is not specifically designed for the basket recommendation task and a mismatch might exist. Therefore, the second type of training objective we consider is the basket recommendation task, which masks the input sequence at the basket-level instead of item-level. Specifically, we mask the last basket and try to predict the items in this basket only based on the historical items (left-side information). Similarly, we propose another two basket-level masking strategies as follows:

1. *All*: This masking strategy can be regarded as optimizing the model for the NBR task. Given a flattened item sequence, we find and mask all items, i.e., both novel items and repeat items in the last basket.

2. *Explore*: This is a NNBR-specific masking strategy. Given a flattened item sequence, we find the items in the last basket, instead of masking all items, we only mask the novel items $i \in I^{novel}$ and remove the *repeat items* $i \in I^{rep}$. The model will be only optimized for finding all novel items in the future based on the historical basket sequence.

Joint task. The pretrain-then-finetune paradigm has been widely adopted in NLP. Item-level masking (the cloze task) and basket-level masking (the basket recommendation task) can also be combined as a joint masking strategy to employ the pretrain-then-finetune paradigm in NNBR, which first uses item-level masking strategy (i.e., self-supervised task) to get item correlations as the pre-train stage and then employ basket-level masking strategy (i.e., supervised task) to finetune it for basket recommendation.

Loss. Following [92], we select minimizing the negative log-likelihood loss as the training objective:

$$\mathcal{L} = \frac{1}{|I^m|} \sum_{i \in I^m} -\log p(i \mid S_u), \tag{3.4}$$

[3]More *'s indicate more training signals. Item-level masking can be seen as self-supervised learning, which is more flexible and can leverage more training signals than basket-level masking. Basket-explore has the fewest training signals as it can only use the novel items in the last basket.

where I^m is the masked item set, $p(i \mid S_u, t)$ is the predicted probability of item i at position t.

Test and prediction. To predict a future basket (a set of items), we only need to add one masked token at the end of the user's item sequence, since items within the same basket share the same position embedding. In the NNBR task, the candidate items are novel items I^{new} that the user has not bought before, thus we use the embedding matrix w.r.t. the novel items of every user to compute the probabilities according to Eq. 3.3. Finally, we select top-K novel items with the highest scores as the recommendation list of the next novel basket.

3.4.3 Swapping strategy

In sequential recommendation, some work [24, 83, 110, 120] argues that the items in a sequence may not be sequentially dependent and different item orders may actually corresponding to the same user intent. Ignoring flexible orders in sequential recommendation might lead to less accurate recommendations for scenarios where many items are not sequentially dependent [83, 110, 126]. In grocery shopping, the items purchased within the different baskets might not have rigid orders. To further understand if considering the flexible orders among items could further improve the performance w.r.t. the NNBR task, we propose the item swapping strategy to create augmentations for the BTBR.

Specifically, as illustrated in Figure 3.3, we randomly select items according to a swap ratio λ and then move them to another basket to enrich the items' interactions within the same basket. Besides, we introduce a hyper-parameter, i.e., swap hop γ, to control the basket distance of the swapping strategy. Note that, we only perform the local swap strategy when using item-level masking (the cloze task) to train the model, since basket-level masking (the basket recommendation task) is designed to respect the sequential order and predict the future basket based on historical information.

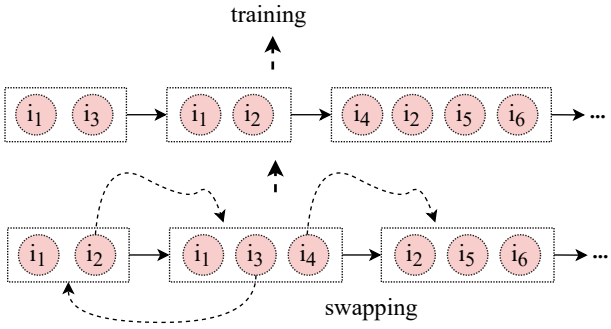


Figure 3.3: An example of the item swapping strategy.

3.5 Experimental Setup

3.5.1 Research questions

To comprehensively understand the next novel basket recommendation task, and evaluate the performance of BTBR with different strategies, we decompose the thesis-level research question **RQ2** into the following questions:

RQ2.1 How do existing NBR models perform w.r.t. the NNBR task? Does training specifically for the NNBR task lead to better performance?

RQ2.2 How does BTBR with different masking strategies perform compared to the state-of-the-art models?

RQ2.3 Does the swapping strategy contribute to the improvements?

RQ2.4 How do the hyper-parameters influence the models' performance and how different masking strategies affect the training dynamics?

RQ2.5 Is the joint masking strategy more robust than using the single masking strategy?

3.5.2 Datasets.

We evaluate the NNBR task on three publicly available grocery shopping datasets (TaFeng,[4] Dunnhumby,[5] and Instacart[6]), which vary in their repetition and exploration ratios. Following [69], we sample users whose basket length is between 3 and 50, and remove the least frequent items in each dataset. We also focus on the fixed size (10 or 20) next novel basket recommendation problem. In our experiments, we split the dataset across users, 80% for training, and 20% for testing, and leave 10% of the training users as the validation set. We repeat the splitting and experiments five times and report the average performance. The statistics of the processed datasets are shown in Table 3.3.

Table 3.3: Statistics of the processed datasets.

Dataset	#items	#users	Avg. basket size	Avg. #baskets per user	repeat ratio	explore ratio
TaFeng	11,997	13,858	6.27	6.58	0.188	0.812
Dunnhumby	3,920	22,530	7.45	9.53	0.409	0.591
Instacart	13,897	19,435	9.61	13.21	0.597	0.403

[4] https://www.kaggle.com/chiranjivdas09/ta-feng-grocery-dataset
[5] https://www.dunnhumby.com/source-files/
[6] https://www.kaggle.com/c/instacart-market-basket-analysis/data

3.5.3 Baselines

We investigate the performance of six NBR baselines, which we select based on their performance on our chosen datasets in the analysis performed in [69]. Importantly, for a fair comparison, we do not include methods that leverage additional information [9, 22, 93].

- **G-TopFreq:** G-TopFreq uses the k most popular items in the dataset to form the recommended next basket, which is widely used in recommendation due to its effectiveness and simplicity.
- **TIFUKNN:** TIFUKNN [52] is a state-of-art method that models the temporal dynamics of frequency information of users' past baskets to introduce personalized frequency information (PIF), then it uses a KNN-based method on the PIF.
- **Dream:** Dream [125] models users' global sequential basket behavior for NBR. It uses a pooling strategy to generate basket representations, which are then fed into an RNN to learn user representations and predict the corresponding next set of items.
- **DNNTSP:** DNNTSP [127] is a state-of-art method that leverages a GNN and self-attention techniques. It encodes item-item relations via a graph and employs a self-attention mechanism to capture temporal dependencies of users' basket sequences.
- **Beacon:** Beacon [62] is a RNN-based method that encodes the basket considering the incorporating information on pairwise correlations among items.
- **CLEA:** CLEA [84] is a state-of-art RNN-based method that posits that not all items contribute to the next move, and uses a contrastive learning model to automatically extract items relevant to the target item and get the representation via a GRU-based encoder.

Note that, for the above baseline models (except G-TopFreq), we have two versions with different training methods, i.e., using all items in the last basket as training labels (Train-all), and only using novel items in the last basket as training labels (Train-explore).

3.5.4 Configurations

For the training-based baseline methods and TIFUKNN, we strictly follow the hyper-parameter setting and tuning strategy of their respective original papers, the embedding size is tuned on $\{16, 32, 64, 128\}$ for all training-based methods based on the validation set to achieve their best performance.

We use PyTorch to implement our model and train it using a TITAN X GPU with 12G memory. For BTBR, we set self-attention layers and their head number to 2, and tune the embedding size on $\{16, 32, 64, 128\}$. The Adam

optimizer with a learning rate of 0.001 is used to update parameters. We set the batch size to 128 for the Tafeng and Dunnhumby datasets, and 64 for the Instacart dataset; we sweep the mask ratio α in $\{0.1, 0.3, 0.5, 0.7, 0.9\}$, local swap ratio in $\{0, 0.1, 0.3, 0.5, 0.7, 0.9\}$ and swap hop γ in $\{1, 3, 5, 7, 9\}$. We share both our processed dataset and the source code.[7]

3.5.5 Metrics

Two widely used metrics for the NBR problem are *Recall@k* and *nDCG@k*. In the NNBR task, *Recall* measures the ability to find all novel items that a user will purchase in the next basket; *NDCG* is a ranking metric that also considers the order of these novel items, i.e.,

$$\text{Recall@}K = \frac{1}{|U|} \sum_{u \in U} \frac{\left|P_u \cap T_u^{novel}\right|}{|T_u^{novel}|}, \tag{3.5}$$

$$\text{nDCG@}K = \frac{1}{|U|} \sum_{u \in U} \frac{\sum_{k=1}^{K} p_k / \log_2(k+1)}{\sum_{k=1}^{\min(K, |T_u^{novel}|)} 1 / \log_2(k+1)}, \tag{3.6}$$

where U is a set of users who will purchase novel items in their next basket, T_u^{novel} is a set of ground-truth novel items of user u, p_k equals 1 if $P_u^k \in T_u^{novel}$, otherwise $p_k = 0$. P_u^k denotes the k-th item in the predicted basket P_u. Note that, some methods might assign high scores w.r.t. the repeat items [69], to generate a novel basket, we fully remove the repeat items, then only rank and select top-k novel items as the recommended basket P_u to ensure a fair comparison, i.e., the recommended basket *only consists top-k novel items*.

3.6 Experimental Results

3.6.1 Train-all and Train-explore

To answer RQ2.1, we employ two training strategies for each baseline method: (i) *Train-all*: we keep both repeat items and explore items as part of the ground-truth labels during training, which means that the model is trained to find all possible items in the next basket; and (ii) *Train-explore*: we remove the repeat items and only keep novel items in the ground-truth labels during training, which means the model is specifically trained to find novel items in the next basket. For the NNBR performance evaluation, we assess the models' ability to find novel items, which means the recommended novel basket consists of top-k novel items and there are no repeat items. We report the experimental results of different baseline methods in Table 3.4. We have three main findings.

First, we can see that no method can consistently outperform all other methods across all datasets. On the Tafeng dataset, several NN-based methods (Dream-all, Dream-explore, Beacon-all, Beacon-explore, DNNTSP-all, CLEA-explore) fall in the top-tier methods group with quite good performance. On the

[7] See `https://github.com/liming-7/Mask-Swap-NNBR`

Table 3.4: Results of methods training for finding novel items, i.e., Train-explore compared against the methods training for finding all items, i.e., Train-all. Boldface and underline indicate the best and the second best-performing performance w.r.t. the NNBR task, respectively. Significant improvements and deteriorations of Train-explore over the corresponding Train-all baseline results are marked with $^{\uparrow}$ and $^{\downarrow}$, respectively. (paired t-test, $p < 0.05$).

Dataset	Metric	Train	G-Pop	TIFUKNN	Dream	Beacon	CLEA	DNNTSP
Tafeng	Recall@10	all	0.0587	0.0714	0.0960	0.0926	0.0870	**0.1024**
		explore	=	0.0911$^{\uparrow}$	<u>0.1021</u>$^{\uparrow}$	0.0967$^{\uparrow}$	0.1010$^{\uparrow}$	0.0940$^{\downarrow}$
	nDCG@10	all	0.0603	0.0662	0.0823	0.0789	0.0755	<u>0.0855</u>
		explore	=	0.0783$^{\uparrow}$	**0.0859**$^{\uparrow}$	0.0819$^{\uparrow}$	0.0857$^{\uparrow}$	0.0767$^{\downarrow}$
	Recall@20	all	0.0874	0.0926	0.1244	0.1252	0.1150	0.1245
		explore	=	0.1157$^{\uparrow}$	0.1244	**0.1257**	<u>0.1253</u>$^{\uparrow}$	0.1168$^{\downarrow}$
	nDCG@20	all	0.0703	0.0738	0.0928	0.0909	0.0861	<u>0.0943</u>
		explore	=	0.0876$^{\uparrow}$	0.0939	0.0929	**0.0952**$^{\uparrow}$	0.0858$^{\downarrow}$
Dunnhumby	Recall@10	all	0.0468	0.0497	0.0494	0.0499	0.0499	0.0514
		explore	=	0.0498	0.0506	**0.0529**$^{\uparrow}$	<u>0.0520</u>$^{\uparrow}$	0.0472$^{\downarrow}$
	nDCG@10	all	0.0397	0.0409	0.0409	0.0411	0.0376	<u>0.0415</u>
		explore	=	0.0411	0.0385	**0.0428**$^{\uparrow}$	0.0404$^{\uparrow}$	0.0378$^{\downarrow}$
	Recall@20	all	0.0701	0.0745	0.0744	0.0804	0.0711	0.0782
		explore	=	0.0746	0.0791	**0.0813**	<u>0.0807</u>$^{\uparrow}$	0.0739
	nDCG@20	all	0.0491	0.0505	0.0505	<u>0.0532</u>	0.0479	0.0524
		explore	=	0.0506	0.0502	**0.0546**	0.0521$^{\uparrow}$	0.0484$^{\downarrow}$
Instacart	Recall@10	all	0.0430	0.0425	0.0440	0.0454	0.0394	0.0414
		explore	=	**0.0494**$^{\uparrow}$	0.0455	0.0460	<u>0.0469</u>$^{\uparrow}$	0.0419
	nDCG@10	all	0.0359	0.0346	0.0356	0.0388	0.0302	0.0335
		explore	=	**0.0400**$^{\uparrow}$	0.0355	<u>0.0387</u>	0.0369$^{\uparrow}$	0.0341
	Recall@20	all	0.0685	0.0649	0.0690	0.0733	0.0626	0.0635
		explore	=	<u>0.0755</u>$^{\uparrow}$	0.0719	0.0741	**0.0764**$^{\uparrow}$	0.0642
	nDCG@20	all	0.0455	0.0431	0.0452	0.0499	0.0394	0.0424
		explore	=	<u>0.0500</u>$^{\uparrow}$	0.0462	**0.0501**	0.0484$^{\uparrow}$	0.0431

Dunnhumby dataset, Beacon-explore achieves the best performance w.r.t. all metrics. On Instacart dataset, TIFUKNN-explore is among the best-performing methods, which means that well-tuned neighbor-based models may outperform complex neural-based methods on some datasets w.r.t. the NNBR task [28, 69]. The performance of G-TopFreq is obviously the worst on the Tafeng and Dunnhumby dataset, however, its performance is quite competitive on the Instacart dataset, which indicates that the popularity information is very important w.r.t. the NNBR task in the scenario with a high repeat ratio.

Second, the improvements of recent methods achieved in NBR task does not always generalize to the NNBR task. Recent proposed methods (TIFUKNN, CLEA, DNNTSP) have surpassed the previous classic baselines (i.e., G-TopFreq, Dream, Beacon) by a large margin in conventional NBR task [52, 69, 84, 127], whereas, the improvements are relatively small or even missing on some datasets when handling the NNBR task. This indicates that the recently proposed methods make limited progress on finding novel items for the user and that their improvements mainly come from the repeat recommendation, which is consistent with the findings in [69].

Third, the NNBR performance changes diversely for different methods when changing from Train-all to Train-explore. Training and tuning existing NBR methods specifically for the NNBR task lead to significant or mild improvements in most cases, since the models do not need to deal with the repetition task and they are more targeted on finding novel items that meet users' preferences. Surprisingly, we find that DNNTSP-explore's performance is much worse than DNNTSP-all on the Tafeng and Dunnhumby datasets. We suspect that the underlying reason for this deterioration is that the repeat items (labels) contain useful item-to-item correlation signals that can be captured by the DNNTSP.[8] Since various NBR methods have distinct architectures, certain methods may gain more from tailored training for exploration, while others can grasp item-item correlations from repeat labels. Consequently, it is unwise to indiscriminately eliminate repeat labels during training.[9]

3.6.2 Effectiveness of BTBR

To answer RQ2.2, we evaluate the overall NNBR task performance of BTBR with different masking strategies, i.e., item-level random masking (item-random), item-level select masking (item-select), basket-level all masking (basket-all) and basket-level explore masking (basket-explore). The results of the comparison

[8] Assume that one user's historical basket sequence is $[[a, b, c], [c, d], [a, c]]$, and next basket is $[b, e]$. Even though b is a repeat item, the model might be able to learn the correlation between b and other items in this historical sequence, which might help with the model's ability of finding novel items.

[9] This finding is important as it helps to avoid the potential issue of poor baselines. To ensure a fair comparison, NNBR practitioners should experiment with both strategies to train their baseline models and achieve best performances, instead of using an intuitive solution, i.e., removing repeat labels.

Table 3.5: Results of BTBR method with different masking strategies compared against the best performance of baseline method training for each metric w.r.t. NNBR task. Boldface and underline indicate the best and the second best performing performance w.r.t. the NNBR task, respectively. Significant improvements and deteriorations of over the best baseline results are marked with $^{\uparrow}$ and $^{\downarrow}$, respectively (paired t-test, $p < 0.05$). ▲% shows the improvements against the best performing baseline.

Dataset	Metric	Best	Item level				Basket level		Joint
			Random	Select	Random swap	Select swap	All	Explore	Pretrain-Finetune
Instacart	Recall@10	0.1024	0.0736$^{\downarrow}$	0.0801$^{\downarrow}$	0.0717$^{\downarrow}$	0.0746$^{\downarrow}$	<u>0.1056</u>$^{\uparrow}$	0.1032	**0.1057**$^{\uparrow}$**(3.2%)**
	nDCG@10	0.0859	0.0597$^{\downarrow}$	0.0651$^{\downarrow}$	0.0587$^{\downarrow}$	0.0605$^{\downarrow}$	<u>0.0869</u>	0.0860	**0.0870 (1.3%)**
	Recall@20	0.1257	0.0977$^{\downarrow}$	0.1036$^{\downarrow}$	0.0895$^{\downarrow}$	0.0911$^{\downarrow}$	<u>0.1292</u>$^{\uparrow}$	0.1271	**0.1353**$^{\uparrow}$**(7.6%)**
	nDCG@20	0.0952	0.0691$^{\downarrow}$	0.0739$^{\downarrow}$	0.0685$^{\downarrow}$	0.0688$^{\downarrow}$	<u>0.0970</u>$^{\uparrow}$	0.0957	**0.0973**$^{\uparrow}$**(2.2%)**
Tafeng	Recall@10	0.0529	0.0548$^{\uparrow}$	0.0572$^{\uparrow}$	0.0553$^{\uparrow}$	<u>0.0592</u>$^{\uparrow}$	0.0524	0.0521	**0.0593**$^{\uparrow}$**(12.1%)**
	nDCG@10	0.0428	0.0439$^{\uparrow}$	0.0461$^{\uparrow}$	0.0443$^{\uparrow}$	**0.0469**$^{\uparrow}$	0.0427	0.0424	<u>0.0468</u>$^{\uparrow}$(9.3%)
	Recall@20	0.0813	0.0847$^{\uparrow}$	0.0891$^{\uparrow}$	0.0867$^{\uparrow}$	**0.0924**$^{\uparrow}$	0.0815	0.0806	<u>0.0915</u>$^{\uparrow}$(12.5%)
	nDCG@20	0.0546	0.0560$^{\uparrow}$	0.0587$^{\uparrow}$	0.0571$^{\uparrow}$	**0.0598**$^{\uparrow}$	0.0540	0.0532	<u>0.0596</u>$^{\uparrow}$(9.2%)
Dunnhumby	Recall@10	0.0494	0.0554$^{\uparrow}$	0.0583$^{\uparrow}$	0.0572$^{\uparrow}$	**0.0600**$^{\uparrow}$	0.0539$^{\uparrow}$	0.0455$^{\downarrow}$	<u>0.0598</u>$^{\uparrow}$(21.1%)
	nDCG@10	0.0400	0.0445$^{\uparrow}$	0.0474$^{\uparrow}$	0.0458$^{\uparrow}$	**0.0486**$^{\uparrow}$	0.0426$^{\uparrow}$	0.0387$^{\downarrow}$	<u>0.0478</u>$^{\uparrow}$(19.5%)
	Recall@20	0.0764	0.0887$^{\uparrow}$	0.0924$^{\uparrow}$	0.0898$^{\uparrow}$	**0.0935**$^{\uparrow}$	0.0846$^{\uparrow}$	0.0734$^{\downarrow}$	<u>0.0934</u>$^{\uparrow}$(22.3%)
	nDCG@20	0.0501	0.0573$^{\uparrow}$	0.0607$^{\uparrow}$	0.0583$^{\uparrow}$	**0.0616**$^{\uparrow}$	0.0551$^{\uparrow}$	0.0485$^{\downarrow}$	<u>0.0613</u>$^{\uparrow}$(22.4%)

with the best baseline performances are shown in Table 3.5.[10] Based on the results, we have several observations. First, BTBR with the basket-all masking strategy (i.e., conventional next basket recommendation task) can significantly outperform the best baselines on the Tafeng and Instacart datasets, and achieve comparable performance on the Dunnhumby dataset. This result indicates that it may not be necessary to introduce basket representations, because only modeling item-to-item correlations is already effective for the NNBR task.

Second, there is no consistent best masking strategy across all datasets. On the Tafeng dataset, it is clear that basket-level masking outperforms item-level masking, where basket-all and basket-explore can respectively outperform and achieve the existing best performances w.r.t. each metric; however, using item-level masking leads to significant deterioration. On the Dunnhumby and Instacart datasets, BTBR with item-level masking strategies can significantly outperform the best performances achieved by baselines by a large margin, and is superior to BTBR with basket-level masking strategies. The above results show that the sequential order of items or baskets on the Tafeng dataset might be more strict than the order on the Dunnhumby and Instacart datasets, so using item-level masking, which fails to fully respect the sequential order and has poor performance on the Tafeng dataset.

Third, we can also observe that item-select masking achieves better performance than item-random masking w.r.t. all metrics across all datasets (paired t-test, $p < 0.05$), i.e., the improvements range from 4.1% to 9.0%, which demonstrates the effectiveness of our specifically designed item-select masking strategy for the NNBR task. In a sequence with many recurring items, the conventional random masking strategy could not ensure there is no masked item remaining in the other positions of the sequence, so the model might learn to predict the masked item based on the items' remaining occurrences, i.e., item self-relations. While the proposed item-select masking strategy will remove all occurrences of the same item, which can ensure that the masked items are novel items w.r.t. the remaining masked sequence, and the model has to infer the masked novel item via learning the masked item's relation with other items.

Finally, it can also be seen that basket-explore masking, which is specifically targeted at the NNBR task, does not lead to any improvements on the Tafeng and Dunnhumby datasets, and results in a decrease in performance on the Instacart dataset, compared with basket-all masking. This result again verifies the findings in Section 3.6.1 and indicates that masking and training BTBR specifically for the NNBR task may be suboptimal, since the repeat item labels may also be helpful with item-to-item correlations modeling.

3.6.3 Effectiveness of the item swapping strategy

To answer RQ2.3, we conduct experiments to verify the effectiveness of the swapping strategy, and the results are shown in Table 3.5. We find that adding

[10]To avoid confusion, we only mark the significant differences for comparison with the baselines in this table. More comparison results among different strategies can be found in the experimental analysis.

a swapping strategy on top of item-random and item-select leads to a decrease in performance on the Tafeng dataset. At the same time, we note that adding a swapping strategy on top of item-random and item-select leads to better performance on the Dunnhumby and Instacart datasets (paired t-test, $p < 0.05$). These results are not surprising, since the swapping strategy will not only enrich the item interactions within the basket, but also have a risk of introducing noise w.r.t. the temporal information. The sequential order is relatively strict on the TaFeng dataset (see Section 3.6.2), and the model can not benefit from the swap strategy.

We further investigate the influence of hyper-parameters of the swapping strategy, i.e., swap ratio and swap hop. Figure 3.4 shows a heatmap w.r.t. Recall@10 on different datasets when swap ratio ranges within $[0.1, 0.3, 0.5, 0.7, 0.9]$ and swap hop ranges within $[1, 3, 5, 7, 9]$. We observe that training with both high swap ratio and swap hop (the upper-right of the heatmap) leads to poor performance on the Tafeng and Dunnhumby dataset. When it comes to the Instacart dataset, better performance is achieved via using a high swap-hop. The repeat ratio on the Instacart dataset is high, which means that the user's interest is relatively stable and swapping across adjacent baskets won't help, so a higher swap hop is preferred to enrich item interactions within the basket on this dataset.

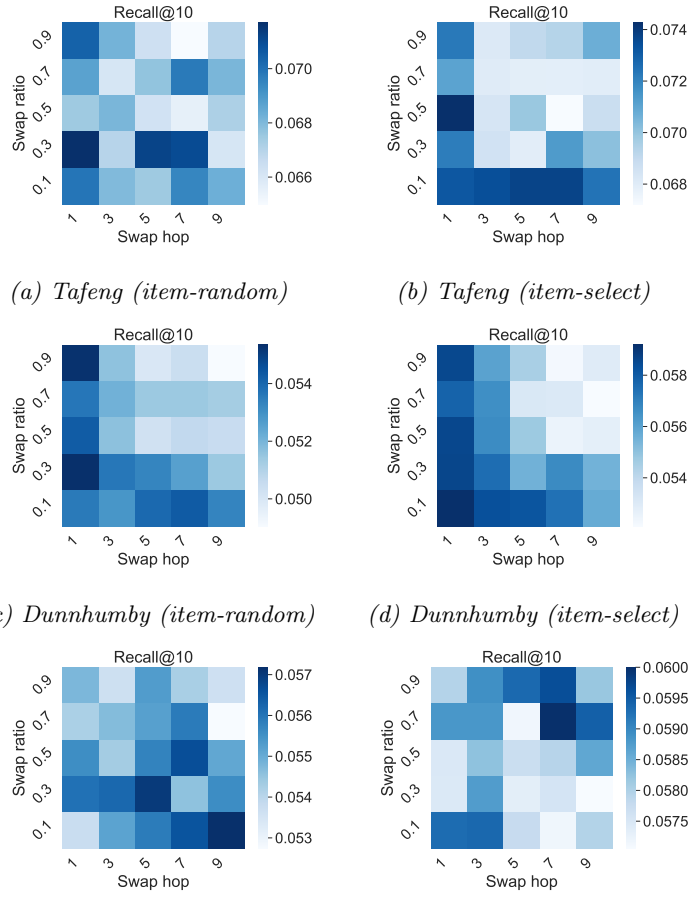
Given the above findings, there is a trade-off between enriching the item interactions within baskets and respecting the original temporal order information, so it is reasonable to search for the optimal swap hyper-parameters to get the highest performance on different datasets in practice.

3.6.4 Effect of mask ratio and training dynamics

To answer RQ2.4, we investigate the effect of mask ratio and analyze how the performance changes as training goes on to further understand the properties of different masking strategies.

Mask ratio. The mask ratio α when using item-level masking is one hyper-parameter that is worth discussing. Figure 3.5 shows the Recall@10 when mask ratio ranges within $[0.1, 0.3, 0.5, 0.7, 0.9]$. We can observe that item-select outperforms item-random with the same mask ratio in most cases. We also see that the optimal mask ratio is 0.1 for item-random and item-select, and the optimal mask ratio is much higher (0.5, 0.7) on the Dunnhumby and Instacart datasets. We suspect that a higher mask ratio is preferred in the NNBR task when the dataset has long interaction records for the users.

Training dynamics. Figure 3.6 shows how the Recall@10 evolves as training goes when using different masking strategies. First, it is obvious that basket-level masking achieves its best performances very fast, and then drops much earlier than item-level masking. This is because the training labels of basket-level masking are static, which can easily lead to overfitting, while the training labels of item-level masking are dynamic, which alleviates overfitting. Second, compared to basket-all masking, basket-explore masking further aggravates the



(a) Tafeng (item-random) *(b) Tafeng (item-select)*

(c) Dunnhumby (item-random) *(d) Dunnhumby (item-select)*

(e) Instacart (item-random) *(f) Instacart (item-select)*

Figure 3.4: Performance heatmap with different swap hops and swap ratios.

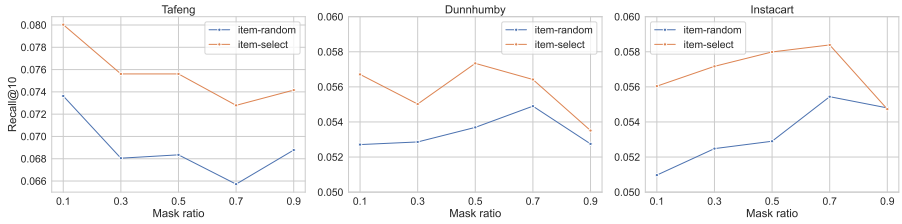


Figure 3.5: Performance of BTBR with item-random strategy and item-select masking strategy with various mask ratios.

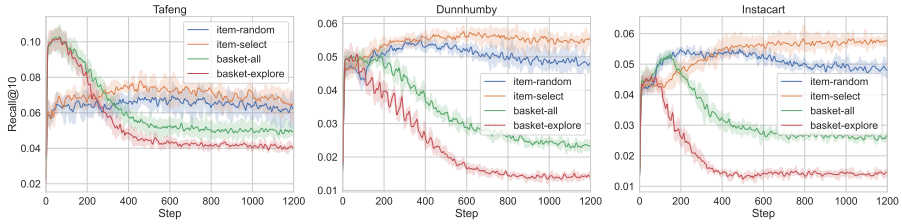


Figure 3.6: The training progress w.r.t. Recall@10 of BTBR with different masking strategies on three datasets.

overfitting problem via removing the repeat items (labels), which might lead to a performance decrease, especially in the scenario with a high repeat ratio. Finally, the performance of item-random and item-select evolves similarly on the Tafeng dataset, since the repeat ratio on it is small. On the Dunnhumby and Instacart datasets, item-random masking results in overfitting earlier than the item-select masking, since the masked item might still exist in other positions of the masked sequence and the model will rely more on the repeat item prediction instead of inferring novel items, as the repetition prediction task is relatively easier [69].

3.6.5 Effectiveness of joint masking

So far, we have built a comprehensive understanding of different masking strategies and realize that no single masking strategy is optimal in all cases, due to the diverse characteristics of datasets. Now, we conduct experiments to evaluate the effectiveness of the joint masking (training), i.e., pre-train the model using item-select masking, then fine-tune the model using basket-all masking. The results are also shown in Table 3.5. We find that BTBR with joint masking can consistently outperform best performances derived by existing baselines across datasets, the improvements range from 1.3% to 7.6% on Tafeng dataset, from 9.2% to 12.5% on Dunnhumby dataset and from 19.5% to 22.4% on Instacart dataset. Although joint masking does not lead to further improvements compared with a single optimal strategy, i.e., basket-all on the Tafeng dataset

and item-select with swap on the Dunnhumby and Instacart datasets, in most cases.[11] The joint masking strategy under the pretrain-then-finetune paradigm is still valuable due to its robustness w.r.t. NNBR task (i.e., can consistently achieve the best performance) on various datasets with different characteristics, which answers RQ2.5.

3.7 Conclusion

In this chapter, we have formulated and investigated the NNBR task, i.e., the task of recommending novel items to users given historical interactions, to answer the thesis-level research question **RQ2**:

> *How to design basket recommendation models targeted at the exploration task, and how to optimize the model to explore items in a scenario with many repetition signals?*

The NNBR task has practical applications, and helps us to evaluate an NBR model's ability to find novel items. To understand the performance of existing NBR methods on the NNBR task, we have evaluated several NBR models with two training methods, i.e., Train-all and Train-explore. To address the NNBR task, we have proposed a bi-directional transformer basket recommendation model (BTBR), which uses a bi-directional transformer to directly model item-to-item correlations across different baskets. To train BTBR, we designed five types of masking strategies and training objectives considering different levels: (i) item-level random masking, (ii) item-level select masking, (iii) basket-level all masking, (iv) basket-level explore masking, and (v) joint masking. To further improve the BTBR performance, we also proposed an item swapping strategy to enrich item interactions.

3.7.1 Main findings

We conducted extensive experiments on three datasets. Concerning existing NBR methods we found that: (i) The performance on the NNBR task differs widely between existing NBR methods; (ii) The performance of existing methods on the NNBR task leaves considerable room for improvement, and the top performing methods on the NNBR task are different from the top performers on the NBR task; (iii) Training specifically for the NNBR task by removing repeat items from the ground truth labels does not lead to consistent improvements in performance. Concerning our newly proposed BTBR method, we found that: (i) BTBR with a properly selected masking and swapping strategy can substantially improve the NNBR performance; (ii) There is no consistent best masking level for BTBR across all datasets; (iii) The proposed item-select masking strategy outperforms the conventional item-random masking strategy on the NNBR

[11] The highest and second-highest scores in Table 3.5 are essentially at the same level and there is no significant difference between the joint training strategy and the single optimal strategy on each dataset in terms of performance.

task; (iv) The item-basket swapping strategy can further improve NNBR performance; and (v) A joint masking strategy is robust on various datasets but does not lead to further improvements compared to a single level masking strategy.

3.7.2 Implications

A broader implication of our work is that blindly training specifically for the proposed recommendation task might lead to sub-optimal performance and it is necessary to consider various training objectives on diverse datasets. Another implication is that it is important to consider the differences between repetition behavior and exploration behavior when designing recommendation models for the grocery shopping scenario.

3.7.3 Limitations

One limitation of this chapter is that we only focus on the grocery shopping scenario. An obvious avenue for future work, therefore, is to extend the proposed item-select masking strategy to sequential item recommendation scenarios and investigate if it can outperform the widely used item-random masking strategy w.r.t. finding novel items.

In the next chapter, we will switch to an item-centered recommendation scenario.

4

Reverse Next-Period Recommendation

Previous chapters have been focusing on *user-centered* recommendations, which take information about a user as input and suggest items based on the user's preferences. In this chapter, we focus on an *item-centered* recommendation task, again in the grocery shopping scenario. In the *reverse next-period recommendation* (RNPR) task, we are given an item and have to identify potential users who would like to consume it in the next period.

Taking repetition and exploration into consideration, we aim to answer the thesis-level research question **RQ3**:

> *How to help a given item find its potential users in an item-centered setting, and how do repetition and exploration influence the design and optimization of the recommendation model?*

4.1 Introduction

Recommendation systems are an important instrument to connect users and items in many online services, like e-commerce [13, 64, 123], grocery shopping [105], music/movie streaming platforms [17, 30], and news [72, 106]. Unlike the top-n recommendation scenario, where the assumption is that there is no temporal information about past interactions [32, 56, 118], sequential recommendation systems keep track of users' historical interactions. This allows the recommender system to model users' preferences over time and recommend items for their next interactions [113]. Various types of sequential recommendation tasks have been well investigated in recent years, such as next-item recommendation [48, 117] and next-basket recommendation [52, 62, 84, 88, 125, 127]. What unites these tasks is their *user-centered* focus: given a user and their profile, these tasks aim to suggest relevant items that meet the user's preferences.

This chapter was published as: M. Li, M. Ariannezhad, A. Yates, and M. de Rijke. Who will purchase this item next? Reverse next period recommendation in grocery shopping. *ACM Transactions on Recommender Systems*, 1(2):Article 10, 2023.

4.1.1 Item-centered recommendations

In this chapter we focus on a less studied *item-centered* task, where the recommender system is given an item and needs to identify users who are most likely to consume it. Examples of such item-centered tasks emerge when advertising products [95], reducing waste [121], or promoting a healthy lifestyle [91]. For example, if a supermarket wants to sell bread that will expire soon to reduce waste, simply recommending the bread to all users will not only lead to high service costs, but it will also harm users' experience for those who do not like bread. As a result, item-centered recommendation algorithms have to identify specific *top-k* users who have an interest and may consume a given item. We define a novel "item-centered" recommendation problem in a sequential setting, namely *reverse next-period recommendation* (RNPR):[1]

> Given an item and historical transactions of all users, the reverse next-period recommendation task is to find potential users who have an interest in the item in the next time period.

Somewhat related to our item-centered focus, Wang et al. [114] have recently formulated the task of selecting potential "adopters" for a free-trial item to increase the exposure of the long-tail items. However, despite the similarity to our item-centered task, Wang et al. still focus on user-side performance.

While some previous studies have considered item-centered recommendation problems, their focus has typically been on improving efficiency in this setting rather than designing recommendation algorithms tailored to the new item-centered setting. Two related approaches that focus on efficiency are reverse maximum inner product search [reverse k-MIPS, 2] and reverse top-k queries [102, 103, 132]. These approaches do not consider the temporal information and assume that the user and item representation vectors are known or can be pre-computed in advance. As a result, they are not able to handle the RNPR task effectively.

In this chapter, we are specifically interested in the grocery shopping scenario, where historical interactions consist of baskets (multi-sets of items), for the following reasons: (i) the demand for item-centered recommendation is very clear in the grocery shopping scenario, e.g., to help reduce food waste[2] or to promote healthy lifestyles;[3] (ii) repetition behavior and exploration behavior both appear in the grocery shopping scenario, which allows us to understand the imbalance between repetition and exploration in the item-centered recommendation scenario. Specifically, we regard the next n baskets after the historical interactions to be "the next time period." We consider three key aspects of the RNPR task in this work: (i) user-centered methods for item-centered tasks, (ii) repetition vs. exploration behavior of users, and (iii) efficiency.

[1] While conventional recommender systems concentrate on recommending items to users, the term "reverse" marks a shift in focus by recommending users to items.

[2] `https://shorturl.at/iST79`

[3] `https://www.aholddelhaize.com/en/sustainability/healthier-choices/`

4.1.2 User-centered methods for item-centered tasks

Various sequential recommendation algorithms [36, 52, 125, 127] have been proposed and shown to achieve good performance in user-centered sequential recommendation. Even though these models are user-centered, they can also be adapted to the RNPR task, for instance by computing item scores from the user's side and ranking users from the item's side. An intuitive solution to adapt these models for the RNPR task is that, for a given item, the model computes a score for every candidate user that reflects the user's preference for it, and then selects the top-k highest-scoring users.

Before we design task-specific solutions for RNPR, it is of interest to answer the following question: what are the performance and limitations of these user-centered sequential recommendation methods in an item-centered RNPR setting? Since we focus on the grocery shopping scenario, we assess and investigate the performance of several representative next-basket recommendation algorithms [52, 127] on the RNPR task, and we find that the performance of state-of-the-art NBR methods does not always generalize to the RNPR task, even though they do model the temporal dependencies present in sequential recommendation (Section 4.8.1).

4.1.3 Repetition vs. exploration

In a user-centered sequential recommendation scenario, namely next-basket recommendation, a recent study [69] separates the candidate items into *repeat items* for a user, that is, items that the user has interacted with before, and *explore items* for a user, which are items that are new for the user. Similarly, for the RNPR task, given an item, we can also split its candidate users into *repeat users*, who have previously interacted with the given item, and *explore users*, who have never interacted with the given item before. We consider three sub-tasks of the RNPR problem:

Expl-RNPR: find possible new users (i.e., *explore users*), who will purchase the given item in the next period;

Rep-RNPR: find possible repeat users (i.e., *repeat users*), who will repurchase the given item in the next period; and

Mixed-RNPR: find all possible users (i.e., both *repeat users* and *explore users*), who will purchase the given item in the next period.

To address the Expl-RNPR task, we propose a *habit-interest fusion* (HIF) model that uses pre-trained embeddings to model a user's interests and employs frequency information to capture the repetition-exploration habits of the user. To train HIF effectively, we use an item-wise pairwise ranking loss and propose two strategies to construct the training samples: positive augmentation and negative adjustment. To address the Rep-RNPR task, we employ a simple time-aware frequency method, which only leverages users' direct interactions with a given item. To address the Mixed-RNPR task, we introduce a *repetition-exploration*

user ranking (REUR) algorithm, which decouples repetition, i.e., recommending users who have purchased the given item before, from exploration, i.e., recommending users who have not purchased the given item, and then tries to find the optimal combination of *repeat users* and *explore users*. Importantly, REUR allows us to investigate the trade-off between recommending repeat users and explore users. We find that recommending repeat users for a given item is much easier than finding potential explore users for a given item (Section 4.8.4).

4.1.4 Efficiency

Real world e-commerce applications usually have a large number of users and items [2], making it computationally expensive to compute every single user's score for a given item in order to identify the top-k users. In addition, an item-centered recommender system needs to operate in an *ad hoc* fashion, where it is not known up front which item needs to be recommended [2]. Therefore, it is important to reduce the computational costs of RNPR. To this end, we propose repetition-based methods to reduce the number of candidate users for a given item. Specifically, we first analyze the statistics of users' repetition behavior on both item and category level, from both the item and user perspective, and then propose two *repetition-rule based candidate filtering methods* (RRBF), which select candidate users for a given item based on users' item level (RRBF-item) and category level (RRBF-cat) repetition behaviors. For the Expl-RNPR task, we propose a *candidate filtering model* (CFM) to predict whether a user will purchase a specific category in the next period based on the temporal category information, which can further reduce the computational costs on top of RRBF-cat. We find that both the rule-based method (RRBF) and the model-based method (CFM) can effectively reduce the computational costs of RNPR (Section 4.8.5).

4.1.5 Main contributions

The main contributions of this chapter are as follows:

- We define and investigate the problem of reverse next-period recommendation (RNPR), introducing the Expl-RNPR, Rep-RNPR and Mixed-RNPR sub-tasks that consider different types of users, i.e., *repeat users* and *explore users*. To the best of our knowledge, this is the first work to study this problem.

- We investigate several sequential NBR recommendation algorithms applied to the RNPR problem, and find that their performance cannot be generalized in some cases for the Expl-RNPR task, and that they are more complex than needed for the Mixed-RNPR task.

- For the Expl-RNPR task, we propose a habit-interest fusion (HIF) model to capture users' habits and interests w.r.t. a given item, and we propose two training sample construction strategies for HIF.

- For the Mixed-RNPR task, we propose a REUR algorithm to decouple the repetition task and exploration task; and we investigate the trade-off between repetition and exploration via the REUR algorithm.
- We analyze users' repetition behavior on different levels from both a user and item perspective, and propose several repetition-based user candidate filtering methods to reduce the computational cost at inference time.
- We conduct experiments on two publicly available grocery shopping datasets, i.e., Dunnhumby and Instacart. The results demonstrate the effectiveness of the strategies we propose in this chapter.

4.2 Related Work

4.2.1 User-centered recommendation

Sequential item recommendation tasks have been investigated for many years. The purpose of such tasks is to consider users and their preferences and to recommend the next item according to those preferences. Recurrent neural networks (RNNs) [25, 41] and transformers [101] have shown strong performance in modeling sequential information, and they have been widely used to learn representations of historical behavior in session-based recommendation. GRU4Rec [49] leverages GRUs to model user sequences and then optimize a ranking-based loss for session-based recommendation. An updated version, GRURec+ [48], has a new ranking loss and sampling strategy. NARM [65] couples a GRU with an attention mechanism to make the recommendation model focus more on recent baskets. SASRec [58] employs a self-attention-based method to capture the temporal dynamics of sequential recommendations in an efficient way.

In addition to RNN- and transformer-based models, several deep learning techniques have also been applied to this area. Memory networks are applied by STAMP [73] to capture a user's general interests and current interests. SR-GNN [117] models a session sequence as a graph and then uses a graph neural network [89] to capture item transactions and learn an accurate item embedding. Tang and Wang [96] propose a CNN-based method to capture general interests and sequential patterns via vertical and horizontal filtering. Yuan et al. [128] introduce a generative model to improve the performance. Pre-trained models (such as BERT) [92] and knowledge graphs are also being applied to user-centered recommendations [53, 115].

In grocery shopping, both the sequences of historical interactions and the output of recommendations are sets (or rather, multisets) of items, so-called baskets, and the *next basket recommendation* (NBR) task is a user-centered sequential recommendation task that caters to this scenario. Over the years, many dedicated NBR methods have been proposed [69]. These include Markov chain (MC)-based methods [88, 108], deep learning-based methods [4, 9, 51, 62, 112, 125, 127], and frequency neighbor-based methods [36, 52]. An analysis conducted by Li et al. [69] assesses and evaluates the NBR performance from

a new repetition and exploration perspective; their comparisons show that recommending *repeat items* (items that a user has interacted with previously) is an easier task than recommending *explore items* (items that a user has never interacted with before), see Chapter 2.

All of the sequential recommendation methods mentioned above focus on the user perspective, whereas we propose the reverse next-period recommendation (RNPR) problem that focuses on the item perspective.

4.2.2 Item-centered recommendations

Item-centered recommendations focus on the item perspective. That is, they aim to recommend suitable users for a given item that are likely to interact positively with it (i.e., purchase it in a grocery shopping setting, listen to it in a music recommendation setting, download and read it in a book recommendation setting, etc.). Early proposals of item-centered recommendation date back at least to the so-called reverse top-k query problem [102, 103, 132]. Early publications on this problem typically consider Euclidean spaces with low-dimensional (often, around 5) user and item vectors.

In recent years, recommender systems have benefited from the development of deep learning techniques, which can construct high-dimensional representations and embeddings of users and items. For example, Amagata and Hara [2] propose reverse top-k maximum inner product search (reverse k-MIPS), which assumes that d-dimensional representations of users and items are obtained via matrix factorization [60]. Interestingly, previous work on item-centered recommendations only focuses on efficiency (i.e., on reducing the computational costs) rather than on improving the performance on the item-centered recommendation task. Furthermore, they do not consider temporal dependencies between historical items, which is a key aspect of the sequential recommendation task. Recently, Wang et al. [114] have formulated a user selection problem for free-trial items, which aims to increase item exposure and retain user-side performance.

Unlike previous work, we formulate the RNPR problem in a sequential setting. We aim to find users who will purchase a given item and focus on item-side performance. Moreover, we do not only focus on the efficiency aspect, but also try to improve the performance on the RNPR task.

4.3 Problem Formulation

In this section, we describe two types of users, i.e., *repeat users* and *explore users*, formalize the reverse next-period recommendation task, and associate three sub-tasks with it, i.e., Expl-RNPR, Rep-RNPR and Mixed-RNPR. We also introduce the candidate filtering task for reverse next-period recommendation. The notation used in this chapter is shown in Table 4.1.

Table 4.1: Notation used in this chapter.

Symbol	Description
U	Set of all users, i.e., $U = \{u_1, u_2, \ldots, u_o\}$.
I	Set of all items, i.e., $I = \{i_1, i_2, \ldots, i_m\}$
C	Set of all categories, i.e., $C = \{c_1, c_2, \ldots, c_q\}$
I^c	Set of all items belongs to category c, i.e., a subset of I.
B_u^t	t-th basket purchased by user u at time t, which is a set of items $i \in I$
S_u^h	Sequence of historical baskets for user u, i.e., $S_u^h = \{B_u^1, B_u^2, \ldots, B_u^t\}$
S_u^n	Sequence of future (next-period) baskets for user u, i.e., $S_u^n = \{B_u^{t+1}, B_u^{t+2}, \ldots, B_u^{t+n}\}$
I_u^h	Set of historical items purchased by user u
I_u^n	Set of items that user u will purchase during next period (n baskets)
C_u^h	Set of categories from which user u has purchased items before
C_u^n	Set of categories from which user u will purchase items during next period (n baskets)
U_i^{rep}	Set of *repeat users* u_i^{rep} who have purchased item i, i.e., $i \in I^h_{u_i^{rep}}$
U_i^{expl}	Set of *explore users* u_i^{expl} who have not purchased item i, i.e., $i \notin I^h_{u_i^{expl}}$
U_c^{rep}	Set of *repeat users* u_c^{rep} who have purchased an item in category c, i.e., $c \in C^h_{u_c^{rep}}$
U_c^{expl}	Set of *explore users* u_c^{expl} who have not purchased an item in category c, i.e., $c \notin C^h_{u_c^{expl}}$
U_i^t	Set of the target users for item i
$\hat{U}_i^t$	Set of the candidate users for item i
T_i	Set of ground-truth users u who will purchase item i in next period, i.e., $i \in I_u^n$
T_i^{rep}	Set of ground-truth *repeat users* $u_i^{rep,*}$ for item i, i.e., $i \in I^n_{u_i^{rep,*}}$ and $i \in I^h_{u_i^{rep,*}}$
T_i^{expl}	Set of ground-truth *explore users* $u_i^{expl,*}$ for item i, i.e., $i \in I^n_{u_i^{expl,*}}$ and $i \notin I^h_{u_i^{rep,*}}$
P_i^n	Predicted top-k users for item i, i.e., $P_i^n = [u_1^p, u_2^p, \ldots, u_k^p]$
$f(\cdot)$	Reverse next period recommendation (RNPR) algorithm
$g(\cdot)$	Candidate filtering algorithm

4.3.1 Reverse next-period recommendation

Assume we have a set of users and a set of items, denoted as $U = \{u_1, u_2, \ldots, u_o\}$ and $I = \{i_1, i_2, \ldots, i_m\}$, respectively. Each item belongs to a category $c \in C = \{c_1, c_2, \ldots, c_q\}$. B_u^t denotes user u's basket at time step t, where B_u^t consists of a set of items $i \in I$. $S_u^h = \{B_u^1, B_u^2, \ldots, B_u^t\}$ represents the sequence of historical interactions for user u, and $S_u^n = \{B_u^{t+1}, B_u^{t+2}, \ldots, B_u^{t+n}\}$ represents the sequence of the next n interactions for user u. Then, $I_u^h = \{i_1, i_2, \ldots, i_z\}$ and $I_u^n = \{i_1, i_2, \ldots, i_e\}$ represent the item set that user u has purchased before and will purchase in the next n baskets, i.e., next period, respectively. $C_u^h = \{c_1, c_2, \ldots, c_v\}$ represents the category set in which user u has purchased items before, $C_u^n = \{c_1, c_2, \ldots, c_w\}$ represents the category set in which user u will purchase items in the next n baskets.

Given a specific item i, the users in U can be divided into *repeat users* and *explore users* based on the historical interaction with the item i:

Repeat users U_i^{rep} for item i are the users u who have purchased the item i before, that is, users u such that $i \in I_u^h$.

Explore users U_i^{expl} for item i are the users u who have not purchased product i before, that is, users u such that $i \notin I_u^h$.

Similarly, given a specific category c, the users in U can also be divided as follows:

Repeat users U_c^{rep} for category c are the users u who have purchased an item in category c before, that is, users u such that $c \in C_u^h$.

Explore users U_c^{expl} for category c are the users u who have not purchased category an item in c before, that is, users u such that $c \notin C_u^h$.

Given a specific item i and historical interactions $S^h = \{S_1^h, S_2^h, \ldots, S_m^h\}$ of target users $u_1, \ldots, u_m \in U_i^t$, the goal of the *reverse next-period recommendation* (RNPR) task is to predict the top-k users $P_i^n \subseteq U_i^t$, who will purchase the given item i in one of the next n baskets. To address the RNPR task, we seek to define a function f that takes item i and historical interactions $S^h = \{S_1^h, S_2^h, \ldots, S_m^h\}$ of target users $u_1, \ldots, u_m$ as input, and returns P_i^n:

$$P_i^n = [u_1^p, u_2^p, \ldots, u_k^p] = f(i, \{S_1^h, S_2^h, \ldots, S_m^h\}). \tag{4.1}$$

where P_i^n is a predicted ranked list, which contains top-k users for item i. Considering the difference types of users that we have defined above, we define three sub-tasks for RNPR:

Expl-RNPR: To find the top-k explore users who are most likely to purchase the given item i, that is, $U_i^t = U_i^{expl}$.

Rep-RNPR: To find top-k repeat users who are most likely to repurchase the given item i, that is, $U_i^t = U_i^{rep}$.

Mixed-RNPR: To find the top-k users who are most likely to purchase the given item i, that is, the target users are simply the set of all users: $U_i^t = U = U_i^{expl} \cup U_i^{rep}$.

4.3.2 Candidate filtering

Given a specific item i and its target users U_i^t for the RNPR task, the goal of *candidate filtering* is to select a subset of candidate users $\hat{U}_i^t \subseteq U_i^t$ based on their historical interactions S^h. More formally, we seek to define a candidate filtering function g such that

$$\hat{U}_i^t = \{u_1^c, u_2^c, \ldots, u_q^c\} = g(i, S_1^h, S_2^h, \ldots, S_m^h). \tag{4.2}$$

Given a filtered set of candidate users $\hat{U}_i^t$ for item i, we only compute item scores for users in this filtered set of users $\hat{U}_i^t$ instead of all candidate users U_i^t.

4.4 Repetition Analysis

People often have regular habits and display repetition behavior in grocery shopping [6, 69, 70, 105]. Li et al. [69] analyze the repetition behavior from the user side on the item level. That is, how many of the items that the user will purchase next are repeat items that they have purchased before. However, repetition behavior at the category level and from the item side remain unknown. In particular, (i) at the *category level*, how many of the categories from which the user will purchase an item are categories that they have previously purchased an item from? And (ii) from the *item side*, among the users who will purchase the given item or from the given category, what is the proportion of users who have already purchased the given item or from the given category in their previous interactions?

To better understand users' repetition behavior in grocery shopping, we analyze both the item and category level repetition behavior from both the item side and the user side.[4] Specifically, we analyze four types of repeat ratio $RepR$, i.e, user-side item-level $RepR_u^{item}$, user-side category-level $RepR_u^{cat}$, item-side item-level $RepR_i^{item}$ and item-side category-level $RepR_i^{cat}$, defined as follows:

$$RepR_u^{item} = \frac{1}{N}\sum_{n=1}^{N} \frac{\#\textit{repeat items } i \in I_{u_n}^h \text{ the user } u_n \text{ will purchase}}{\#\text{all items the user } u_n \text{ will purchase}} \tag{4.3}$$

$$RepR_u^{cat} = \frac{1}{N}\sum_{n=1}^{N} \frac{\#\textit{repeat categories } c \in C_{u_n}^h \text{ the user } u_n \text{ will purchase}}{\#\text{all categories the user } u_n \text{ will purchase}} \tag{4.4}$$

$$RepR_i^{item} = \frac{1}{M}\sum_{m=1}^{M} \frac{\#\textit{repeat users } u \in U_i^{rep} \text{ who will purchase item } i_m}{\#\text{all users who will purchase item } i_m} \tag{4.5}$$

[4] We perform this analysis by splitting the data into historical baskets and future baskets, which is the same as the experimental setting in Section 4.7.2

Table 4.2: Repeat ratios at the item level and category level, from the item perspective and the user perspective.

Dataset	Instacart		Dunnhumby	
Perspective	Item level	Category level	Item level	Category level
User side	0.6822	0.8791	0.4264	0.8737
Item side	0.6111	0.7751	0.4374	0.6649

$$RepR_i^{cat} = \frac{1}{Q}\sum_{q=1}^{Q}\frac{\#\textit{repeat users } u \in U_c^{rep} \text{ who will purchase category } c_q}{\#\text{all users who will purchase category } c_q}, \quad (4.6)$$

where N, M, and Q are the number of users, items and categories, respectively. We compute these four ratios for each of the two datasets that we will be using in this chapter, Instacart and Dunnhumby (see Section 4.7.2). See Table 4.2.

From the user side (the first row in Table 4.2), we can observe that both the item level repeat ratio $RepR_u^{item}$ and category level repeat ratio $RepR_u^{cat}$ are high, ranging from 0.4264 to 0.8791. The results indicate that a large proportion of items/categories the users will purchase in the next period is made up from items/categories that the users have purchased before. The category level repeat ratio $RepR_u^{cat}$ is relatively high, which shows that the repetition behavior at the category level is more stable than item level in grocery shopping. For example, a user might like to buy fruits every time, but the user might alternate between different types of fruits as time passes.

From the item side, we can also see that both datasets have considerable repeat ratios, i.e., $RepR_i^{item}$ and $RepR_i^{cat}$, ranging from 0.4374 to 0.7751. Similarly, the category level repeat ratio $RepR_i^{cat}$ is also higher than the item level repeat ratio $RepR_i^{item}$. The results indicate that a considerable proportion of the users in our datasets are *repeat users*.

The results presented above indicate that in grocery shopping people have regular habits and that repetition behavior is a strong signal that can be used to address the item-centered RNPR problem from several angles: (i) with the habit module to model users' category level exploration behavior in HIF model (Section 4.1.3); (ii) with the REUR algorithm, which decouples the repetition task and exploration task (Section 4.5.3); and (iii) with the repetition-rule based candidate filtering methods (Section 4.6.1) and the CFM model to model category level repetition behavior (Section 4.6.2) for reducing candidate users.

4.5 Reverse Next-Period Recommendation

In this section, we introduce the habit-interest fusion (HIF) model and corresponding training strategies for the Expl-RNPR task, describe a simple time-aware frequency model for the Rep-RNPR task, and finally describe the REUR algorithm for the Mixed-RNPR task.

4.5.1 Habit-interest fusion model for Expl-RNPR

The objective of a user-centered recommendation model is to rank positive items higher than the negative items, i.e., giving a higher prediction score to positive items, where the prediction score for an item may not represent the user's absolute preference on this item, as this prediction score can be influenced by other items in the catalog and the item distribution in the dataset, e.g., popularity. User-centered recommendation models usually only take a user's historical interactions and learn the general interest of the user and but may not track the users' interest w.r.t. a specific given item as time goes by. To achieve accurate item-centered recommendations, there are two things that should be taken into consideration: (i) the prediction of the model should be appropriate and meaningful for ranking users for a given item; and (ii) apart from a user's historical interactions, the recommendation model should also take the given item as input and be able to track user's interests or habits w.r.t. the given item as time goes by.

Model

Recall from Section 4.3.1 that the Expl-RNPR task is to find the top-k explore users who are most likely to purchase a given item i. To address the Expl-RNPR task, we propose a habit-interest fusion (HIF) model, which leverages frequency information to model category-level repetition and exploration habits, and pre-trained item representations to model user's interests. Figure 4.1 illustrates the architecture of the HIF model.

Pre-trained embedding In the context of NLP, the skip-gram framework [76, 77] has been proposed to learn word representations by predicting the surrounding words within the context. Several recent publications [10, 42, 105] leverage skip-gram techniques to learn item/product representations in an e-commerce scenario. In this chapter, we assume that the items within the same basket share similar semantics and use basket-level skip-grams to derive the embeddings of items. We regard a particular item as a target item $i \in I$ and regard the other items in the same basket as context items $i' \in I_b^i$. Then, the learning objective is to maximize the following function:

$$\mathcal{L} = \sum_{i \in I} \sum_{i' \in I_b^i} \log p(i' \mid i), \tag{4.7}$$

where $p(i' \mid i)$ denotes the probability of observing a context item $i' \in I_b^v$ given the current/target item i. It is defined by a softmax function:

$$p(i' \mid i) = \frac{\exp(Emb_i^T \cdot Emb_{i'})}{\sum_{m=1}^{M} \exp(Emb_i^T \cdot Emb_{i'_m})}, \tag{4.8}$$

where Emb_i and $Emb_{i'}$ are vector representations of the current item i and the context item i', respectively. M represents the number of items in the item

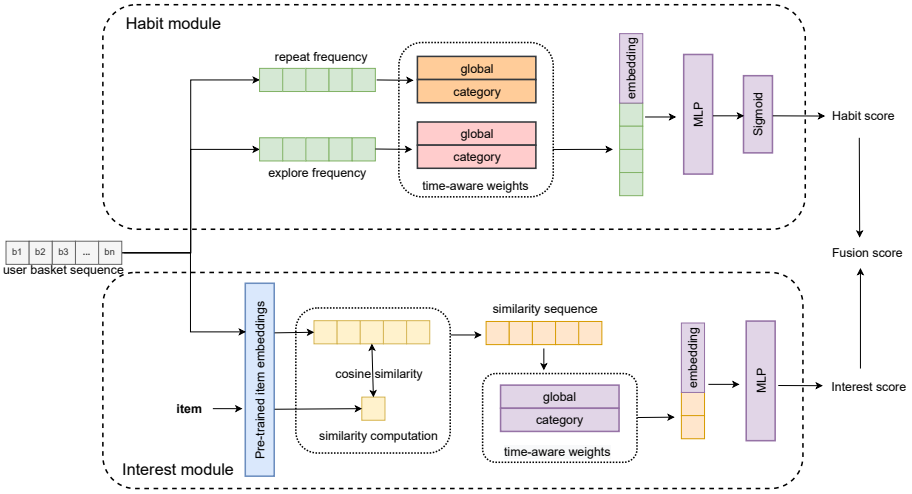


Figure 4.1: Overall architecture of the HIF model.

catalog. After pre-training on historical data, we can get a vector representation (a.k.a. embedding) of each item.

Interest module Suppose that a user u has a sequence of historical baskets $S^h = \{B^1, B^2, \ldots, B^t\}$. We first get pre-trained item embeddings Emb_i for each item i within each basket B^t. Note that baskets may have different sizes, so we aggregate item embeddings within the same basket by a pooling strategy (max pooling or average pooling) to generate the basket representation Emb_b^t at each timestamp t. Given the target item i we want to recommend, we compute the cosine similarity $Sim_{u,i}^t$ between its embedding Emb_i and basket embedding Emb_b^t at each timestamp, and then get the similarity vector $Sim_{u,i}$, which reflects user's interests in the given item i across different timestamps. That is:

$$Emb_b^t = \text{Pooling}\left(Emb_{i_1^t}, Emb_{i_2^t}, \ldots, Emb_{i_n^t}\right) \tag{4.9}$$

$$Sim_{u,i}^t = \cos\left(Emb_i, Emb_b^t\right) = \frac{Emb_i \cdot Emb_b^t}{|Emb_i||Emb_b^t|} \tag{4.10}$$

$$Sim_{u,i} = \left[Sim_{u,i}^1, Sim_{u,i}^2, \ldots, Sim_{u,i}^t\right]. \tag{4.11}$$

To model users' dynamic interests, we introduce two types of time-aware weight embeddings, i.e., (i) a category specific time-aware weight embedding TW_e^c, which can only be trained by the samples of the corresponding category c, and (ii) a global time-aware weight embedding TW_e^g, which is shared across categories and can be trained by all training samples.[5] For a given item i and

[5]Note that, the time-aware weight embeddings throughout out this chapter will be first initialized using a uniform distribution and then updated during the training process.

user $u \in U_i^{expl}$, we compute the dot products of the similarity vector $Sim_{u,i}$ and two time-aware weight embeddings, i.e., TW_e^c and TW_e^g, to get time-aware interests features, i.e., $SimF^c_{u,i}$ and $SimF^g_{u,i}$. Finally, we concat $SimF^c_{u,i}$ and $SimF^g_{u,i}$ with a trainable category embedding Emb_c^{inte} to get a hybrid representation, which will be fed into a two layer fully-connected network to get the final interests score $Score_{u,i}^{inte}$, that is:

$$SimF^c_{u,i} = TW_e^c \cdot Sim_{u,i} \tag{4.12}$$

$$SimF^g_{u,i} = TW_e^g \cdot Sim_{u,i} \tag{4.13}$$

$$Score_{u,i}^{inte} = \text{FFN}(SimF^c_{u,i} \oplus SimF^g_{u,i} \oplus Emb_c^{inte}). \tag{4.14}$$

Habit module In the Expl-RNPR task, we aim to find possible *explore users* for a given item. However, there are no direct interactions between the given item i and *explore users* U_i^{expl}, so we cannot directly model *explore users*' habits w.r.t. the item i. In the grocery shopping scenario, every item belongs to a category, and a category can contain many items. We notice that if an item i will be purchased by the user $u \in U_i^{expl}$, it indicates that the user u will purchase and explore the items in category c^i in the next period. Therefore, we aim to model users' repetition and exploration habits w.r.t. the target category c^i of the given item i.

The users' repetition habits within a category can be dynamic across time. Besides, the purchase frequency within a category can also indicate demands of the user. Specifically, to capture the user's repetition habits, we create a category-level repetition frequency vector $RepVec$ for category $c^i \in C$ for the user u by considering both temporal information and frequency information. That is,

$$RepVec_{u,c^i} = \left[\sqrt{|I^{c^i} \cap B^1|}, \sqrt{|I^{c^i} \cap B^2|}, \ldots, \sqrt{|I^{c^i} \cap B^t|}\right], \tag{4.15}$$

where I^{c^i} is the item set within category c^i; B^t is a set of items (basket) that user u purchased at timestamp t. Note that the square root operation is applied to address the problem of varying sizes of baskets in recommendation systems. By taking the square root, the impact of baskets that are too large is reduced, leading to more equitable and balanced frequency information. Then, we derive time-aware category repetition feature $RepF^c_{u,c^i}$ and global repetition feature $RepF^g_{u,c^i}$ as follows:

$$RepF^c_{u,c^i} = TW^c_{rep} \cdot RepVec_{u,c^i} \tag{4.16}$$

$$RepF^g_{u,c^i} = TW^g_{rep} \cdot RepVec_{u,c^i}, \tag{4.17}$$

where TW^c_{rep} and TW^g_{rep} are a category time-aware weight embedding and a global time-aware weight embedding, respectively, for modeling repetition behavior.

Note that the user might be loyal to a specific item [105] and uninterested in exploring new items within the same category, e.g., someone might only

purchase a specific brand of milk. To model a user's exploration habits within a category, we also create an exploration frequency vector $ExplVec_{u,c^i}$ considering the temporal orders, that is:

$$ExplVec_{u,c^i} = \left[\sqrt{|I^{c^i} \cap B^1_{expl}|}, \sqrt{|I^{c^i} \cap B^2_{expl}|}, \ldots, \sqrt{|I^{c^i} \cap B^t_{expl}|}\right], \quad (4.18)$$

where B^t_{expl} is a set of *explore items* (new items) that the user u purchased at timestamp t.

Similarly, we compute the category exploration feature $ExplF^c_{u,c^i}$ and global exploration feature $ExplF^g_{u,c^i}$ as follows:

$$ExplF^c_{u,c^i} = TW^c_{expl} \cdot ExplVec_{u,c^i} \quad (4.19)$$

$$ExplF^g_{u,c^i} = TW^g_{expl} \cdot ExplVec_{u,c^i}, \quad (4.20)$$

where TW^c_{expl} and TW^g_{expl} are the category time-aware weight embedding and the global time-aware weight embedding, respectively, for modeling exploration behavior. Finally, we concatenate repetition features, i.e., $RepF^c_{u,c^i}$ and $RepF^g_{u,c^i}$, exploration features, i.e., $ExplF^c_{u,c^i}$ and $ExplF^g_{u,c^i}$, and a trainable category specific embedding Emb^{hab}_c to get a feature vector, which will be fed into a two-layer fully-connected network to get the habit score $Score^{hab}_{u,i}$. That is,

$$Score^{hab}_{u,i} = \text{FFN}(RepF^c_{u,c^i} \oplus RepF^g_{u,c^i} \oplus ExplF^c_{u,c^i} \oplus ExplF^g_{u,c^i} \oplus Emb^{hab}_c). \quad (4.21)$$

Finally, we compute the fusion score by:

$$Score^{fusion}_{u,i} = \text{Sigmoid}(Score^{hab}_{u,i}) \cdot Score^{inte}_{u,i}. \quad (4.22)$$

Training

In a conventional *user-centered* scenario, a recommendation model is optimized based on a user-wise loss, which is computed based on all items for each user. Since we focus on *item-centered* recommendations to rank users for the given item, we propose an item-wise ranking loss to train our model. Specifically, positive users and negative users are sampled for each item, and then the training objective is to minimize the following loss function:

$$\mathcal{L}_i = -\frac{1}{N}\sum_{k=1}^{N} \log\left(\frac{1}{1 + e^{-(Score^k_{pos,i} - Score^k_{neg,i})}}\right), \quad (4.23)$$

where $Score_{pos,i}$ and $Score_{neg,i}$ represent the predicted fusion scores for positive users and negative users, respectively. By minimizing the proposed item-wise ranking loss, the model will maximize the difference in predicted preference (fusion) scores between the positive and negative users, such that positive users are ranked higher in the predicted user ranking list. Even though the definition of item-wise ranking loss is straight forward, we identify two major issues w.r.t. the training process of the Expl-RNPR model using item-wise ranking loss.

Table 4.3: The number of training samples for Expl-RNPR.

Dataset	Avg. #negative samples per item	Avg. #positive samples per item	Avg. #positive samples per item after positive augmentation
Instacart	1,307.9	5.8	31.9
Dunnhumby	667.9	6.0	73.6

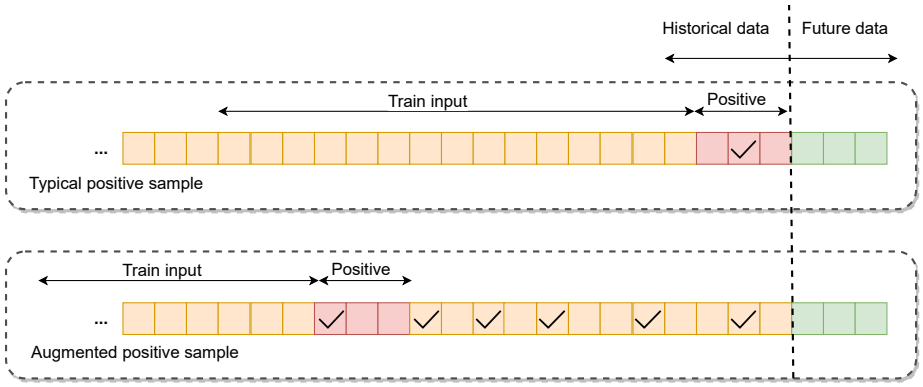
First, as illustrated in Figure 4.2a, a typical positive sample for Expl-RNPR is an *explore user* who only purchased the given item i in the last period of the historical sequence. However, as shown in Table 4.3, items have a small number of such positive samples (i.e., new users) if we only consider the last period of the historical sequence. Therefore, we need to augment the positive samples, i.e., include more users who explore the target item for the first time. According to an intuitive reading of the Expl-RNPR task, we should not select a *repeat user* who has already purchased the given item as a positive training sample for the given item, since Expl-RNPR is targeting *explore users*. However, *repeat users* of the target item should have a sub-sequence of interactions, i.e., a basket sequence before their first purchase, that could be regarded as a positive sample for Expl-RNPR training (shown in Figure 4.2a).

Second, a typical negative sample for Expl-RNPR is an *explore user* who did not purchase the given item i in the last period sequence. However, as illustrated in Figure 4.2b, if a user u is a new user of a given item i, i.e., $i \in I_u^n$, the user has not purchased this item in previous interactions, i.e., $i \notin I_u^h$, and this means that the user should also be regarded as a negative sample for the item i during the training process. In this case, when we use a leave-one/few-out splitting strategy to construct a historical (training) dataset and a future (test) dataset, the positive samples (i.e., the ground-truth) in the test set will be the negative sample in the training set, even though they may share a long overlap between two input sequences. To avoid the negative impact of this case, we propose a negative sample adjustment strategy, which eliminates the potential overlap between positive and negative sequences by truncating a sub-sequence from the original negative samples. Note that we perform the truncation action on all negative samples, since we do not know which one is the positive sample in the future (test) dataset.

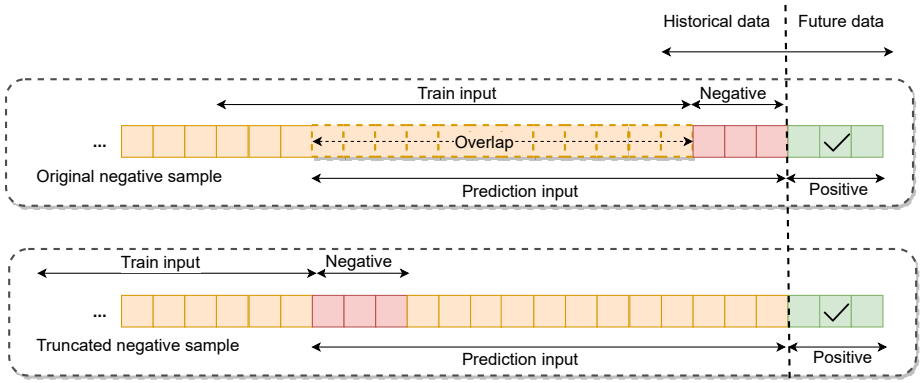
4.5.2 Time-aware frequency model for Rep-RNPR

The task of Rep-RNPR is to help a given item find *repeat users*. Different from *explore users*, *repeat users* have previously had direct interactions with the item that we want to recommend. Therefore, we employ a simple time-aware frequency model, which only uses users' direct interactions with the given item instead of using item-item correlations and complex representations. Formally,

$$Score_{u,i} = \sum_{j=1}^{m} F_{u,i,j} \cdot \beta^{T-j}, \tag{4.24}$$



(a) Positive sample augmentation strategy.



(b) Negative sample adjustment strategy.

Figure 4.2: Training samples construction strategies.

where $F_{u,i,j}$ denotes the user's u purchase frequency of item i at timestamp j, β denotes the time-decay factor, which emphasizes the impact of recent interactions. We find the optimal β based on the historical data.

4.5.3 The REUR algorithm for Mixed-RNPR

Different from the Expl-RNPR task in Section 4.5.1 and the Rep-RNPR task in Section 4.5.2, the Mixed-RNPR task considers both *repeat users* and *explore users* for the items that are to be recommended. Theoretically, a model for Expl-RNPR can also be applied to Mixed-RNPR without excluding *repeat users* in the final prediction stage. A recent analysis [69] shows that the repetition and exploration tasks in the (user-centered) NBR problem have different levels of difficulty, where the repetition task, i.e., recommending repeat items to a user, is a much easier task. In an item-centered recommendation scenario, we mainly use item-to-item relations to infer *explore users*' interests for the target item,

since *explore users* do not have any previous interactions with it. Yet, we can address *repeat users* prediction via the users' direct interactions with the target item.

Considering the above differences between the repetition and exploration tasks in Mixed-RNPR, we propose a *repetition-exploration user ranking* (REUR) algorithm to decouple the two tasks and investigate the trade-off between repetition and exploration in an item-centered setting. Specifically, as is shown in Algorithm 1, we use separate models for the repetition and exploration tasks.[6] Note that the models designed in Section 4.5.1 and Section 4.5.2 can be used for ranking *explore users* and ranking *repeat users*, respectively. For a given item, we rank *repeat users* and *explore users* according to the scores derived from the repetition model M^{rep} and exploration model M^{expl}, respectively. Then, REUR generates the final ranked list of users P_i^n by combining the above two ranked lists. We define a combination (repeat) ratio α, which controls the proportions of *repeat users* and *explore users*. Assume that we want to recommend a given item to k users. Then REUR first selects the top-m highest-score *repeat users* and then fills any remaining slots with the top-n highest-scoring *explore users*, where $m = k \cdot \alpha$ and $n = k - m$.

As we will see, one simple way to achieve good performance on the Mixed-RNPR task is to find a global optimal combination ratio α for all items. We notice that different items might have different repurchase tendencies, i.e., the repurchase tendency of a pan is likely to be smaller than that of the milk that is cooked in it, which might influence the optimal combination ratio. The repurchase tendency is defined by:

$$RT_i = \frac{\#\text{users who repurchase item } i}{\#\text{users who bought item } i \text{ before}}. \tag{4.25}$$

We also cluster items according to their repurchase tendency $RT_i \in [0, 0.2)$, $[0.2, 0.4)$, $[0.4, 0.6)$, $[0.6, 0.8)$, $[0.8, 1.0]$, and try to find the optimal combination ratio α for each cluster. We sweep the combination ratio α and select the optimal α based on the performance of the validation set.

Apart from achieving good performance on the Mixed-RNPR task, another important task is to investigate the trade-off between repetition and exploration so as to make sure that we gain an in-depth understanding of the potential imbalances in the RNPR task. With the REUR algorithm, we can also easily investigate the trade-off by setting different combination ratios (see Section 4.8.4).

4.6 Candidate Filtering

Given a specific item i, an intuitive solution to an item-centered recommendation task is to compute scores for every user in the target user set U_i^t. However, this is usually computationally expensive. The goal of candidate filtering is to select a group of users as candidates, which is a subset of target users, i.e.,

[6] For a given item, the repetition task is to find repeat users, whereas the exploration task is to find new users.

Algorithm 1: Repetition-exploration user ranking algorithm

Data: User set U, item set I, basket sequences S, user list size k, combination/repeat ratio α

Result: Predicted top-k users P_i^n for item i.

```
Get repetition model M^rep by finding optimal time-decay factor β on the dataset;
Get exploration model M^expl via training the HIF model on the dataset until converge;
for each given item i do
    Get repeat users U_i^rep and explore users U_i^expl;
    Rank repeat users u ∈ U_i^rep via M^rep(S_u, i);
    Rank explore users u ∈ U_i^expl via M^expl(S_u, i);
    Decide number of repeat users, i.e., m = k · α, and explore users, i.e, n = k − m;
    Construct the top-k users P_i^n using top-m repeat users and top-n explore users;
end
```

$\hat{U}_i \in U_i^t$. After candidate filtering, we can reduce the computational costs by only considering users within the candidate set.

4.6.1 Repetition-rule based candidate filtering

According to the repetition analysis in Section 4.4, users have regular habits in grocery shopping and category-level repetition behavior seems more stable and prominent than item-level repetition behavior. Next, we first propose two repetition rule-based candidate filtering methods, i.e., RRBF-item and RRBF-cat. Formally,

RRBF-item: For a given item i, we only select the *repeat users* U_i^{rep} to form up the candidate set, that is, $\hat{U}_i = U_i^{rep}$. This method is designed for Mixed-RNPR, which helps to reduce Mixed-RNPR to Rep-RNPR.[7]

RRBF-cat: For a given item i, we get its corresponding category c^i and only select the explore users U_i^{expl} who have previously purchased items from c^i to form the candidate set, that is, $\hat{U}_i = U_c^{rep} \cap U_i^{expl}$. Note that RRBF-cat is used in the Expl-RNPR task.

4.6.2 Model-based candidate filtering

Repetition-rule based filtering (RRBF) does not consider the temporal information and is static. To further reduce the number of candidates on top of

[7] RRBF-item cannot be used for Expl-RNPR, since it only selects repeat users as candidates.

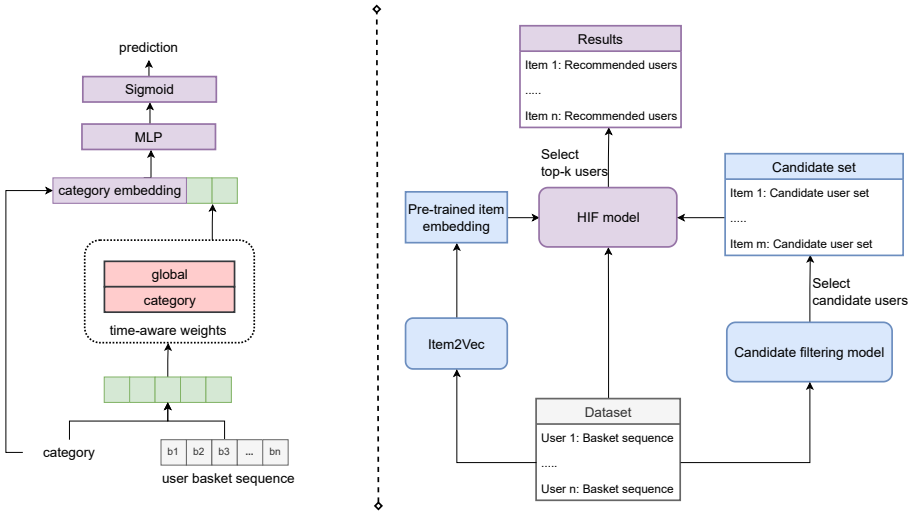


Figure 4.3: The architecture of candidate filtering model (left) and the overall pipeline (right) of Expl-RNPR task.

RRBF-cat, we propose a *candidate filtering model* (CFM) to predict whether a user likes to repurchase the category or not, then select the users who would like to purchase the category of the given item. The architecture of the candidate filtering model is shown in Figure 4.3. Note that the category catalog is relatively stable and small compared to the item catalog, and the items within the same category can share the same set of candidate users.

Model

CFM predicts users' repurchase behavior by modeling their dynamic demands within the target category. For the sake of simplicity, we do not consider dependencies among different categories. Specifically, we use the category-level repetition frequency vector $RepVec$ of category $c \in C$ (defined in Eq. 4.15), which contains both temporal information and frequency information.

Different categories might have different characteristics w.r.t. repurchase behavior. For example, daily necessities like fruit might be purchased during every visit to the grocery store, whereas users are less likely to repurchase household items like dish soap right after their previous purchase of dish soap. Therefore, we introduce a category specific time-aware weight embedding TW^c_{cf} to model temporal dependencies. In addition, we also introduce a global time-aware weight embedding TW^g_{cf} which can be shared and trained by all categories, so that different categories can benefit from the training samples of each other. Given a use u and a category c, we first derive the category specific repetition feature $RepF^c_{u,c}$ and the general repetition feature $RepF^g_{u,c}$, then we concatenate $RepF^c_{u,c}$ and $RepF^g_{u,c}$ with a category embedding Emb^c_{cf}, and feed it to a

two layer fully-connected neural network, that is:

$$RepF^{c}_{u,c} = TW^{c}_{cf} \cdot RepVec_{u,c} \tag{4.26}$$

$$RepF^{g}_{u,c} = TW^{g}_{cf} \cdot RepVec_{u,c} \tag{4.27}$$

$$p_{u,c} = \text{Sigmoid}(\text{FFN}(RepF^{c}_{u,c} \oplus RepF^{g}_{u,c} \oplus Emb^{c}_{cf})), \tag{4.28}$$

where $p_{u,c}$ is the probability that user u will purchase items within the category c; $\oplus$ denotes the concatenation operation.

Once we obtain the repurchase probabilities $p_{u,c}$ of the target users, we can set a filtering threshold λ to filter candidates. For a given item $i \in I^c$, we only select users whose $p_{u,c}$ is above the filtering threshold λ as candidates. The overall pipeline of Expl-RNPR is shown in Figure 4.3.

Training

In the training set, every user has a ground-truth label for each category, i.e., whether the user has repurchased from the category or not. Conversely, for a specific category, we can split users into positive users and negative users. However, positive users and negative users within a category might be imbalanced. Besides, the positive users are unevenly distributed across all categories, e.g., a popular category is likely to have more positive users than a less popular category.

To overcome the problems of imbalanced data listed above, we sample the same number of training instances (users) U^c for each category in every epoch instead of using all users, and balance the number of positive users and negative users within each category. Then, we use binary cross-entropy loss to train our model:

$$\mathcal{L}_c = -\frac{1}{|U^c|} \sum_{u \in U^c} y_{u,c} \log(p_{u,c}) + (1 - y_{u,c}) \log(1 - p_{u,c}), \tag{4.29}$$

where $y_{u,c}$ and $p_{u,c}$ denote the ground-truth of category c and the probability of c being purchased by user u in the next period.

4.7 Experimental Setup

In this section, we describe our experimental settings, including our research questions, datasets, constructed baselines, parameter settings, and evaluation metrics.

4.7.1 Research questions

To better understand the RNPR problem and investigate the effectiveness of the proposed methods, we intend to answer the following questions through our experiments:

RQ3.1 How do user-centered state-of-the-art NBR methods perform on the RNPR task?

RQ3.2 What is the effectiveness of our newly proposed methods? Do they outperform existing baselines?

RQ3.3 What is the effectiveness of our training strategies for the HIF model in Expl-RNPR?

RQ3.4 What are the differences and trade-offs between the repetition and exploration tasks in Mixed-RNPR?

RQ3.5 Do the proposed candidate filtering strategies help to reduce computational costs at inference time? How does the candidate filtering process influence the performance of our models?

4.7.2 Datasets

In order to ensure the reproducibility of our study, we conduct our experiments on two publicly available real-world datasets:

Dunnhumby: covers two years of household-level transactions at a retailer from a group of 2,500 households. All products bought by the same customer in the same transaction are treated as a basket.[8]

Instacart: contains over three million grocery orders of Instacart users. We treat all items purchased in the same order as a basket.[9]

In each dataset, we sample active users with at least 30 baskets in the dataset, and truncate a basket sequence to 100 baskets. We follow a strategy that is similar to the widely used leave-one-out approach to split the dataset. Specifically, for each user, the last 10 baskets are regarded as future baskets, and all remaining baskets are regarded as historical baskets. All training is conducted using the historical data. We select target items according to their frequency in the ground-truth (a.k.a. future data) to ensure there are new users for this item in the next period, otherwise we can only add zero to the evaluation metrics. Note that using the explore users' frequency in the ground-truth instead of the repeat users' not only allows us to address the Expl-RNPR task, but also to make a fair comparison between the Expl-RNPR task and the Mixed-RNPR task. Considering the number of users in each dataset, the minimum number of future new users' for an item is set to 50 for the Instacart dataset, and 10 for the Dunnhumby dataset. Since we use category information in our model, splitting across items has a risk of information leakage. So we split the items into validation and test dataset according to their corresponding category [36], 50% categories for validation, and 50% categories for testing. We repeat the split 5 times and report the average performance w.r.t. metrics over the 5 splits.

[8] `https://www.dunnhumby.com/source-files/`
[9] `https://www.kaggle.com/c/instacart-market-basket-analysis/data`

Table 4.4: Dataset statistics after preprocessing.

Dataset	Users	Categories	Target items	Avg. #items per basket	Avg. #baskets per user	Avg. #item per user	Avg. #target users per item
Instacart	30,134	134	1,369	10.19	49.47	142.38	442.47
Dunnhumby	1,991	307	866	11.77	77.44	528.53	39.27

4.7.3 Baselines

We construct three simple methods, i.e., Random, I-TopFreq and C-TopFreq, two pre-training based methods, i.e., Basket2Vec and User2Vec, and select three SOTA NBR methods according to [69] and [4], i.e., DNNTSP [127], TIFUKNN [52] and ReCANet [4], as baseline methods for comparison:

Simple methods

We select three simple baseline models:

- Random selects the users for the given item at random.
- I-TopFreq ranks users according to their historical purchase frequency of the given input item, and selects users with the top-k highest purchase frequency. Since this method can only select *repeat users*, it will not be used in the Expl-RNPR task.[10]
- C-TopFreq ranks users according to their historical purchase frequency of different items within the given item's category, and selects users who prefer to purchase a lot of different items within the input item's category. This method is designed for the Expl-RNPR task, and it will not be used in the Rep-RNPR task.

Pre-training based methods

We select two pre-training based methods:

- Basket2Vec [105] pre-trains the item embeddings at the basket level, uses the average embedding of the items in historical baskets to represent the user, computes similarity between the user and the given item, and finally selects the top-k users who have the highest similarity with the given item.
- User2Vec [105] pre-trains the item embeddings on the user level, uses the average embedding of the items in historical baskets to represent the user, computes similarity between the user and the given item, and finally selects the top-k users who have the highest similarity with the given item.

[10]Repeat users play a vital role in Rep-RNPR and Mixed-RNPR, and item frequency serves as a reliable signal for identifying them for a particular item. We evaluated a similar category frequency-based method, but found that it did not outperform I-TopFreq w.r.t. Rep-RNPR and Mixed-RNPR, since using category frequency could result in noise and dilute the significance of item frequency information.

Next basket recommendation methods

We select three NBR methods as baselines:

- DNNTSP [127] is a state-of-the-art NBR method, which encodes the item-item relation via a GNN, and models temporal dependencies via self-attention techniques. After training DNNTSP, we first derive the target users' purchase probability of the items, and then select the top-k users with the highest purchase probability of the given item.[11]
- TIFUKNN [52] is a state-of-the-art NBR method, which constructs personal item frequency information (PIF) for each user, and leverages a KNN-based method based on PIF. We first get target users' scores of the item, then select the top-k users with the highest purchase score of the given item.
- ReCANet [4] is a state-of-the-art NBR method, which builds a neural-based temporal model to focus on recommending repeat items to the user in the NBR task. After training ReCANet, we first derive the target users' purchase probabilities of items, and then select the top-k users with the highest purchase probability of the given item.

4.7.4 Metrics

To assess the performance of RNPR methods, we extend three widely used user-centered metrics to the item-centered setting and arrive at the following metrics: *Recall*@K, *nDCG*@K and *IHN*@K.

Recall measures the ability to find all users who will purchase the given item in the next period:

$$Recall@K = \frac{1}{N}\sum_{j=1}^{N}\frac{\left|P_{i_j} \cap T_{i_j}\right|}{\left|T_{i_j}\right|}, \tag{4.30}$$

where P_{i_j} are the top-K predicted users for item i_j, and T_{i_j} denotes ground-truth users for item i_j.

nDCG is a ranking metric that also considers the order of the users:

$$nDCG@K = \frac{1}{N}\sum_{j=1}^{N}\frac{\sum_{k=1}^{K} p_k/\log_2(k+1)}{\sum_{k=1}^{\min(K,|T_{i_j}|)} 1/\log_2(k+1)}, \tag{4.31}$$

[11] When training DNNTSP, we use the binary cross entropy loss from the original paper [127] rather than devising a new user-centered loss compatible with their approach. There are several reasons for this. First, we want to explore the performance of user-centered model in the item-centered setting. Second, our proposed item-wise loss will only sample some of the users for training, which would be unfair for the DNNTSP model, since the HIF model can use all available users in its pre-training stage. Third, DNNTSP will learn and update item embeddings during the training process, which makes it hard to pre-compute user features to speed up training process and item-wise loss can only use one item's loss information at each backpropagation step, so training DNNTSP model via the proposed item-wise loss is extremely slow.

where p_k equals 1 if $P_{i_j}^k \in T_{i_j}$, otherwise $p_k = 0$. $P_{i_j}^k$ denotes the k-th user in the predicted user list P_{i_j} for item i_j.

IHN represents the average number of correct users the model can find for each item, that is:

$$IHN@K = \frac{1}{N} \sum_{j=1}^{N} \left| P_{i_j} \cap T_{i_j} \right|. \tag{4.32}$$

Since the two datasets that we use, Instacart and Dunnhumby, have different numbers of users, when we evaluate the Expl-RNPR and Mixed-RNPR performance, the value of K for the Instacart dataset is set to 100 and 200, the value of K for the Dunnhumby dataset is set to 50 and 100. Since the number of repeat user candidates is limited, the K is set to 20 and 50 for Rep-RNPR task.

4.7.5 Configurations

For all experiments, we set the next period size to 5 and 10 baskets. For Basket2Vec and User2Vec the embedding size is set to 100 for all datasets. For TIFUKNN, the number of neighbors is selected from $[100, 200, 300, 400, 500]$, the fusion weight and time-decay factor are selected from $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]$, and the window size is set to the next period size, i.e., 5 or 10.

DNNTSP and ReCANet are used with the same parameter settings as in [127] and [4], respectively. For HIF, we use the same pre-trained item embeddings as Basket2Vec to make a fair comparison with it. For both the HIF model and CFM model, the hidden layer of the fully connected network is set to 32, the maximum user sequence is set to 30. For REUR, the time-decay factor β is chosen from $[0.5, 0.6, 0.7, 0.8, 0.9]$, we sweep the combination ratio α in $[0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]$ to investigate the trade-off and find the optimal α for Mixed-RNPR. We use Adam optimizer with 0.001 as the learning rate and 256 as the batch size to train our models.

We use PyTorch to implement our model and train it using a TITAN X GPU with 12G memory. We repeat our experiments 5 times and report the average results. We share the code and parameters in a public GitHub repository.[12]

4.8 Experimental Results

In this section, we report on the experimental results to answer the research questions listed in Section 4.7.1.

4.8.1 Performance comparison of "user-centered" methods

Table 4.5, Table 4.6 and Table 4.7 show the performance of all approaches for the Expl-RNPR task, Rep-RNPR task and the Mixed-RNPR task, respectively. We have several findings based on the experimental results. As expected, the performance of Random is always among the lowest in terms of all metrics for

[12] `https://github.com/liming-7/RNPR-Rec`

Table 4.5: Exlp-RNPR results of HIF compared against the baselines. Boldface and underline indicate the best performing model and the best performing baseline, respectively. Significant improvements of HIF over the best performing baseline results are marked with † (paired t-test, $p < 0.05$). ▲% indicates the relative improvements of HIF against the best performing baseline.

Dataset			Baselines						Ours	
Period	Metric	Random	C-TopFreq	Basket 2Vec	User 2Vec	DNNTSP	TIFUKNN	HIF-max (▲%)	HIF-mean (▲%)
Instacart										
5	Recall@100	0.0036	0.0068	0.0211	0.0119	0.0035	0.0148	0.0161	**0.0251**† (19.0)
	nDCG@100	0.0055	0.0079	0.0276	0.0146	0.0055	0.0232	0.0213	**0.0346**† (25.4)
	IHN@100	0.4909	0.6932	2.3655	1.2823	0.4962	2.0027	1.9109	**2.9315**† (23.9)
	Recall@200	0.0067	0.0135	0.0388	0.0237	0.0070	0.0268	0.0307	**0.0444**† (14.4)
	nDCG@200	0.0067	0.0114	0.0359	0.0207	0.0071	0.0275	0.0281	**0.0427**† (15.9)
	IHN@200	0.9708	1.4125	4.4617	2.5971	1.0163	3.6394	3.6685	**5.2537**† (17.8)
10	Recall@100	0.0033	0.0096	0.0206	0.0115	0.0039	0.0147	0.0158	**0.0237**† (15.0)
	nDCG@100	0.0092	0.0134	0.0453	0.0237	0.0104	0.0389	0.0412	**0.0566**† (24.9)
	IHN@100	0.9142	1.1718	4.4471	2.4161	1.0094	3.7418	3.9003	**5.3562**† (20.4)
	Recall@200	0.0072	0.0182	0.0376	0.0225	0.0071	0.0266	0.0295	**0.0423**† (12.5)
	nDCG@200	0.0107	0.0173	0.0490	0.0274	0.0109	0.0396	0.0431	**0.0582**† (18.7)
	IHN@200	1.9561	2.2505	8.3443	4.8073	1.9528	6.8362	7.2453	**9.6829**† (16.0)
Dunnhumby										
5	Recall@50	0.0292	0.0420	0.0623	0.0286	0.0284	0.0647	0.0756	**0.0841**† (30.0)
	nDCG@50	0.0193	0.0270	0.0410	0.0174	0.0185	0.0442	0.0530	**0.0600**† (35.7)
	IHN@50	0.4690	0.5893	1.0022	0.4506	0.4548	1.0305	1.1617	**1.3001**† (26.2)
	Recall@100	0.0569	0.0786	0.1134	0.0637	0.0560	0.1146	0.1323	**0.1514**† (32.1)
	nDCG@100	0.0304	0.0416	0.0620	0.0320	0.0298	0.0648	0.0763	**0.0877**† (26.1)
	IHN@100	0.9154	1.1224	1.8280	1.0329	0.9055	1.8261	2.0620	**2.3643**† (22.7)
10	Recall@50	0.0290	0.0423	0.0584	0.0305	0.0278	0.0642	0.0753	**0.0848**† (32.1)
	nDCG@50	0.0247	0.0351	0.0493	0.0244	0.0233	0.0578	0.0657	**0.0773**† (33.7)
	IHN@50	0.8952	1.3482	1.7667	0.9280	0.8506	1.9304	2.1937	**2.5056**† (29.8)
	Recall@100	0.0562	0.0791	0.1128	0.0643	0.0566	0.1168	0.1319	**0.1490**† (27.6)
	nDCG@100	0.0374	0.0569	0.0761	0.0410	0.0369	0.0829	0.0912	**0.1082**† (30.5)
	IHN@100	1.7369	2.681	3.4384	1.9602	1.7557	3.4876	3.9814	**4.4191**† (26.7)

all tasks, as it just randomly selects users. C-TopFreq outperforms Random on the Expl-RNPR task since it models users' interests w.r.t. the category of the target item. I-TopFreq achieves competitive or even better performance compared to other approaches on both the Rep-RNPR and Mixed-RNPR task, which confirms that item repetition frequency information is important and only considering *repeat users* can achieve quite good performance on the Mixed-RNPR task.

Among the two pre-training based methods, Basket2Vec always achieves better performance than User2Vec in all cases, which indicates that basket-level pre-training can get better item representations than user-level pre-training. We suspect that this is because users' interests are dynamic and items purchased at the same time have more similarity. Basket2Vec is the best performing approach on the Expl-RNPR task, but Basket2Vec is inferior to I-TopFreq on both the Rep-RNPR and Mixed-RNPR task, which suggests that the item-item correlation is less important than the item repetition frequency information in these two tasks.

Surprisingly, the state-of-the-art DNNTSP method performs poorly on the Expl-RNPR task; its performance is in the same range as that of Random. DNNTSP is supposed to capture item-item correlations effectively, as it leverages advanced techniques, i.e., a GNN and self-attention mechanism. The ReCANet method achieves quite good performance on the Rep-RNPR task and the Mixed-RNPR task, which further emphasizes the importance of modeling repeat users in the Mixed-RNPR task. However, ReCANet does not outperform the SimpleTF method w.r.t. these two item-centered tasks. A plausible reason is that the user-wise binary cross entropy loss tries to distinguish items for a given user, which is sub-optimal for "item-centered" recommendations, where the goal is to distinguish users for a given item. Compared to DNNTSP, the relatively simple TIFUKNN is more robust on the Expl-RNPR task and even achieves the best performance (amongst the baselines) on the Dunnhumby dataset and the second best performance on the Instacart dataset. For the Rep-RNPR task and Mixed-RNPR task, TIFUKNN achieves good performance on the Instacart dataset, as it adopts personal item frequency information. However, it performs worse than the simple I-TopFreq approach on Dunnhumby; we suspect that the underlying reason is that the KNN module has a negative impact on finding repeat users in the Rep-RNPR task and Mixed-RNPR task.

To sum up, the performance of state-of-the-art NBR methods does not always generalize to the Expl-RNPR task, and they are overly complex for the Rep-RNPR task and the Mixed-RNPR task. This section answers RQ3.1.

4.8.2 Performance of our proposed methods

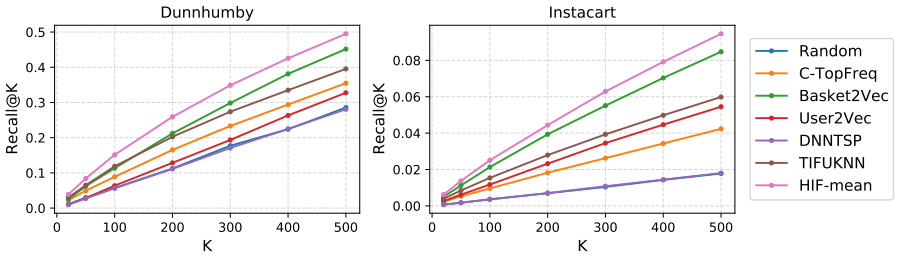
For the Expl-RNPR task, the performance of the HIF model is shown in Table 4.5 and Figure 4.4a. We can make two main observations based on the results. First, HIF with the average pooling strategy (HIF-mean) significantly outperforms all existing approaches on both datasets across all metrics, with improvements ranging from 14.4% to 35.7%, since HIF models users' interests

Table 4.6: Rep-RNPR results of the simple time-aware frequency model (SimpleTF) and the baselines. Boldface and underline indicate the best and the second best performing model, respectively. Significant improvements of SimpleTF over the best performing baseline results are marked with † (paired t-test, $p < 0.05$). ▲% indicates the relative improvements of SimpleTF against the best performing baseline.

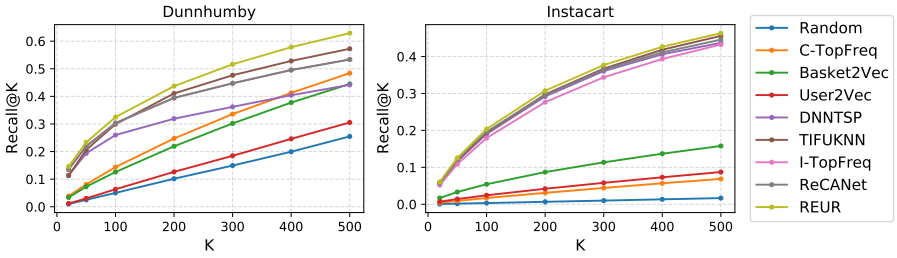
Dataset											
Period											
	Metric	Random	I-TopFreq	Basket 2Vec	User 2Vec	DNNTSP	TIFUKNN	ReCANet	SimpleTF (▲%)
Instacart											
5	Recall@20	0.0262	0.0975	0.0535	0.0408	0.1020	0.1050	0.1092	**0.1129**† (3.4)
	nDCG@20	0.1949	0.6928	0.4528	0.3589	0.7545	0.7598	0.7692	**0.7972**† (3.6)
	IHN@20	3.9027	13.3096	8.5759	6.8230	14.5037	14.6265	14.8567	**15.3445**† (3.2)
	Recall@50	0.0661	0.1987	0.1165	0.0919	0.2071	0.2124	0.2191	**0.2264**† (3.3)
	nDCG@50	0.1983	0.6319	0.4125	0.3296	0.6845	0.6886	0.7024	**0.7249**† (3.2)
	IHN@50	9.7476	29.2964	19.1395	15.3787	31.8664	32.0061	32.5761	**33.6926**† (3.4)
10	Recall@20	0.0265	0.0806	0.0486	0.0383	0.0808	0.0842	0.0893	**0.0922**† (3.3)
	nDCG@20	0.2714	0.7795	0.5421	0.4441	0.8205	0.8292	0.8458	**0.8617** (1.8)
	IHN@20	5.4067	15.1624	10.4104	8.5298	15.9418	16.1461	16.4628	**16.9079**† (2.7)
	Recall@50	0.0656	0.1715	0.1083	0.0883	0.1718	0.1778	0.1838	**0.1897**† (3.2)
	nDCG@50	0.2707	0.7230	0.5005	0.4136	0.7566	0.7644	0.7841	**0.7998** (2.0)
	IHN@50	13.4393	34.4646	23.7418	19.7032	36.1090	36.4683	37.2163	**38.2577**† (2.8)
Dunnhumby											
5	Recall@20	0.0845	0.2674	0.1374	0.1027	0.2205	0.2183	0.2670	**0.2821**† (5.4)
	nDCG@20	0.1072	0.4047	0.1973	0.1394	0.3457	0.3252	0.4051	**0.4283**† (5.7)
	IHN@20	1.9041	6.4474	3.3213	2.3951	5.4764	5.2513	6.4471	**6.8007**† (5.5)
	Recall@50	0.2081	0.4516	0.2944	0.2383	0.3983	0.4116	0.4500	**0.4734**† (4.8)
	nDCG@50	0.1580	0.4481	0.2534	0.1944	0.3896	0.3825	0.4479	**0.4711**† (5.1)
	IHN@50	4.7051	11.5646	7.1853	5.7145	10.1565	10.1866	11.4604	**12.0355**† (4.1)
10	Recall@20	0.1076	0.2854	0.1618	0.1219	0.2379	0.2436	0.2867	**0.3049**† (6.3)
	nDCG@20	0.1575	0.4767	0.2544	0.1785	0.4017	0.3875	0.4783	**0.5077**† (6.1)
	IHN@20	2.9360	7.8469	4.4430	3.2380	6.6009	6.5305	7.8477	**8.3416**† (6.2)
	Recall@50	0.2647	0.4860	0.3564	0.2932	0.4430	0.4587	0.4856	**0.5096**† (4.8)
	nDCG@50	0.2206	0.5097	0.3197	0.2463	0.4463	0.4442	0.5092	**0.5379**† (5.5)
	IHN@50	7.1424	14.3655	9.7911	7.8676	12.7583	12.9982	14.3637	**14.9963**† (4.3)

Table 4.7: Mixed-RNPR results of the REUR algorithm and the baselines. Boldface and underline indicate the best and the second best performing model, respectively. Significant improvements of REUR over the best performing baseline results are marked with † (paired t-test, $p < 0.05$). ▲% indicates the relative improvements of REUR against the best performing baseline.

Dataset	Period	Metric	Random	I-TopFreq	C-TopFreq	Basket 2Vec	User 2Vec	DNNTSP	ReCANet	TIFUKNN	REUR (▲%)
Instacart	5	Recall@100	0.0034	0.1794	0.0170	0.0542	0.0243	0.1904	0.1933	0.1941	**0.2037**† (4.9)
		nDCG@100	0.0158	0.5490	0.0650	0.2332	0.1223	0.5933	0.5983	0.5999	**0.6270**† (5.9)
		IHN@100	1.5835	50.8376	6.2624	20.7245	10.9495	54.7954	55.2954	55.4311	**57.9105**† (4.5)
		Recall@200	0.0066	0.2764	0.0309	0.0871	0.0420	0.2920	0.2965	0.2982	**0.3074**† (3.1)
		nDCG@200	0.0162	0.4929	0.0630	0.2030	0.1098	0.5288	0.5316	0.5360	**0.5568**† (3.9)
		IHN@200	3.1285	83.8501	11.5067	34.6381	19.2747	89.7975	90.1346	91.0450	**94.3099**† (3.6)
	10	Recall@100	0.0033	0.1409	0.0163	0.0448	0.0208	0.1446	0.1463	0.1477	**0.1560**† (5.6)
		nDCG@100	0.0238	0.6511	0.0956	0.2829	0.1500	0.6794	0.6809	0.6871	**0.7206**† (4.9)
		IHN@100	2.3681	61.5801	9.2279	25.6278	13.6983	64.0579	64.2261	64.8748	**68.1811**† (5.1)
		Recall@200	0.0065	0.2262	0.0294	0.0736	0.0369	0.2328	0.2356	0.2380	**0.2471**† (3.8)
		nDCG@200	0.0239	0.5685	0.0889	0.2434	0.1341	0.5913	0.5946	0.5999	**0.6252**† (4.2)
		IHN@200	4.7845	104.8940	16.9723	43.7287	24.6122	108.8507	109.6201	110.6156	**115.1574**† (4.1)
Dunnhumby	5	Recall@50	0.0254	0.2190	0.0810	0.0726	0.0302	0.1937	0.2187	0.2054	**0.2333**† (6.5)
		nDCG@50	0.0250	0.2612	0.0774	0.0772	0.0310	0.2271	0.2613	0.2234	**0.2821**† (8.0)
		IHN@50	0.9308	7.5279	2.6521	2.7403	1.2303	6.7633	7.5267	6.8094	**8.0548**† (7.0)
		Recall@100	0.0508	0.3025	0.1436	0.1259	0.0633	0.2597	0.3022	0.2996	**0.3248**† (7.4)
		nDCG@100	0.0357	0.2910	0.1051	0.0990	0.0456	0.2505	0.2907	0.2615	**0.3150**† (8.2)
		IHN@100	1.8140	10.8553	4.7957	4.7572	2.5121	9.7879	10.8014	10.4737	**11.7091**† (7.9)
	10	Recall@50	0.0256	0.2037	0.0758	0.0677	0.0292	0.1784	0.2042	0.1921	**0.2171**† (6.6)
		nDCG@50	0.0356	0.3162	0.1018	0.1015	0.0427	0.2746	0.3166	0.2723	**0.3402**† (7.5)
		IHN@50	1.6067	11.7383	4.3820	4.3947	2.0023	10.4649	11.7391	10.6945	**12.5645**† (7.0)
		Recall@100	0.0496	0.2876	0.1369	0.1206	0.0620	0.2420	0.2870	0.2868	**0.3089**† (7.4)
		nDCG@100	0.0451	0.3280	0.1262	0.1191	0.0571	0.2802	0.3274	0.2977	**0.3536**† (7.8)
		IHN@100	3.1421	17.3898	8.0764	7.8467	4.1466	15.4814	17.3895	16.8825	**18.7001**† (7.5)



(a) Expl-RNPR task.



(b) Mixed-RNPR task.

Figure 4.4: The Recall@K performance of various methods with K ranging from 20 to 500.

via item-item correlations and models habits via frequency information at the same time.[13] Second, the performance of HIF is different when using different basket pooling strategies, and the average pooling strategy has a clear advantage over the max pooling strategy on both datasets. This result indicates that average pooling retains more useful information and leads to better basket representations in the HIF model.

For the Rep-RNPR task, the performance of SimpleTF is shown in Table 4.6.[14] Surprisingly, the SimpleTF method outperforms many complex neural methods, with improvements ranging from 1.8% to 6.3%. This result indicates that not all user-centered techniques, e.g., learning item representations and leveraging neighbor information, are helpful in the item-centered recommendation setting.

For the Mixed-RNPR task, the performance of REUR is shown in Table 4.7 and Figure 4.4b. REUR achieves the best performance on both datasets through the combination of *repeat users* ranking (derived from a simple time-aware frequency model) and *explore users* ranking (derived from the HIF model). The improvements range from 3.1% to 8.2% over the best baseline methods. Be-

[13] All occurrences of HIF, unless otherwise stated, refer to HIF-mean, i.e., the HIF model with average pooling strategy.

[14] It is not meaningful to evaluate the Recall@K for large values of K in the Rep-RNPR task, as the number of repeat user candidates is limited.

Table 4.8: Expl-RNPR ablation study results. Boldface indicates the best performing model, i.e., HIF with average pooling. Significant deteriorations compared to HIF are marked with ‡ (paired t-test, $p < 0.05$). ▼% indicates the relative drop in performance compared to HIF.

Dataset	Metric	HIF	w/o hab (▼%)	w/o aug-pos (▼%)	w/o adj-neg (▼%)	w/o aug-pos adj-neg(▼%)
Instacart	Recall@100	**0.0251**	0.0237‡(5.9)	0.0233‡(7.2)	0.0198‡(21.1)	0.0206‡(17.9)
	nDCG@100	**0.0346**	0.0311‡(10.1)	0.0310‡(10.4)	0.0282‡(18.5)	0.0289‡(16.5)
	IHN@100	**2.9315**	2.6538‡(9.5)	2.6365‡(10.1)	2.3985‡(18.2)	2.4994‡(14.7)
	Recall@200	**0.0444**	0.0422‡(5.0)	0.0426‡(4.1)	0.0363‡(18.2)	0.0390‡(12.2)
	nDCG@200	**0.0427**	0.0394‡(7.7)	0.0401‡(6.1)	0.0350‡(18.0)	0.0369‡(13.6)
	IHN@200	**5.2537**	4.8525‡(7.6)	4.9452‡(5.9)	4.4325‡(15.6)	4.7102‡(10.3)
Dunnhumby	Recall@50	**0.0841**	0.0718‡(14.6)	0.0833 (1.0)	0.0791‡(5.9)	0.0808‡(3.9)
	nDCG@50	**0.0600**	0.0480‡(20.0)	0.0589 (1.8)	0.0565‡(5.8)	0.0583‡(2.8)
	IHN@50	**1.3001**	1.1472‡(11.8)	1.2984 (0.1)	1.2388‡(4.7)	1.2919 (0.6)
	Recall@100	**0.1514**	0.1281‡(15.4)	0.1456‡(3.8)	0.1376‡(9.1)	0.1436‡(5.1)
	nDCG@100	**0.0877**	0.0710‡(19.0)	0.0845‡(3.6)	0.0806‡(8.1)	0.0832‡(5.1)
	IHN@100	**2.3643**	2.0351‡(13.9)	2.2834‡(3.4)	2.1659‡(8.4)	2.2777‡(3.6)

sides achieving the best performance, an important advantage of the REUR algorithm is that we can explicitly investigate the trade-off between repetition and exploration, which will be discussed in Section 4.8.4. Note that REUR decouples repetition and exploration, which allows it to benefit from the candidate filtering part by only considering *repeat users*, which we will discuss later in Section 4.8.5.

To sum up, the HIF model, SimpleTF model and REUR algorithm are the state-of-the-art methods on the Expl-RNPR, Rep-RNPR and Mixed-RNPR tasks, respectively. This section answers RQ3.2.

4.8.3 Ablation study of HIF

To analyze the effectiveness of our proposed training strategies and answer RQ3.3, we conduct an ablation study on the two datasets. Specifically, we compare the performance of the full HIF model with the following four settings:

1. No habits module (HIF w/o hab).
2. No positive augmentation strategy (HIF w/o aug-pos).
3. No negative adjustment strategy (HIF w/o adj-neg).
4. No postive augmentation strategy and no negative adjustment strategy (HIF w/o aug-pos and adj-neg).

The results of the ablation study are shown in Table 4.8. The results show that both the habits capturing module and the positive augmentation strategy and negative adjustment strategy are beneficial for the HIF because removing

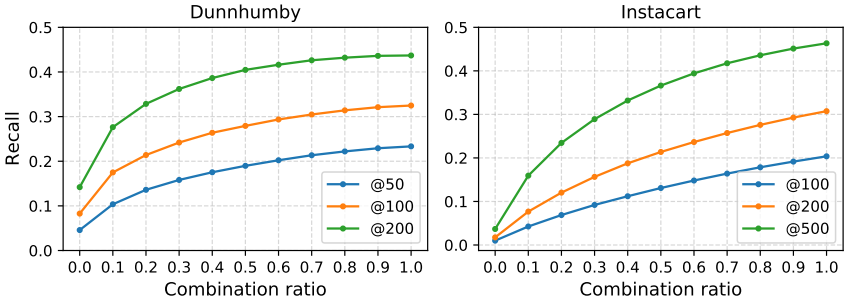
any of them will lead to a decrease in performance. Without habits module, the performance of HIF decreases, ranging from 5% to 20%, which indicates that the frequency information is valuable and the designed habits module is able to leverage this information to model users' shopping habits. Furthermore, HIF w/o hab still outperforms the Basket2Vec baseline (in Table 4.5), which indicates that the interest module of HIF can capture users' dynamic interests by modeling the dynamic user preferences.

When we employ the positive augmentation strategy, *repeat users* will be sampled and truncated for training. Without positive augmentation, the performance of HIF drops significantly on the Instacart dataset w.r.t. all metrics, ranging from 4.1% to 10.4%, while the drop is not significant in terms of several metrics, i.e., Recall@50, nDCG@50 and IHN@50, on the Dunnhumby dataset. A plausible reason for this result is that the Instacart dataset has more users to be ranked, which means that the Expl-RNPR task is more difficult on Instacart dataset, so HIF can benefit more from the augmented postive samples on the Instacart dataset but the original training samples are enough for finding the top-50 users on the Dunnhumby dataset. Training HIF without negative adjustment strategy results in 4.7% to 21.1% drops in performance. This indicates that avoiding the overlap between training input and prediction input is important and effective in training the HIF model for the Expl-RNPR task. Interestingly, training HIF without both sampling strategies (HIF w/o aug-pos and adj-neg) achieves better performance than training without negative adjustment strategy. The positive augmentation strategy can help us generate more positive samples, which means that the HIF model will also leverage more negative samples during training, and this will increase the probability of using the historical data of the ground-truth users. As a result, negative adjustment is especially important when using positive augmentation strategy, and HIF w/o adj-neg is inferior to other variants of HIF.

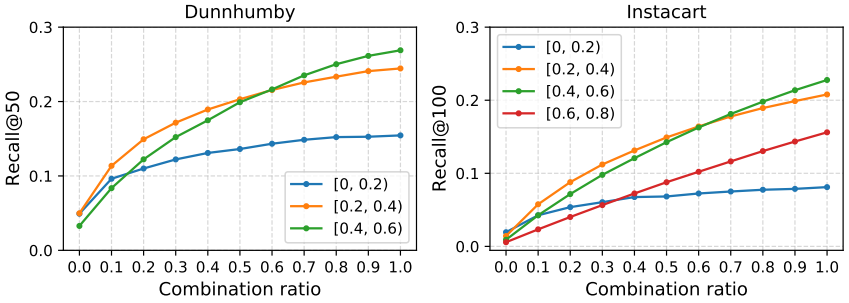
4.8.4 Trade-off analysis for Mixed-RNPR

To investigate the trade-off between repetition and exploration in the Mixed-RNPR task (RQ3.4), we use the proposed REUR algorithm and sweep its combination (repeat) ratio α. Figure 4.5a shows the repetition and exploration trade-off on different datasets when using the same combination ratio α. As the proportion of *repeat users* increases in the recommendation, the performance of REUR on the Mixed-RNPR task increases. REUR achieves its best performance when the combination ratio is 1.0, which indicates that REUR has more confidence in the last user in the *repeat user* ranking than in the first user in the *explore user* ranking. This suggests that the repetition task is much easier than the exploration task in the "item-centered" recommendation that we consider in this chapter.

As mentioned before, the repurchase tendency RT of an item can potentially influence the trade-off. To further understand the repetition and exploration trade-off, we also investigate this trade-off on different groups items. To ensure there are enough items within the group, we explore three groups of items, i.e,



(a) Overall analysis.



(b) Group-level analysis.

Figure 4.5: Repetition-exploration trade-off analysis.

$RT \in [0, 0.2)$, $[0.2, 0.4)$, $[0.4, 0.6)$ on the Dunnhumby dataset and four groups $RT \in [0, 0.2)$, $[0.2, 0.4)$, $[0.4, 0.6)$, $[0.6, 0.8)$ on the Instacart dataset. The results are shown in Figure 4.5b. Interestingly, we observe the same trend in all groups of items, the performance increases when the repeat ratio α increases, even in the group with a very low repurchase tendency, i.e., $RT \in [0, 0.2)$. This result once again confirms the large gap in difficulty between the repetition task and the exploration task.

4.8.5 Candidate filtering

To answer RQ3.5, we first evaluate the effectiveness of candidate filtering, and then discuss the influence on the Expl-RNPR task and insights on the Mixed-RNPR task.

Effectiveness of candidate filtering

To reduce the computational costs at inference time, the candidate filtering process selects a subset of users from the whole user set, which might remove some of the ground-truth users. A good candidate filtering strategy should reduce

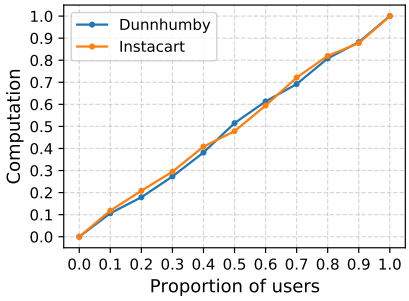


Figure 4.6: The relation of actual computation cost and proportion of users that HIF model computes.

Table 4.9: The proportion of ground-truth users (PoG) and computational costs (CP) of RRBF.

Methods	RRBF-cat for Expl-RNPR		RRBF-item for Mixed-RNPR	
Dataset	PoG	Computation (CP)	PoG	Computation (CP)
Instacart	77.70%	60.08%	61.11%	4.5%
Dunnhumby	87.54%	76.12%	43.74%	11.3%

computational costs, i.e., exclude users who have a low purchase probability w.r.t. the given item, while keeping a large proportion of the ground-truth users (PoG), i.e., retain as many ground-truth users as possible. Therefore, we analyze the proportion of ground-truth users (PoG) among the candidate users that are left after filtering to evaluate their performance, that is:

$$PoG = \frac{1}{|I^t|} \sum_{i \in I^t} \frac{|T_i \cap \hat{U}_i^t|}{|T_i|},$$

where I^t is a set of items we want to recommend, T_i denotes the ground-truth users, $\hat{U}_i^t$ denotes the set of candidate users for item i after candidate filtering. Note that the computation CP represents a percentage of the original computational costs, that is:

$$CP = \frac{\sum_{i \in I^t} |U_i^t|}{\sum_{i \in I^t} |\hat{U}_i^t|}.$$

Intuitively, we expect the computational costs to increase linearly with the number of item-user scores the model needs to calculate. To validate this, we conduct experiments to investigate the correlation between actual inference times and the defined computational costs CP. As shown in Figure 4.6, a linear correlation between the actual inference times and CP is generally observed, and we believe that the minor deviations are a result of the varying lengths

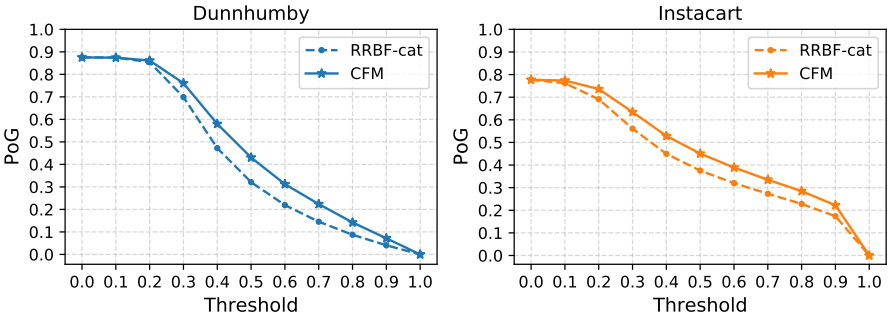


Figure 4.7: Proportion of ground-truth users with the CFM filtering threshold changing from 0 to 1. RRBF-cat represent the random strategy within the RRBF-cat candidates.

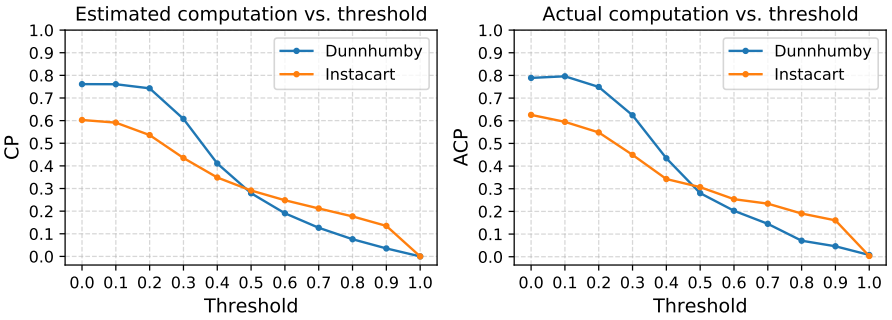


Figure 4.8: Computational costs (left) and actual computational cost (right) with the CFM filtering threshold changing from 0 to 1.

of users' historical baskets.[15] Table 4.9 shows the experimental results for the two repetition rule-based candidate filtering methods introduced in Section 4.6, i.e., RRBF-cat and RRBF-item. Theoretically, the *PoG* of a random candidate filtering strategy, i.e., randomly selecting a subset of target users as candidates, should be proportional to its computational cost *CP*. Clearly, both RRBF-cat and RRBF-item are effective, since they have a higher left proportion of ground-truth users *PoG* in the candidate user set than using a random strategy. RRBF-cat reaches 77.7% and 87.54% w.r.t. Expl-RNPR *PoG* on Instacart and Dunnhumby, respectively. This result indicates that a large proportion of ground-truth *explore users*, who will explore the given item in the next period, should have already purchased some other items within the given item's category. RRBF-item retains a high proportion of ground-truth users *PoG* w.r.t. Mixed-RNPR on both datasets, while it only has 4.5% and 11.3% of the original

[15]The computation cost is normalized by the total computation cost by using all users as candidates for each item.

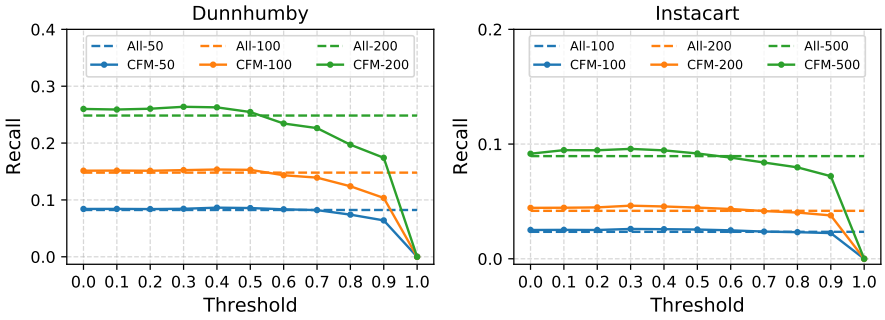


Figure 4.9: The Recall performance of HIF with the CFM filtering threshold changing from 0 to 1. The dashed line is the performance of using all target users.

computational costs on the Instacart and Dunnhumby datasets, respectively.

In practice, if the prediction results of the CFM model cannot be reused in other tasks in the platform, the computational costs of the CFM model should also be factored into the item-centered recommendation computational costs. We also conduct experiments to compare the actual inference time of the HIF model and CFM model. We find that the average inference times for the CFM model to compute one user-category score is $0.15ms$ on the Instacart dataset and $0.21ms$ on the Dunnhumby dataset, whereas the average inference time for the HIF model to compute one item-user score is $3.3ms$ on the Dunnhumby dataset and $2.1ms$ on the Instacart dataset. Considering that the number of categories is limited and the CFM model is much lighter than the HIF model, we also evaluated the actual computational costs compared to the original computational costs without candidate filtering and find that the total inference time of the CFM model constitutes only 0.31% and 0.78% of the original inference time on the Instacart and Dunnhumby datasets, respectively. Figure 4.7 and 4.8 show the results of model-based candidate filtering (CFM) on the Expl-RNPR task. As the filtering threshold increases, both the computation CP and the left proportion of ground-truth users PoG decreases, since more ground-truth users are removed from the candidate set with a higher filtering threshold. We can also observe that CFM has a higher Expl-RNPR PoG than using a random strategy within the candidate set of RRBF-cat in both datasets. This result indicates that CFM can further filter candidates effectively on top of RRBF-cat by considering temporal information.

Influence on the Expl-RNPR task

To understand the influence of candidate filtering on the performance of HIF for the Expl-RNPR task, we analyze the sensitivity of the performance of HIF w.r.t. the candidate filtering threshold of CFM. The experimental results are shown in Figure 4.9. Note that the performance of HIF when using RRBF-

cat is equal to the performance when using CFM with the filtering threshold 0, since they have the same set of candidates. We can observe that HIF with RRBF-cat can achieve same or even higher performance than computing on all target users, while HIF with RRBF-cat only needs 60.08% and 76.12% of the original computational costs on Instacart and Dunnhumby, respectively.

Increasing the CFM filtering threshold leads to a decrease in computational costs (see Figure 4.9), however, the performance of HIF remains at the same level as RRBF-cat until 0.5, and then decreases gradually as there are fewer candidate users left for each item. This result suggests that the CFM model can help HIF to remove a lot of users who have low probability of purchasing a given item by considering category-level repurchase behavior. With 0.5 as candidate filtering threshold, the CFM model is able to further reduce by 50% (Instacart) and 60.5% (Dunnhumby) the computational costs on top of RRBF-cat, while achieving the same level of performance as using RRBF-cat candidates or, indeed, using all users.

Insights on the Mixed-RNPR task

The trade-off analysis in Section 4.8.4 suggests that the repetition task is much easier than the exploration task, and that *repeat users* dominate the final recommendation set in REUR. Table 4.9 shows that only considering the *repeat users*, i.e., RRBF-item, can substantially reduce the computational costs. Besides, the repetition task can be solved by a simple time-aware frequency model, however the exploration task requires a model to infer users' interests in the given item by using item-item correlations and complex representations. Considering the above facts, it is reasonable to ask: can we ignore *explore users* and only consider *repeat users* in the Mixed-RNPR task? The answer might be different depending on different assumptions: (i) If there is no special demand for addressing the need of *explore users* and only focusing on optimizing the accuracy aspect, we believe that reducing the Mixed-RNPR task to the Rep-RNPR task, which simply recommends users who have purchased the given item before is an efficient and effective solution.[16] We believe this is an important point to make, even though this makes the Mixed-RNPR a less challenging algorithmic problem. (ii) An important goal of a recommender system is to connect items with new customers, if beyond accuracy metrics need to be considered, e.g., the long-term recommendation effect, the explore users should not be ignored and we should consider the balance between repeat users and explore users in the recommendation.[17]

[16]For instance, some bread might be close to its "best by date", and the supermarket may want to sell the bread without caring whom they sell it to.

[17]For instance, the supermarket wants to promote a certain product, so the item-centered recommender model not only needs to find repeat users, which is important for the short-term profit, but also find potential new users, who might account for long-term profit.

4.9 Conclusion

In this chapter, we have studied an item-centered sequential recommendation problem, i.e., reverse next-period recommendation (RNPR), which aims to help a given item find top-k users in the next period, to answer the thesis-level research question **RQ3**:

> *How to help a given item find its potential users in an item-centered setting, and how do repetition and exploration influence the design and optimization of the recommendation model?*

We have introduced three subtasks for RNPR, i.e., Expl-RNPR, Rep-RNPR and Mixed-RNPR, considering the differences in types of target users. For the Expl-RNPR task, we propose a habit-interest fusion model (HIF), which leverages frequency information to model users' habits and pre-trained embeddings to model users' interests. For training the HIF model, we propose two strategies to construct training samples, i.e., a positive augmentation strategy and a negative adjustment strategy, to construct training samples. For the Rep-RNPR task, we employ a simple time-aware frequency model, which only uses the users' direct interactions with the given item. For the Mixed-RNPR task, we propose a repetition-exploration ranking user (REUR) algorithm to decouple the repetition task and exploration task, and generate the final ranking by combining two ranked lists. In addition, we also examined how to reduce the computational costs of our approaches without losing performance. Specifically, we proposed two repetition-rule based filtering methods, i.e., RRBF-cat and RRBF-item, and a model-based candidate filtering method (CFM) to further reduce the computational costs of the HIF model during inference.

4.9.1 Main findings

We have performed extensive experiments on two publicly available grocery shopping datasets and the experimental results demonstrate the effectiveness of our proposed methods and strategies. The repetition analysis shows that people display stable repetition behavior at the category level in grocery shopping, which is a strong indicator that can be used to find potential top-k users for a given item. Our experiments further show that filtering out some target users using candidate filtering methods, i.e., RRBF and CFM, can effectively reduce computational costs without sacrificing performance.

Apart from proposing solutions, we have investigated the performance of state-of-the-art user-centered NBR models on the RNPR problem and found that their performance cannot always be generalized to the RNPR task, even though they are good at helping users find top-k items. This result suggests that we should not directly use user-centered recommendation (NBR) methods for the item-centered recommendation (RNPR) task, and that task-specific algorithms should be designed to cater for the RNPR task.

With the proposed REUR algorithm, we also investigated the trade-off between repetition and exploration in Mixed-RNPR. We found that the repetition

task, i.e., finding repeat users, is much easier than the exploration task, i.e., finding explore users, in item-centered sequential recommendation, and only recommending *repeat users* is an effective and efficient approach for the Mixed-RNPR task. This imbalance in difficulty can also be found in the user-centered NBR task [69]. A broader implication of this finding is that it is necessary to consider and investigate the differences and trade-offs between repetition and exploration in various recommendation scenarios.

4.9.2 Limitations

Despite the effectiveness of the proposed methods, one limitation of the HIF model is that it cannot be applied to find users for cold-start items currently. Even though we avoid using item-specific trainable parameters in the HIF model, it is still difficult to derive meaningful representations for cold-start items with limited interactions. One possible solution for cold-start items in the RNPR task is to employ a cold-start item representations learning method [see, e.g., 94].

4.9.3 Future work

Work on the RNPR task can be extended in a number of ways. For simplicity, we have employed a simple time-aware frequency model for the repetition task in this chapter. We intend to consider item characteristics and correlations to further improve the repetition performance. Second, we have used a conventional skip-gram algorithm to obtain pre-trained item embeddings in this chapter. Instead, we want to try and use recent graph neural networks to learn better representations for RNPR as a potential future work. Third, different items might have different numbers of potential users in the next period, so it is interesting to address the RNPR task with a dynamic number of users to recommend. Fourth, we have concentrated on maximizing accuracy, specifically in identifying the correct users for a given item, without considering the potential impact of recommending either repeat users or explore users. It would be interesting to examine the differences in the causal effect between repetition and exploration and to contemplate a way to achieve a balance between them. Finally, we have only focused on the RNPR problem itself in this chapter; it is of interest to investigate additional dimensions, such as its potential influence on user satisfaction or fairness among items.

In the next chapter, we will look into a more general recommendation setting, i.e., sequential recommendation (SR).

5 Sequential Item Recommendation

In the previous chapters, the importance of repetition and exploration has been discovered and studied. In this chapter, we move on to a more general recommendation setting, i.e., sequential recommenders (SR), where we aim to infer a user's preferences and suggest the next item for them to interact with based on their historical interaction sequences. There has not been a systematic analysis of sequential recommenders from the perspective of repetition and exploration. As a result, it is unclear how these models, that are typically optimized for accuracy, perform in terms of repetition and exploration, as well as the potential drawbacks of deploying them in real applications.

In this chapter, we address the thesis-level research question **RQ4**:

> *How do sequential recommendation models perform, and how should we evaluate item exposure from the perspective of repetition and exploration?*

5.1 Introduction

Recommender systems have become an essential instrument for connecting users and items on many online platforms [131]. Users may have dynamic interests over time, and sequential recommender systems aim to learn from users' historical interaction sequences to infer their preferences and suggest an appropriate next item for them to interact with [37, 85, 110]. Advances in deep learning have led to the development of numerous sequential recommendation models that employ deep learning techniques such as RNNs [48, 49], CNNs [96], GNNs [89, 117], contrastive learning [120], attention mechanisms [65, 73], and self-attention [58, 92, 101].

This chapter was published as: M. Li, A. Vardasbi, A. Yates, and M. de Rijke. Repetition and exploration in sequential recommendation. In *SIGIR 2023: 46th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2532–2541. ACM, July 2023.

5.1.1 Repetition and exploration

The default focus of sequential recommendation is on increasing accuracy, that is, to find relevant or correct next items that meet the preferences of users. In the next basket recommendation (NBR) scenario, Li et al. [69] distinguish between *repetition*, i.e., when the next item the user interacts with is present in the user's historical interaction sequence, and *exploration*, i.e., when the user first interacts with an item they have not previously interacted with. The authors find very large differences in performance when recommending repeat items vs. explore items, with the task of recommending repeat items being far easier and achieving far higher accuracy scores. As repetition and exploration behavior coexist in many sequential recommendation scenarios, such as item repurchase [12, 19], song relistening [3], and POI revisits [23], a natural question to ask is:

> *How do sequential recommendation models perform from the repetition and exploration perspective?*

5.1.2 Sequential recommendation: A user-centered perspective

To address the question highlighted above, we first adopt a user-centered perspective. We select a diverse set of highly-cited sequential recommendation models [49, 58, 87, 92, 96, 117] to examine if a similar imbalance between repetition performance and exploration performance as was found for NBR is also observed from the point of the users who are being served sequential recommendations. We consider the case where each user purchases one item and that item can either be a *repeat item* (an item the user has bought before) or an *explore item* (an item the user has not purchased before).

We find that users who prefer repetition over exploration get noticeably higher recommendation accuracy than users who prefer to explore. We also find that a higher overall accuracy (aggregated over all users) can be achieved by sacrificing the performance for users who prefer to explore.

These findings matter because the average overall accuracy can be achieved by sacrificing the quality of recommendations for a large proportion of users, which challenges the widely adopted usage of average accuracy with significance test in SR research for evaluation.

5.1.3 Sequential recommendation: An item-centered perspective

The user-centered evaluation summarized above only provides a partial perspective on the capabilities of a recommendation algorithm. There is at least one more side to the (sequential) recommendation task: items. *Item exposure* refers to how often an item is recommended by a recommendation algorithm. Item exposure can have a significant impact on the user experience and the overall effectiveness of the system [34, 39].

Previous studies regarding item exposure often focus on fairness [116, 119]. In this chapter, we generalize the highlighted question in two ways: from the

next basket recommendation scenario to the sequential recommendation scenario and also from a user-centered perspective to an item-centered perspective. Specifically, we distinguish between *item repeat exposure* and *item explore exposure*: The former refers to the number of times the item is exposed to repeat users, i.e., users who have purchased it before, whereas the latter refers to the number of times an item gets exposed to new users, i.e., users who have not purchased it before. The motivation for this perspective is that if an item has been purchased by a large proportion of new users in the future, then this item should probably be recommended to many new users.

We first analyze the distribution of items' next target users (in historical interaction logs) and observe that for most items, there exists a large proportion of purchases that are made by their new users. However, our analysis reveals that sequential recommendation models do not provide enough explore exposure to all items. Surprisingly, we find that some items receive zero explore exposure (i.e., these items will only be recommended to repeat users).

These findings matter because many sequential recommendation models suffer from the issue of *zero/less explore exposure*, which can influence long-term performance from the item perspective, i.e., it is unlikely to get such items exposed to new users.

5.1.4 Repetition "shortcuts" and inherent repetitive bias

Our consistent observation in the above analyses suggests that repeat-next users (that is, users who prefer to purchase a repeat item next) may act as a "shortcut" [40] to the optimization goal of sequential recommendation models, leading those model to recommend repetitive items even for explore-next users, i.e., repetitive bias. To investigate the potential impact of this shortcut on explore-next users, we design a counterfactual experiment: we remove all repeat-next users from the dataset and only train models based on explore-next users (that is, users who prefer to purchase a explore item next), so that there will be no shortcut during training and the model is optimized exclusively for the explore-next users, which can be seen as a pure exploration scenario.

We find that removing the shortcuts results in a higher degree of novelty of the recommendation (meaning that less repeat items are recommended to explore-next users). This confirms the existence of shortcuts biases sequential recommendation (SR) models towards recommending repetitive items. Surprisingly, we also find that *sequential recommendation models will still recommend repeat items to users even in datasets with users who will only explore.* This means that SR models often fail to capture the simple characteristics of the pure exploration datasets and have an *inherent repetitive bias issue.*

Our analysis identifies the usage of *shared item embeddings* in the prediction layer as one potential cause of worsening the repetitive bias, as representations of user preferences inferred by the SR model tend to be highly similar to the item embeddings present in the input item sequence. We find that replacing shared item representations with independent item embeddings in the prediction layer alleviates this issue, thereby increasing the novelty of recommendations.

To complete our study and analysis of repetitive bias, we propose a remedy called the (3R) strategy, i.e., **r**emove **r**epeat items **r**ule, that simply removes repeat items from the predicted recommendation results. With this remedy, the accuracy of existing SR models in pure exploration scenario can be improved by a large margin.

Our findings matter because the issue of inherent repetitive bias impacts the performance of SR models in the pure exploration scenario. Future models should be evaluated more rigidly so as to determine where observed improvements come from.

5.1.5 Main contributions

The main contributions of this chapter are:

- We analyze the accuracy of SR models through the lens of repeat and explore items. We confirm that the imbalance in performance and difficulty between the repetition task and exploration task known from other recommendation tasks also exists in the SR scenario. We point out evaluation issues of only using the overall average performance (in terms of accuracy) with a significance test.
- We generalize the perspective on repetition and exploration by adopting both a user-centered and an item-centered perspective. To the best of our knowledge, we are the first to propose *item explore exposure* and *item repeat exposure* to analyze the exposure allocation at a more fine-grained level.
- We demonstrate the importance of considering item explore exposure and show that several state-of-the-art SR models suffer from the problem of zero/few item explore exposure.
- We analyze the outcomes of our study by uncovering two key phenomena: (i) the impact of repetition "shortcuts": SR models may skew the recommendation towards repeat items by exploiting shortcuts, which leads to a repetitive bias for explore-next users, and (ii) inherent repetitive bias. We investigate the differences between using shared item representations and independent item representations in the prediction layer and propose a remedy to eliminate the repetitive bias issue.

5.2 Related Work

In this section, we describe several empirical research lines in recommender systems that serve as the background of this work.

5.2.1 Sequential recommendation

Sequential item recommendation has been extensively studied. Several models employing deep learning techniques have been proposed [48, 49, 58, 65,

73, 92, 96, 117], such as RNN [48, 49], CNN [96], GNN [89, 117], contrastive learning [120], attention [65, 73], and self-attention [58, 92, 101, 126]. SASRec [58] was the first sequential recommendation model that employed a self-attention mechanism [101]. BERT4Rec [92] later upgraded the left-to-right training scheme in SASRec by using a bi-directional transformer with a Cloze task [98]. In addition, flexible orders [83], capturing repetition and exploration [87], and a consistent representation space [50] have all been found to improve the accuracy of the sequential recommendation.

5.2.2 Accuracy

There are several reproducibility and empirical studies focusing on accuracy-related metrics for recommender systems. Jannach and Ludewig [54] compare the performance of neural-based sequential models with nearest neighbor-based models; Petrov and Macdonald [82] evaluate the performance of BERT4Rec with different versions of implementations; Fang et al. [37] investigate several factors that influence the GRU4Rec performance; and, Zhao et al. [133] investigate the influence of different dataset splitting methods. However, these accuracy-oriented studies have primarily focused on the average performance. They do not provide a detailed assessment of performance for different user groups. More recently, Li et al. [69] have introduced a new evaluation perspective on the NBR task by differentiating between repetition (recommending items that users have purchased before) and exploration (recommending items that are new to the user) tasks in NBR. The authors highlight the difficulty of striking a balance between the two tasks. They analyze existing methods in NBR and conclude that the performance of many existing methods is mainly due to a (strong) bias towards the repetition task, at the expense of their ability to explore. Building on this study, the NBR models proposed in [4, 59] are designed to exploit those insights and improve their effectiveness.

However, the performance of sequential recommendation models w.r.t. repetition and exploration is still unexplored. This chapter fills this gap by analyzing the impact of repetition and exploration in a sequential recommendation scenario.

5.2.3 Beyond accuracy

Apart from accuracy, diversity is another aspect to satisfy users' diversified demand [20, 85, 111, 130]. Recently, similar empirical and revisit studies [75, 124] have been performed to investigate the trade-off between accuracy and diversity. The notion of item exposure is used to measure item-side performance. It has become an important factor that models need to consider, as items and producers are important participants within a recommender system and the ecosystem in which it is deployed. Existing research w.r.t. item exposure is mostly focused on individual or group fairness, either on the customer-side, i.e., adopting a user-centered perspective [15], or on the provider-side, i.e., adopting an item-centered perspective [78, 129], or two-sided [79, 116, 119].

Instead of analyzing the general exposure an item or group gets, as most prior work does, we are specifically interested in how sequential recommendation models allocate exposure in relation to repetition and exploration behavior.

5.3 Problem Formulation and Definitions

5.3.1 Sequential recommendation task

We use $\mathcal{I}$ and $\mathcal{U}$ for the sets of all items and users, respectively. Given a user $u \in \mathcal{U}$ and her historical item sequence $I_u = [i_1, i_2, \ldots, i_t]$, where i_t denotes the item that the user interacted with at timestamp t, the sequential recommendation model $\mathcal{M}_{sq}$ infers the user's preferences from the historical sequence I_u and predicts the next items as recommendation results p_u^{t+1} at timestamp $t+1$. Formally:

$$p_u^{t+1} = \mathcal{M}_{sq}(I_u), \tag{5.1}$$

where p_u^{t+1} is a score distribution over the items. Usually, p_u^{t+1} is used to extract a ranked list of k items as the most probable items that u may interact with at timestamp $t+1$. Similarly to I_u, we use U_i to denote the sequence of users who have interacted with a given item $i \in \mathcal{I}$. When no confusion is possible, we use the same notations I_u and U_i for the set (instead of sequence) of historical items and users, respectively. In those cases, we define $\bar{I}_u = \mathcal{I} \setminus I_u$ and $\bar{U}_i = \mathcal{U} \setminus U_i$.

5.3.2 Repeat vs. explore

Using the historical interaction sequences I_u and U_i, for each user u and item i, we can divide both users and items as follows:

User-centered perspective. For a given user u, the items can be divided into *repeat items* I_u and *explore items* $\bar{I}_u$ (i.e., items that user u has not interacted with before).

Item-centered perspective. For a given item i, the users can be divided into *repeat users* U_i and *explore users* $\bar{U}_i$ (i.e., users who have not interacted with item i before).

Repeat-next user vs. explore-next user. Given the item the users will purchase next, the users can be divided into *repeat-next users* U_* (i.e., users who will purchase a repeat item in the next step) and *explore-next users* $\bar{U}_*$ (i.e., users who will purchase an explore item in the next step).

5.3.3 Explore exposure vs. repeat exposure

Item exposure. Conventional item exposure measures the number of times an item is recommended to users or the chance of an item being examined by the user. Besides, the position of an item in a recommendation list can influence its exposure, e.g., items at the top of the list are likely to receive more exposure than items at the bottom of the list. Usually, a click model C, which measures

the likelihood that the user will examine the item in each position, is used in computing the item exposure, that is:

$$E_i@K = \sum_{u \in U_i} \mathcal{C}_K(r_{u,i}), \tag{5.2}$$

where $r_{u,i}$ is the position of item i in the recommendation list shown to user u. In this study, we use the exposure model from the discounted cumulative gain (DCG) formula, i.e., $\mathcal{C}_K(r_{u,i}) = (\mathbb{I}(r_{u,i} \leqslant K))/(\log_2(r_{u,i}+1))$.

The conventional definition of item exposure provided above does not account for the allocation of exposure to different types of users. From the repetition and exploration perspective, it is possible to evaluate the exposure allocation of the item to different types of users, i.e., *repeat users* and *explore users*. Formally, we propose item explore exposure and item repeat exposure as follows:

Item repeat exposure refers to the accumulated exposure that item i get from its repeat users U_i, that is:

$$RE_i@K = \sum_{u \in U_i} \mathcal{C}_K(r_{u,i}). \tag{5.3}$$

Item explore exposure refers to the accumulated exposure that item i gets from its explore users $\bar{U}_i$, that is:

$$EE_i@K = \sum_{u \in \bar{U}_i} \mathcal{C}_K(r_{u,i}). \tag{5.4}$$

Using the above two definitions, we define the Explore exposure ratio as the proportion of a given item's explore exposure from the total exposure it gets in the recommender system, that is:

$$EEr_i@K = \frac{EE_i@K}{E_i@K}. \tag{5.5}$$

This metric provides an individual-level assessment of exposure allocation. In the extreme case where $EEr_i@K = 0$, item i is only recommended to users who have purchased it before and will not be recommended to explore users.

5.4 Experimental Setup

5.4.1 Research questions

In this study, we decompose the thesis-level research question **RQ4** into the following questions:

RQ4.1 How do the SR models perform w.r.t. repetition and exploration? Does the imbalance between repetition and exploration reported in prior work on NBR also exist in SR scenarios?

RQ4.2 Should we consider item explore exposure in the SR? How do the sequential recommendation models perform w.r.t. item explore exposure and item repeat exposure?

RQ4.3 Does the repetition "shortcut" impose the SR models to recommend repeat items for explore-next users?

RQ4.4 Does the repetitive bias of the sequential recommendation model still exist in a pure exploration scenario?

RQ4.5 How can we avoid the potential effect of this repetitive bias?

5.4.2 Datasets

As our goal is to investigate the performance from the repetition and exploration perspective, we select two widely used sequential datasets with both repetition and exploration behavior:

- **Diginetica** is a widely used dataset released in CIKM2016 Challenge, which includes user e-commerce search sessions with unique ids.[1]

- **Yoochoose** is a widely used dataset released in the RecSys2015 Challenge, which contains a collection of sessions from a retailer, and each session in the dataset represents a series of click events performed by a user during the session.[2]

We follow the widely used preprocessing procedure in previous works, i.e., "5-core". Specifically, we remove items that are purchased/viewed less than 5 times and remove users whose interaction sequence length is less than 5. We set the maximum length of a sequence to 50 and any sequences longer than 50 are truncated. We split each dataset into train, validation, and test partitions using a leave-one-out strategy: for each item sequence, we hold the final interaction for the test set, the second last interaction for the validation set, and the third last interaction for the train set. The statistics of the pre-processed datasets are shown in Table 5.1.

Table 5.1: Statistics of the processed datasets. * RNU denotes repeat-next users; † ENU denotes explore-next users.

Dataset	#items	#users	# RNU*	# ENU†	ENU proportion
Diginetica	35,042	75,739	22,610	53,129	61.9%
Yoochoose	30,833	1,878,967	715,518	1,163,449	70.2%

[1] https://competitions.codalab.org/competitions/11161
[2] https://www.kaggle.com/datasets/chadgostopp/recsys-challenge-2015

5.4.3 Methods

Methods selection. The purpose of this study is to provide insights w.r.t. performance evaluation and model design from a novel angle, rather than to track and confirm the best or latest sequential recommendation model. Thus, we consider the following aspects to select the methods we want to analyze:

- **Influential**: the selected method should be highly-cited and influential, which continue serving as competitive baselines in sequential recommendation research.
- **Representative**: the selected methods should have diverse representation techniques, which continue serving as the backbone of various sequential recommendation models.
- **Consistency**: the selected methods should follow the same paradigm of modeling, which only takes users' historical item sequence as input to generate the users' preference representation.[3]

Methods. Following the criteria listed above, we select several highly-cited methods with representative techniques (i.e., RNN, CNN, GNN, transformers, BERT) as follows:

- **GRU4Rec** is a representative method that uses a recurrent neural network (i.e. a GRU) to model users' sequential behavior [49].
- **Caser** is a representative method that uses a CNN to model users' sequential behavior [96].
- **SRGNN** is a representative method that uses a graph neural network (GNN) to model user historical sequence [117].
- **SASRec** is a representative method that employs a left-to-right Transformer model to capture users' sequential behavior [58].
- **BERT4Rec** is a representative method that employs a bi-directional transformer model and introduces the Cloze task to train the model [62].
- **RepeatNet** is a representative method that models the users' preference w.r.t. repetition and exploration, and uses separate decoders for repeat item and explore item prediction [87].

5.4.4 Configurations

For the neural-based sequential recommendation methods listed above, we use the implementations in the Recbole open-source project and then integrate them into our pipeline. We follow the hyper-parameter settings suggested in Recbole.

[3]Note that we do not include sequential recommendation methods with a user embedding or any additional information as input to maintain fairness and consistency.

The embedding size is tuned on $\{32, 64, 128\}$ for all methods based on the validation set to achieve their best performance. For BERT4Rec and SASRec, we use two stacked transformer layers with 8 heads. For all methods, the dropout ratio is set to 0.1 and the Adam optimizer is employed with a learning rate of 0.001.

All the training is performed using TITAN X Pascal GPUs with 12G memory. We repeat our experiments 5 times and report the average performance. We share both our dataset processing scripts, the source code, and the hyperparameters we use in an anonymous repository.[4]

5.4.5 Metrics

We use three widely used metrics for the sequential recommendation problem, i.e., Recall@K, MRR@K, and NDCG@K, to measure accuracy. In the sequential recommendation task, *Recall* measures the ability to find a relevant item that meets the user's preference; $NDCG$ and MRR are metrics that also consider the order of the relevant items. For these three accuracy-oriented metrics, the higher the value, the better the performance.

We also use Novelty$_u$@K to measure the novelty of the recommendation, that is:

$$\text{Novelty}_u@K = \frac{\sum_{r=1}^{K} h(u,r) \cdot \log_2(r+1)}{\sum_{r=1}^{K} \log_2(r+1)} \tag{5.6}$$

where $h(u,r) = 1$ if the r^{th} item in the recommended list to user u is a explore item, otherwise $h(u,r) = 0$. Explore-next users prefer higher novelty, while repeat-next users prefer lower novelty. We will later describe our proposed metrics in later sections to remain focused. In this chapter, we consider the metrics with $K \in \{1, 3\}$, as a higher K will lead to a passive increase w.r.t. the novelty and the item explore exposure we will discuss below.[5]

5.5 Repetition Accuracy and Exploration Accuracy

Evaluation. To answer RQ4.1, we aim to gain an understanding of accuracy from the repetition and exploration angle and find potential issues w.r.t. only using average overall accuracy. Specifically, apart from the average overall accuracy, we also examine a more fine-grained level, analyzing the accuracy (Recall, NDCG, MRR) and novelty (Novelty) w.r.t. two user groups, i.e., repeat-next users U_* and explore-next users $\bar{U}_*$.

Results. The experimental results w.r.t. different accuracy metrics are shown in Figure 5.1 and Table 5.2. We have the following observations: (i) there is a large imbalance in recommendation accuracy between repeat-next users and

[4] `https://github.com/liming-7/Repetition-exploration-SR`

[5] For example, if the length of a user's historical sequence is 5, there are at most 5 different items that could be regarded as repeat items, so the recommendation list with a size of 10 will always contain at least 5 explore items.

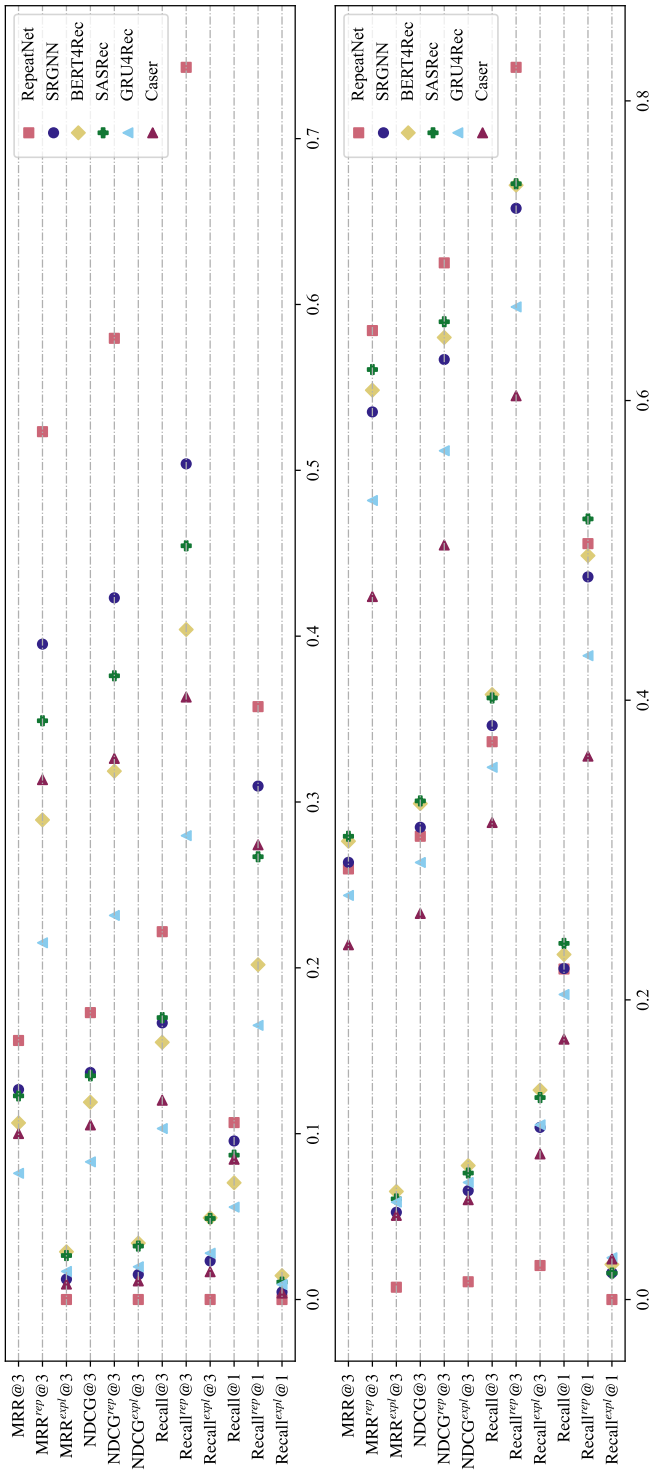


Figure 5.1: The recommendation accuracy for all users, repeat-next users and explore-next users. Right: Diginetica; left: Yoochoose.

explore-next users, where all models achieve noticeably higher recommendation accuracy (across different metrics) w.r.t. repeat-next users than explore-next users; (ii) compared to explore-next users, repeat-next users account for a relatively small proportion of the user population, whereas they contribute to a large proportion of the average performance; (iii) the absolute difference in recommendation accuracy between different methods w.r.t. explore-next users is smaller than the difference w.r.t. repeat-next users; and (iv) a higher average overall accuracy does not necessarily link to the improvement w.r.t. the recommendation accuracy across all users, e.g., RepeatNet achieves the best overall accuracy in most cases on Diginetica, whereas it has the lowest accuracy on both datasets w.r.t. explore-next users.

Table 5.2: The contribution of repeat-next users to the average overall performance w.r.t. Recall@1.

Dataset	GRU4Rec	Caser	SRGNN	BERT4Rec	SASRec	RepeatNet
Diginetica	71.5%	81.1%	88.0%	78.5%	86.5%	100%
Yoochoose	90.5%	89.2%	94.3%	92.9%	94.8%	100%

The above results answer RQ1 and confirm that the findings w.r.t. the imbalance between repetition and exploration in NBR setting generalize to the sequential item recommendation scenario.

In general, the expected novelty for repeat-next users is 0, meaning that only repeat items are recommended, while the expected novelty for explore-next users is 1, indicating that only explore items are recommended. The experimental results w.r.t. the novelty over different types of users are shown in Table 5.3. We have the following observations: (i) different methods exhibit diverse performance in terms of the novelty of the recommendation; for instance, GRU4Rec has relatively high novelty, while RepeatNet and SRGNN have much lower novelty compared to GRU4Rec; and (ii) the novelty of recommendations to explore-next users is slightly higher than for repeat-next users in most cases, indicating that these sequential recommendation models have the ability to identify user preferences towards repetition and exploration to some degree.

Sacrificing the performance for specific users. When there is a huge imbalance between the recommendation performance w.r.t. different groups of user, using the average overall performance to represent the performance of a method has a risk of hiding and sacrificing the performance for users for whom the recommendation task relatively (more) difficult (e.g., users who prefer to explore, in this chapter). For instance, compared to BERT4Rec and SASRec, RepeatNet and SRGNN can achieve higher overall performance by sacrificing the performance for a large proportion of users who prefer to explore on the Diginetica dataset.

Significance testing. In recommender system research, we often use a significance test when comparing the performance of models. If the p-value is less than a pre-determined level of significance (usually 0.05 or 0.01), we usually

Table 5.3: The novelty of the recommendation for repeat-next users and explore-next users.

	Diginetica				Yoochoose			
	Novelty@1		Novelty@3		Novelty@1		Novelty@3	
Method	RNU	ENU	RNU	ENU	RNU	ENU	RNU	ENU
GRU4Rec	0.567	0.675	0.725	0.779	0.223	0.362	0.488	0.577
Caser	0.237	0.304	0.634	0.660	0.265	0.436	0.496	0.597
SRGNN	0.145	0.208	0.447	0.479	0.099	0.172	0.408	0.476
SASRec	0.231	0.285	0.511	0.556	0.085	0.134	0.423	0.492
BERT4Rec	0.385	0.483	0.530	0.599	0.110	0.183	0.409	0.479
RepeatNet	0	0	0.010	0.032	0	0	0.108	0.150

claim something like "The proposed model A significantly outperforms baseline B". Based on the findings above, we want to caution against over-reliance on the successful outcomes of a significance test in the context of SR. Our concern stems from our experimental results that show that RepeatNet achieves higher accuracy scores than SASRec, with a paired significance test p-value below 0.05. However, SASRec performs better for users who prefer to explore, who account for over 60% of the users.

Lessons. The findings concerning reproducibility that we have listed above, confirm that the analysis of repetition and exploration is also important, but neglected, in SR scenarios, just as in NBR scenarios, which motivates us to perform a deeper analysis of SR models from several angles w.r.t. repetition and exploration.

Furthermore, upon drilling down, we have found that, when comparing sequential recommendation models on a dataset containing users who prefer to repeat, we should be aware that the widely-used average overall accuracy with a significance test may not fully represent the models' recommendation accuracy for all user groups. Instead, we should also: (i) evaluate the accuracy for repeat-next users and explore-next users separately and perform separate significance tests on these two averages; and (ii) check the actual accuracy distribution to know whether the method favors a specific user group over another.

5.6 Explore Exposure and Repeat Exposure

To answer RQ4.2, we first illustrate the importance of analyzing item explore exposure and then perform an analysis of this property.

5.6.1 Importance of item explore exposure

In order to demonstrate the importance of analyzing item explore exposure and answer RQ2, we perform the following two analyses:

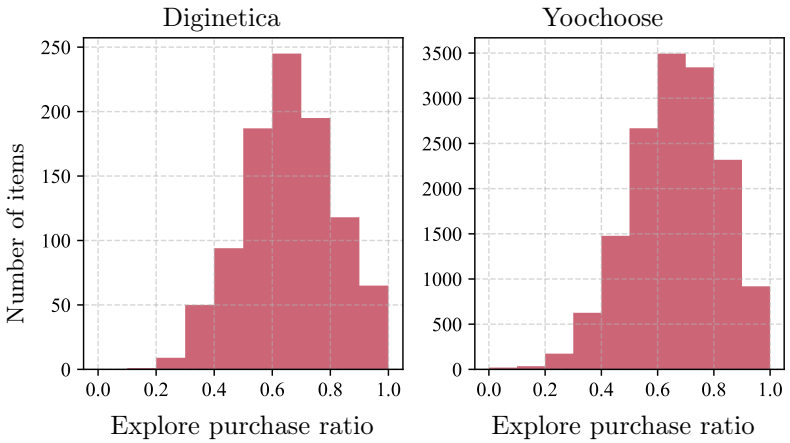


Figure 5.2: Distribution of items across different exploratory purchase ratios.

Exploratory purchase. The number of exploratory purchases for an item reflects the expected number of times that the item will be purchased by a new user u_i^{novel} in the future. A large number of exploratory purchases of an item indicates that it should be recommended to a large number of explore users, i.e., require explore exposure, otherwise, it will lose a large potential user purchase and never be known by these potential explore users. We rank the items based on their total purchases and find a substantial number of exploratory purchases across different items in both datasets, as shown in Figure 5.3.

Exploratory purchase ratio. The exploratory purchase ratio (EPr) of an item refers to the proportion of exploratory purchases within all future purchases of this item. For statistical analysis of the exploratory purchase ratio, we ignore items with less than 10 future purchases for the sake of confidence. From Figure 5.2, we observe the distribution is right-skewed, i.e., the EPr is more spread out towards a higher value, which indicates a large proportion of the future purchases are made by explore users. An item with a high EPr should be recommended more to potential explore users than repeat users (a.k.a the item should get more explore exposure than repeat exposure). An extreme case with $EPr_i = 1$ indicates that all future purchases will be made by explore users for the item i, so it is meaningless for giving repeat exposure to this item.

5.6.2 Less/zero explore exposure issue

Evaluation. The exploratory purchase ratio (EPr) provides an expected exposure distribution between explore exposure and repeat exposure, which can be seen as the expected explore exposure ratio. Therefore, we can evaluate whether the SR models provide items with enough explore exposure compared to what is expected by computing the difference between the exploratory purchase ratio

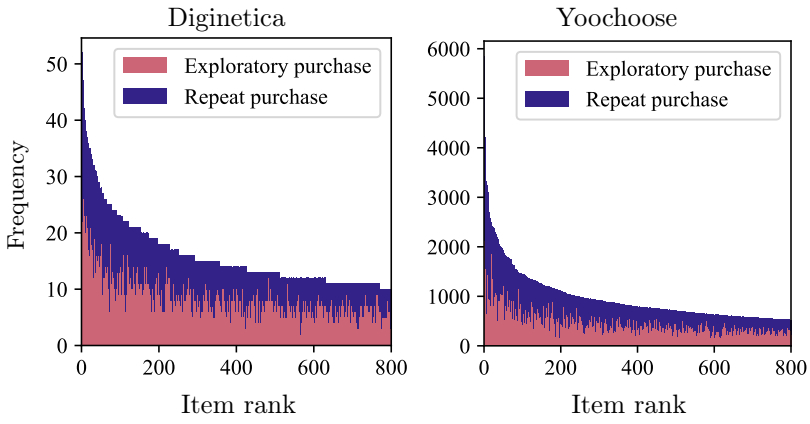


Figure 5.3: Distribution of purchases across different items.

(EPr) and the explore exposure ratio (EEr), that is:

$$\Delta_i^{\mathrm{E}}@K = EPr_i@K - EEr_i@K.$$

Specifically, we first rank items according to their future total purchases and then calculate the average Δ_i^{E} of items with top-Q total purchases and non-zero total exposure for the following reasons: (i) items in the catalog are not equally important, and fewer total purchases of item indicate that only a small number of users will prefer this item, and (ii) we focus on the item exposure distribution, and analyzing the repeat exposure and explore exposure distribution of items with zero exposure is meaningless.

Table 5.4: Proportion of items with zero explore exposure; analyzed on top-500 popular next items with $K = 1$.

Dataset	GRU4Rec	Caser	SRGNN	BERT4Rec	SASRec	RepeatNet
Diginetica	11.8%	16.2%	38.2%	19.4%	35.7%	100%
Yoochoose	6.7%	2.8%	16.7%	18.7%	33.9%	99.7%

Findings. To answer RQ2, we re-evaluate the performance of SR models w.r.t. their exposure allocation to items. The results of our analysis are shown in Figure 5.4. For all methods on both datasets, we observe that the average Δ^{E} is negative, which indicates that items tend to receive less explore exposure than expected, and their exposure is biased towards repeat exposure.

Note that RepeatNet has the lowest Δ^{E} score, which can be seen as the lower-bound of Δ^{E}.[6] Moreover, SRGNN and SASRec are very close to this

[6]Even equipped with a module to identify whether a user prefers to explore or repeat, RepeatNet can only recommend repeat items to the user (Novelty = 0), which also means the item will only be recommended to users who purchased them before.

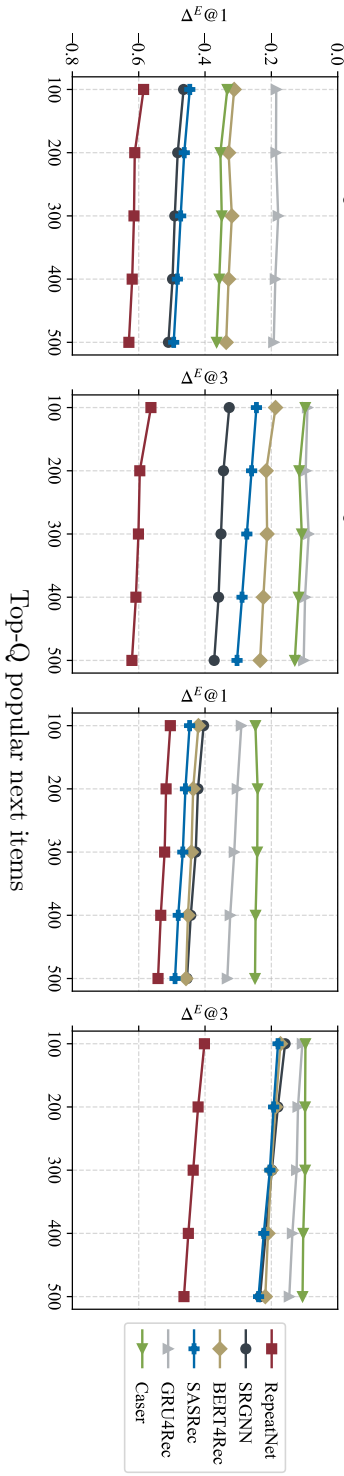


Figure 5.4: Difference between expected EEr and actual EEr of the over different item groups.

lower-bound w.r.t. $\Delta^{\mathrm{E}}@1$ on both datasets. To further investigate the potential issues w.r.t. SR models, we analyze the proportion of items that will only be recommended to repeat users (i.e., zero explore exposure) in the sequential next-item recommendation scenario (i.e. $K = 1$). The results are shown in Table 5.4. Surprisingly, we find that a non-negligible proportion of items suffer from the zero explore exposure issue, which is a severe problem as these items may never be discovered or seen by their potential new users.

Lessons. According to the findings in this section, we should be aware that items need explore exposure and that SR models suffer from a less/zero explore exposure issue w.r.t. a neglected proportion of items. Instead of only analyzing the overall item exposure from an item-centered perspective, we should also analyze the exposure distribution w.r.t. repetition and exploration.

5.7 Repetitive Bias

Pure exploration. An important task of recommender systems is to connect users with items that they have never seen; there is a large proportion of explore-next users, who would like to explore items. From the analysis in Section 5.5, the imbalance in difficulty between the repetition task and the exploration task suggests that the existence of repeat-next users is a "shortcut" to the optimization goal of accuracy-oriented SR models, leading those models to recommend repetitive items even for explore-next users.

Specifically, we remove all repeat samples (i.e., repeat-next users) from the train, validation, and test set to ensure there will be no shortcut during training and optimization, so that the model will be specifically trained and optimized for explore-next users. Note that this constructed subset can be regarded as a pure exploration scenario, as all the training, validation, and test ground-truth labels in this constructed subset are explore items.

Influence of removing repetition shortcuts. From Figure 5.5, we observe that removing shortcuts leads to a higher novelty for all methods on both datasets. This illustrates that the presence of repeat-next users makes the model more likely to recommend repeat items even for explore-next users, this answers RQ4.3.

A counterintuitive finding. In this pure exploration scenario, we surprisingly find that *some sequential recommendation models will still recommend repeat items to users even in datasets with pure exploration.* This finding is counterintuitive since humans can easily notice the basic characteristics of this scenario, i.e., there are no repeat items in the ground-truth labels in the training, validation, and test set. Whereas, many complex models fail to detect this simple pattern of the dataset and have an inherent repetitive bias issue, which is a serious pitfall that results in poor user experience.[7]

[7] RepeatNet can avoid the inherent repetitive issue since it has an indicator to identify repetition and exploration. BERT4Rec employs a self-supervised training objective, so it is not that surprising to have repetitive bias.

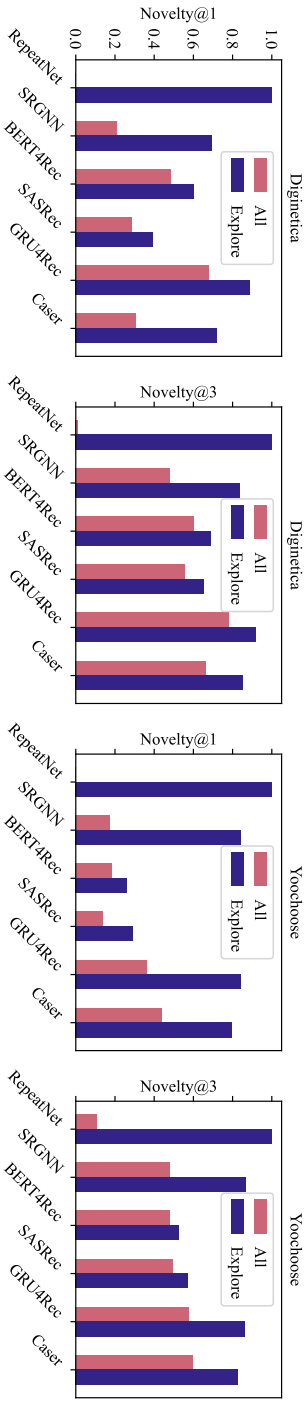


Figure 5.5: The recommendation novelty for explore-preferred users: train using all vs. using pure exploration.

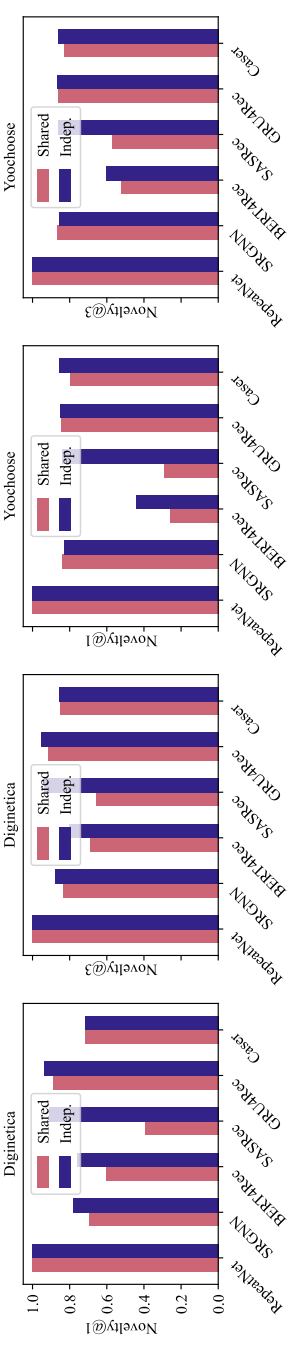


Figure 5.6: The recommendation novelty w.r.t. explore-next users: shared embedding vs. independent embedding.

Table 5.5: The recommendation accuracy w.r.t. explore-next users of SR models with 3R strategy. S and I denote using shared and independent embeddings, respectively. We exclude RepeatNet here since it does not have repetitive bias.

		Diginetica-Expl.		Yoochoose-Expl.	
Method	Mode	Recall@1	NDCG@3	Recall@1	NDCG@3
SRGNN	S	0.0107	0.0197	0.0716	0.1155
	S+3R	0.0135	0.0232	0.0823	0.1302
	I	0.0104	0.0180	0.0640	0.1061
	I+3R	0.0125	0.0203	0.0754	0.1212
BERT4Rec	S	0.0140	0.0309	0.0270	0.0886
	S+3R	0.0259	0.0493	0.1102	0.1767
	I	0.0158	0.0323	0.0399	0.0942
	I+3R	0.0230	0.0431	0.1021	0.1643
SASRec	S	0.0119	0.0323	0.0330	0.0947
	S+3R	0.0300	0.0520	0.1035	0.1681
	I	0.0234	0.0377	0.0824	0.1356
	I+3R	0.0256	0.0403	0.0991	0.1562
GRU4Rec	S	0.0091	0.0161	0.0614	0.1021
	S+3R	0.0106	0.0183	0.0725	0.1172
	I	0.0066	0.0118	0.0577	0.0961
	I+3R	0.0073	0.0130	0.0699	0.1123
Caser	S	0.0053	0.0108	0.0415	0.0740
	S+3R	0.0072	0.0131	0.0595	0.0980
	I	0.0042	0.0087	0.0480	0.0817
	I+3R	0.0058	0.0107	0.0548	0.0935

Shared embeddings vs. independent embeddings. We suspect that the user's preference representation inferred by the SR model will be similar to the item embeddings within the model's input sequence. Therefore, repeat items will be ranked high in the recommendation list when using the dot product between user representation and item embedding as the prediction layer. The prediction layer typically shares item embeddings with the input layer, which reduces model size and may reduce overfitting [58, 92, 117]. To understand whether using shared embeddings in the prediction layer will exacerbate bias towards repetitive items, we replace the shared item representation with an independent item representation to conduct another group of experiments.

From the results in Figure 5.6, we find that replacing shared item embeddings with independent item embeddings in the prediction layer can indeed alleviate this repetitive issue and increase the novelty of recommendations for explore-

next users. The results confirm that using shared embeddings in the prediction layer contributes to the issue of repetitive bias of SR models, which answers RQ4.4.

A simple remedy: the 3R strategy. However, we can also see that using independent embeddings does not entirely address the repetitive bias issue of SR models. A straightforward method to eliminate the repetitive bias in the pure exploration scenario is to post-process recommendation results according to the scenario's characteristics. We propose a remedy called the 3R strategy, i.e., **r**emove **r**epeat item from **r**ule, which simply removes the repeat items in the recommendation in the pure exploration scenario. From the experiment results in Table 5.5, we observe that: (i) simply adopting the 3R strategy can easily bring substantial improvements across all methods, and (ii) shared embeddings with the 3R strategy outperforms independent embeddings with the 3R strategy in terms of recommendation accuracy for all methods on both datasets. 3R strategy is the answer to RQ4.5 in the pure exploration scenario.

Lessons. According to the analysis above, when comparing SR models in a pure exploration scenario, we should be aware that: (i) many complex SR models have the inherent repetitive bias issue, which negatively impacts their performance in a pure exploration SR scenario, (ii) the SR models may perform differently when the input item embeddings are not shared with the prediction layer, and using shared embedding may exacerbate the inherent repetitive bias issue, and (iii) a higher recommendation accuracy can be easily achieved by addressing the repetitive bias using the 3R strategy to post-process recommendations for a pure exploration scenario.

Additionally, it is important for future models to be rigorously evaluated to check where improvements truly come from.

5.8 Conclusion

In this chapter, we have investigated and revisited sequential recommendation from the repetition and exploration perspective to answer the thesis-level research question **RQ4**:

> *How do sequential recommendation models perform, and how should we evaluate item exposure from the perspective of repetition and exploration?*

Taking lessons learned in a NBR scenario as our starting point, we analyzed several representative SR models in multiple ways: (i) from a user-centered perspective, where we analyze the accuracy and novelty w.r.t. repeat-next users and explore-next users, and (ii) from an item-centered perspective, where we define explore exposure and repeat exposure to measure exposure allocation. We have also investigated the repetitive bias of SR models w.r.t. the recommendation for explore-next users from the following aspects: (i) the repetition "shortcuts," and (ii) shared embedding and independent embedding.

5.8.1 Main findings

We arrive at several important findings and discover some issues w.r.t. SR models: (i) as in NBR, in SR too there is a huge imbalance between repetition and exploration, and SR models perform much better for repeat-next users than explore-next users; (ii) a higher average performance can be achieved by sacrificing the performance for a large proportion of users, which indicates that our widely used evaluation strategy, i.e., "overall performance with significance test", hides important details about the effectiveness of SR models; (iii) many SR models suffer from a less/zero explore exposure issue, i.e., items are mostly (or even only) recommended to their repeat users; (iv) the existence of repetition "shortcuts" increases the repetitive bias w.r.t. the recommendation for explore-next users; (v) many SR models suffer from an inherent repetitive bias (i.e., they still recommend repeat items even in the pure exploration scenario), and using shared embeddings will exacerbate this inherent repetitive bias; and (vi) a simple strategy for post-processing the recommendations of SR models may lead to substantial improvements in a pure-exploration scenario.

5.8.2 Implications

Our work highlights the following important lessons that practitioners and researchers should follow: (i) in a SR scenario with both repetition and exploration, instead of only relying on the average overall accuracy with a significance test, we should also evaluate the performance of repeat-next users and explore-next users separately, and check the distribution of performance results across users; (ii) in a pure exploration SR scenario, we should be aware of the inherent repetitive bias issue, and use the 3R strategy to post-process SR models when using them as baselines; and (iii) on a two-sided platform, SR practitioners should also check the explore exposure and the exposure allocation of items to ensure that items will not only get exposed to their repeat users and have explore exposure to reach potential exploring consumers.

Our analyses show that using the repetition "shortcut" in SR scenarios with repetition behavior and addressing the repetitive bias in SR scenarios with pure exploration may lead to substantial improvements w.r.t. recommendation accuracy of SR models. Given that many recent SR models were evaluated without separately considering repetition and exploration performance, it is unclear whether the improvements observed come from improving the model overall or from leveraging shortcuts that improve repetition at the expense of exploration. For evaluations conducted in an exploration scenario, it is unclear which improvements would remain after mitigating the models' repetitive bias with the 3R strategy.

5.8.3 Limitations and future work

Our analyses mainly focus on the neural-based SR models that have been published in recent years, ignoring classic machine learning-based and neighbor-based methods. Another limitation is that we only focus on repetition and

exploration, but there might be other factors that also lead to a performance imbalance in the SR scenario. We focus on analyzing the exposure distribution of items and uncover the limited item explore exposure issue; it would be interesting to consider how to avoid this issue.

Next, we will conclude the thesis.

6 Conclusions

In this thesis, we have studied recommendation tasks from a repetition and exploration perspective. Specifically, we have looked into the evaluation and optimization of recommendation models from both user-centered and item-centered perspectives. In this chapter, we first summarize the main findings in terms of the thesis-level research questions listed in Chapter 1, and then outline potential future directions for follow-up research.

6.1 Main Findings

RQ1 *How to evaluate the next basket recommendation performance from the perspective of repetition and exploration?*

This question has been answered in Chapter 2, where we reproduce and investigate various types of next basket recommendation (NBR) models. Specifically, we propose a set of metrics that measure the repetition/exploration ratio and performance of NBR models. Using these new metrics, we provide a second analysis of state-of-the-art NBR models from several perspectives: (i) the overall performance on different scenarios; (ii) the basket components; (iii) the repeat and explore performance; (iv) the contribution of repetition and exploration to the overall performance; (v) the treatment effect for different user groups; (vi) the potential limitations of the average metrics; and (vii) the treatment effect for different items.

Through our experiments, we arrived at several important findings: (i) No state-of-the-art NBR method shows the best performance across datasets with various levels of repetition behavior. (ii) There is a clear distinction in difficulty and trade-off between the repetition task and the exploration task within NBR. Recommending repetitions is significantly easier than recommending exploration. (iii) Being biased towards the easier repetition task is an important strategy that helps to boost the overall NBR performance. (iv) Some NBR methods might achieve better overall performance at the detriment of users who would like to explore new items. (v) Deep learning-based NBR methods do not effectively utilize the distinction between users' repetition behavior and exploration behavior.

These results help us gain a clearer understanding of the progress made by existing NBR methods and the reasons behind any observed improvements. In summary, this thesis sheds light on the evaluation challenges of NBR, introduces a new evaluation protocol, and offers valuable insights for designing models for this task.

RQ2 *How to design basket recommendation models targeted at the exploration task, and how to optimize the model to explore items in a scenario with many repetition signals?*

This question has been answered in Chapter 3, where we formulate and investigate the next novel basket recommendation (NNBR), i.e., an exploration recommendation task. We evaluate how existing NBR methods perform on the NNBR task and find that, so far, limited progress has been made w.r.t. the NNBR task. To address the NNBR task, we propose a simple **b**i-directional **t**ransformer **b**asket **r**ecommendation model (BTBR), which is focused on directly modeling item-to-item correlations within and across baskets instead of learning complex basket representations. To properly train BTBR in a scenario with both repetition and exploration signals, we propose and investigate several masking strategies and training objectives: (i) item-level random masking, (ii) item-level select masking, (iii) basket-level all masking, (iv) basket-level explore masking, and (v) joint masking. In addition, an item-basket swapping strategy is proposed to enrich the item interactions within the same baskets.

We conduct extensive experiments on three open datasets with various characteristics. The results demonstrate the effectiveness of BTBR and our masking and swapping strategies for the NNBR task. BTBR with a properly selected masking and swapping strategy can substantially improve the NNBR performance.

RQ3 *How to help a given item find its potential users in an item-centered setting, and how do repetition and exploration influence the design and optimization of the recommendation model?*

This question has been answered in Chapter 4, where we formulate and investigate the reverse next-period recommendation (RNPR), i.e., an item-centered recommendation task. Considering the repetition and exploration, we formulate three sub-tasks of the overall RNPR task, (i) Expl-RNPR, (ii) Rep-RNPR, and (iii) Mixed-RNPR, where we consider different types of target users, i.e., (i) explore users, who are new to a given item, (ii) repeat users, who previously purchased a given item, and (iii) both explore users and repeat users. To address the Expl-RNPR task, we propose a habit-interest fusion model that employs frequency information to capture the repetition-exploration habits of users and that uses pre-trained item embeddings to model the user's interests. For the Mixed-RNPR task, we propose a repetition-exploration user ranking algorithm to decouple the repetition and exploration task, and investigate the trade-off between targeting different types of users for a given item. Furthermore, to reduce computational costs at inference, we analyze the repetition behavior from

both user and item perspectives and then introduce a repetition-based candidate filtering method for each sub-task. We conduct experiments on two public grocery shopping datasets. Our experimental results not only demonstrate the difference between repetition and exploration, but also the effectiveness of the proposed methods.

RQ4 *How do sequential recommendation models perform, and how should we evaluate item exposure from the perspective of repetition and exploration?*

This question has been answered in Chapter 5, where we reproduce and investigate various types of sequential recommendation (SR) models. Specifically, we examine whether repetition and exploration are still important dimensions in the sequential recommendation scenario. We consider this generalizability question both from a user-centered and an item-centered perspective. For user-centered perspective, we group users into repeat-next users and explore-next users, and then analyze the recommendation performance w.r.t. the two groups separately. Towards the latter, we define *item repeat exposure* and *item explore exposure*, which consider the exposure distribution on different types of users.

We conduct experiments to examine the recommendation performance of sequential recommendation models in terms of both accuracy and exposure from the perspective of repetition and exploration. We find that (i) there is an imbalance in accuracy and difficulty w.r.t. repetition and exploration in SR scenarios, (ii) using the conventional average overall accuracy with a significance test does not fully represent a model's recommendation accuracy, (iii) accuracy-oriented sequential recommendation models may suffer from less/zero item explore exposure issue, where items are mostly (or even only) recommended to their repeat users and fail to reach their potential new users, (iv) the existence of repetition "shortcuts" increases the repetitive bias w.r.t. the recommendation for explore-next users; (v) many SR models suffer from an inherent repetitive bias (i.e., they still recommend repeat items even in the pure exploration scenario), and using shared embeddings will exacerbate this inherent repetitive bias; and (vi) a simple strategy for post-processing the recommendations of SR models may lead to substantial improvements in a pure-exploration scenario.

6.2 Future Directions

In this thesis, we have investigated the topics of repetition and exploration in recommendation by answering four main research questions using four chapters. The specific limitations and future directions are discussed at the end of each corresponding research chapter. With many inspiring findings and insights derived from this thesis, we believe there are several future directions for the entire thesis in general.

6.2.1 Other recommendation scenarios

Throughout this thesis, we have observed that the distinction between repetition and exploration exists in several recommendations tasks. Repetition and exploration behaviors co-exist in various scenarios and platforms. However, in a single thesis, we are not able to cover all aspects and scenarios of recommendation research. It is worth extending our analysis methodology w.r.t. repetition and exploration to more recommendation settings. Specifically, we think that the perspective of repetition and exploration would also bring new insights to the following recommendation scenarios and tasks:

- Repetition and exploration in reinforcement learning-based recommendation (RL4Rec). The goal of RL4Rec is to optimize a policy that can receive maximum cumulative reward over multiple sequential recommendations. In this thesis, we have found that optimizing the recommendation model in a scenario with both repetition and exploration behaviors introduces several challenges, e.g., the repetition might be regarded as a "shortcut" during the optimization process. It is important and interesting to investigate the potential influence of repetition and exploration signals when optimizing RL4Rec methods in a such scenario.

- Causal effect of recommendation. The recommended items might have been clicked or purchased even without recommendations, and recommending such items would not increase positive interactions. To increase user engagement, we need to focus on the change caused by the recommendation, which is called the causal effect of recommendation. We expect to see a difference in recommendation effects between repetition recommendations and exploration recommendations.

- Long-term reward. As the repetition task is much easier than the exploration task, simply ignoring the exploration task and being biased towards the easier repetition task is an intuitive way of boosting performance when considering the short-term reward. However, people might get tired of repetition over time. In that sense, the exploration task needs to be taken into account when targeting long-term rewards.

6.2.2 Evaluation

Throughout this thesis, we have found that the average overall performance can only partially reflect performance, but is not able to represent actual performance w.r.t. different user groups. Recommendation models could leverage a shortcut to increase overall performance, which is at the sacrifice of a certain group of users. Recently, more researchers have realized the potential issues of relying on average overall performance to evaluate recommendation performance and argued that the evaluation of recommendation methods should consider the distribution of utility between and within different user groups to more deeply understand recommendation performance [35]. We believe the repetition and

exploration perspective is just one aspect that influences the difficulty of the recommendation task. Beyond this perspective, we believe there are more angles and meta-features worthy of investigating, e.g., differences in the length of the user interaction sequence, different levels of popularity and types of historical items, etc. Besides, it is also worth performing fine-grained level analysis using different metrics (e.g., diversity, fairness, etc) and their trade-offs. Ultimately, we believe it is important to design an evaluation platform and library that can support the analysis of evaluation metrics across user groups, item groups, recommendation tasks, and assumptions.

Bibliography

[1] M. Akcay, I. S. Altingovde, C. Macdonald, and I. Ounis. On the additivity and weak baselines for search result diversification research. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, pages 109–116. ACM, 2017. (Cited on page 14.)

[2] D. Amagata and T. Hara. Reverse maximum inner product search: How to efficiently find users who would like to buy my item? In *Proceedings of the 15th ACM Conference on Recommender Systems*, pages 273–281, 2021. (Cited on pages 70, 72, and 74.)

[3] A. Anderson, R. Kumar, A. Tomkins, and S. Vassilvitskii. The dynamics of repeat consumption. In *Proceedings of the 23rd international conference on World Wide Web*, pages 419–430, 2014. (Cited on pages 1, 23, and 108.)

[4] M. Ariannezhad, S. Jullien, M. Li, M. Fang, S. Schelter, and M. de Rijke. ReCANet: A repeat consumption-aware neural network for next basket recommendation in grocery shopping. In *SIGIR 2022: 45th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1240–1250. ACM, July 2022. (Cited on pages 40, 46, 49, 73, 90, 91, 92, and 111.)

[5] M. Ariannezhad, M. Li, S. Jullien, and M. de Rijke. Tutorial: Complex item set recommendation. In *SIGIR 2023: 46th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3444–3447. ACM, July 2023.

[6] M. Ariannezhad, M. Li, S. Schelter, and M. de Rijke. A personalized neighborhood-based model for within-basket recommendation in grocery shopping. In *WSDM 2023: The Sixteenth International Conference on Web Search and Data Mining*, pages 87–95. ACM, February 2023. (Cited on pages 39 and 77.)

[7] T. G. Armstrong, A. Moffat, W. Webber, and J. Zobel. Improvements that don't add up: Ad-hoc retrieval results since 1998. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 601–610. ACM, 2009. (Cited on page 14.)

[8] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. (Cited on page 52.)

[9] T. Bai, J.-Y. Nie, W. X. Zhao, Y. Zhu, P. Du, and J.-R. Wen. An attribute-aware neural attentive model for next basket recommendation. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1201–1204, 2018. (Cited on pages 11, 14, 15, 17, 45, 49, 57, and 73.)

[10] O. Barkan and N. Koenigstein. Item2vec: Neural item embedding for collaborative filtering. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2016. (Cited on page 79.)

[11] Z. Batmaz, A. Yurekli, A. Bilge, and C. Kaleli. A review on deep learning for recommender systems: Challenges and remedies. *Artificial Intelligence Review*, 52:1–37, 2019. (Cited on page 12.)

[12] A. R. Benson, R. Kumar, and A. Tomkins. Modeling user consumption sequences. In *Proceedings of the 25th International Conference on World Wide Web*, pages 519–529, 2016. (Cited on pages 1 and 108.)

[13] R. Bhagat, S. Muralidharan, A. Lobzhanidze, and S. Vishwanath. Buy it again: Modeling repeat purchase recommendations. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 62–70, 2018. (Cited on page 69.)

[14] A. Biswas, G. K. Patro, N. Ganguly, K. P. Gummadi, and A. Chakraborty. Toward fair recommendation in two-sided platforms. *ACM Transactions on the Web (TWEB)*, 16 (2):1–34, 2021. (Cited on page 34.)

[15] J. Bobadilla, R. Lara-Cabrera, Á. González-Prieto, and F. Ortega. Deepfair: Deep learning for improving fairness in recommender systems. *arXiv preprint arXiv:2006.05255*, 2020. (Cited on page 111.)

[16] T. Breuer, N. Ferro, N. Fuhr, M. Maistro, T. Sakai, P. Schaer, and I. Soboroff. How to measure the reproducibility of system-oriented ir experiments. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 349–358. ACM, 2020. (Cited on page 14.)

[17] O. Celma. Music recommendation. In *Music Recommendation and Discovery*, pages

43–85. Springer, 2010. (Cited on page 69.)

[18] J. Chen, C. Wang, and J. Wang. Will you "reconsume" the near past? fast prediction on short-term reconsumption behaviors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015. (Cited on page 23.)

[19] J. Chen, C. Wang, J. Wang, and S. Y. Philip. Recommendation for repeat consumption from user implicit feedback. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):3083–3097, 2016. (Cited on pages 1, 23, and 108.)

[20] W. Chen, P. Ren, F. Cai, F. Sun, and M. de Rijke. Improving end-to-end sequential recommendations with intent-aware diversification. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 175–184, 2020. (Cited on page 111.)

[21] Y. Chen, J. Li, C. Liu, C. Li, M. Anderle, J. McAuley, and C. Xiong. Modeling dynamic attributes for next basket recommendation. In *Proceedings of the 15th ACM Conference on Recommender Systems*, 2021. (Cited on pages 15 and 17.)

[22] Y. Chen, J. Li, C. Liu, C. Li, M. Anderle, J. McAuley, and C. Xiong. Modeling dynamic attributes for next basket recommendation. *arXiv preprint arXiv:2109.11654*, 2021. (Cited on pages 45, 49, 50, and 57.)

[23] Z. Chen, H. Cao, H. Wang, F. Xu, V. Kostakos, and Y. Li. Will you come back/check-in again? understanding characteristics leading to urban revisitation and re-check-in. In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, pages 1–27, 2020. (Cited on page 108.)

[24] M. Cheng, F. Yuan, Q. Liu, X. Xin, and E. Chen. Learning transferable user representations with sequential behaviors via contrastive pre-training. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 51–60, 2021. (Cited on pages 48 and 55.)

[25] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014. (Cited on pages 50 and 73.)

[26] M. Crane. Questionable answers in question answering research: Reproducibility and variability of published results. *Transactions of the Association for Computational Linguistics*, 6:241–252, 2018. (Cited on page 14.)

[27] M. F. Dacrema, P. Cremonesi, and D. Jannach. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 101–109, 2019. (Cited on pages 12, 14, 16, 17, and 19.)

[28] M. F. Dacrema, P. Cremonesi, and D. Jannach. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 101–109, 2019. (Cited on page 60.)

[29] M. F. Dacrema, S. Boglio, P. Cremonesi, and D. Jannach. A troubling analysis of reproducibility and progress in recommender systems research. *ACM Transactions on Information Systems (TOIS)*, 39(2):1–49, 2021. (Cited on pages 12 and 14.)

[30] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston, and D. Sampath. The youtube video recommendation system. In *Proceedings of the 4th ACM conference on Recommender systems*, pages 293–296, 2010. (Cited on page 69.)

[31] S. Deng, O. Sprangers, M. Li, S. Schelter, and M. de Rijke. Domain generalization in time series forecasting. *ACM Transactions on Knowledge Discovery from Data*, To appear.

[32] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004. (Cited on page 69.)

[33] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. (Cited on pages 47, 50, and 52.)

[34] M. D. Ekstrand, R. Burke, and F. Diaz. Fairness and discrimination in retrieval and recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 1403–1404, 2019. (Cited on page 108.)

[35] M. D. Ekstrand, B. Carterette, and F. Diaz. Distributionally-informed recommender

system evaluation. *ACM Transactions on Recommender Systems*, 2023. (Cited on page 134.)

[36] G. Faggioli, M. Polato, and F. Aiolli. Recency aware collaborative filtering for next basket recommendation. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*, pages 80–87, 2020. (Cited on pages 12, 14, 17, 18, 36, 49, 71, 73, and 89.)

[37] H. Fang, D. Zhang, Y. Shu, and G. Guo. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *ACM Transactions on Information Systems*, 39(1):Article 10, 2020. (Cited on pages 1, 107, and 111.)

[38] N. Ferro and D. Kelly. Sigir initiative to implement acm artifact review and badging. *SIGIR Forum*, 52(1):4–10, June 2018. (Cited on page 14.)

[39] Y. Ge, S. Liu, R. Gao, Y. Xian, Y. Li, X. Zhao, C. Pei, F. Sun, J. Ge, W. Ou, and Y. Zhang. Towards long-term fairness in recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, page 445–453, 2021. (Cited on page 108.)

[40] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2 (11):665–673, 2020. (Cited on page 109.)

[41] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471, 2000. (Cited on pages 50 and 73.)

[42] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp. E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1809–1818, 2015. (Cited on page 79.)

[43] M. C. Hao, M. Hsu, U. Dayal, S. F. Wei, T. Sprenger, and T. Holenstein. Market basket analysis visualization on a spherical surface. In *Visual Data Exploration and Analysis VIII*, volume 4302, pages 227–233. International Society for Optics and Photonics, SPIE, 2001. (Cited on page 11.)

[44] C. Hauff, D. Hiemstra, and F. de Jong. A survey of pre-retrieval query performance predictors. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 1419–1420. ACM, 2008. (Cited on page 14.)

[45] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. (Cited on page 52.)

[46] D. Hendrycks and K. Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *arXiv preprint arXiv:1606.08415*, 2016. (Cited on page 52.)

[47] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004. (Cited on pages 1 and 46.)

[48] B. Hidasi and A. 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. (Cited on pages 49, 69, 73, 107, 110, and 111.)

[49] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015. (Cited on pages 1, 49, 73, 107, 108, 110, 111, and 115.)

[50] Y. Hou, B. Hu, Z. Zhang, and W. X. Zhao. Core: Simple and effective session-based recommendation within consistent representation space. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 1796–1801, 2022. (Cited on page 111.)

[51] H. Hu and X. He. Sets2sets: Learning from sequential sets with neural networks. In *Proceedings of the 25th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1491–1499, 2019. (Cited on pages 12, 15, 16, 17, 18, 19, 45, 47, 49, 50, and 73.)

[52] H. Hu, X. He, J. Gao, and Z.-L. Zhang. Modeling personalized item frequency information for next-basket recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1071–1080, 2020. (Cited on pages 12, 14, 16, 17, 18, 19, 36, 49, 57, 60, 69, 71, 73, 90,

and 91.)

[53] X. Huang, Q. Fang, S. Qian, J. Sang, Y. Li, and C. Xu. Explainable interaction-driven user modeling over knowledge graph for sequential recommendation. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 548–556, 2019. (Cited on page 73.)

[54] D. Jannach and M. Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 306–310, 2017. (Cited on page 111.)

[55] D. Jannach, G. de Souza P. Moreira, and E. Oldridge. Why are deep learning models not consistently winning recommender systems competitions yet? a position paper. In *Proceedings of the Recommender Systems Challenge 2020*, RecSysChallenge '20, pages 44–49. ACM, 2020. (Cited on pages 12 and 14.)

[56] S. Kabbur, X. Ning, and G. 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. (Cited on page 69.)

[57] M. Kaminskas and D. Bridge. Diversity, serendipity, novelty, and coverage: A survey and empirical analysis of beyond-accuracy objectives in recommender systems. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 7(1):1–42, 2016. (Cited on pages 1 and 46.)

[58] W.-C. Kang and J. McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining*, pages 197–206, 2018. (Cited on pages 1, 14, 49, 51, 52, 73, 107, 108, 110, 111, 115, and 126.)

[59] O. Katz, O. Barkan, N. Koenigstein, and N. Zabari. Learning to ride a buy-cycle: A hyper-convolutional model for next basket repurchase recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 316–326, 2022. (Cited on pages 46, 49, and 111.)

[60] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009. (Cited on page 74.)

[61] W. Krichene and S. Rendle. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1748–1757, 2020. (Cited on page 14.)

[62] D.-T. Le, H. W. Lauw, and Y. Fang. Correlation-sensitive next-basket recommendation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2808–1814, 2019. (Cited on pages 1, 12, 14, 15, 16, 17, 18, 19, 22, 45, 47, 49, 50, 57, 69, 73, and 115.)

[63] Y. Leng, L. Yu, J. Xiong, and G. Xu. Recurrent convolution basket map for diversity next-basket recommendation. In *International Conference on Database Systems for Advanced Applications*, pages 638–653. Springer, 2020. (Cited on pages 15 and 49.)

[64] L. Lerche, D. Jannach, and M. Ludewig. On the value of reminders within e-commerce recommendations. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, pages 27–35, 2016. (Cited on page 69.)

[65] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1419–1428, 2017. (Cited on pages 1, 49, 73, 107, 110, and 111.)

[66] M. Li, M. Ariannezhad, A. Yates, and M. de Rijke. Masked and swapped sequence modeling for next novel basket recommendation in grocery shopping. In *RecSys 2023: 17th ACM Conference on Recommender Systems*, pages 35–46. ACM, September 2023.

[67] M. Li, M. Ariannezhad, A. Yates, and M. de Rijke. Who will purchase this item next? Reverse next period recommendation in grocery shopping. *ACM Transactions on Recommender Systems*, 1(2):Article 10, 2023. (Cited on page 40.)

[68] M. Li, J. Huang, and M. de Rijke. Repetition and exploration in offline reinforcement learning-based recommendations. In *DRL4IR workshop@CIKM 2023*, October 2023.

[69] M. Li, S. Jullien, M. Ariannezhad, and M. de Rijke. A next basket recommendation reality check. *ACM Transactions on Information Systems*, 41(4):Article 116, 2023. (Cited on pages 45, 46, 49, 56, 57, 58, 60, 65, 71, 73, 77, 84, 90, 106, 108, and 111.)

[70] M. Li, A. Vardasbi, A. Yates, and M. de Rijke. Repetition and exploration in sequential

recommendation. In *SIGIR 2023: 46th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2532–2541. ACM, July 2023. (Cited on page 77.)

[71] J. Lin. The neural hype and comparisons against weak baselines. *SIGIR Forum*, 52(2): 40–51, January 2019. (Cited on page 14.)

[72] J. Liu, P. Dolan, and E. R. Pedersen. Personalized news recommendation based on click behavior. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, pages 31–40, 2010. (Cited on page 69.)

[73] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang. Stamp: Short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1831–1839, 2018. (Cited on pages 1, 49, 73, 107, and 111.)

[74] Y. Liu, M. Li, M. Ariannezhad, M. Mansoury, M. Aliannejadi, and M. deRijke. Measuring item fairness in next basket recommendation: A reproducibility study. In *ECIR 2024: 46th European Conference on Information Retrieval*. Springer, April 2024.

[75] M. Ludewig and D. Jannach. Evaluation of session-based recommendation algorithms. *User Modeling and User-Adapted Interaction*, 28:331–390, 2018. (Cited on page 111.)

[76] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. (Cited on page 79.)

[77] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *Proceedings of the 43rd Advances in Neural Information Processing Systems*, 2013. (Cited on page 79.)

[78] M. Morik, A. Singh, J. Hong, and T. Joachims. Controlling fairness and bias in dynamic learning-to-rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 429–438, 2020. (Cited on page 111.)

[79] M. Naghiaei, H. A. Rahmani, and Y. Deldjoo. Cpfair: Personalized consumer and producer fairness re-ranking for recommender systems. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 770–779, 2022. (Cited on page 111.)

[80] H. Oosterhuis and M. de Rijke. Optimizing ranking models in an online setting. In *ECIR 2019: 41st European Conference on Information Retrieval*, pages 382–396. Springer, April 2019. (Cited on page 14.)

[81] G. K. Patro, A. Biswas, N. Ganguly, K. P. Gummadi, and A. Chakraborty. Fairrec: Two-sided fairness for personalized recommendations in two-sided platforms. In *Proceedings of 2020 The Web Conference*, pages 1194–1204, 2020. (Cited on page 34.)

[82] A. Petrov and C. Macdonald. A systematic review and replicability study of bert4rec for sequential recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 436–447, 2022. (Cited on page 111.)

[83] M. Qian, X. Gu, L. Chu, F. Dai, H. Fan, and B. Li. Flexible order aware sequential recommendation. In *Proceedings of the 2022 International Conference on Multimedia Retrieval*, pages 109–117, 2022. (Cited on pages 48, 55, and 111.)

[84] Y. Qin, P. Wang, and C. Li. The world is binary: Contrastive learning for denoising next basket recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 859–868, 2021. (Cited on pages 12, 14, 17, 18, 19, 22, 45, 47, 49, 50, 57, 60, and 69.)

[85] M. Quadrana, P. Cremonesi, and D. Jannach. Sequence-aware recommender systems. *ACM Computing Surveys*, 51(4):1–36, 2018. (Cited on pages 1, 107, and 111.)

[86] T. Raeder and N. V. Chawla. Market basket analysis with networks. *Social Network Analysis and Mining*, 1(2):97–113, 2011. (Cited on page 11.)

[87] P. Ren, Z. Chen, J. Li, Z. Ren, J. Ma, and M. de Rijke. Repeatnet: A repeat aware neural recommendation machine for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4806–4813, 2019. (Cited on pages 108, 111, and 115.)

[88] S. Rendle, C. Freudenthaler, and L. 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. (Cited on pages 1, 11, 15, 17, 49, 69, and 73.)

[89] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008. (Cited on pages 1, 49, 73, 107, and 111.)

[90] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014. (Cited on page 52.)

[91] V. Subramaniyaswamy, G. Manogaran, R. Logesh, V. Vijayakumar, N. Chilamkurti, D. Malathi, and N. Senthilselvan. An ontology-driven personalized food recommendation in iot-based healthcare system. *The Journal of Supercomputing*, 75(6):3184–3216, 2019. (Cited on page 70.)

[92] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450, 2019. (Cited on pages 1, 49, 51, 52, 53, 54, 73, 107, 108, 111, and 126.)

[93] L. Sun, Y. Bai, B. Du, C. Liu, H. Xiong, and W. Lv. Dual sequential network for temporal sets prediction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1439–1448, 2020. (Cited on pages 12, 15, 17, 45, 47, 49, 50, and 57.)

[94] J. Tagliabue, B. Yu, and F. Bianchi. The embeddings that came in from the cold: Improving vectors for new and rare products with content-based inference. In *Proceedings of the 14th ACM Conference on Recommender Systems*, pages 577–578, 2020. (Cited on page 106.)

[95] P. J. Tan, A. Tanusondjaja, A. Corsi, L. Lockshin, C. Villani, and S. Bogomolova. Audit and benchmarking of supermarket catalog composition in five countries. *International Journal of Advertising*, pages 1–28, 2022. (Cited on page 70.)

[96] J. Tang and K. Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, pages 565–573, 2018. (Cited on pages 1, 49, 73, 107, 108, 111, and 115.)

[97] N. Tax, S. Bockting, and D. Hiemstra. A cross-benchmark comparison of 87 learning to rank methods. *Information Processing & Management*, 51(6):757–772, 2015. (Cited on page 14.)

[98] W. L. Taylor. "cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433, 1953. (Cited on pages 47, 49, 52, and 111.)

[99] V. Vančura. Neural basket embedding for sequential recommendation. In *Fifteenth ACM Conference on Recommender Systems*, pages 878–883, 2021. (Cited on pages 47 and 50.)

[100] S. Vargas and P. Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the 5th ACM conference on Recommender systems*, pages 109–116, 2011. (Cited on pages 12 and 23.)

[101] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. (Cited on pages 1, 47, 49, 50, 51, 52, 73, 107, and 111.)

[102] A. Vlachou, C. Doulkeridis, Y. Kotidis, and K. Nørvåg. Reverse top-k queries. In *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*, pages 365–376, 2010. (Cited on pages 70 and 74.)

[103] A. Vlachou, C. Doulkeridis, Y. Kotidis, and K. Norvag. Monochromatic and bichromatic reverse top-k queries. *IEEE Transactions on Knowledge and Data Engineering*, 23(8): 1215–1229, 2011. (Cited on pages 70 and 74.)

[104] E. M. Voorhees and D. K. Harman, editors. *TREC: Experiment and Evaluation in Information Retrieval*. MIT Press, 2005. (Cited on page 14.)

[105] M. Wan, D. Wang, J. Liu, P. Bennett, and J. McAuley. Representing and recommending shopping baskets with complementarity, compatibility and loyalty. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1133–1142, 2018. (Cited on pages 47, 49, 69, 77, 79, 81, and 90.)

[106] H. Wang, F. Zhang, X. Xie, and M. Guo. Dkn: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*, pages 1835–1844, 2018. (Cited on page 69.)

[107] L. Wang and T. Joachims. User fairness, item fairness, and diversity for rankings in two-sided markets. In *Proceedings of the 44th ACM SIGIR International Conference on Theory of Information Retrieval*, pages 23–41, 2021. (Cited on page 34.)

[108] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng. Learning hierarchical representation model for next basket recommendation. In *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 403–412, 2015. (Cited on pages 15 and 73.)

[109] P. Wang, Y. Zhang, S. Niu, and J. Guo. Modeling temporal dynamics of users' purchase behaviors for next basket prediction. *Journal of Computer Science and Technology*, 34 (6):1230–1240, 2019. (Cited on pages 15 and 49.)

[110] S. Wang, L. Hu, Y. Wang, L. Cao, Q. Z. Sheng, and M. Orgun. Sequential recommender systems: Challenges, progress and prospects. *arXiv preprint arXiv:2001.04830*, 2019. (Cited on pages 1, 48, 55, and 107.)

[111] S. Wang, L. Hu, Y. Wang, Q. Z. Sheng, M. Orgun, and L. Cao. Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks. In *International Joint Conference on Artificial Intelligence*, pages 3771–3777, 2019. (Cited on page 111.)

[112] S. Wang, L. Hu, Y. Wang, Q. Z. Sheng, M. Orgun, and L. Cao. Intention nets: Psychology-inspired user choice behavior modeling for next-basket prediction. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 6259–6266, 2020. (Cited on pages 12, 14, 15, 17, 49, and 73.)

[113] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, and D. Lian. A survey on session-based recommender systems. *ACM Computing Surveys (CSUR)*, 54(7):1–38, 2021. (Cited on page 69.)

[114] S. Wang, C. Gao, M. Gao, J. Yu, Z. Wang, and H. Yin. Who are the best adopters? user selection model for free trial item promotion. *arXiv preprint arXiv:2202.09508*, 2022. (Cited on pages 70 and 74.)

[115] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 5329–5336, 2019. (Cited on page 73.)

[116] H. Wu, B. Mitra, C. Ma, F. Diaz, and X. Liu. Joint multisided exposure fairness for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 703–714, 2022. (Cited on pages 108 and 111.)

[117] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan. Session-based recommendation with graph neural networks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 346–353, 2019. (Cited on pages 1, 49, 69, 73, 107, 108, 111, 115, and 126.)

[118] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*, pages 153–162, 2016. (Cited on page 69.)

[119] Y. Wu, J. Cao, G. Xu, and Y. Tan. Tfrom: A two-sided fairness-aware recommendation model for both customers and providers. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 1013–1022, 2021. (Cited on pages 108 and 111.)

[120] X. Xie, F. Sun, Z. Liu, S. Wu, J. Gao, J. Zhang, B. Ding, and B. Cui. Contrastive learning for sequential recommendation. In *2022 IEEE 38th International Conference on Data Engineering*, pages 1259–1273, 2022. (Cited on pages 48, 49, 55, 107, and 111.)

[121] F. Yalvaç, V. Lim, J. Hu, M. Funk, and M. Rauterberg. Social recipe recommendation to reduce food waste. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, pages 2431–2436, 2014. (Cited on page 70.)

[122] W. Yang, K. Lu, P. Yang, and J. Lin. Critically examining the "neural hype": Weak baselines and the additivity of effectiveness gains from neural ranking models. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1129–1132. ACM, 2019. (Cited on page 14.)

[123] G.-E. Yap, X.-L. Li, and P. S. Yu. Effective next-items recommendation via personalized sequential pattern mining. In *Proceedings of the International Conference on Database Systems for Advanced Applications*, pages 48–64, 2012. (Cited on page 69.)

[124] Q. Yin, H. Fang, Z. Sun, and Y.-S. Ong. Understanding diversity in session-based recommendation. *arXiv preprint arXiv:2208.13453*, 2022. (Cited on page 111.)

[125] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 729–732, 2016. (Cited on pages 1, 11, 12, 14, 15, 16, 17, 18, 19, 22, 45, 47, 49, 50, 57, 69, 71, and 73.)

[126] J. Yu, H. Yin, X. Xia, T. Chen, J. Li, and Z. Huang. Self-supervised learning for recommender systems: A survey. *arXiv preprint arXiv:2203.15876*, 2022. (Cited on pages 49, 55, and 111.)

[127] L. Yu, L. Sun, B. Du, C. Liu, H. Xiong, and W. Lv. Predicting temporal sets with deep neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1083–1091, 2020. (Cited on pages 1, 12, 14, 15, 16, 17, 18, 19, 36, 45, 47, 49, 50, 57, 60, 69, 71, 73, 90, 91, and 92.)

[128] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He. A simple convolutional generative network for next item recommendation. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, pages 582–590, 2019. (Cited on page 73.)

[129] M. Zehlike and C. Castillo. Reducing disparate exposure in ranking: A learning to rank approach. In *Proceedings of The Web Conference 2020*, page 2849–2855, 2020. (Cited on page 111.)

[130] M. Zhang and N. Hurley. Avoiding monotony: Improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 123–130, 2008. (Cited on page 111.)

[131] S. Zhang, L. Yao, A. Sun, and Y. Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys*, 52(1):1–38, 2019. (Cited on pages 1 and 107.)

[132] Z. Zhang, C. Jin, and Q. Kang. Reverse k-ranks query. *Proceedings of the VLDB Endowment*, 7(10):785–796, 2014. (Cited on pages 70 and 74.)

[133] W. X. Zhao, J. Chen, P. Wang, Q. Gu, and J.-R. Wen. Revisiting alternative experimental settings for evaluating top-n item recommendation algorithms. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2329–2332, 2020. (Cited on page 111.)

Summary

Recommender systems serve as an essential bridge for connecting users and items on a digital platform, which helps users find relevant items and helps items reach their potential users. Over the years, numerous recommendation models targeted at various domains have been designed.

People exhibit both regular habits and curiosity, demonstrating repetition behaviors as well as exploration behaviors when engaging with platforms. The coexistence of repetition and exploration imposes challenges for performance evaluation and designing recommendation models. Thus, this thesis aims to provide insights and understand the repetition and exploration in various recommendation tasks. Considering the repetition and exploration in recommendation, this thesis covers two research aspects: (i) evaluate recommendation performance, and (ii) optimize and design recommendation methods.

Specifically, this thesis investigates four recommendation tasks, (i) next basket recommendation, (ii) next novel basket recommendation, (iii) reverse next-period recommendation, and (iv) sequential recommendation. In terms of recommendation evaluation, this thesis introduces a comprehensive set of evaluation metrics that take into account both user-side and item-side aspects, which allow for a more detailed understanding of recommendation performance. Additionally, the thesis presents guidelines for evaluating recommendation models in a scenario with both repetition and exploration behaviors. In the context of recommendation optimization and design, the thesis puts forward a range of training strategies and recommendation model designs based on the insights derived from the analysis of repetition and exploration.

To sum up, this thesis uncovers the key differences between repetition and exploration in recommendation, and highlights the importance of evaluating and optimizing recommendation models from the perspective of repetition and exploration.

Samenvatting

Aanbevelingssystemen dienen als een essentiële brug voor het verbinden van gebruikers en items op een digitaal platform, waardoor gebruikers relevante items kunnen vinden en items hun potentiële gebruikers kunnen bereiken. Door de jaren heen zijn er talloze aanbevelingsmodellen ontworpen, gericht op verschillende domeinen.

Mensen vertonen zowel reguliere gewoonten als nieuwsgierigheid, waarbij ze zowel herhalingsgedrag als verkenningsgedrag vertonen wanneer ze met platforms omgaan. Het naast elkaar bestaan van herhaling en verkenning brengt uitdagingen met zich mee voor het evalueren en het ontwerpen van aanbevelingsmodellen. Het doel van dit proefschrift is dan ook om inzichten te verschaffen en inzicht te krijgen in de herhaling en verkenning van verschillende aanbevelingstaken. Gezien de herhaling en verkenning in aanbevelingen, behandelt dit proefschrift twee onderzoeksaspecten: (i) evaluatie van de prestaties van aanbevelingen, en (ii) optimalisatie en ontwerp van aanbevelingsmethoden.

Concreet onderzoekt dit proefschrift vier aanbevelingstaken: (i) *next basket recommendation*, (ii) *next novel basket recommendation*, (iii) *reverse next-period recommendation*, en (iv) *sequential recommendation*. Op het gebied van de evaluatie van aanbevelingen introduceert dit proefschrift een uitgebreide set evaluatiemetrieken die zowel rekening houden met aspecten aan de gebruikerszijde als aan de itemzijde, waardoor een gedetailleerder inzicht in de prestaties van aanbevelingen mogelijk wordt. Daarnaast presenteert het proefschrift richtlijnen voor het evalueren van aanbevelingsmodellen in een scenario met zowel herhalings- als verkenningsgedrag. In de context van optimalisatie en ontwerp van aanbevelingen presenteert het proefschrift een reeks trainingsstrategieën en ontwerpen van aanbevelingsmodellen, gebaseerd op de inzichten verkregen uit de analyse van herhaling en verkenning.

Samenvattend onthult dit proefschrift de belangrijkste verschillen tussen herhaling en verkenning bij aanbevelingen, en benadrukt het het belang van het evalueren en optimaliseren van aanbevelingsmodellen vanuit het perspectief van herhaling en verkenning.