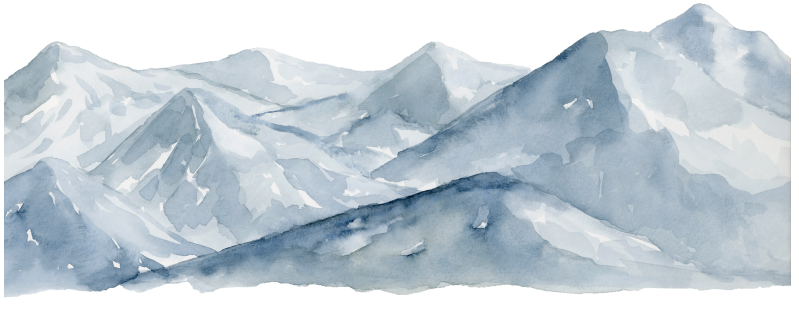


User-oriented Recommender Systems in Retail

Mozhdeh Ariannezhad

User-oriented Recommender Systems in Retail

Mozhdeh Ariannezhad

User-oriented Recommender Systems in Retail

Mozhdeh Ariannezhad

User-oriented Recommender Systems in Retail

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in
de Agnietenkapel
op donderdag 14 december 2023, te 16.00 uur

door

Mozhdeh Ariannezhad

geboren te Ahvaz

Promotiecommissie

Promotor:	prof. dr. M. de Rijke	Universiteit van Amsterdam
Co-promotor:	dr. ing. S. Schelter	Universiteit van Amsterdam
Overige leden:	prof. dr. H. Haned	Universiteit van Amsterdam
	dr. M. Lalmas	Spotify
	prof. dr. ir. C.I. Sánchez Gutiérrez	Universiteit van Amsterdam
	dr. E.V. Shutova	Universiteit van Amsterdam
	prof. dr. E. Yilmaz	University College London

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

The work described in this thesis has been primarily carried out at the Information Retrieval Lab of the University of Amsterdam and in part during an internship at Bloomberg AI. The research carried out at the University of Amsterdam was funded by Ahold Delhaize.

Cover by Kunrus Nakwijitpong
Printed by Ridderprint, The Netherlands

ISBN: 978-94-6483-591-5

Acknowledgments

I was lucky. I enjoyed my PhD journey, and that is mostly because I got to know and work with so many amazing people. I want to take this opportunity to thank them.

Maarten, I am grateful for having you as my supervisor. I will always look up to you and aspire to be more like you. Your ability to bring out the best in people, including me, is truly inspiring. Thank you!

Sebastian, I am thankful to you for your supervision and the way you have cared for me as an individual. You made the difficult days remarkably better.

Clarisa, Emine, Hinda, Katia, and Mounia, I am incredibly fortunate to have you on my PhD committee. Thank you for dedicating your valuable time to read and discuss my thesis.

I cannot think of a better place to do meaningful research than IRLab. I was surrounded by brilliant people throughout these years. I want to thank everyone who was part of the lab during my time there: Ali A, Ali V, Amin, Ana, Andrew, Anna, Antonis, Arezoo, Arian, Barrie, Bob, Carsten, Chang, Christof, Chuan M, Chuan W, Clara, Clemencia, Dan, David S, David V, Federico, Gabriel, Gabrielle, Georgios, Evangelos, Hamid, Harrie, Hinda, Hongyu, Ilias, Ilya, Iman, Ivana, Jiahuan, Jie, Jin, Jingfen, Julien, Justine, Maarten M, Maartje, Mahsa, Maria, Mariya, Marta, Marzieh, Maurits, Ming, Mohammad, Mostafa, Mounia, Negin, Nikos, Olivier, Oscar, Pablo, Panagiotis, Peilei, Pengjie, Petra, Philipp, Pooya, Rolf, Romain, Ron, Roxana, Ruben, Ruqing, Saedeh, Sam, Sami, Sebastian, Shaojie, Shashank, Shubha, Songgaojun, Spyretta, Svitlana, Thilina, Thong, Vaishali, Vera, Wanyu, Weija, Xinyi, Yang, Yangjun, Yibin, Yifei, Yuanna, Yuanxing, Yuyue, Zahra, Zihan, and Ziming. Thank you for making the lab a great place to work!

Sami, thank you for being my friend. We got to go through this journey together and collaborate on many different things, for which I am grateful.

Mariya, thank you for always being there for me. For all the talks, all the laughs, for everything we got to share along the way.

Ali V, Arezoo, and Mohammad, I am happy I found you along my PhD journey! We went from colleagues to friends throughout these years. I am grateful for your friendship.

Sam, thank you for always being there to talk, for all the worries and joys we shared.

Ming, I enjoyed our collaborations from the day you joined the lab until the very end. Thank you for all your hard work!

Clara and Jinfeng, we got to sit together in the best office of LAB 42. Thank you for being the best office mates. Clara, I am grateful to you for being my paranymph as well!

I was part of AIRLab during my PhD, and because of that I had the chance to work with the kind, supportive, and professional people in Albert Heijn and Ahold Delhaize. Anca, Bart, Bassem, Bouke, Dung, Michelle, and Min, thank you for making this PhD possible. AIRLab gave me the opportunity to work with Paul and Stefan from INDE Lab. I am thankful to for this opportunity.

During my PhD, I did an internship with Bloomberg AI in London back in 2020. I had an amazing experience working with a great team. Anju, Diego, Edgar, and Mohamed, I learned a lot from you. Thank you for supporting me during my time at

Bloomberg.

I did a second internship with Booking.com in 2022. I worked with the best and enjoyed my time there so much that I decided to go back for a full time job. Hassan, Mathijs, Radu, Rafail, and Shubhika, thank you for showing me what it is like to do impactful research in industry.

I was lucky to be surrounded with my wonderful friends here in Amsterdam. Atieh, Mahsa, Masoud, Morteza, Sara, Zahra, and Zeynab, thank you for all the moments we shared. Getting to the finish line wouldn't have been possible without your support.

Even from far away, I always felt supported by my incredible friends from all over the world. Afsoon, Choobi, Ela, Hamidia, Hanieh, Mahshid, Mahya, Mosi, and Negar, I am grateful for knowing that I can always count on you.

Finally, I would like to thank my family. No words can describe my gratitude to you. Baba, Maman, Negar, and Shayan, I am blessed to have you by my side every step of the way. I also extend my appreciation to my mother and father-in-law for their support.

Ali, you are everything. You gave meaning to this journey, like you do to life. Thank you.

Mozhdeh
Amsterdam
November 2023

Contents

1 Introduction

A primary objective of every service provider system is to fulfill its users' needs [57, 140]. Irrespective of the nature of the service, encompassing domains such as media, entertainment, products, or information, user satisfaction remains a key consideration [22, 33, 60, 132]. While the goal of satisfying users is the same across different domains and services, a universal solution that disregards domain-specific characteristics often proves suboptimal [124]. As an example, consider the difference between grocery shopping and home appliance shopping. While both fall under the retail category, the differences in user behavior between the two are significant: it is safe to assume that supermarket customers would repurchase the same set of items in short periods of times, say, on a weekly basis [12]. However, this assumption does not hold true for items with a longer life time, such as washing machines or electric kettles [149]. The same situation extends to other seemingly similar services: music and movie streaming, both belonging to the media sector. Listening to the same song over and over again is a typical pattern [121], whereas it is not as common for users to re-watch the same movie multiple times [6]. Consequently, it is important to take domain-specific characteristics into account when designing solutions to improve user satisfaction.

Different means are helpful in ensuring users have a positive experience with a given system. Recommendation is one: a successful recommender system helps users find what they need faster and discover relevant items offered by the provider [73, 152, 169]. Accurate demand forecasting is another, which guarantees timely availability of such items for users [11]. One of the main sources of data to facilitate both tasks is *user interactions* with the system [52]. On the one hand, analyzing interaction data from a user level perspective helps to understand users better, which is crucial for providing attractive personalized recommendation [100, 141]. On the other hand, studying data from an item level to see how they have been consumed in the past is an important signal for predicting the demand in the future [36, 162].

In this thesis, we investigate how to learn from domain-specific user interaction data and thus improve the user experience with service provider systems. We focus on recommendation as our main task, and retail as our main domain — hence the thesis title. To diversify and to analyze whether our methodology and findings generalize to other tasks and domains, we explore the finance domain and the demand forecasting task as additional directions.

There is an extensive and growing body of research on understanding and analyzing

user interaction data across various fields, including banking, web search, mobile search, emails, and software suites [5, 26, 153, 156]. Furthermore, the volume of research in retail and e-commerce domains is significant, particularly focusing on single-channel retail setups, where the data comes either from online clicks, or offline in-store transactions [32, 60, 77, 100, 141, 151, 159, 162, 170]. It is worth noting that in many cases, these studies have not had a direct impact on shaping the design of subsequent models. In fact, many state-of-the-art models are designed to be domain-agnostic, and their performance tends to be less impressive compared to models specifically tailored to a particular domain [11, 13, 16]. Our goal is to address these limitations by taking a two-pronged approach. Firstly, we aim to understand and learn from user interactions in a *multi-channel* retail setting, where interaction data comes from both digital (i.e., online) and in-store (i.e., offline) shopping. Secondly, we intend to explicitly incorporate the insights gained from domain-specific studies of user behavior into the design of models for downstream tasks. This approach seeks to enhance the effectiveness and applicability of our models in their respective domains.

With the rise of e-commerce in recent years, retailers with physical stores feel the need to provide the opportunity of online shopping for their customers in addition to in-store shopping [55]. The online channel is not used solely for shopping; customers may use it as a means to explore the product inventory and check products before shopping offline [31, 68]. The introduction of an online sales channel does not result in the isolation of the offline retail channel. Instead, it creates a multi-channel shopping experience for customers in retail sectors like grocery, cosmetics, and apparel [8, 69]. While these channels can operate with different strategies, for example in marketing and communication with customers, treating them separately as distinct units has downsides from both customer and business points of view. Such an approach adds friction for customers who seek a seamless shopping experience across channels and leads to operational inefficiencies in supply chain and replenishment for the business [2, 69, 106].

To avoid these drawbacks, we focus on gaining a holistic view of multi-channel retail in the first part of this thesis. We first study demand forecasting in multi-channel retail, and we further provide a comprehensive understanding of user behavior in this setting, with insights that can inform the design of user-oriented recommender systems [151]. We continue on this route in the second part of this thesis, and focus on gaining actionable insights from user behavior understanding that are then used by our proposed recommender algorithms.

As part our journey in this thesis, we take a step back from the retail domain, and tap into the finance domain. For users of financial information systems, timely access to the right information is crucial [21, 53]. However, our understanding about how users interact with financial information systems is limited, where existing literature mostly addresses the stock recommendation task [30, 34, 164]. We take a first step to fill this gap by analyzing a large-scale log data from user interactions with a financial information system, and by designing recommender systems for financial documents. Next, we focus on retail, in particular grocery shopping. In retail, recommendation systems help users to find the items that they need from large inventories. Grocery shopping has two main characteristics that make it distinct from other retail domains: 1) users shop for items repeatedly and on a regular basis, and 2) grocery items have a short life time and are repurchased frequently by the same user [97]. As a result, historical purchases of

users offer strong signals for recommending relevant items for their future purchases. We study the repeat consumption behavior and design recommender systems that are able to explicitly model users personal preferences, as indicated by their history.

1.1 Research Outline and Questions

Each chapter in this thesis is organized around one or more of the following research themes: 1) Characteristics of multi-channel retail (Chapter 2 and 3); 2) Understanding user behavior (Chapter 3, 4, and 5); and 3) Designing user-oriented recommender systems (Chapter 4, 5, and 6).

Below, we list our main research questions. Each chapter investigates one of them. In the first two chapters, we focus on multi-channel retail and study demand forecasting and user behavior in this setting.

RQ1 How can we effectively utilize data from multiple shopping channels to improve the accuracy of demand forcasting models?

Predicting the amount of sales in the future is a fundamental problem in the replenishment process of retail companies. Models for forecasting the demand of an item typically rely on influential features and historical sales. However, the values of some influential features (to which we refer as *non-plannable features*) are only known during model training (for the past), and not for the future at prediction time. Examples of such features include sales in other channels, such as other stores in supermarket chains. Existing forecasting methods ignore such non-plannable features or wrongly assume that they are also known at prediction time. To answer **RQ1**, we consider non-plannable features to be privileged information, i.e., information that is available at training time but not at prediction time, and design a neural network to leverage this source of data accordingly. We present a dual branch neural network architecture that incorporates non-plannable features at training time, with a first branch to embed the historical information, and a second branch, the *privileged information* (PI) *branch*, to predict demand based on privileged information. Next, we leverage a single branch network at prediction time, which applies a simulation component to mimic the behavior of the PI branch, whose inputs are not available at prediction time.

RQ2 What are the characteristics of user behavior in multi-channel retail settings?

The growing popularity of online shopping drives traditional retailers with physical stores to allow their customers to shop online as well as offline, that is, in-store. Increasingly, customers can browse and purchase products across multiple shopping channels. Understanding how customer behavior relates to the availability of multiple shopping channels is an important prerequisite for many downstream machine learning tasks, such as recommendation and purchase prediction. However, previous work in this domain is limited to analyzing single-channel behavior only. To answer **RQ2**, we provide the first insights into multi-channel customer behavior in retail based on a large sample of 2.8 million transactions originating from 300,000 customers of a food retailer in Europe. We further investigate

the performance of a next basket recommendation model under multi-channel settings.

In the next three chapters, we focus on designing recommender systems that are informed by insights from user behavior understanding studies. We first consider the finance domain, and then move on to retail.

RQ3 What are the characteristics of financial information seeking behavior in user interactions with company filings and how can they be used in designing user-oriented filing recommender systems?

Publicly-traded companies are required to regularly file financial statements and disclosures. Analysts, investors, and regulators leverage these filings to support decision making, with high financial and legal stakes. Despite their ubiquity in finance, little is known about the information seeking behavior of users accessing such filings. To answer **RQ3**, we analyze 14 years of logs of users accessing company filings of more than 600K distinct companies on the U.S. Securities and Exchange Commission's (SEC) Electronic Data Gathering, Analysis, and Retrieval (EDGAR) system, the primary resource for accessing company filings. We provide an analysis of the information-seeking behavior for this high-impact domain. Understanding user interactions with EDGAR can suggest ways to enhance the user journey in browsing filings, e.g., via filing recommendation. Based on our observations, we define two filing recommendation tasks that correspond to either less or more frequent users, i.e., within-session and next-session recommendation. We design recommendation models that build on insights from our user behavior analysis.

RQ4 What are the characteristics of repeat consumption behavior in the grocery shopping domain and how can they be used in desiging next basket recommender systems?

Retailers such as grocery stores or e-marketplaces often have vast selections of items for users to choose from. Predicting a user's next purchases has gained attention recently, in the form of next basket recommendation (NBR), as it facilitates navigating extensive assortments for users. Neural network-based models that focus on learning basket representations are the dominant approach in the recent literature. However, these methods do not consider the specific characteristics of the grocery shopping scenario, where users shop for grocery items on a regular basis, and grocery items are repurchased frequently by the same user. To answer **RQ4**, we first gain a data-driven understanding of users' repeat consumption behavior through an empirical study on six public and proprietary grocery shopping transaction datasets. We discover that a Next Basket Recommendation (NBR) model with a strong focus on previously purchased items can potentially achieve high performance. We introduce *ReCANet*, a repeat consumption-aware neural network that explicitly models the repeat consumption behavior of users in order to predict their next basket.

RQ5 How can we design an effective and scalable within-basket recommender system that explicitly considers users' personal preferences?

Users of online shopping platforms typically purchase multiple items at a time in the form of a shopping *basket*. Personalized within-basket recommendation is the task of recommending items to complete an incomplete basket during a shopping session. In contrast to the related task of session-based recommendation, where the goal is to complete an ongoing anonymous session, we have access to the shopping history of the user in within-basket recommendation. Previous studies have shown the superiority of neighborhood-based models for session-based recommendation and the importance of personal history in the grocery shopping domain. But their applicability in within-basket recommendation remains unexplored. To answer **RQ5**, we propose PerNIR, a *personalized nearest neighbor-based model for within-basket recommendation* that explicitly models the personal history of users for within-basket recommendation in grocery shopping. The main novelty of PerNIR is in modeling the short-term interests of users, which are represented by the current basket, as well as their long-term interests, which is reflected in their purchasing history. In addition to the personal history, neighboring users are utilized to capture the collaborative purchase behavior.

1.2 Main Contributions

This thesis makes different types of contributions: algorithmic, empirical, and concerning resources.

Algorithmic contributions

- SB-PIANN and DB-PIANN: single and dual branch neural demand forecasting models capable of leveraging privileged information, i.e., information that is available during training time and not available at prediction time (Chapter 2).
- CIK-S-POP, CIK-encoder, and CIK-P-POP: two-stage recommender systems for different filing recommendation tasks in the finance domain (Chapter 4).
- ReCANet: a repeat consumption-aware neural network for next basket recommendation in grocery shopping (Chapter 5).
- PerNIR: a personalized nearest neighbor-based model for within-basket recommendation that explicitly considers users' personal preferences (Chapter 6).

Empirical contributions

- An empirical verification of the effectiveness of PIANN for demand forecasting in retail in comparison with state-of-the-art (Chapter 2).
- Insights from a descriptive study on understanding multi-channel customer behavior in retail (Chapter 3).
- Insights from a descriptive study on understanding financial information seeking behavior from user interactions with company filings (Chapter 4).
- Insights from a descriptive study on understanding customers' repurchasing behavior in grocery shopping (Chapter 5).

- An empirical verification of the effectiveness of ReCANet for next basket recommendation in grocery shopping in comparison with state-of-the-art (Chapter 5).
- An empirical verification of the effectiveness of PerNIR for within-basket recommendation in grocery shopping in comparison with state-of-the-art (Chapter 6).

Resource contributions

- The source code for PIANN (Chapter 2) released under an open source license.
- The source code for ReCANet (Chapter 5) released under an open source license.
- The source code for PerNIR (Chapter 6) released under an open source license.

1.3 Thesis Overview

This thesis consists of an introductory chapter, five research chapters divided into two parts, and a concluding chapter. In each research chapter, one of the main research questions (defined in Section 1.1) is investigated. Additionally, Chapters 2, 5, and 6 answer more fine-grained research questions that concern chapter-specific contributions.

The current chapter, Chapter 1, introduces the subject of this thesis. It outlines its research questions and the main contributions, together with the origins for each chapter.

Part I, titled *Characteristics of Multi-Channel Retail*, contains two research chapters. Chapter 2 looks at improving the performance of demand forecasting in a multi-channel retail scenario. Specifically, this chapter provides a solution to make use of sales in multiple shopping channels in order to improve the performance of demand forecasting in a target channel. Chapter 3 studies user behavior in a multi-channel retail setting. This chapter provides insights about the channel-specific properties of user behavior, and their effects on the performance of recommender systems.

Part II, titled *From User Behavior to Recommendation*, contains three research chapters. Chapters 4 and 5 both include two main sections: user behavior analysis and proposed recommender systems. Chapter 4 delves into the finance domain, providing insights on financial information seeking behavior in user interactions with company filings. Chapter 5 focuses on understanding repurchasing behavior in retail, specifically in the grocery shopping domain. Both chapters propose domain-specific user-oriented recommender systems that are informed by the finding of the aforementioned user behavior analysis. Chapter 6 leans more on scalable recommendation in retail: it proposes an efficient and easy to scale recommender system that explicitly models users' personal preferences that are reflected in their purchasing history.

Finally, Chapter 7 concludes the thesis by summarizing the findings and outlining directions for future work.

All chapters are based on separate published papers, are self-contained, and can be read independently. We aim to avoid creating alternate versions of published work that deviate from the originals. Unavoidably, this results in some overlap in the problem formulations, the description of datasets and some baseline methods in different chapters.

1.4 Origins

Below we list the publications that form the basis for each chapter in this thesis and briefly describe the roles of co-authors.

Chapter 2 is based on the following paper:

> M. Ariannezhad, S. Schelter, and M. de Rijke. Demand forecasting in the presence of privileged information. In *Advanced Analytics and Learning on Temporal Data - 5th ECML PKDD Workshop, AALTD 2020, Ghent, Belgium, September 18, 2020*, pages 46–62. Springer, 2020.

MA proposed the idea, designed the model, implemented it, ran experiments and analyzed most results. SSC and MdR helped with the model design and intermediate result analysis. All authors contributed to the writing.

Chapter 3 is based on the following paper:

> M. Ariannezhad, S. Jullien, P. Nauts, M. Fang, S. Schelter, and M. de Rijke. Understanding multi-channel customer behavior in retail. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1–5, 2021*, pages 2867–2871. ACM, 2021.

MA proposed the idea, implemented it, ran experiments and analyzed most results. SJ, SSC and MdR helped with the intermediate result analysis. PN and MF facilitated access to data. All authors contributed to the writing.

Chapter 4 is based in part on the following paper:

> M. Ariannezhad, M. Yahya, E. Meij, S. Schelter, and M. de Rijke. Understanding financial information seeking behavior from user interactions with company filings. In *Companion of The Web Conference 2022, Virtual Event / Lyon, France, April 25–29, 2022*, pages 586–594. ACM, 2022.

This work was done during an internship at Bloomberg AI in 2020. EM proposed the initial idea. MA formed the final idea, prepared the data, ran experiments and analyzed most results. MY and EM helped with technical details. MY, EM, SSC and MdR helped with intermediate result analysis. All authors contributed to the writing.

Chapter 5 is based on the following paper:

> 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 '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11–15, 2022*, pages 1240–1250. ACM, 2022.

MA proposed the idea, designed the model, implemented it, ran experiments and analyzed most results. SJ, ML, SSC and MdR helped with the model design and intermediate result analysis. MF facilitated access to data. All authors contributed to the writing.

Chapter 6 is based on the following paper:

> M. Ariannezhad, M. Li, S. Schelter, and M. de Rijke. A personalized neighborhood-based model for within-basket recommendation in grocery shopping. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023, Singapore, 27 February 2023 – 3 March 2023*, pages 87–95. ACM, 2023.

MA proposed the idea, designed the model, implemented it, ran experiments and analyzed most results. ML, SSC and MdR helped with the model design and intermediate result analysis. All authors contributed to the writing.

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

- 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*. ACM, 2023.
- M. Li, S. Jullien, M. Ariannezhad, and M. de Rijke. A next basket recommendation reality check. *ACM Trans. Inf. Syst.*, 41(4), 2023.
- M. Ariannezhad, M. Li, S. Jullien, and M. de Rijke. Complex item set recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23–27, 2023*, pages 3444–3447. ACM, 2023.
- S. Jullien, M. Ariannezhad, P. Groth, and M. de Rijke. A simulation environment and reinforcement learning method for waste reduction. *Trans. Mach. Learn. Res.*, 2023, 2023.
- M. Li, M. Ariannezhad, A. Yates, and M. de Rijke. Who will purchase this item next? Reverse next period recommendation in grocery shopping. *ACM Trans. Recomm. Syst.*, 1(2), 2023.
- S. Schelter, M. Ariannezhad, and M. de Rijke. Forget me now: Fast and exact unlearning in neighborhood-based recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23–27, 2023*, pages 2011–2015. ACM, 2023.
- S. Jullien, M. Ariannezhad, P. Groth, and M. de Rijke. GLDQN: Explicitly parameterized quantile reinforcement learning for waste reduction. *CoRR*, abs/2205.-15455, 2022.

- M. Ariannezhad. Understanding and learning from user behavior for recommendation in multi-channel retail. In *Advances in Information Retrieval – 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022*, pages 455–462. Springer, 2022.
- T. van de Looij and M. Ariannezhad. Cluster-based forecasting for intermittent and non-intermittent time series. In *Advanced Analytics and Learning on Temporal Data – 6th ECML PKDD Workshop, AALTD 2021, Bilbao, Spain, September 13, 2021*, pages 139–154. Springer, 2021.
- M. Ariannezhad, A. Montazeralghaem, H. Zamani, and A. Shakery. Improving retrieval performance for verbose queries via axiomatic analysis of term discrimination heuristic. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7–11, 2017*, pages 1201–1204. ACM, 2017.
- M. Ariannezhad, A. Montazeralghaem, H. Zamani, and A. Shakery. Iterative estimation of document relevance score for pseudo-relevance feedback. In *Advances in Information Retrieval – 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8–13, 2017*, pages 676–683, 2017.
- H. R. Zagheli, M. Ariannezhad, and A. Shakery. Negative feedback in the language modeling framework for text recommendation. In *Advances in Information Retrieval – 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8–13, 2017*, pages 662–668, 2017.
- R. Rahimi, A. Shakery, J. Dadashkarimi, M. Ariannezhad, M. Dehghani, and H. N. Esfahani. Building a multi-domain comparable corpus using a learning to rank method. *Nat. Lang. Eng.*, 22(4):627–653, 2016.

Part I

Characteristics of Multi-Channel Retail

2

Demand Forecasting in Multi-Channel Retail

In this chapter, we address **RQ1**: How can we effectively utilize data from multiple shopping channels to improve the accuracy of demand forcasting models? Given a certain shopping channel as target, we propose to model the sales data from other channels as privileged information. We design a neural forecasting approach that is able to make use of this source of data.

2.1 Introduction

Demand forecasting aims to predict future sales and has the potential to significantly improve supply chain management. An accurate forecast prevents overstocking and reduces costs, waste, and losses. At the same time, it also avoids understocking and thereby helps to prevent unfulfilled orders and unsatisfied customers [28, 144]. In practice, demand forecasting is modeled as a time series forecasting (TSF) problem, where the goal is to predict future sales volumes based on historical sales and influential features [25]. Following the success of deep neural networks in sequence-to-sequence tasks such as machine translation [139], recent work has studied the effectiveness of neural networks for TSF in general [81, 116, 127], and demand forecasting in particular [36, 50, 115].

We divide influential features into two categories, based on their availability at the time of forecast. *Plannable features* are known for the past and the future; examples are time-dependent features such as calendar events, or static features such as product characteristics. *Non-plannable features* are time-dependent and only known for the past, e.g., sales in other stores. Many neural-based approaches use an encoder-decoder structure to map the history of a time series to its future. A common strategy when treating influential features is to feed their historical values to the encoder, either alongside the historical sales data [50], or to a different layer that is responsible for encoding them [36]. The decoder then either uses their future values to produce the forecast [50, 155], or assumes that they are unknown [36]. Neither of these schemes

This chapter was published as M. Ariannezhad, S. Schelter, and M. de Rijke. Demand forecasting in the presence of privileged information. In *Advanced Analytics and Learning on Temporal Data - 5th ECML PKDD Workshop, AALTD 2020, Ghent, Belgium, September 18, 2020*, pages 46–62. Springer, 2020.

is ideal for non-plannable features. Using future values of influential features in the decoding stage makes sense for plannable features, but the approach is not applicable for non-plannable features. A model trained in that way is not applicable in a real-world setting as the non-plannable features are not known at prediction time. Simply ignoring non-plannable features – both at training and prediction time – is not optimal either as they carry important information that should be leveraged at the time of training the model.

In this chapter, we therefore propose an indirect approach to model the effects of non-plannable features, and treat them as *privileged information* [101], i.e., information that is available at the time of training the model but not at prediction time (Section 2.3.1). We introduce a neural network architecture to capture the effect of these features at training time, and use this effect to produce a forecast at prediction time (Section 2.3.2). To this end, we propose *two different network architectures for training and prediction*. At the time of training, the network has two different branches. The first branch is responsible for modeling the effect of historical sales and plannable features, while the second branch uses non-plannable features as input to produce a forecast based on them. At prediction time, the second branch is not available, and is *replaced by a simulation network*, which is trained to mimic its behavior (Section 2.3.3).

Our experimental evaluation demonstrates that the proposed model outperforms state-of-the-art baselines on several datasets in terms of mean absolute error and symmetric mean absolute percentage error. Our experiments show that the proposed network is not only able to learn from non-plannable features at training time, but can also embed this type of information for use at prediction time.

We summarize the contributions of our research as follows:

- We categorize the influential features for demand forecasting into two categories, based on their availability for the time of forecast. We propose to treat the non-plannable features as privileged information (Section 2.3.1).
- We design a novel neural network architecture that is able to leverage privileged information. To this end, we propose to use two different networks for training and prediction time, where the network at prediction time simulates the effect of the unknown privileged information (Section 2.3.2).
- We conduct extensive experiments on two publicly available datasets. Our experimental results show that our approach outperforms state-of-the-art baselines for demand forecasting in the majority of cases in terms of mean absolute error and symmetric mean absolute percentage error metrics (Section 2.5).

2.2 Related Work

Time series forecasting. Prior work on time series forecasting (TSF) mostly focuses on linear approaches [125], such as Auto-Regressive Integrated Moving Average (ARIMA) model [27], with a solid underlying theory and relatively few parameters. However, linear methods cannot model non-linear temporal dependencies and complex relationships between different dimensions of a time series. Recently, neural

network-based approaches have found their way into TSF. The most dominant type of network used are recurrent neural networks (RNNs) and long short-term memory networks (LSTMs) [3, 19, 116, 127, 131]. Convolutional neural networks (CNNs) are also considered in the literature [78, 130], and some work studies networks with both recurrent and convolutional components [81, 158].

For the task of demand forecasting, state-of-the-art approaches mostly employ neural-based models. TADA [36] uses different LSTM layers to model different kinds of influential features, and the multimodal-attention model proposed in [50] uses a bidirectional LSTM with an attention mechanism to better capture latent patterns in historical data. To incorporate the impact of substitutable products with respect to the target product, DSF [115] uses a sequence-to-sequence structure with gated recurrent units.

Our focus is on modeling non-plannable features. In previous work, non-plannable features are either treated as plannable, i.e., with the unrealistic assumption that their future values are known at prediction time, or are only used as historical data. None of these approaches is able to incorporate non-plannable features in a realistic manner. In contrast, we propose a neural architecture, that 1) is capable of modeling non-plannable features, and 2) has different components for historical sales and influential features, which are usually treated in the same way in previous work.

Learning under privileged information. The learning under privileged information (LUPI) framework was originally proposed for support vector machines [147]. The idea was again popularized in [101], where it was unified with knowledge distillation for neural networks. Most of the work done in this area is in the field of computer vision [39, 64, 82], with teacher-student networks as the dominant approach. Teacher-student networks are mostly based on distilling knowledge from a 'teacher' network to a 'student' network at training time through the loss function, which makes sense for classification problems [101], where class probabilities produced by the teacher network are treated as 'soft targets' for training the student network.

To the best of our knowledge, LUPI has never been used in time series forecasting and utilizing existing LUPI frameworks is not straight-forward in a forecasting scenario. In this work, we propose a network architecture to leverage non-plannable features. We achieve this with two different networks at training and prediction time. In contrast to common teacher-student networks, our second network is not guided by a loss component, but learns a simulation component that mimics the output of the missing branch.

2.3 A Privileged Information-Aware Neural Network

We present a dual branch neural network architecture for demand forecasting. It incorporates non-plannable features as privileged information (PI) at training time, with a first branch (the *historical branch*) that embeds historical information and plannable features via dilated causal convolutional layers, and a second branch, the *PIBranch*, which leverages fully-connected feed-forward layers to predict sales based on privileged information. At prediction time, we apply a slightly different single branch network with a simulation component to mimic the behavior of the PIBranch, whose inputs are

not available at prediction time.

2.3.1 Problem statement

The goal of a demand forecasting model is to predict the amount of sales in the future. Many different settings exist for building a forecasting model. Without loss of generality, we consider the case of multiple stores and multiple items, and forecast the demand of each item per store. Formally, for an arbitrary target product in a target store, the goal of the forecasting model is to predict

$$\{\hat{y}_t\}_{t=T+1}^{T+\Delta} = \{\hat{y}_{T+1}, \ldots, \hat{y}_{T+\Delta}\},$$

where T is the length of history being considered, Δ is the forecast horizon, and $\hat{y}_t \in \mathbb{R}$ denotes the predicted sales of the target item in the target store at time t. Future sales are affected by both the history of sales in the past, and other influential features. Influential features can be divided into two categories. *Plannable features* are known both for the past and the future; they can be static, such as product-dependent features like category and brand, or time-dependent, such as promotional campaigns and calendar events. Future values of *non-plannable features* are not known at the time of forecast; examples include behavior data from users on an online shopping website, sales of similar items in the same store, or sales of the same item in other stores.

We design a forecasting model that incorporates both types of feature alongside the historical sales. Formally:

$$\{\hat{y}_t\}_{t=T+1}^{T+\Delta} = F\big(\{y_t\}_{t=1}^{T}, \{\mathbf{x}_t^p\}_{t=1}^{T+\Delta}, \{\mathbf{x}_t^{np}\}_{t=1}^{T}\big), \tag{2.1}$$

where $F(\cdot)$ is a non-linear mapping function that we learn, and

$$\{y_t\}_{t=1}^{T} = \{y_1, y_2, \ldots, y_T\} \tag{2.2}$$
$$\{\mathbf{x}_t^p\}_{t=1}^{T+\Delta} = \{\mathbf{x}_1^p, \mathbf{x}_2^p, \ldots, \mathbf{x}_{T+\Delta}^p\} \tag{2.3}$$
$$\{\mathbf{x}_t^{np}\}_{t=1}^{T} = \{\mathbf{x}_1^{np}, \mathbf{x}_2^{np}, \ldots, \mathbf{x}_T^{np}\} \tag{2.4}$$

denote the historical sales of the target item, and the corresponding plannable features $\mathbf{x}^p \in \mathbb{R}^n$ and non-plannable features $\mathbf{x}^{np} \in \mathbb{R}^m$ for the item, respectively, with the corresponding feature dimensions n and m.

For training our model, we adopt the 'walk-forward' training schema that is a common choice for time series data [36, 80, 135]. In this approach, which we illustrate in Fig. 2.1, we apply a sliding window to divide the data into history and forecast horizon, and shift this sliding window Δ steps from training time to prediction time. In other words, assuming the length of the whole dataset is $T + 2\Delta$ and $T \gg \Delta$ is the length of the sliding window, the $[1, T]$ interval is used as the history training time, the $[T+1, T+\Delta]$ interval is used as the forecast horizon at training time (i.e., the validation window), the $[\Delta+1, T+\Delta]$ interval is used as the history at prediction time and finally, the $[T+\Delta+1, T+2\Delta]$ interval is used as the forecast horizon at prediction time.

We evaluate our model on the forecast horizon at prediction time, for which the non-plannable features are not available. For training and hyperparameter selection, we leverage the history and the validation window, for which non-plannable features

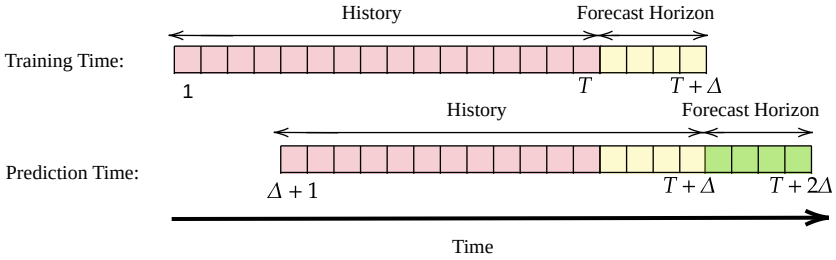


Figure 2.1: The 'walk-forward' training scheme. A history of length T is used for training, and validation is performed on $[T+1, T+\Delta]$. For predication time, the history is shifted Δ steps, and $[T+\Delta+1, T+2\Delta]$ is used as forecast horizon.

are known. This assumption is inline with real world cases, where a forecasting model is trained on the past data, for which the values of all influential features are already known. We rephrase Eq. 2.1 into Eq. 2.5 at training time and into Eq. 2.6 at prediction time in order to account for this setup:

$$\{\hat{y}_t\}_{t=T+1}^{T+\Delta} = F_1\left(\{y_t\}_{t=1}^{T}, \{\mathbf{x}_t^p\}_{t=1}^{T+\Delta}, \{\mathbf{x}_t^{np}\}_{t=1}^{T+\Delta}\right) \tag{2.5}$$

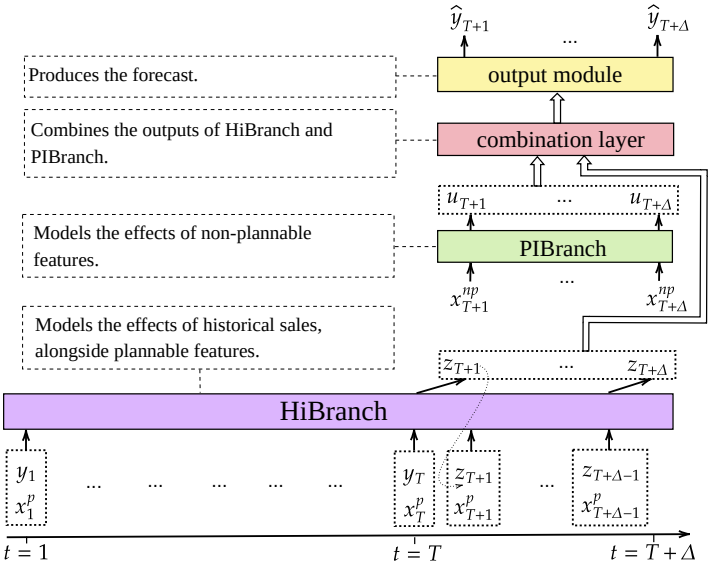
$$\{\hat{y}_t\}_{t=T+\Delta+1}^{T+2\Delta} = F_2\left(\{y_t\}_{t=\Delta+1}^{T+\Delta}, \{\mathbf{x}_t^p\}_{t=\Delta+1}^{T+2\Delta}\right), \tag{2.6}$$

where $F_1(\cdot)$ is the function that we learn at training time and $F_2(\cdot)$ is the function that we apply at prediction time to produce the forecast.

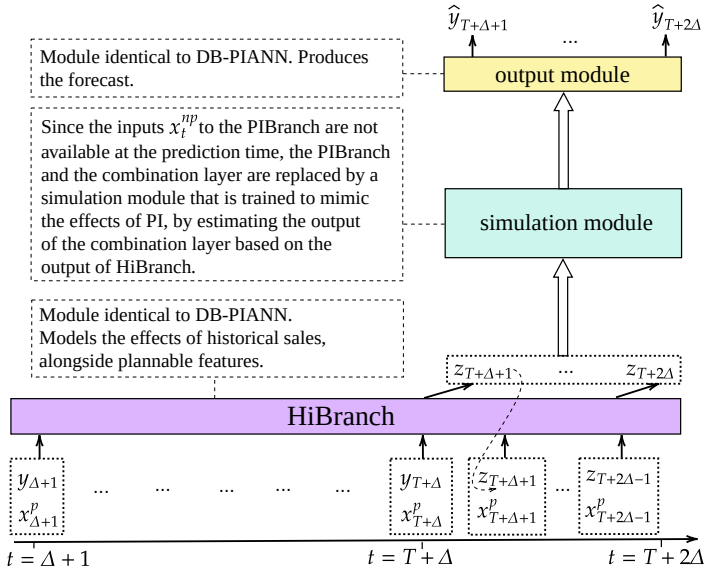
2.3.2 Architecture overview

We model non-plannable features as privileged information (PI), i.e., information that is available at the training time but not available at prediction time. This approach requires different forecasting models for training and prediction time. We therefore propose a *Dual Branch PI Aware Neural Network* (DB-PIANN) to embed both the historical sales and PI at training time, and a *Single Branch PI Aware Neural Network* (SB-PIANN) to produce the forecast at prediction time. Fig. 2.2a illustrates the architecture of DB-PIANN. Historical sales and plannable features are fed into one branch of the network, while privileged information is fed into a different branch. The outputs of these two branches are then merged with a combination layer, and fed into the output module to produce the forecast. The unavailablity of the PI for the future implies that we cannot produce a forecast with DB-PIANN at prediction time. However, only the input to the privileged information branch (PIBranch) is missing at prediction time, and we can still utilize the historical branch (HiBranch) of DB-PIANN.

We tackle this challenge by *training an additional simulation network to mimic the behavior of the missing PI Branch*. This network takes the output of the historical branch as input, and learns to reproduce the output of the combination layer. Fig. 2.2b details the architecture of the SB-PIANN. The difference to DB-PIANN is that the PIBranch and the combination layer are replaced with the simulation network; the other branch is identical. DB-PIANN and SB-PIANN model $F_1(\cdot)$ and $F_2(\cdot)$ in Eq. 2.5 and Eq. 2.6, respectively.



(a) Dual Branch PI Aware Neural Network (DB-PIANN) (dual branch)



(b) Single Branch PI Aware Neural Network (SB-PIANN) (single branch)

Figure 2.2: Neural network architectures for PI-aware demand forecasting. Fig. 2.2a illustrates the architecture of DB-PIANN applied at training time and Fig. 2.2b details the architecture of the SB-PIANN, which we leverage at prediction time.

2.3.3 Architecture details

Next, we introduce the details of DB-PIANN and SB-PIANN. As illustrated in Fig. 2.2a, DB-PIANN consists of four modules: 1) a *historical branch* (HiBranch), which is responsible for modeling the effects of historical sales data, along with plannable features, 2) a *privileged information branch* (PIBranch), which takes care of non-plannable features, 3) a *combination layer*, which combines the outputs of the two previous branches, and 4) an *output module*, which produces the final forecasts.

Based on these definitions, we break up Eq. 2.5 as follows:

$$\{\hat{y}_t\}_{t=T+1}^{T+\Delta} = F_1\big(\{c_t\}_{t=T+1}^{T+\Delta}\big), \tag{2.7}$$

where F_1 is the output module, and $\{c_t\}_{t=T+1}^{T+\Delta}$ is the output of the combination layer, defined as:

$$\{c_t\}_{t=T+1}^{T+\Delta} = \{z_t + u_t\}_{t=T+1}^{T+\Delta}, \tag{2.8}$$

where $\{z_t\}_{t=T+1}^{T+\Delta}$ and $\{u_t\}_{t=T+1}^{T+\Delta}$ are the outputs of the HiBranch and PIBranch, respectively (Fig. 2.2a).

Historical branch. The input of this branch, namely N_1, consists of the historical sales of an item y_t with of length T, and the plannable features $\mathbf{x_t^P}$ of length $T + \Delta$ corresponding to that item. The output of this branch, z_t, has the length of Δ, and will be fed to the combination layer. Formally:

$$\{z_t\}_{t=T+1}^{T+\Delta} = N_1\big(\{y_t \oplus \mathbf{x_t^P}\}_{t=1}^{T}, \{\mathbf{x_t^P}\}_{t=T+1}^{T+\Delta}\big), \tag{2.9}$$

where $\oplus$ denotes the concatenation operation. With this definition, HiBranch can be applied in SB-PIANN at prediction time, where we shift the time Δ steps. However, one cannot include non-plannable features as well at prediction time, since their values are unknown for the forecast horizon.

We choose a stack of dilated causal 1D convolution layers [146] for the historical branch, followed by two fully-connected feed-forward layers to produce the final output of the branch. Formally:

$$N_1\big(\{y_t \oplus \mathbf{x_t^P}\}_{t=1}^{T}, \{\mathbf{x_t^P}\}_{t=T+1}^{T+\Delta}\big) = W_2^h \mathrm{ReLU}(W_1^h c_{L_1} + b_1^h) + b_2^h, \tag{2.10}$$

where c_{L1} is the output of the last convolutional layer, and W_1^h, b_1^h, W_2^h, and b_2^h are the weights and biases of the first and second feed-forward layer.

Causal convolutions are typically faster to train compared to RNNs and LSTMs (the common choice for sequence to sequence problems), since they do not have any recurrent connections [17]. Dilated convolutions allow us to process a long sequence without exponentially increasing the number of layers, by skipping input values with a certain step size. This is especially useful for the case of demand forecasting, where the history of the data is relatively long. More formally, for a sequence input x and a filter f with size k, a causal dilated convolution operation g on element s of the sequence is defined as:

$$g(s) = (x *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot x_{s-d \cdot i}, \tag{2.11}$$

where d is the dilation factor. Following [146], we increase d exponentially with the depth of the network, i.e., $d = O(2^j)$ at level j of the network, where $j \in \{0, \ldots, L_1 - 1\}$ and L_1 denotes the number of layers. All parameters are shared across the whole forecast horizon.

The historical branch N_1 operates on a rolling basis, meaning that the output $z_t \in \mathbb{R}$ is produced one step at a time and is concatenated to the history in the input, which is is then shifted one step forward, and fed back into the first convolutional layer, until the end of the forecast horizon is reached (see Fig. 2.2a).

Privileged information branch. The input of this branch, namely N_2, are the non-plannable features with a length of Δ, and the output of this branch, u_t, (also of length Δ), will be fed to the combination layer. To make use of all the available training data, we apply a sliding window of length Δ and move it forward one step at a time, until we reach the end of the training interval, i.e., $T + \Delta$. For the PIBranch N_2, we leverage fully-connected feed-forward layers with dropout [134] applied after each hidden layer. We use a ReLU activation function for the hidden layers, and a linear fully-connected layer for the output. Formally:

$$\{u_t\}_{t=i}^{i+\Delta} = N_2(\{\mathbf{x^{np}}_t\}_{t=i}^{i+\Delta}) = W_{L_2}^{np}\mathrm{ReLU}(W_l^{np}x_l + b_l^{np}) + b_{L_2}^{np}, \tag{2.12}$$

where $1 \leqslant i \leqslant T$ is the start of the sliding window, L_2 denotes the number of layers, and W_l^{np} and b_l^{np} are the weights and biases of the l-th layer for $l \in \{1, \ldots, L_2\}$; x_l is the input of the l-th layer, which is the output of the previous layer for $l \in \{2, \ldots, L_2\}$, and equal to $\{\mathbf{x^{np}}_t\}_{t=T+1}^{T+\Delta}$ for $l = 1$.

Note that the PIBranch cannot be used in SB-PIANN at prediction time, since its input values are unknown for the forecast horizon.

Output module. In SB-PIANN, this module consumes the output of the combination layer, and produces the predicted sale values for the validation period. We again apply fully-connected feed-forward layers:

$$\{\hat{y}_t\}_{t=T+1}^{T+\Delta} = F_1(\{u_t + z_t\}_{t=T+1}^{T+\Delta}) = W_{L_3}^{o}\mathrm{ReLU}(W_l^{o}s_l + b_l^{o}) + b_{L_3}^{o}, \tag{2.13}$$

where W_l^o and b_l^o are the weights and biases of the l-th layer for $l \in \{1, \ldots, L_3\}$, s_l is the input of l-th layer, which is the output of the previous layer for $l \in \{2, \ldots, L_3\}$, and equal to $\{u_t + z_t\}_{t=T+1}^{T+\Delta}$ for $l = 1$.

Together, the definitions of the aforementioned modules complete the architecture of DB-PIANN, according to Eq. 2.7.

SB-PIANN. Next, we define our second architecture SB-PIANN, which is going to produce the forecast at prediction time. We break down Eq. 2.6 as follows:

$$\begin{aligned}\{\hat{y}_t\}_{t=T+\Delta+1}^{T+2\Delta} &= F_2\left(\{y_t\}_{t=\Delta+1}^{T+\Delta}, \{\mathbf{x^p}_t\}_{t=\Delta+1}^{T+2\Delta}\right) \\ &= F_1\left(S(\{z_t\}_{t=T+\Delta+1}^{T+2\Delta})\right) \\ &= F_1\left(S(N_1\big(\{y_t \oplus \mathbf{x_t^p}\}_{t=\Delta+1}^{T+\Delta}, \{\mathbf{x_t^p}\}_{t=T+\Delta+1}^{T+2\Delta}\big))\right),\end{aligned} \tag{2.14}$$

where S is the simulation module, which we define in Eq. 2.15 below, N_1 refers to the historical branch defined in Eq. 2.10, and F_1 depicts the output module, defined in Eq. 2.13.

Simulation module. The purpose of the *simulation module* is to replace the missing PIBranch of DB-PIANN. As we cannot use the PIBranch at prediction time, one of the inputs of the combination layer is not available at prediction timeas well (see Fig. 2.2). Therefore, we train a feed-forward neural network to estimate the output of the combination layer based on the output of HiBranch. Formally:

$$S(\{z_t\}_{t=T+\Delta+1}^{T+2\Delta}) = W_{L_4}^s \mathrm{ReLU}(W_l^s p_l + b_l^s) + b_{L_4}^s, \tag{2.15}$$

where W_l^s and b_l^s are the weights and biases of the l-th layer for $l \in \{1, \ldots, L_4\}$. p_l is the input of l-th layer, which is the output of the previous layer for $l \in \{2, \ldots, L_4\}$, and equal to $\{z_t\}_{t=T+\Delta+1}^{T+2\Delta}$ for $l = 1$.

2.3.4 Learning process

Many options exist for training a forecasting model for a collection of time series, such as the sales per item per store in our case. Following previous work [36, 127], we train a single model for all of the items; each training sample contains the sales of a single item in a single store, along with its corresponding influential features.

We train the PIBranch N_2 and the historical branch N_1 separately, and then train DB-PIANN using the learned models, as outlined in the previous section. We subsequently train the simulation network S in isolation. With this scheme, SB-PIANN does not actually need training and can be composed from the previously learned modules, i.e., HiBranch, the simulation module and the output module. We leverage the mean absolute error between the predicted values and the actual values as objective function to train all the components of our architecture. All of our proposed modules are smooth and differentiable, which allows us to learn their parameters by standard back propagation. With mean absolute error as the objective function, we define the loss for each module as follows:

$$\mathcal{L} = \frac{1}{N}\Big(\sum_{n=1}^{N}\sum_{t=T+1}^{T+\Delta} \mid y_{nt}^a - y_{nt}^p \mid \Big), \tag{2.16}$$

where N is the number of training samples, n is an index for the samples, y_{nt}^a is the label for sample n at time t, and y_{nt}^p is the output of the corresponding module. Specifically, when training the HiBranch, PIBranch, and output module, y_{nt}^a is equal to the actual sales of the training sample n at time t, and y_{nt}^p is equal to the corresponding value of z_t, u_t, and $\hat{y}_t$ for sample n. For the simulation module, y_{nt}^a translates to the corresponding values of $z_t + u_t$ of sample n.

2.4 Experimental Setup

Datasets. Unfortunately, most of the sales datasets from existing work are not publicly available. For example, we could neither obtain the 'One Stop Warehouse' dataset from [36], the Amazon Demand Forecasting dataset from [155], nor the JD50K Online Sales Forecasting dataset from [50].

Table 2.1: Dataset statistics.

Dataset	Items	Stores	#time series	Time series' length
Favorita	1,656	54	11,614	365
Dunnhumby	1,101	26	8,825	117

We therefore evaluate the performance of SB-PIANN on two publicly available datasets to ensure reproducibility. Both datasets contain sales numbers at the product and store level; we therefore set the goal of predicting the sales per product per store for a certain forecast horizon. Here, for a target item in a target store, we treat its sales in other stores as the privileged information. The *Favorita* dataset contains daily sales of thousands of items across 54 stores located in Ecuador.[1] We use the data from the 15th of August 2016 to the 15th of August 2017, and only consider items that have less than five days of sales data missing.[2] We also use the *The Complete Journey* dataset published by *Dunnhumby*.[3] This dataset contains around 300 million transactions for $\sim$5,000 items across $\sim$760 distinct stores, spanning more than two years of history. We randomly select a subset of items and stores for our experiments, and aggregate sales on a weekly basis to reduce the sparsity. The statistics of the datasets are shown in Table 2.1. Not all of the items are sold in all of the stores, so the total number of time series, i.e., training samples, is different from the number of items multiplied by the number of stores.

Influential features. Our design of the network architecture does not restrict the types of influential features that our approach can incorporate. However, we work with non-plannable features only in the experiments, as their effect on the forecast is the focus of this chapter. For such non-plannable features, we rely on the sales of an item in other stores. In many demand forecasting datasets, including both of the datasets that we use, the retail company has more than one store, and the sales of a target item in other stores can help to forecast the demand in the target store. Formally:

$$\mathbf{x^{np}}_t = \{(x_t^{s_0}, x_t^{s_1}, \dots, x_t^{s_n})\}, \tag{2.17}$$

where n is the number of stores and $x_t^{s_i}$ denotes the sales for the target item in store s_i at time t.

Training and testing. Analogous to previous work, we use the walk-forward strategy [36, 135] for the train-test split, as detailed in Section 2.3.3. Following this strategy, we split the data according to the time dimension, and not based on stores and items. Given a time series with a total size of T for the history of sales data and Δ steps to predict, we use $[1, T]$ as the training data, $[T+1, T+\Delta]$ for validation and $[T+\Delta+1, T+2\Delta]$ for testing (as illustrated in Fig. 2.1). We experiment with $\Delta \in \{2, 8, 16\}$.

[1] `https://www.kaggle.com/c/favorita-grocery-sales-forecasting/data`

[2] The same dataset is used in [36], however the authors do not mention how they created a subset of the original dataset in their paper. Nevertheless, we achieve a comparable performance with their method on our version of the data.

[3] `https://www.dunnhumby.com/careers/engineering/sourcefiles`

Implementation details. Our models are implemented using the Keras framework [41] with TensorFlow as backend [1]. We use mini-batch stochastic gradient descent (SGD) together with the Adam optimizer [76] to train the models. We set the batch size to 32, and stop training when the loss on the validation set converges. All the methods run on a Linux server with an Intel Xeon CPU, and a GeForce GTX 980 (Maxwell GM204) GPU. The GPU code is implemented using CUDA 9.

Metrics. We consider two metrics for evaluation: mean absolute error (MAE) and symmetric mean absolute percentage error (SMAPE):

$$\textbf{MAE} = \frac{1}{N \times \Delta} \sum_{n=1}^{N} \sum_{t=T+\Delta+1}^{T+2\Delta} |y_t - \hat{y}_t|, \tag{2.18}$$

$$\textbf{SMAPE} = \frac{100}{N \times \Delta} \sum_{n=1}^{N} \sum_{t=T+\Delta+1}^{T+2\Delta} \frac{|y_t - \hat{y}_t|}{(|y_t| + |\hat{y}_t|)/2}, \tag{2.19}$$

where y_t and $\hat{y}_t$ denote real and predicted sales, respectively.

Parameter settings. For all of the parameters of the networks, we conduct a grid search and leverage the parameters with the best performance on the validation set. For the feed-forward modules, we conduct a grid search over $\{1, 2, 3, 4\}$ for the number of hidden layers, $\{16,\ 32,\ 64,\ 128,\ 256,\ 512\}$ for the number of nodes in each hidden layer, and $\{0.1, 0.2, 0.3, 0.4\}$ for the dropout rate. For the convolutional layers, we search over $\{16, 32, 64\}$ for the number of filters, $\{2, 4, 8, 16\}$ for the filter size, and $\{4, 5, 6, 7, 8\}$ for the number of layers.

2.5 Experimental Results

We investigate the ability of SB-PIANN to capture the effects of privileged information in Section 2.5.1, and analyse its performance compared to the state-of-the-art for demand forecasting in Section 2.5.2.

2.5.1 Capturing the effects of privileged information

Our first set of experiments addresses the following research question: *To what extent is the proposed model, SB-PIANN, able to capture the effect of privileged information?* We conduct experiments with different components of our proposed model to answer our first research question. The purpose of these experiments is to reveal the importance of privileged information and to quantify its impact on the final performance of SB-PIANN. We compare its performance to that of the HiBranch and PIBranch in isolation, as well to the "oracle" performance of DB-PIANN. DB-PIANN's performance is only reported for the sake of comparison; the model cannot be used for prediction in real world cases due to the unavailability of the values of non-plannable features at prediction time.

Results and discussion. Table 2.2 shows the performance of HiBranch, PIBranch, SB-PIANN and DB-PIANN in terms of MAE and SMAPE, on testing time intervals, i.e., $[T + \Delta + 1, T + 2\Delta]$. The results are reported for all Δ settings on the Favorita and Dunnhumby datasets. We compare the performance of two different branches

Table 2.2: Performance of SB-PIANN and DB-PIANN and their components on the Favorita and Dunnhumby datasets. * indicates that a component is significantly outperformed by SB-PIANN; $^-$ indicaties that there is no significant difference between DB-PIANN and SB-PIANN (student t-test, $\alpha = 0.05$).

Method	$\Delta = 2$		$\Delta = 8$		$\Delta = 16$	
	MAE	SMAPE	MAE	SMAPE	MAE	SMAPE
Favorita dataset						
HiBranch	6.126*	38.840*	6.494*	38.563*	7.278*	38.794*
PIBranch	6.415*	38.235*	6.664*	39.692*	7.080*	38.431*
SB-PIANN	6.069	37.139	6.277	38.102	6.868	38.241
DB-PIANN	5.985$^-$	36.956$^-$	6.052$^-$	37.304$^-$	6.752$^-$	37.314$^-$
Dunnhumby dataset						
HiBranch	3.593	43.042*	3.661*	44.348*	4.079*	50.289*
PIBranch	3.669*	44.108*	3.640	44.006	3.691	45.059
SB-PIANN	3.558	42.567	3.612	43.506	4.059	49.540
DB-PIANN	3.555$^-$	42.531$^-$	3.601$^-$	43.384$^-$	3.899$^-$	47.817$^-$

for different forecast horizons. We observe that the performance of the PIBranch is comparable to that of the HiBranch for shorter forecast horizons, i.e., 2 and 8, and outperforms HiBranch for the longest forecast horizon. This shows that the PIBranch is capable of predicting the sales at least as well as HiBranch, and is more robust with respect to forecast horizon. We attribute the ability of PIBranch to predict future sales to two aspects: First, we note that the historical sales data are not taken into account in the PIBranch, and the forecast is produced solely based on the privileged information. This indicates the usefulness of this information for demand forecasting. Second, the obtained performance points out the effectiveness of the proposed network to model the effects of privileged information. In other words, a deep feed-forward neural network is capable of producing a forecast based on the sales of products in other stores, and this forecast is comparable to and in some cases superior to the one based on the historical sales.

We also report the performance of DB-PIANN, while noting (again) that it cannot be used in a real world setting, as it requires the privileged information, which is not available at prediction time. DB-PIANN outperforms individual branches on both datasets and for all of the forecast horizons, which shows that it is able to leverage both the historical sales and the privileged information to produce the final forecast. The performance gain also indicates that both branches contribute positively to the final forecast; their contribution is not overlapping, and they are both essential to achieve the highest performance.

Finally, we observe that the performance of SB-PIANN is better than that of HiBranch and is relatively close to DB-PIANN, on both datasets and for all forecast horizons. Recall that SB-PIANN is not using the inputs of the PIBranch, and indirectly models the effects of privileged information via the simulation component. The close

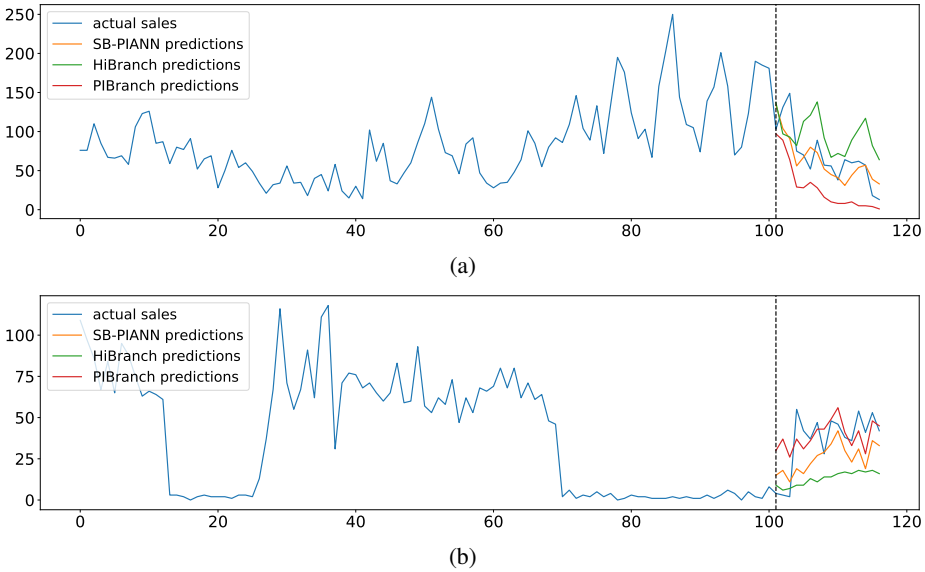


Figure 2.3: Forecasts produced by HiBranch, PIBranch and SB-PIANN, along with the actual sales, for sample products. 2.3a is taken from Favorita and 2.3b is selected from Dunnhumby. y axis shows the sales and x axis is the time. 2.3a and 2.3b show the improved forecasts using PI.

performance of SB-PIANN to that of DB-PIANN shows that the proposed approach is able to embed the PI from train to test time effectively. In other words, the simulation component is capable of transferring the effects of privileged information absorbed in the train time to the prediction time. The superiority of SB-PIANN compared to the HiBranch supports the hypothesis that utilizing the privileged information leads to a better performance than a forecast that only relies on the historical data.

To gain more insights into forecasts by our proposed method, we visualize a few examples in Fig. 2.3. Each figure shows the demand for a product in a store, taken from both Favorita and Dunnhumby. We show 101 days of history for both datasets. The history of sales as well as the actual sales in the future are plotted, along with forecasts made by PIBranch, HiBranch, and SB-PIANN. We observe that the forecast based on PI has a different trend from the one based on the history of sales. This might be a result of an event that affects the sales of the product in the stores, and leads to a better prediction for the future sales, compared to the forecast of HiBranch, which is based on the history of sales in the target store. We can also recognize that SB-PIANN picks-up on this difference, and its forecast is superior to that of HiBranch.

2.5.2 Comparison to existing approaches for demand forecasting

Next, we focus on the following question: *How effective and accurate is SB-PIANN for the task of demand forecasting?* We compare SB-PIANN to the following state-of-the-

art neural network-based forecasting models:

DA-RNN [116]. This dual-stage attention-based RNN method is proposed for time series prediction. It uses an encoder-decoder structure with two different attention mechanisms. In this model, the PI are available also for the future. However, this is an unrealistic assumption; we use this model as the baseline to evaluate the ability of our model in making use of PI.

LSTNet [81]. The long- and short-term time-series network uses both CNNs and RNNs to capture both short-term and long-term trending patterns of the time series. It also has an auto-regressive component. The architecture is proposed for multi-variate TSF, therefore we build a model for each item and forecast for the target stores simultaneously. For LSTNet, we only make use of the historical values of PI for both training and test time.

TADA [36]. An encoder-decoder based architecture that uses two LSTM branches for encoding different types of features. Since the method was tested on the Favorita dataset, we also report their results based on their original division of features for this dataset. For the Dunnhumby dataset, we randomly divide the data from other stores into two categories. We again only make use of the historical values of PI available for both training and test time.

All baselines report superior performance compared to auto-regressive models, decision tree models, and simpler neural network-based approaches. We therefore omit these methods from our evaluation. Aside from their state-of-the-art performance, we chose our baselines to cover a wide range of possible approaches. Specifically, they apply different training schemes; DA-RNN trains a model per time-series, LSTNet trains a model per store, and TADA uses an approach similar to ours, training a single model for all of the data. They also differ in the ways they treat the PI. For DA-RNN, the assumption is that these features are available for both history and the future (which is not true in real-world settings), while the other two only use the historical values of the PI.

Implementation details. For the baseline implementation, we asked the authors of TADA and DA-RNN for the code and they kindly provided us with their own implementation. For LSTNet, we leverage the code which is made publicly available by the authors.[4] We use the same testing environment as SB-PIANN for the baselines, as outlined in Section 2.4.

Results and discussion. Table 2.3 shows the performance of SB-PIANN compared to the baselines for different forecast horizons on Favorita and Dunnhumby, respectively. The best performance is highlighted with bold face. In almost all cases, SB-PIANN outperforms the baselines in terms of MAE and SMAPE. We observe that the performance of all methods starts to drop with the increase of the forecast horizon, but SB-PIANN is more robust with respect to the length of the forecast horizon. In all cases, the performance is better on Favorita than on Dunnhumby in terms of SMAPE. This might be due to the fact that the sales data in Dunnhumby is more scarce.

According to the results, SB-PIANN even outperforms DA-RNN, which makes the unrealistic assumption of having the PI available at prediction time. The fact

[4] https://github.com/laiguokun/LSTNet

Table 2.3: Comparison of neural network-based forecasting methods on the Favorita and Dunnhumby datasets. * indicates that a method is significantly outperformed by SB-PIANN (student t-test, $\alpha = 0.05$).

Method	Uses PI		$\Delta = 2$		$\Delta = 8$		$\Delta = 16$	
	Train	Test	MAE	SMAPE	MAE	SMAPE	MAE	SMAPE
Favorita dataset								
DA-RNN	+	+	9.343*	48.864*	8.583*	46.174*	8.709*	43.132*
LSTNet	−	−	6.619*	40.509*	7.481*	43.920*	8.332*	45.016*
TADA	−	−	6.428*	**36.744**	7.431*	41.389*	8.463*	43.165*
SB-PIANN	+	−	**6.069**	37.139	**6.277**	**38.102**	**6.868**	**38.241**
Dunnhumby dataset								
DA-RNN	+	+	4.535*	52.779*	3.995*	47.154*	4.168*	**48.942**
LSTNet	−	−	4.134*	48.484*	4.818*	55.310*	5.673*	59.864*
TADA	−	−	4.367*	49.018*	5.777*	63.834*	8.251*	78.055*
SB-PIANN	+	−	**3.558**	**42.567**	**3.612**	**43.506**	**4.059**	49.540

that SB-PIANN outperforms even DA-RNN shows the effectiveness of our proposed approach to leverage privileged information. The lower performance of DA-RNN might also be a result of its training scheme, i.e., building a forecast model per time series. In this scenario, the model cannot learn from the similarity and differences between different products and stores.

While the special type of PI that we experiment with, i.e., sales in other stores, suggests the use of a multi-variate TSF model, our experiments show that compared to LSTNet, a state-of-the-art model proposed for the multi-variate TSF, SB-PIANN, performs better in all cases. On the Favorita dataset, we compare the performance of SB-PIANN with the original version of TADA, i.e., using the definition that the authors propose for internal and external features. In this version, a set of 13 attributes are used as features, such as the location of the stores and the oil price. In our approach, we only rely on the sales of the items in other stores as features, and we observe that our model performs significantly better in terms of both error metrics, while using no manual categorization of features. The gain in performance is more significant on the Dunnhumby dataset, which indicates the limitation of TADA when the division of features into internal and external as required by TADA is not straightforward. In other words, the performance of TADA degrades when influential features cannot be characterized as internal and external, and accordingly, cannot be fed into the corresponding LSTM layers.

2.6 Conclusion

Demand forecasting is a fundamental problem in the replenishment process of retail companies. Models for forecasting the demand for a product usually consider patterns in the history of sales data and influential features. Such influential features can be

divided into two categories: plannable features that are known for the past and the future, and non-plannable features, for which the future values are unknown. Neural forecasting models usually ignore non-plannable features when predicting the amount of sales in the future.

In this chapter, we answered **RQ1**: we consider non-plannable features as privileged information and design a novel neural network to utilize them. We propose two different network architectures for training and prediction time. At the time of training, the network has two different branches to model the effect of historical sales and non-plannable features. At prediction time, the second branch is not available, and is replaced by a simulation network that is trained to mimic its behavior. We extensively evaluate our proposed approach on two real-world forecasting datasets, and find that it outperforms state-of-the-art baselines in terms of mean absolute error and symmetric mean absolute percentage error metrics.

While our proposed architecture is capable of leveraging plannable features, our focus in this chapter is on modeling non-plannable features. In future work, it will be appealing to study different approaches to incorporate plannable features in the model; aside from treating them as an extra dimension of the historical sales, they can be fed into the PIBranch, or an extra dedicated branch. Moreover, although we make no specific assumptions about the type of non-plannable features in our model, we rely on a single type of non-plannable features in our experiments. We also aim to investigate the impact of more sources of privileged information.

In the next chapter, we continue our study on multi-channel retail with a focus on understanding user behavior and next basket recommendation.

3
Multi-Channel Customer Behavior

In this chapter we stay in the same as domain as Chapter 2, that is, multi-channel retail, but we change our task settings from demand forecasting to understanding user behavior and next basket recommendation. We address **RQ2**: What are the characteristics of user behavior in multi-channel retail settings? We conduct a descriptive study on a real-world sample of transactions from large retailer to understand user behavior in this scenario.

3.1 Introduction

The emergence of e-commerce in recent years has encouraged retailers with physical stores to provide the possibility of online shopping for their customers in addition to in-store (offline) shopping. Customers do not only buy goods via multiple shopping channels, but they can also leverage the online channel for exploring the product inventory, comparing products, and saving products for later purchases, before shopping offline. In addition, the data generated via online shopping provides a further opportunity for personalizing the shopping experience through recommendation [2, 106].

The ease of use of online shopping translates into a rapidly growing market share of e-commerce solutions, even in industries such as fashion. This growth is not cannibalistic – most customers who purchase goods online also purchase in-store (offline) [31]. The addition of an online channel does not isolate the offline channel. Instead, it creates a multi-channel shopping experience for customers in retail sectors like grocery, cosmetics, and apparel.

Understanding customer behavior in retail serves as a basis for many downstream machine learnings tasks, such as recommending products and predicting purchases. While there are numerous studies examining user behavior in online shopping platforms, little is known about multi-channel customer behavior. Previous work relies mainly on click stream data [32, 60, 100, 141, 159, 162, 170]. Other sources of customer behavior include transaction data [151], digital receipts of online purchases extracted from emails [77], transaction logs of a bank [156], or search logs of a commercial

This chapter was published as M. Ariannezhad, S. Jullien, P. Nauts, M. Fang, S. Schelter, and M. de Rijke. Understanding multi-channel customer behavior in retail. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1–5, 2021*, pages 2867–2871. ACM, 2021.

product search engine [136]. However, these studies utilize data from a single shopping channel only, and do not explore multi-channel customer behavior. So far, multi-channel customer behavior has mostly been studied in the marketing and retail research literature [2, 31, 55, 68, 69]. These studies rely on perceptions gathered via interviews and customer surveys, in order to model, e.g., lock-in effects or physical store surface needs. Yet, perceptions are often different from actions, and these works do not consider actual transaction data from customers.

In this chapter, we provide the first study on multi-channel customer behavior in retail. Based on a sample of 2.8 million transactions from 300,000 customers, gathered from a food retailer with multiple physical stores and two online platforms, we provide a first picture of customer behavior in a multi-channel retail setting. To this end, we group the customers into three groups, namely *online-only*, *offline-only*, and *multi-channel* customers, based on their choice of shopping channels (Section 3.2). We first compare the shopping behavior of these customer groups in Section 3.3. We find that the tendency to purchase previously bought products, defined as repeat behavior ratio, is higher for online-only customers. Zooming in on multi-channel customers, our analysis reveals that there is little overlap in online and offline baskets of multi-channel customers; they use each channel for different sets of items. Our analysis further indicates that online baskets are larger, and contain items from more diverse product categories.

The observed differences between customer groups can affect the performance of models that are designed for downstream prediction tasks based on customer behavior (Section 3.4). As a case study, we investigate the Next Basket Recommendation (NBR) task, where the goal is to predict items that a customer will purchase in their future basket, given their previous shopping baskets. Existing approaches to NBR rely on data from a single shopping channel only, and do not consider multi-channel settings [66, 148, 165]. We examine the performance of a standard NBR model, namely PersonalTopK, for different customer groups and different channels of the future basket. NBR performance differs significantly for different customer groups, with the online-only customers receiving the best and offline-only customers receiving the worst overall performance. For multi-channel customers, the performance heavily depends on the channel type for the target basket, and choosing the correct target channel has the potential to boost NBR performance. Overall, our experiments indicate that the NBR task is not trivial for multi-channel settings; a single model is not able to achieve the same performance for different customer groups. Our findings serve as a call for follow-up research on designing recommendation models that explicitly model the characteristics of multi-channel retail.

In summary, we provide the following contributions.

- To the best of our knowledge, we provide the first study of multi-channel customer behavior in retail, based on a large sample of 2.8 million transactions from 300,000 customers of a food retailer in Europe with physical stores, an online shop and a mobile application (Section 3.2).
- Our analysis of the transaction logs of different customer groups indicates that online-only, offline-only, and multi-channel customers have different shopping behavior across multiple dimensions, such as repeat ratio and basket size (Section 3.3).

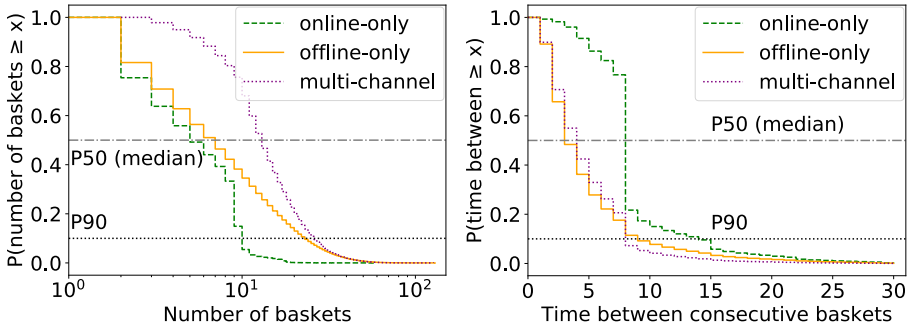


Figure 3.1: Cumulative distribution of the number of baskets (left) and the time between consecutive baskets (in days, right) per customer.

- In experiments with a standard next basket recommendation model, it underperforms for multi-channel customers (compared to online-only customers), indicating the need for approaches that explicitly model the multi-channel context (Section 3.4).

3.2 Dataset Description

Our work is based on proprietary transaction data from a large food retailer in Europe, with a number of physical stores, an online website, and a mobile application. The same product inventory is offered on all channels. Customers can get a loyalty card either in-store or online, and can use that card for their shopping across all channels. A customer can be tracked across offline and online channels if they use their loyalty card at the cashier in-store, and use the same card when buying online. A customer needs to be identified to fulfill the order in the online channel, while this is not the case for in-store shopping (offline).

Sampling of customers based on channel preferences. In order to understand multi-channel customer behavior, we select an eight week period of time as the *transaction period*, and define three groups of customers based on their channel preferences during the transaction period: 1) *offline-only customers* who conducted only offline transactions and no online transaction, 2) *online-only customers* who have only online transactions and no offline transactions, and 3) *multi-channel customers* who conducted both online and offline transactions.

Extraction of transaction data. We first filter the data to retain only customers with a loyalty card so as to be able to track customers, both within and across channels. Next, we sample 100,000 customers at random from each group, and extract their transactions during our period of interest. Each transaction in the data is marked as either online or offline, and represents a basket with one or multiple products, purchased by a customer. The sample of online-only customers has 500K corresponding transactions, while the sample for offline-only customers comprises of 900K transactions. The multi-channel customers undertook 1.4M transactions with a similar online/offline ratio.

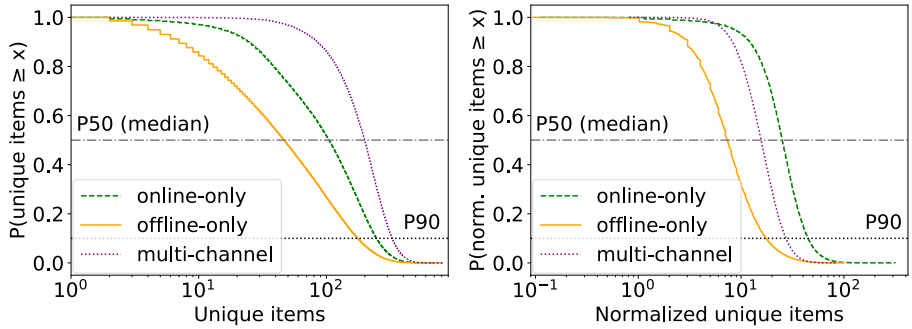


Figure 3.2: Cumulative distribution of the number of unique items (left), and normalized unique items (right) per customer.

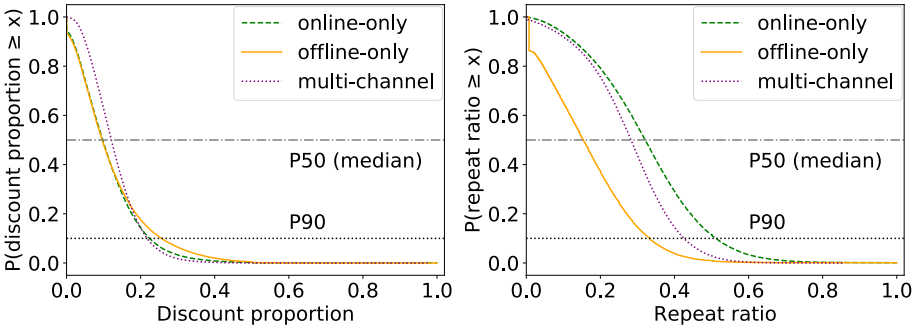


Figure 3.3: Cumulative distribution of the discount proportion (left) and the repeat ratio (right) for different groups of customers.

3.3 Understanding Customer Behavior

We first focus on similarities and differences in the behavior of all three customer groups, and then consider multi-channel customers. We also study the characteristics of online and offline baskets.

Comparison of customer groups. We consider the users' purchase frequency in both channels. Fig. 3.1 depicts the cumulative distribution of the number of baskets and the time between consecutive baskets (in days) for online-only, offline-only, and multi-channel customers. We observe that multi-channel customers have the highest number of baskets, and online-only customers have the lowest. Offline-only and multi-channel customers have a similar behavior with respect to shopping times, with a median of three days between consecutive baskets, and a 90% percentile of seven days. The distribution is different for online-only customers: the average time between consecutive baskets is longer, with a median of seven days. This seven day interval is frequent for a large number of online customers, partly because of the possibility of selecting a fixed day for delivery for a series of online baskets.

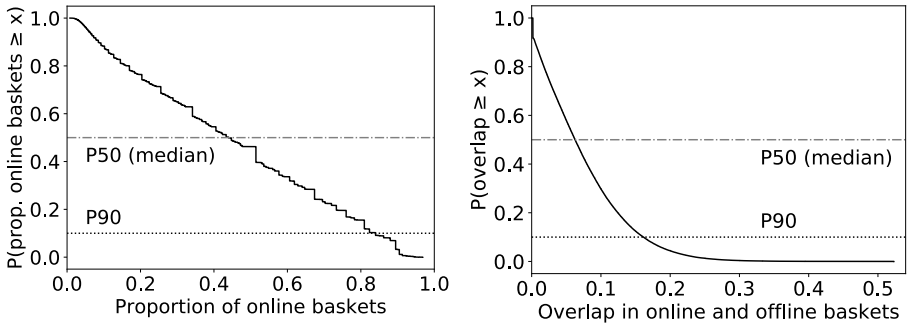


Figure 3.4: Cumulative distribution of the proportion of online baskets (left) and the overlap in items purchased online and offline (right) for multi-channel customers.

On top of the number of purchases, we also seek insights into the quantity and variety of items purchased. Fig. 3.2 shows the cumulative distribution of the number of unique items that a customer has purchased during their shopping history, and the normalized unique items, defined as the number of unique items divided by the number of baskets. Offline-only customers have the smallest number of unique items and normalized items; while multi-channel customers have the highest number of unique items, online-only customers have purchased more unique items when the number of baskets is considered. We conjecture that easy access to the full catalogue plus the ease of home delivery leads to this increase in diversity.

As the loyalty card allows for discounts, it is of interest to know how different customer groups behave with regard to promotions. Fig. 3.3 shows the cumulative distribution of the discount proportion (defined as the amount of discount per basket divided by basket value) and the repeat ratio (defined as the number of unique items divided by the total number of items purchased across baskets) for different groups of customers. All customers are similar w.r.t. the discount proportion, online-only customers have the highest repeat ratio, and offline-only customers have the lowest.

Behavior of multi-channel customers. Do multi-channel users prefer a shopping channel? Fig. 3.4 (left) depicts the cumulative distribution of the proportion of online baskets, defined as the number of online baskets divided by the total number of both online and offline baskets, per customer. For roughly half of the multi-channel customers, online baskets are in the majority, while for the other half offline baskets are dominant, so there is no clear preference.

Next, we want to know whether multi-channel customers use the two channels for the same purchases. Fig. 3.4 (right) plots the cumulative distribution of the overlap in items purchased online and offline for multi-channel customers, calculated as the Jaccard index of online and offline item sets. The overlap between online and offline baskets is minimal for the majority of customers; for 90% of them, the overlap is less than 0.16. This implies that while multi-channel customers purchase both online and offline baskets, they use each channel for purchasing a separate set of products.

Finally, we investigate when multi-channel customers are more susceptible to prefer

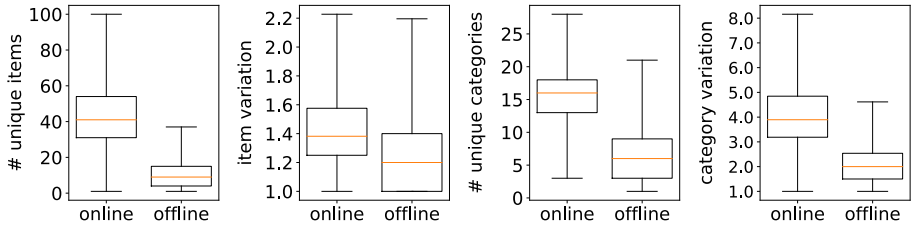


Figure 3.5: Distribution of the variation in terms of items and item categories for online and offline baskets.

one channel over the other. We find that offline shopping peaks on Fridays and Saturdays, while online shopping has a uniform distribution across week days. We also observe a decline in shopping on Sundays for both online and offline channels. We further look at the patterns underlying the selection of a channel for shopping. We consider the channel switch probability for multi-channel customers. The probability is exactly equal to 0.5 for 20% of customers; these customers may or may not select a different channel for their next basket with an equal chance. Roughly 40% of customers change their channel with a probability of less than 0.5, and 10% of customers change their channel almost after every basket. This indicates that predicting the next channel for the customer is not trivial.

Comparison of online and offline baskets. Each transaction in the data represents a basket of products purchased at a time by a customer. How do baskets purchased online compare to those purchased offline? Fig. 3.5 shows the distribution of the number of unique items and unique item categories in a basket, the basket variation (defined as the basket size divided by the number of unique items), and the basket category variation (defined as the basket size divided by the number of unique item categories) for online and offline baskets. While online baskets contain more unique items and more unique item categories, there is not much difference in basket variation and basket category variation; this is a hint that the difference in the distribution of the number of unique items and the number of unique item categories in online and offline baskets is mostly caused by the larger basket size in the online channel.

3.4 Next Basket Recommendation

We have observed important differences in purchasing behavior between the three groups of customers. We hypothesize that ignoring the shopping channel can hurt the performance of downstream machine learning tasks. In this section, we present a case study to investigate this hypothesis. In particular, we study the performance of a prominent Next Basket Recommendation (NBR) model in the multi-channel context. The goal of an NBR model is to predict the set of items that a customer will purchase in their next basket, given their purchase history [66, 148, 165]. Formally, given the history of baskets for customer u defined as $B^u = \{B_1^u, B_2^u, \ldots, B_n^u\}$, where B_i^u is a basket of items defined as $B_i^u = \{x_1, x_2, \ldots, x_t\}$, and $x_i \in X$ denotes an item from

Table 3.1: Experimental results for next basket recommendation using the PersonalTopK model.

	Prediction target	Recall	nDCG	PHR
k = 10	Online-only customers	0.1582	0.5873	0.9743
	Offline-only customers	0.1773	0.2716	0.7331
	Multi-channel customers	0.1282	0.3696	0.7688
	Multi-channel customers, online target basket	0.1431	0.5946	0.9816
	Multi-channel customers, offline target basket	0.1163	0.1891	0.5981
	Multi-channel customers, target channel known	0.1373	0.3808	0.7882
k = 20	Online-only customers	0.2459	0.4993	0.9882
	Offline-only customers	0.2435	0.2664	0.7998
	Multi-channel customers	0.1950	0.3292	0.8242
	Multi-channel customers, online target basket	0.2265	0.5068	0.9916
	Multi-channel customers, offline target basket	0.1697	0.1867	0.6899
	Multi-channel customers, target channel known	0.2027	0.3387	0.8369
k = 50	Online-only customers	0.3988	0.4564	0.9939
	Offline-only customers	0.3448	0.2951	0.8664
	Multi-channel customers	0.3085	0.3124	0.8838
	Multi-channel customers, online target basket	0.3677	0.4372	0.9968
	Multi-channel customers, offline target basket	0.2609	0.2123	0.7931
	Multi-channel customers, target channel known	0.3095	0.3191	0.8846

the whole item set X, the goal is to predict the items in the next basket of the customer, i.e., B^u_{n+1}. For the basket history B^u, the recommendation model assigns a score to all items $x_i \in X$, and the top-k items are returned as the candidate items for the next basket recommendation.

We investigate the NBR task for different types of customers in multi-channel retail. We use a simple but powerful recommendation model, namely PersonalTopK, that has been shown to perform on par with complex state-of-the-art methods [66]. PersonalTopK recommends the most frequent k items that appear in the past baskets of a given customer as the prediction for the next basket.

Experimental setup. We experiment with our transaction logs, pick a week as the test week, and consider all customers that have a basket in the test week as candidate test customers, with their first basket in the test week as the target basket to predict. We only consider customers that have over 10 baskets in the previous seven weeks as our test customers, and leverage the baskets from the previous seven weeks as training data. We partition the test customers into three groups based on their shopping channels in the training data. We are left with 15K offline-only, 52K multi-channel, and 3K online-only customers. For the multi-channel customers, around 50% of the target baskets are online and 50% are offline.

We evaluate the NBR performance using Recall@k, nDCG@k, and PHR@k [66, 166]. Recall is widely used in NBR, measuring how many of the items in the target

basket are present in the predicted items for the next basket. nDCG is a ranking based measure which takes into account the order of items. PHR (Personal Hit Rate) measures the ratio of customers whose predicted baskets contain the items appearing in the target basket, evaluating the performance at customer level. All measures are averaged across the predicted baskets for all test users. We report the metrics for $k \in \{10, 20, 50\}$.

Results and discussion. Table 3.1 contains the results of the PersonalTopK recommendation model. PersonalTopK performs surprisingly well across all metrics and basket sizes, compared to common datasets used for evaluating the NBR task [66, 166]. This highlights the importance of personal history in our dataset. The performance for online-only customers is superior to offline-only customers, across all metrics and basket sizes, except Recall@10. The difference in performance is substantial; improvements range from 55% to 116% in terms of nDCG and 15% to 33% in terms of PHR. The performance for multi-channel customers is lower than for online-only customers with a large margin across metrics and basket sizes. The decrease in performance ranges from 19% to 23% in recall, 32% to 37% in nDCG, and 11% to 21% in PHR. The performance for multi-channel customers is higher than for offline-only customers, except in Recall. This is in line with our finding on the repeat ratio from the customer behavior analysis in the previous section; online-only customers have the most repeated behavior, followed by multi-channel and offline-only customers. The importance of personal history differs for different types of customers.

Next, we zoom in on multi-channel customers, and distinguish between multi-channel customers with an online vs. an offline target basket. In cases where the target basket is online, NBR performance is substantially better across all metrics and basket sizes, ranging from 23% to 41% in recall, 106% to 214% in nDCG and 26% to 64% in PHR. This is inline with the previous results; online baskets are easier to predict for the PersonalTopK model due to stronger repeated behavior. The NBR performance for multi-channel customers with an offline target basket is lower than for offline-only customers with a large margin (ranging from 24% to 34% in recall, 28% to 30% in nDCG and 8% to 18% in PHR), across all metrics and basket sizes. This means that the online baskets of these customers are not very helpful in predicting the items in the offline target basket. This is probably caused by the fact that the candidate items pool becomes large with the addition of items purchased in the online channel, which adds noise in predicting the offline target basket. However, this is not the case for the customers with an online target basket in general. While some metrics (Recall@10, Recall@20, Recall@50 and nDCG@50) are marginally lower for multi-channel customers with an online basket compared to online-only customers, the rest of the metrics are improved. This indicates that the offline baskets of multi-channel customers are not necessarily misleading the recommendation of online baskets.

We further consider an oracle version of PersonalTopK for the multi-channel customers, where we assume that the channel for the target basket of customers is known, and where we only consider data from the correct target channel for computing the most frequent items. NBR performance in this context is superior to the performance for multi-channel customers. This means that knowing the target channel can improve the performance of an NBR model, even for one as simple as PersonalTopK. We still observe a large difference compared to the best performing group, namely online-only

customers. Hence, there is a huge potential for improving the performance of NBR for multi-channel customers.

3.5 Conclusion

We answered **RQ2**: What are the characteristics of user behavior in multi-channel retail settings? by presenting the first study on customer behavior in a multi-channel setting in retail, where customers can use both online (web shop, mobile application) and offline (physical store) channels for shopping. We based our analysis on a sample of 2.8 million transactions originating from 300,000 customers of a food retailer in Europe. We revealed significant differences in customer behavior across online and offline channels, for example w.r.t. basket size and the repeat ratio of item purchases. Based on these findings, we investigated the performance of a downstream prediction task under multi-channel settings, namely next basket recommendation.

The recommendation performance differs significantly for customers based on their choice of shopping channel. This strongly indicates that future research on recommenders in this area should take into account the particular characteristics of multi-channel retail shopping.

The next chapter starts Part II of the thesis, with a shift in domain from retail to finance. We continue with user behavior understanding in finance, and focus on relevant recommendation tasks for this domain.

Part II

From User Behavior to Recommendation

4

Understanding and Learning from Financial Information Seeking Behavior

In this chapter, we switch domains, from retail to finance, and answer **RQ3**: What are the characteristics of financial information seeking behavior in user interactions with company filings and how can they be used in designing user-oriented filing recommender systems? We address this question using public large-scale logs from a financial information system.

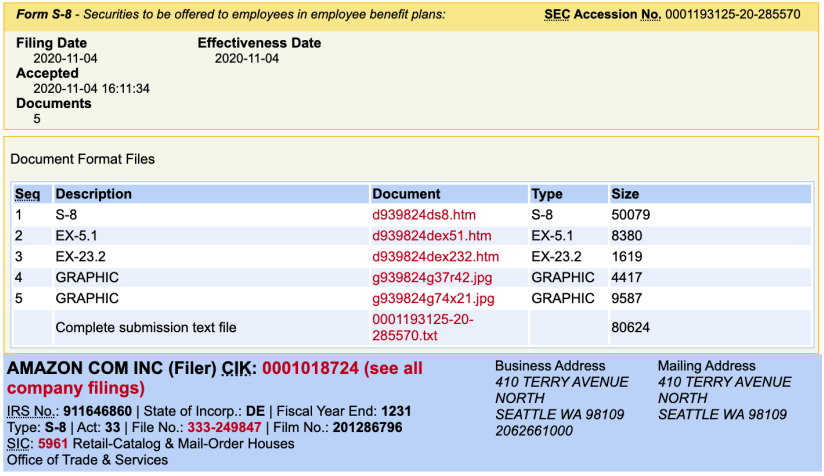
4.1 Introduction

Finance is a domain characterized by a large number of financial and regulatory documents, which analysts and investors need to consume as part of their regular workflows. Advanced techniques for search, filtering, and recommendation are crucial to ensure timely access to the right information [21, 53]. Despite this, information retrieval (IR) research focused on the financial domain is still in its early days. While there are some studies on stock recommendation [30, 34, 164], financial entity extraction [107], financial event representation learning [40], ranking [54], and prediction [161], we understand relatively little about how users interact with financial information systems. In this work we take a first step in this direction by focusing on company filings.

Company filings are financial statements of companies or disclosures made by parties tied to these companies. They are a primary source for investors, analysts, advisors, and regulators to acquire information about a company. In the US, all public companies are required to submit filings to the Securities and Exchange Commission (SEC). These filings are exposed to users through the Electronic Data Gathering, Analysis, and Retrieval system (EDGAR) database [86]. With an average of over 3,000 filings being submitted per day and hundreds of thousands of daily filing views by users, EDGAR plays a central role in the collection and distribution of financial information.

The Securities and Exchange Commission (SEC) made the EDGAR Log File Dataset (EDGAR-LFD) publicly available. The dataset records access to company

This chapter (except for Section 4.6) was published as M. Ariannezhad, M. Yahya, E. Meij, S. Schelter, and M. de Rijke. Understanding financial information seeking behavior from user interactions with company filings. In *Companion of The Web Conference 2022, Virtual Event / Lyon, France, April 25–29, 2022*, pages 586–594. ACM, 2022.



Form S-8 *- Securities to be offered to employees in employee benefit plans:* **SEC Accession No.** 0001193125-20-285570

Filing Date 2020-11-04
Effectiveness Date 2020-11-04
Accepted 2020-11-04 16:11:34
Documents 5

Document Format Files

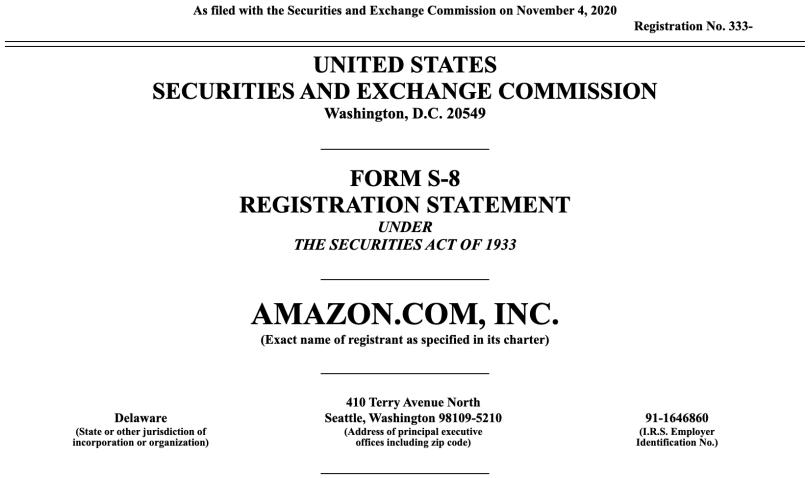
Seq	Description	Document	Type	Size
1	S-8	d939824ds8.htm	S-8	50079
2	EX-5.1	d939824dex51.htm	EX-5.1	8380
3	EX-23.2	d939824dex232.htm	EX-23.2	1619
4	GRAPHIC	g939824g37r42.jpg	GRAPHIC	4417
5	GRAPHIC	g939824g74x21.jpg	GRAPHIC	9587
	Complete submission text file	0001193125-20-285570.txt		80624

AMAZON COM INC (Filer) CIK: 0001018724 (see all company filings)
IRS No.: **911646860** | State of Incorp.: **DE** | Fiscal Year End: **1231**
Type: **S-8** | Act: **33** | File No.: **333-249847** | Film No.: **201286796**
SIC: **5961** Retail-Catalog & Mail-Order Houses
Office of Trade & Services

Business Address
410 TERRY AVENUE NORTH
SEATTLE WA 98109
2062661000

Mailing Address
410 TERRY AVENUE NORTH
SEATTLE WA 98109

(a) Filing index page linking to filing and supplementary files. Image source: `https://www.sec.gov/edgar/search-and-access`



As filed with the Securities and Exchange Commission on November 4, 2020

Registration No. 333-

UNITED STATES
SECURITIES AND EXCHANGE COMMISSION
Washington, D.C. 20549

FORM S-8
REGISTRATION STATEMENT
UNDER
THE SECURITIES ACT OF 1933

AMAZON.COM, INC.
(Exact name of registrant as specified in its charter)

Delaware	410 Terry Avenue North Seattle, Washington 98109-5210	91-1646860
(State or other jurisdiction of incorporation or organization)	(Address of principal executive offices including zip code)	(I.R.S. Employer Identification No.)

(b) Content of an S-8 filing (corresponds to the first row in Fig. 4.1a). Image source: `https://www.sec.gov/edgar/search-and-access`

Figure 4.1: Example EDGAR filing index page and content.

filings in the period between February 14, 2003 and June 30, 2017.[1] It captures access to individual filings from different users, alongside meta-information about the filing that is being accessed. The availability of EDGAR-LFD has led to numerous studies in the finance literature [45, 46, 87, 93, 102, 150], which shows the importance of this dataset for research in finance. The focus in such publications is on revealing correlations between the information acquisition of EDGAR users at an aggregate level and financial variables such as stock returns [46, 87, 150]. Different from previous work in the finance literature, we aim to understand how users interact with EDGAR with the focus on information access itself.

In this chapter, we provide a comprehensive picture of EDGAR-LFD (Section 4.3) and analyse how users interact with financial company filings (Section 4.5). We study the interactions of users with EDGAR on a session level, on a user level, and from a temporal perspective. We find that more than half of the users are one-timers with a single session, while the 10% most active users account for 75% of all sessions. Further, most sessions are focused on filings from a small number of companies. Specifically, 66% of all sessions contain filings from 1–2 companies, suggesting that users tend to focus on filings of a single or of a pair of companies in a session. In the top 10% most frequent users, 50% are interested in six companies or less and 90% of them are interested in 37 or less unique companies over their whole life-cycle, defined as the time between their first and last session on EDGAR. This shows that the most frequent users of the system only focus on a small portfolio of companies.

Our user behavior analysis serves as a stepping stone for the community to tackle retrieval and recommendation tasks for the high-impact financial domain. Our findings have the potential to help financial information providers such as the SEC and commercial providers to better understand the user journey in browsing filings, and suggest ways to enhance the user experience, e.g., via filing recommendation. As a concrete use case to benefit from our analysis, we identify two variations of the filing recommendation task that correspond to the different usage patterns observed in EDGAR, namely next-filing and next-session recommendation.

Next-filing and next-session recommendation. As a use-case for the user behavior analysis, we study filing recommendation as a downstream task. More active users will benefit from different forms of filing recommendations than less active ones based on how much we know about them. For infrequent users, for whom we have no prior history beyond the current session, we investigate how to best recommend the next filing to look at based on their ongoing session of looked-at filings, i.e., next-filing recommendation, which is equivalent to session-based recommendation [70]. For more frequent users for whom we have a rich history, we investigate a next-session recommendation model that predicts a full list of filings for their next session, which is analogous to next basket recommendation [65]. We propose custom two-stage recommendation models for these tasks (Section 4.6), where we predict the company(s) of the next filing in the first stage, and rank the filings of the predicted companies in the second stage so as to recommend the top ranking filings to the user. Our experimental results show that our custom two-stage recommendation models are more effective than state-of-the-art direct filing recommendation models in terms of standard ranking evaluation metrics, such as recall

[1] `https://www.sec.gov/dera/data/edgar-log-file-data-set.html`

and mean reciprocal rank.

In summary, we provide the following contributions:

- We provide a detailed description and statistics for EDGAR-LFD, which will inform anyone interested in exploring this dataset (Section 4.3).
- We provide the first analysis on the information seeking behavior of EDGAR users. Our analysis reveals that user sessions focus on filings from a small number of companies, and that individual users are typically only concerned with a small fraction of the set of all companies (Section 4.5).
- We design custom two-stage recommendation models for session-based and next-session filing recommendation, based on the usage patterns uncovered in our behavioral analysis. We evaluate our models on EDGAR-LFD, where the experiments show their effectiveness (Section 4.6).

4.2 Related Work

Previous work on EDGAR-LFD. While this dataset has not been studied by the IR community, there is a considerable amount of work on it in the finance literature. Loughran and McDonald [102] study the consumption of financial information in filings by analyzing the distribution of daily filing requests. Activity on EDGAR is correlated with poor stock performance [45], reactions of the stock market to earnings announcements [93], and predictive of firm performance [46] and stock returns [87, 150]. Co-searches of companies by the same users on EDGAR are used to identify economically related peer firms [86]. These studies examine the usage of EDGAR at an aggregate level, and do not look into user level activities; the focus is on financial variables such as stock returns, and correlations with market events. Unlike previous work, in this chapter we analyze the EDGAR-LFD from an information access perspective, in order to understand user behavior and enhance the performance of filing recommendation systems.

Analyzing information seeking behavior. There is a large volume of work on analyzing and learning from interaction logs. Interaction logs are studied to characterize information seeking behavior in different settings such as web search [94], mobile search [153], email search [5], library search [79], and search in productivity software suites [26]. What we add to the work listed above is a comprehensive picture of the information seeking behavior in the finance domain.

Information retrieval in finance. IR in finance has gained attention in the recent years. The FinIR workshop [53] introduces and explores challenges and potential research directions in this area. The FinWeb workshop[2] further explores the usefulness of information on the Web for financial technology. Plachouras et al. [114] and Liu et al. [96] propose search systems specifically designed for financial data. Other related work includes methods to rank financial tweets [29], entity extraction and disambiguation in finance [63], extracting summaries from annual financial reports [4], risk ranking from financial reports [143], and financial document classification [51]. Complementary to

[2] https://sites.google.com/nlg.csie.ntu.edu.tw/finweb2021

Table 4.1: Descriptions of the most frequently accessed filing (form) types on EDGAR.

Type	Description
10-K	Report on a company's performance over a financial year
4	Statement of changes in beneficial ownership of a company
8-K	Report of unscheduled material events or corporate changes at a company
10-Q	Report on a company's performance over a financial quarter

existing work, we focus on understanding user behavior in interaction with a financial information system and enhancing the user experience with recommender systems.

Recommendation. To the best of our knowledge, there is no prior work on filing recommendation. We propose two variations of the filing recommendation task based on the usage patterns on EDGAR, namely session-based and next-session. Session-based recommendation is the task of predicting the next item that a user will interact with in their current session [122]. Popular approaches to session-based recommendation include nearest neighbors [70], neural network [122] and factorization based methods [103]. While most recent work focuses on deep neural networks [88, 95, 122], nearest neighbor based methods are shown to often perform equally well or significantly better, despite their simplicity [103]. Our proposed next session filing recommendation task parallels next basket recommendation. Similar to session-based recommendation, neural network based models are gaining attraction [18, 65, 165], while a nearest neighbor based model is shown to be more effective [66, 92]. We report on, and analyze, the performance of state-of-the-art methods for session-based and next session filing recommendation.

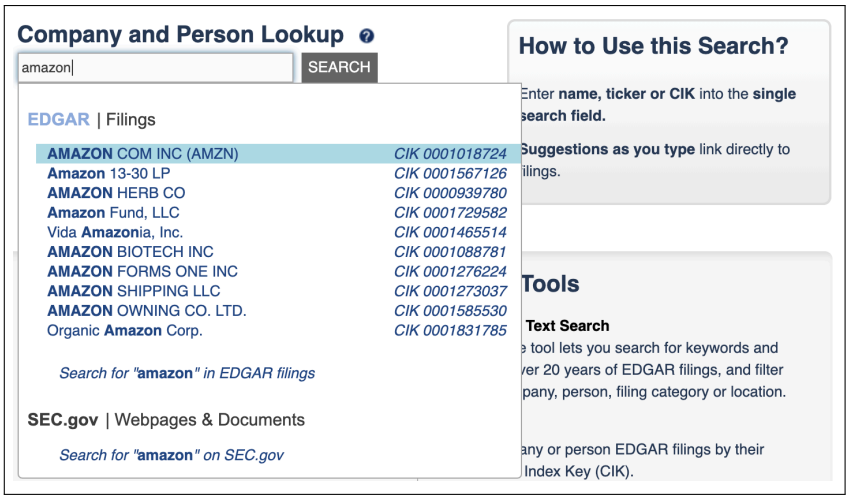
4.3 EDGAR & The Log File Dataset

We start by describing the EDGAR system and EDGAR-LFD, which are the main focus of this chapter. At the center of EDGAR are *filings*, financial statements of companies and disclosures made by parties tied to these companies. Fig. 4.1a shows the index page for a specific type S-8 filing issued by Amazon. An S-8 filing is made by companies when they issue equity to their employees. Other examples of filing (form) types are shown in Table 4.1.[3] A company, like Amazon in our example, is identified by a *unique central index key* (CIK). The concrete filing is uniquely identified by an *accession number*. As shown in Fig. 4.1a, the filing itself is composed of multiple files; these are typically the filing document itself in various formats, and supplementary material such as graphics and relevant correspondences. Fig. 4.1b shows the top of a filing document itself, which corresponds to what a user sees after clicking on a document in the first row of Fig. 4.1a.

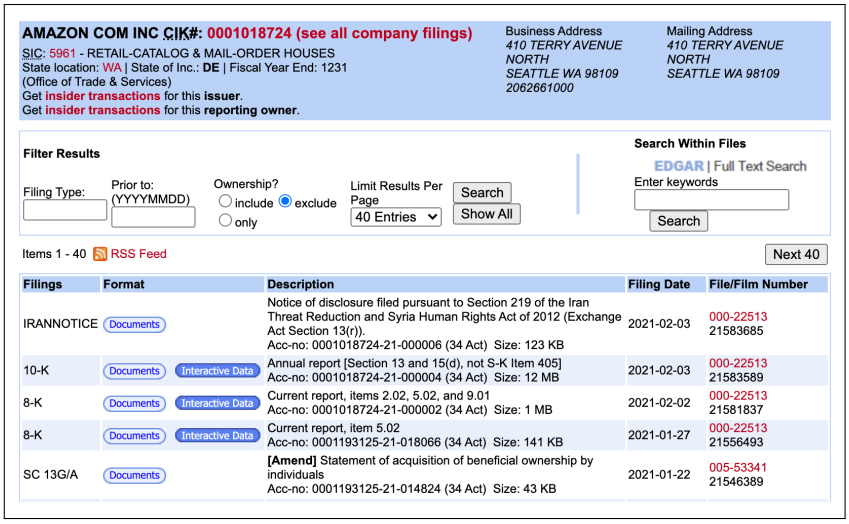
The EDGAR system provides various interfaces for accessing filings. These interfaces can be divided into the following categories:

(1) Company lookup (Fig. 4.2a), to list all filings of that specific company, as shown in Fig. 4.2b.

[3]Descriptions of different filing types can be found under `https://www.sec.gov/forms`

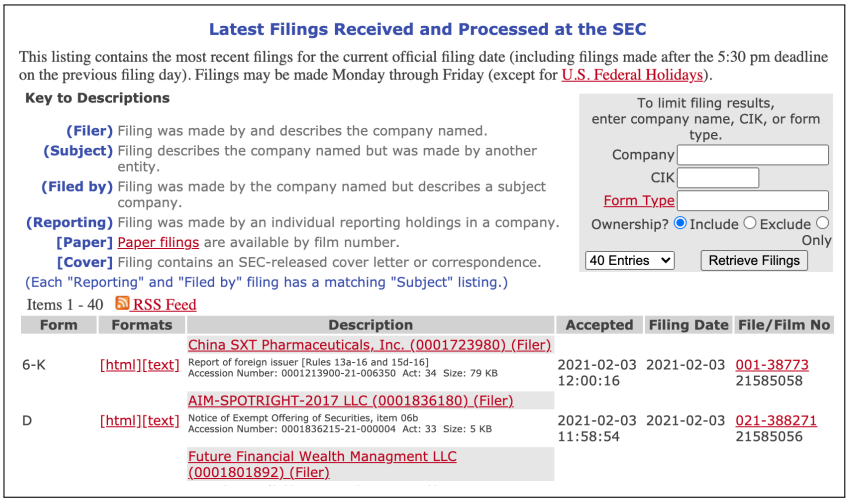


(a) Users can search EDGAR for specific companies. Image source: `https://www.sec.gov/edgar/searchedgar/companysearch`



(b) Chronological list of filings for a company. Image source: `https://www.sec.gov/cgi-bin/browse-edgar?CIK=1018724`

Figure 4.2: Examples of specific parts of the SEC's EDGAR website.



Latest Filings Received and Processed at the SEC

This listing contains the most recent filings for the current official filing date (including filings made after the 5:30 pm deadline on the previous filing day). Filings may be made Monday through Friday (except for U.S. Federal Holidays).

Key to Descriptions

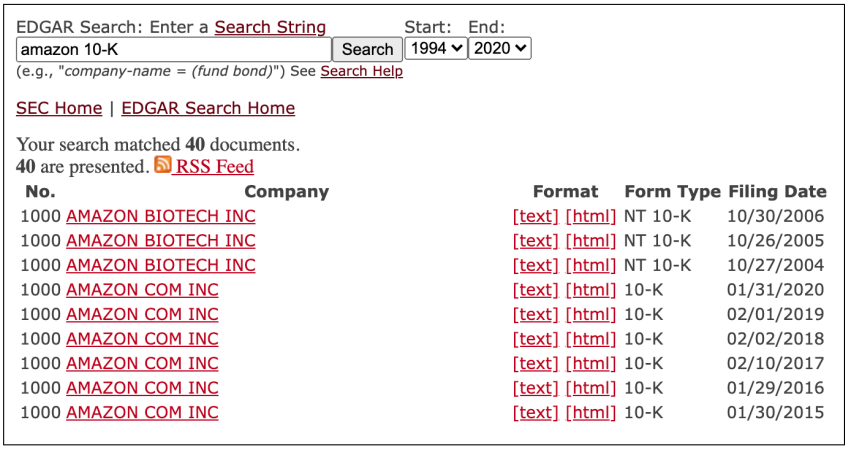
(Filer) Filing was made by and describes the company named.
(Subject) Filing describes the company named but was made by another entity.
(Filed by) Filing was made by the company named but describes a subject company.
(Reporting) Filing was made by an individual reporting holdings in a company.
[Paper] Paper filings are available by film number.
[Cover] Filing contains an SEC-released cover letter or correspondence.
(Each "Reporting" and "Filed by" filing has a matching "Subject" listing.)

To limit filing results, enter company name, CIK, or form type.
Company
CIK
Form Type
Ownership? Include Exclude Only
40 Entries Retrieve Filings

Items 1 - 40 RSS Feed

Form	Formats	Description	Accepted	Filing Date	File/Film No
		China SXT Pharmaceuticals, Inc. (0001723980) (Filer)			
6-K	[html][text]	Report of foreign issuer [Rules 13a-16 and 15d-16] Accession Number: 0001213900-21-006350 Act: 34 Size: 79 KB	2021-02-03 12:00:16	2021-02-03	001-38773 21585058
		AIM-SPOTRIGHT-2017 LLC (0001836180) (Filer)			
D	[html][text]	Notice of Exempt Offering of Securities, item 06b Accession Number: 0001836215-21-000004 Act: 33 Size: 5 KB	2021-02-03 11:58:54	2021-02-03	021-388271 21585056
		Future Financial Wealth Managment LLC (0001801892) (Filer)			

(a) Latest EDGAR submissions, chronologically ordered. Image source: `https://www.sec.gov/cgi-bin/browse-edgar?action=getcurrent`



EDGAR Search: Enter a Search String Start: End:
amazon 10-K Search 1994 2020
(e.g., "*company-name = (fund bond)*") See Search Help

SEC Home | EDGAR Search Home

Your search matched **40** documents.
40 are presented. RSS Feed

No.	Company	Format	Form Type	Filing Date
1000	AMAZON BIOTECH INC	[text] [html]	NT 10-K	10/30/2006
1000	AMAZON BIOTECH INC	[text] [html]	NT 10-K	10/26/2005
1000	AMAZON BIOTECH INC	[text] [html]	NT 10-K	10/27/2004
1000	AMAZON COM INC	[text] [html]	10-K	01/31/2020
1000	AMAZON COM INC	[text] [html]	10-K	02/01/2019
1000	AMAZON COM INC	[text] [html]	10-K	02/02/2018
1000	AMAZON COM INC	[text] [html]	10-K	02/10/2017
1000	AMAZON COM INC	[text] [html]	10-K	01/29/2016
1000	AMAZON COM INC	[text] [html]	10-K	01/30/2015

(b) EDGAR full-text search. Image source: `https://www.sec.gov/cgi-bin/srch-edgar`

Figure 4.3: Examples of specific parts of the SEC's EDGAR website.

Table 4.2: Yearly statistics of EDGAR-LFD after preprocessing.

Year	#valid days	#sessions	#accesses	#unique filings	#unique companies
2003	303	4,421,280	12,377,278	1,430,943	142,960
2004	366	7,377,624	24,697,866	2,012,168	194,072
2005	243	4,505,386	14,290,505	1,817,180	178,416
2006	235	3,971,556	15,424,208	2,232,270	193,222
2007	365	6,703,874	28,643,702	3,467,975	265,258
2008	366	8,351,487	34,072,032	3,469,548	243,016
2009	365	12,111,097	45,109,128	3,244,679	227,435
2010	365	13,633,080	51,405,694	3,403,496	235,163
2011	365	15,543,090	60,586,259	3,598,266	256,870
2012	344	15,835,972	60,817,982	4,745,665	287,858
2013	365	25,724,152	112,685,058	8,517,826	398,015
2014	365	31,280,275	119,560,348	8,158,452	385,115
2015	365	33,084,153	108,967,228	5,993,998	336,679
2016	366	45,364,178	142,694,568	8,211,940	448,807
2017	181	20,282,374	95,057,605	8,586,242	480,512
Total	4,959	248,189,578	926,389,461	11,596,247	607,426

(2) Latest filings, for listing filings made in the past few days (Fig. 4.3a).

(3) EDGAR archive, when the user knows the CIK (and possibly the accession number) of the filing.[4]

These interfaces for reaching a specific filing are reflected in the EDGAR-LFD entries for a specific filing.

With the EDGAR Log File Dataset (EDGAR-LFD), the SEC made access logs for EDGAR filings publicly available.The dataset captures access to individual filings between February 14, 2003 and June 30, 2017, where each record in the data corresponds to a single access by a user to a single filing. A single access corresponds to a single user viewing the contents of a filing (Fig. 4.1b), or a filing index page (Fig. 4.1a). Each record contains the date and time of the access and the obfuscated IP address of the user accessing the filing, as well as other details such as the company's CIK and the filing's accession number. An example of an EDGAR-LFD record is shown in Table 4.3. Additional information about the filings, such as the filing type and the date that the filing was submitted to EDGAR can be inferred using the accession number.[5] Section 4.4 gives more details about the fields of an EDGAR-LFD entry.

We preprocess the dataset and group individual accesses into sessions. The preprocessing and sessionization steps are described in Section 4.4. The yearly dataset statistics after the preprocessing steps are shown in Table 4.2. The processed dataset

[4] `https://www.sec.gov/Archives/edgar/data/1018724`

[5] From: `https://www.sec.gov/Archives/edgar/full-index/`

Table 4.3: Example of an access record in EDGAR-LFD.

IP	Date	Time	CIK	...
203.24.7.aba	2009-10-01	12:09:41	001018724	

...	Accession	Find	Crawler	Extension
	0001193125-09-154174	3	0	.htm

contains more than 926M accesses to more than 11M filings from 600K companies, grouped into more than 248M sessions, which span across more than 14 years of history.

4.4 Data Preparation

Available fields. EDGAR-LFD records access to EDGAR filings between February 14, 2003 and June 30, 2017. The dataset captures access to individual filings, both for the index page of a filing (Fig. 4.1a) and the individual documents that belong to that filing (Fig. 4.1b). The dataset is available in CSV format, and is based on Apache web server access logs. Next, we describe the fields captured by each EDGAR-LFD entry that are relevant to the analyses performed in this chapter.[6]

- **IP** An obfuscated version of the IP address from which a filing was accessed. Obfuscation is done by replacing the last octet of the original IP address with a three character string that preserves the uniqueness of the IP address across the entire dataset.
- **Date** Date of the access in YYYY-mm-dd format.
- **Time** Time of the access in HH:MM:SS format.
- **CIK** Identifier for the company that the filing accessed in this record belongs to.
- **Accession** A unique identifier for the filing being accessed.
- **Find** A number between zero and 10 that indicates how the user arrived at the filing, e.g., internal EDGAR search.
- **Crawler** Indicates whether the user self-identifies as a web crawler.
- **Extension** The extension of the filing page being accessed. Corresponds to the "Document" column in Fig. 4.1a. Used to identify whether a filing document has been accessed, or its index page.

Initial cleanup. After downloading the dataset, we remove records that have an HTTP response code that indicates unsuccessful requests, which are those not in the 2xx class. We also remove records that self-identify as crawlers. We first remove entries from

[6] `https://www.sec.gov/files/EDGAR_variables_FINAL.pdf` for the full list of fields.

dates between September 23, 2005, and May 10, 2006 that were labeled by the SEC as "lost or damaged", as mentioned in [102]. We then further manually examine the days that have significantly less records than the surrounding days for no apparent reason, such as holidays. This results in marking the dates between 2003-02-14 to 2003-03-01, 2003-12-13 to 2003-12-15, and 2012-02-08 to 2012-02-29 as damaged. We remove the damaged dates from the data.

Session definition. In order to better understand the user journeys on EDGAR, we take the individual filing access records in EDGAR-LFD and group these into semantically meaningful sessions. Following [109, 154], we define a session as a sequence of actions performed by a single IP address, where the difference in time between subsequent actions is not larger than a predefined threshold. We rely on [59] to find the threshold, based on fitting a Gaussian mixture model to the histogram of the time between subsequent accesses by the same IP address, i.e., same user. A three component Gaussian mixture model is fitted to the histogram using expectation maximization. After that, the point where inter-activity time is equally likely to be within the Gaussians fit with sub-hour means (within-session) and Gaussians fit with means beyond an hour (between-session) is selected as the threshold.

We apply the above method to EDGAR-LFD in order to create sessions from the individual accesses per IP address. To this end, we sample 10,000 IP addresses that we are confident come from interactions of a single human user with the system based on heuristics from [102] from each year. An IP address corresponding to a single human user is assumed to not have more than 50 requests per day. We plot the histogram of differences consecutive actions by the same IP address in seconds on a log scale. We find 35 minutes to be the optimal threshold for session breaks, which is in line with the thresholds in datasets from other domains [59].

Valid sessions. EDGAR is accessed by both humans and bots, but not all bots self-identify as such; solely relying on the `crawler` attribute of a log entry is unreliable and following previous work [102], we take additional measures to remove bot accesses. We filter out sessions based on three thresholds: 1) the number of accesses in a session, 2) the duration of a session measured in hours, and 3) the average time per access in a session. We keep the sessions that are either a) less than four hours and have a time-per-access of more than two seconds, or b) have less than four accesses in total.

Handling successive requests for the same filing. Each filing on EDGAR has an index page (see Fig. 4.1a), and files that belong to the filing. The data contains access to all documents and index pages. Our focus is on how users interact with different filings; in-filing browsing is out of scope. Hence, for each session, we ignore all successive views to the same accession number, and we remove records corresponding to access to the index pages of filings.

4.5 User Behavior Analysis

We study financial information seeking behavior by analyzing EDGAR-LFD. We look into temporal access patterns, session-level and user-level behavior. We use the data from all years (bottom row, Table 4.2). We aim to understand the user behavior and

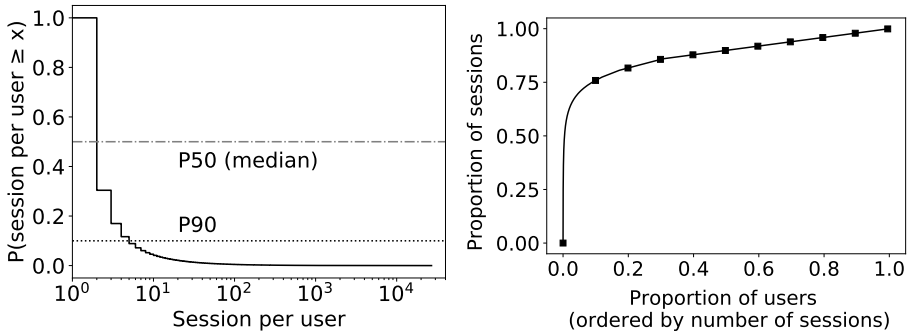


Figure 4.4: Cumulative distribution of (Left:) number of sessions per user, and (Right:) sessions across users.

gain insights that can help to improve the user experience in accessing the filings.

4.5.1 User-level analysis

We focus on user-specific aspects of the filing views. Fig. 4.4 (Left) shows the cumulative distribution of the number of sessions per user. More than half of the users in the data are one-timers with a single session. The 90th percentile for the number of sessions per user is equal to four, and out of 50.2M users in the data, 5.8M have at least four sessions. We will refer to these top 10% users as *frequent users* in the rest of our analysis. Fig. 4.4 (Right) displays the cumulative distribution of sessions across users, showing that the 10% most active users account for roughly 75% of the sessions.

The analysis shows that the proportion of infrequent users is considerable. For such users, historical behavior beyond the current session is scarce. For downstream tasks such as recommendation, this implies that instant recommendation scenarios such as session-based recommendation, where only the information from the current session is considered, are suitable for a general EDGAR user. We also notice that frequent users are responsible for over 75% of all the sessions in the data, showing that for such users we can rely on rich historical behavior for downstream tasks, and other recommendation scenarios that rely on historical behavior (such as sequential recommendation) could be considered.

We further look into the companies and form types of interest to all users and to the subset of frequent users. Fig. 4.5 shows the cumulative distribution of unique number of companies and form types that users have interacted with over their whole life cycle. Of the frequent users, 50% are interested in six or less companies and 90% of them are interested in 37 or less unique companies in total; frequent EDGAR users are concerned with only a small subset of companies that have their filings available on the website. The median and 90th percentile are much less if we consider all users. A similar observation holds for the number of unique form types. Out of 505 different form types available, 50% of the frequent users are interested in five or less and 90% are interested in at most 17 unique form types.

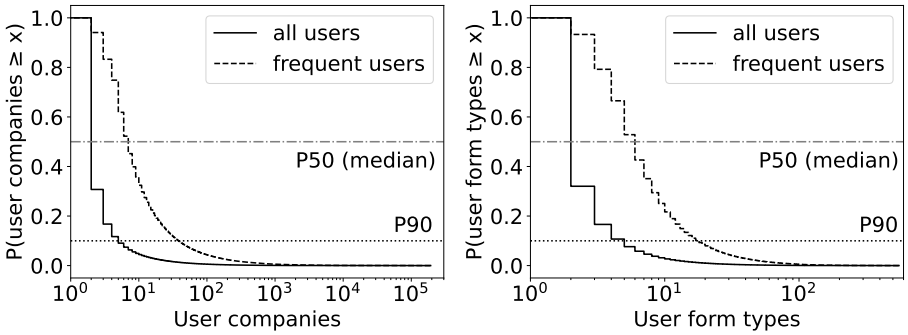


Figure 4.5: Cumulative distribution of (Left:) number of unique companies, and (Right:) number of unique form types for all users and frequent users.

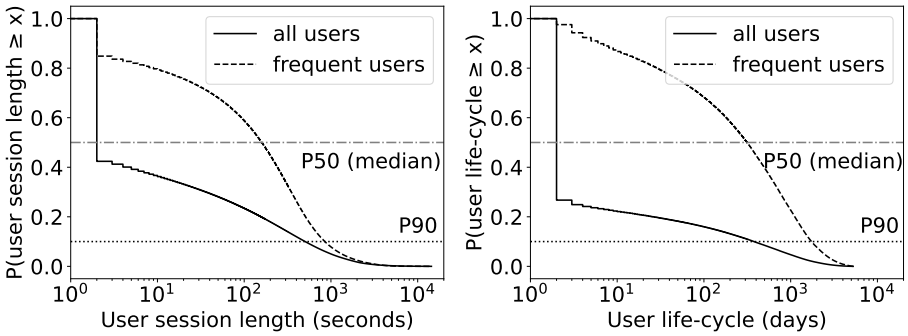


Figure 4.6: Cumulative distribution of (Left:) average session length per user, and (Right:) life-cycle of users in the whole time frame of the data, for all users and frequent users.

This means that each user is concerned with a very small fraction of the data that is available on EDGAR. This implies that in downstream tasks for improving user journey on EDGAR, such as recommendation, focusing on companies and filing types that a user has previously shown interest in will be effective. Such a focus can drastically reduce the search space in different retrieval and recommendation scenarios.

We study the average session length, defined as time between the first and last access in a session, for all users and for frequent users. Fig. 4.6 (Left) shows the cumulative distribution. The session length is longer for frequent users, with 9 and 15 minutes for the 90th percentile of all and frequent users, respectively.

Our analysis reveals that most of the sessions on EDGAR are short; users tend to focus on a single task in a session, that can translates to accessing a particular filing of interest. There is a notable difference between the session length distribution of frequent users and all users; frequent users have longer sessions, which could indicate that they are a specific group of users, for example analysts. It is worth mentioning that EDGAR-

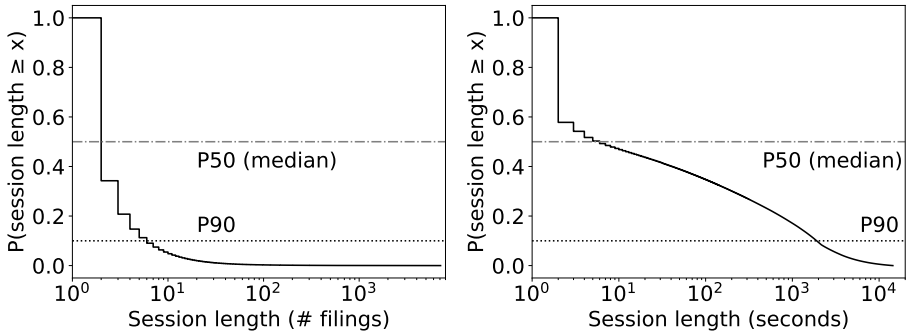


Figure 4.7: Cumulative distribution of (Left:) session length in terms of number of filings accessed in a session, and (Right:) session length measured in seconds.

LFD does not provide any information about the type of users who are accessing the filings, but the usages patterns could help to identify different user groups [46].

Fig. 4.6 (Right) shows the cumulative distribution of life-cycle of users, defined as the number of days between the first and the last session. While the data spans 14 years, the frequent users have a median life-cycle of roughly a year and when considering all users, the 90th percentile for life-cycle is around a year. We observe that frequent users have a much longer life-cycle. For such users, in cases where the historical information is used for a downstream task, the change in the interests of a user in time needs to be accounted for; a very old historical behavior could be less relevant than a recent one when predicting the future interactions of a user.

4.5.2 Session-level analysis

We analyze the characteristics of sessions on EDGAR in this section. Fig. 4.7 shows the cumulative distribution of session length in terms of number of filings viewed and session length measured in seconds between the first and the last access in the session. We observe that 68% of all sessions contain only one filing, suggesting that often times users are interested in one filing at a time. This means that most sessions are focused on a single information need, corresponding to a small number of filings at a time.

The session length distribution shows that 53% of all sessions take less than 10 seconds, corresponding to the sessions with one filing. On the other hand, around 40% of sessions take longer than 100 seconds, which shows that rapid consumption of information is not always possible. It is worth mentioning that since we are measuring the time between the first and the last access, this does not reflect the exact session duration; the user may still be reading the last accessed filing.

We further study sessions that contain more than one filing in terms of the number of unique companies and form types that they contain. Fig. 4.8 shows the distributions. Most sessions are focused on a small number of companies. Specifically, 66% of all sessions contain filings from one or two companies, suggesting that EDGAR users tend to focus on filings of a single or a pair of companies in a session. The average number

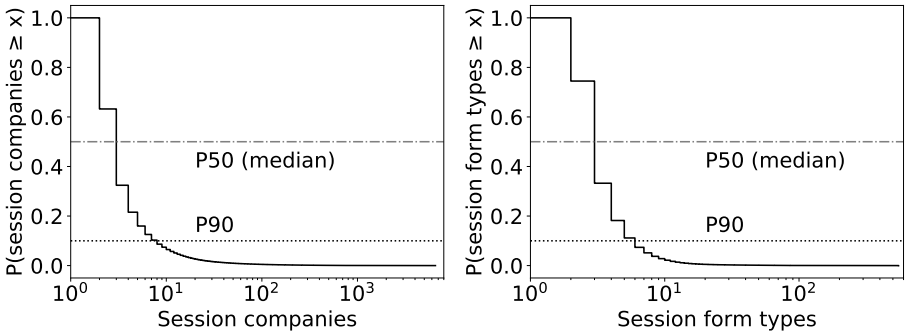


Figure 4.8: Cumulative distribution of (Left:) number of unique companies, and (Right:) number of unique form types in sessions with more than one filing accessed.

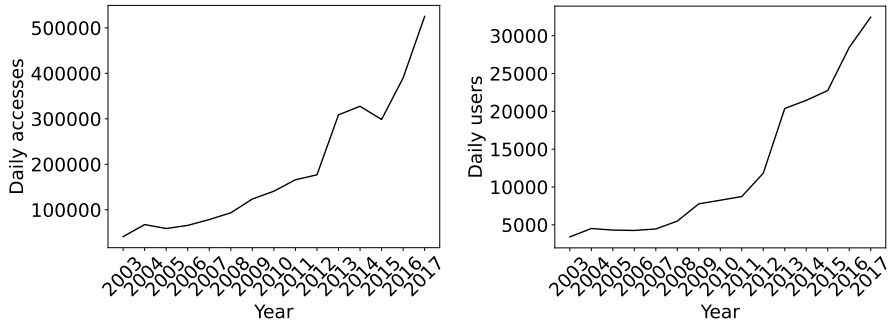


Figure 4.9: Average (Left:) accesses per day, and (Right:) users per day for different years.

of unique form types in sessions is 2.78, with 25% of sessions containing a single form type, suggesting that users are interested in browsing multiple form types in a session. In case of session-based recommendation, these insights can be used to limit the search space to certain filing types from the companies accesses up until now in the current session.

We also look at the referrers to the first access in the session, which tells us how people come to interact with filings in EDGAR. It is worth mentioning that EDGAR-LFD only records accesses to filings; accesses to other pages on the SEC website are not visible in the dataset. We observe that 53% of all sessions start from an unknown referrer, including accesses from outside of the SEC website, such as web search and web links. The remaining sessions start from the various pages shown in Fig. 4.1a, 4.2 and 4.3: 82.5% of those start from the page containing all of the filings of a specific company (Fig. 4.2b), suggesting that the company page is the starting point for browsing filings for most of the sessions. Further categories of known referrers are index page (8.8%, Fig. 4.1a), the EDGAR archive (5.0%), filing data (3.3%), search (0.3%), and the latest filings page (0.2%). Only 0.3% of all sessions initiate from search, which shows

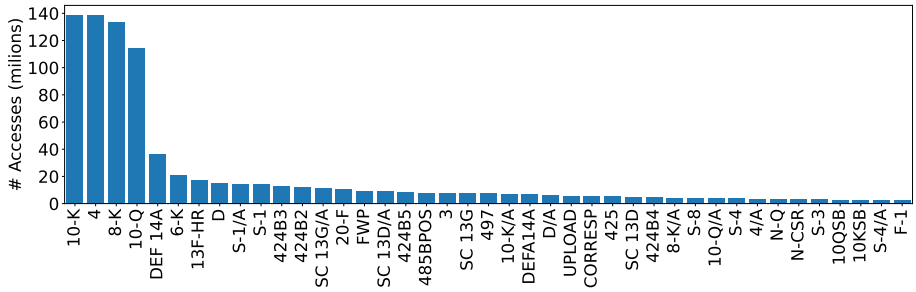


Figure 4.10: Most accessed filing types in the dataset.

that EDGAR users rarely use search to directly find filings; the majority of sessions start from a company page, which indicates that users probably use search to find the companies and use the company filing page (Fig. 4.2b) to access the filings.

4.5.3 Temporal analysis

In this part we focus on the temporal aspects of filing views. Since the data is available for more than 14 years of history, we study the changes in user access across all these years. Fig. 4.9 shows the average of daily accesses and daily users for each year. We observe that there is an exponential growth in both, which shows that more people are using EDGAR as a source of information year by year.

We further study the distribution of the time spent on the filings and the time between accesses and the filing dates for different filing types. We first study the number of accesses per form type. Fig. 4.10 shows the 41 filing types that are responsible for 90% of all accesses on EDGAR, out of 505 available form types. Over half of all accesses are to four filing types, namely 4, 10-K, 10-Q, and 8-K. We look into the time spent on filings by EDGAR users. Fig. 4.11 (Left) shows the cumulative distribution of the time spent on filings for different filing types. Users spent the least time on filing type 4, which is essentially a table, with a median of four and 90th percentile of 100 seconds. Users spent considerably more time on 10-Q and 10-K forms; digesting information in these forms requires more time, as they report on a public company's performance over a financial year or quarter.

Fig. 4.11 (Right) shows the cumulative distribution of the difference between the filing date of a filing and the date that it is being viewed in days, for the top four filing types and in total. EDGAR contains filings from 1993, and EDGAR-LFD covers accesses from 2003 to 2017, so users that we study have access to filings from 10 to 24 years of history. However, 50% of accesses happen within a year of the filing dates. We further notice the difference between filing types. For filing type 4, roughly 35% of the accesses occur in a day from the filing date, and 50% happen in 12 or less days, showing that users are mostly interested in more recent filings of this type. The median is around a year for the 10-K filings, showing that 10-Ks, which cover a company's annual performance, are interesting for the users for a longer period of time.

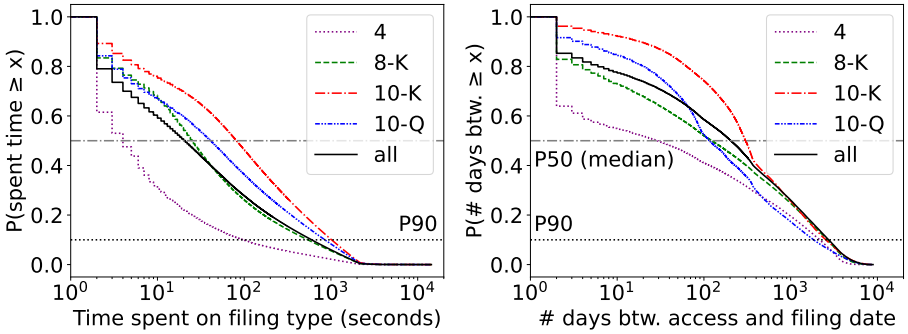


Figure 4.11: Cumulative distribution of (Left:) the time spent on a filing by users, for different filing types (Right) the time between access and filing date.

4.5.4 Main findings

We further highlight some of the findings in our user behavior analysis that will inform our recommendation models.

The user-level analysis shows that there are two main types of users on EDGAR, less active users with a few sessions (Fig. 4.4 (Left)), and the top 10% more active users who account for 75% of all the sessions (Fig. 4.4 (Right)). More active users will benefit from different forms of filing recommendation than less active ones based on how much we know about them. For infrequent users, for whom we have no prior history beyond the current session, we investigate how to best recommend the next filing to look at based on their ongoing session of looked-at filings, i.e., next-filing recommendation. For more frequent users for which we have a rich history, we investigate a next-session recommendation model that predicts a full list of filings for their next session.

4.6 Filing Recommendation

As shown in Section 4.3, users can access filings through either a search interface or listings of filings. The user behavior analysis suggests that there is value in recommending filings to users based on the ones they have already accessed in the short or long term. Such recommendations would reduce the burden on users to move between filings, facilitating quicker access overall. We now discuss two variations of the filing recommendation task that are informed by our user analysis. Subsequently, we propose custom recommendation models based on the results of our analysis, and compare their performance to state-of-the-art approaches and standard baselines. We would like to note that the recommendation use case presented here is just one possible task informed by our analysis. As another example, the user interactions captured in EDGAR-LFD can further be used for learning company representations that are beneficial for downstream tasks, such as identifying economically related peer firms [86].

4.6.1 Next-filing recommendation

Our first recommendation task is *next-filing recommendation*, where the goal is to predict the next filing that a user will view, based on the filings that they have already viewed in the current session. More precisely, this is the task of recommending an accession number (see Table 4.3). Analogous to session-based recommendation [122], next-filing recommendation is particularly useful for EDGAR since the majority of users are not frequent users. As shown in Section 4.5.1, the 90th percentile for the number of sessions per user is equal to four, which makes reliance on historical behavior for a specific user impractical. A successful next-filing recommendation system will save users time, and will help them to have a more comprehensive picture on the subject that they are seeking information about.

Given a session $s = \{x_{s,1}, x_{s,2}, \ldots, x_{s,t}\}$, defined as a sequence of interactions with items corresponding to actions performed by a user, where $x_{s,i} \in X$ is an item from the universe X, the goal of the recommendation model is to predict $x_{s,t+1}$, the next item that the user will interact with in s. For the session s, the recommendation model assigns a score to all items $x_i \in X$, and the top-K items are returned as the candidate items for recommendation.

Our proposed approach. Building on our observations in the user behavior analysis, we decompose the filing recommendation task into 1) predicting the company of the target filing, and 2) ranking the filings of the predicted company. The initial step of narrowing down the candidate set of companies is motivated by our user behavior analysis, which showed that sessions usually contain filings from a small number of companies only (see Fig. 4.8). We propose two company predictors, i.e., CIK-predictors, to address the first problem:

- CIK-S-POP: Our user behavior analysis shows sessions contain filings from a small number of unique companies. Hence, we select the most frequently viewed company in the current session as the company for the next filing.
- CIK-encoder: We use a three layer neural network to encode the sequence of companies accessed in the input session and predict the next company. The first layer is an embedding layer of size 128, the second layer is an LSTM with 128 units, and the final layer is a linear transformation with softmax activation to convert the LSTM output to predicted next-CIK probabilities. We limit the input sequence to (the most recent) 128 companies.

We use the filings of the top prediction of the CIK-predictors as the recommendation candidates. To rank the candidate filings, we use a ranking function similar to [70]. The approach learns from a training set of sessions (not necessarily tied to the user in question), and makes a prediction using the most similar such sessions. Specifically, all candidate filings x_c are ranked using:

$$sim(s_i, x_c) = \sum_{s \in N_{s_i}} sim(s_i, s) \times I(x_c, s), \tag{4.1}$$

where N_{s_i} is the set of top $k_{session}$ similar sessions to the current session s_i, and $I(x_c, s)$ is equal to one if $x_c \in s$ and zero otherwise. Following [70], we set $k_{session} = 500$

in our experiments. To find the top $k_{session}$ similar sessions to the current session, all sessions are ranked based on their similarity to the current session. Similarity is defined as the Jaccard index where sessions are encoded as sets of filings. Formally:

$$sim(s_1, s_2) = \frac{|s_1 \cap s_2|}{|s_1| \times |s_2|}, \tag{4.2}$$

where $|\cdot|$ denotes the number of filings in a session.

Experimental setup. We use the data from 2003 in our experiments and split the data temporally to obtain training and test sessions. We use a week of data for testing, and the 16 weeks prior to that as training data. We repeat this process four times to make sure we are not overfitting on a particular period. Each split comprises of approximately 700K training sessions, 40k test sessions, and contains more than 800k unique items, i.e., filings. We run all models on all four splits, and report aggregated results on the four one-week test periods. We generate input sequences and labels from sessions following previous work [88, 122]. The results are reported on all test samples from the four splits combined. For evaluation, we use Recall@k and MRR@k, following [88, 122]. We report the metrics for $k \in \{10, 20\}$.

Baseline methods used for comparison. We compare our methods with classic recommendation methods as well as state-of-the-art models. Through this comparison, we aim to examine the effectiveness of the recommendation models that are designed based on our understanding of the user behavior, versus powerful existing recommendation models that are not designed for EDGAR. The following baseline models are evaluated:

- S-POP: Recommends the most popular filings of the current session. Despite its simplicity, S-POP can achieve a competitive performance [122].
- Item-KNN [43]: Filings similar to the filings already viewed in the current session are recommended by this baseline.
- Session-KNN [70]: This baseline relies on the similarity score between the current session and the training sessions. All sessions are ranked based on their similarity to the current session, and filings in the top similar sessions are ranked based on their similarity to the filings in the current session.
- RepeatNet [122]: A state-of-the-art neural model with an encoder-decoder architecture. The recommendation probability for each filing is learned via two decoders, to account for two different modes in user behavior: exploration and repeated behavior.

Baseline result analysis. Table 4.4 shows the results for the baselines and our own models. S-POP performs poorly; users rarely access the same filing more than once in a session. Nearest neighbor-based models have the best performance among the baselines on all metrics. This is in line with previous work showing the effectiveness of nearest neighbor-based approaches for recommendation [42, 70]. Session-KNN outperforms Item-KNN, which shows the effectiveness of measuring session similarity compared to individual filing similarity for the task of next-filing recommendation.

The performance of RepeatNet is inferior to the nearest neighbor-based models, by a large margin. We hypothesize that this is due to our huge item space. While RepeatNet

Table 4.4: Experimental results for next-filing recommendation. The best baseline results are underlined, the best overall results are shown in **bold**. Significant improvements over the best baseline are marked with * (t-test, $p < .05$).

Category	Method	Recall		MRR	
		@10	@20	@10	@20
Baselines	S-POP	0.0241	0.0284	0.0135	0.0138
	Item-KNN	0.1095	0.1358	0.0540	0.0558
	Session-KNN	0.1415	0.1771	0.0680	0.0704
	RepeatNet	0.0652	0.0787	0.0348	0.0358
Ours	CIK-S-POP	0.1686*	0.2110*	0.0800*	0.0829*
	CIK-encoder	**0.2047***	**0.2574***	**0.0961***	**0.0998***

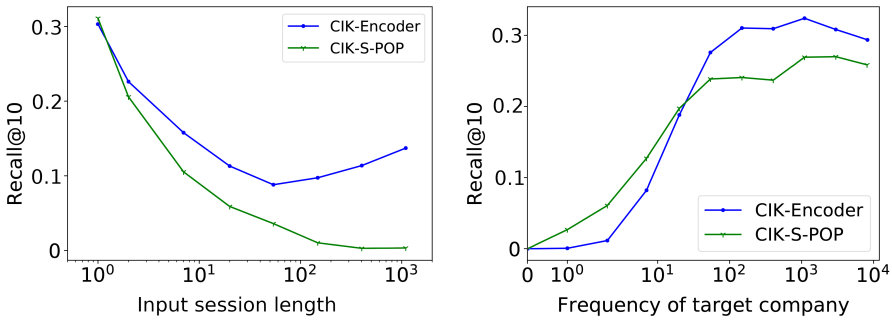


Figure 4.12: Performance of the proposed next-filing recommendation models w.r.t. session length (Left) and frequency of target company in the training data (Right).

has been shown to be effective on e-commerce and music datasets with around 40K unique items [122], we have around 800K unique items in each training set; even after removing the items that appear less than 5 times in the training data [122], around 780K unique items remain; our dataset is much sparser than the ones that RepeatNet has been trained and tested on originally.

Discussion of the proposed approach. We analyze our proposed methods. Both proposed models significantly outperform the best baseline with a large margin. This confirms the effectiveness of the two-step recommendation model, where the search space for the next filing is limited to the filings of the companies of interest to the user. The superior performance of CIK-S-POP as compared to the baselines shows that a simple model that is informed by the user behavior analysis can result in significant improvements in performance.

Additionally, we calculate Precision@1 for both CIK predictors. This metric shows to what extent the models are able to identify the company of the target filing, which directly affects the final results. The Precision@1 is equal to 0.2957 and 0.3613 for CIK-S-POP and CIK-encoder, respectively. The CIK-encoder model is more accurate,

which is reflected in the filing recommendation performance. The higher accuracy in prediction of the company for the next filing is the reason that CIK-Encoder outperforms CIK-S-POP in the task of filing recommendation.

We further look into the sensitivity of each of the models to the input session length. Fig. 4.12 (Left) shows the Recall@10 metric for different input session lengths. For very short input sessions (length one or two) where there is not enough history for CIK-Encoder to capture, CIK-S-POP is the best performing model. As the session length increases, the performance of CIK-Encoder dominates. As the session grows in length and the number of candidate companies to choose from increases, the performance of both models degrades. For extremely long sessions CIK-Encoder rises in performance again, which shows the effectiveness of the neural architecture to learn representations for long input lengths.

Finally, we consider the sensitivity of the models to the frequency of the target company in the training data. Fig. 4.12 (Right) shows the Recall@10 metric for both models w.r.t. the target company frequency. The performance of both models improves with the increase in the frequency of the target company. We notice that CIK-Encoder is unable to outperform CIK-S-POP when the target company has extremely low frequency in the training data.

Summary. We have proposed two-stage filing recommendation models that are designed based on our understanding of user behavior. Our proposed models significantly outperform state-of-the-art baselines, and are more effective for shorter sessions and companies that are more frequent in the data.

4.6.2 Next-session recommendation

We consider the problem of *next-session recommendation* in the context of EDGAR filings, where the session items are filing accesses during a session. The goal of the next-session recommendation task is to recommend a list of filings to the user every time they visit the website, based on the filings they have viewed in the past. Such recommendations would reduce the burden on users to proactively find the filings of interest every time they visit EDGAR, and facilitates quicker access to their information need.

Recommending filings for the next session will be most helpful to the frequent users of EDGAR for whom enough historical behavior is available. Although the frequent users are not the majority of users, they are responsible for most of the activity, justifying the design of recommendation models tailored for them. As shown in Section 4.5.1, the 10% most active users account for roughly 75% of the sessions on EDGAR.

Given $S^u = \{S^u_1, S^u_2, \ldots, S^u_n\}$ as the sessionized history of interactions for user u, where S^u_i is a session of interactions defined as $S^u_i = \{x_1, x_2, \ldots, x_t\}$ and $x_i \in X$ is an item from the whole item set X, the goal is to predict the items in the next session of the user, i.e., S^u_{n+1}. For the session history S^u, the recommendation model assigns a score to all items $x_i \in X$, and the top-K items are returned as the candidate items for the next-session recommendation. This problem definition is equivalent to the task of Next Basket Recommendation (NBR) [123].

Our proposed approach. We propose a simple yet effective recommendation model

built on our observations from the user behavior analysis. We know from our user behavior analysis that a session usually contains filings from a small number of companies (see Fig. 4.8). In particular, 85% of target sessions in our training set contain filings from a single company. As a result, predicting the companies that will have filings in the next session can reduce the search space tremendously. Similar to next-filing recommendation, we propose a two-stage recommendation model, where in the first step we predict the company(s) for the filings in the next session, and in the second step we rank them. Our user behavior analysis also reveals that users are usually interested in a small number of companies during their life time. In 46% of the training cases the companies of the next session of a user are already present in their history.

We define CIK-P-POP such that for each user in the test data, we take the most viewed company in their history in terms of number of sessions that contain it as the predicted company for their next session. To rank the filings of the predicted company, we rely on the fact that users are interested in the most recent filings, and rank the filings of the predicted company in reverse chronological order of filing date.

Experimental setup. We use the data from 2003 in our experiments and split the data temporally to obtain training and test sessions. We select the period from 2003-01-01 to 2003-11-31 as our training period. We keep the users that have more than 10 sessions in this period. We set the week of 2003-12-01 to 2003-12-07 as our test period, and our goal is to predict the first session of our training users in this period. As a result, we train on 41,743 users, evaluate on 13,900 users, with an average number of 63.87 sessions per user with an average length of 3.22. For evaluation, we use Recall@k and nDCG@k for $k \in \{10, 20\}$ [66]. Both measures are calculated across the predicted next session for all test users.

Baseline methods for comparison. We compare our method with classic methods as well as a state-of-the-art model to examine the effect of using the insights from user behavior in designing a recommendation model for EDGAR. Based on [92], the following baseline models are evaluated:

- P-POP: Recommends the most popular filings in the user history.
- User-KNN: following [66], we merge all the filings in the historical sessions of users as a set, calculate the user similarity using the similarity function defined in Eq. 4.2, and sort the filings in the top k_{user} similar users with the scoring function defined in Eq. 4.1. We set $k_{user} = 100$ in our experiments.
- TIFU-KNN [66]: A state-of-the-art nearest neighbor-based model for NBR, which has been shown to outperform deep recurrent neural networks. The model relies on the similarity of the target user with other users and the history of the target user.

Baseline result analysis. From the results in Table 4.5 we observe that P-POP performs reasonably well; recommending filings that a user has accessed before is effective. The User-KNN model does not outperform P-POP on this dataset, which is consistent with the observations on NBR datasets [66, 92], emphasising the importance of personal history of users compared to the similarity with other users. TIFU-KNN achieves the highest performance among the baselines, which shows that combining users' personal preferences with the history of similar users is the most effective.

Table 4.5: Experimental results for next-session recommendation. The best results are shown in **bold**, the best baseline results are underlined. Significant improvements over the best baseline are marked with * (t-test, $p < .05$).

Category	Method	Recall		nDCG	
		@10	@20	@10	@20
Baselines	P-POP	0.0444	0.0629	0.0285	0.0337
	User-KNN	0.0374	0.0433	0.0288	0.0305
	TIFU-KNN	<u>0.0666</u>	<u>0.0862</u>	<u>0.0440</u>	<u>0.0494</u>
Ours	CIK-P-POP	**0.0789***	**0.0965***	**0.0558***	**0.0609***

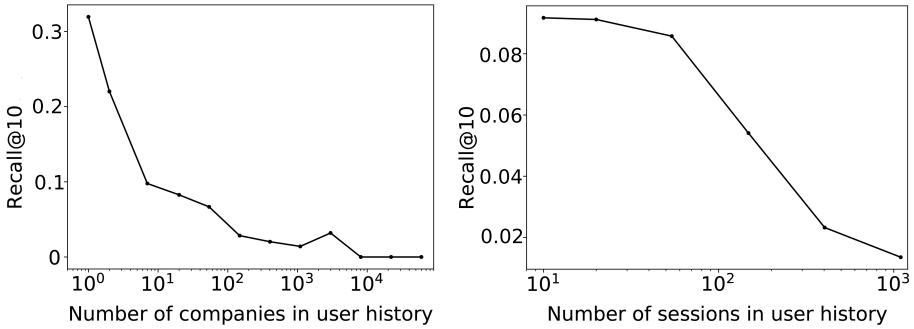


Figure 4.13: Performance of CIK-P-POP w.r.t. the number of companies (Left) and sessions (Right) in the user history.

Discussion of the proposed approach. We observe that despite its simplicity, CIK-P-POP is able to significantly outperform the baselines, particularly the state-of-the-art TIFU-KNN. This, again, highlights the importance of the analysis presented in Section 4.5 in understanding user interaction with EDGAR and using it to facilitate more streamlined interaction. We analyze the performance of CIK-P-POP. For 18.15% of the test users, the companies in the target session contain the predicted company; thus, there is a large room for improvement of the company predictor. We also examine the sensitivity of CIK-P-POP to the number of unique companies in the history of the users. Fig. 4.13 (Left) shows that the model is very sensitive to this parameter, with a range of 30% Recall@10. It performs best when users have accessed a small number of companies throughout their history, which makes it more probable for CIK-P-POP to select the correct one for the next session. We also consider the sensitivity of the model to the number of sessions in the user history. Fig. 4.13 (Right) shows the results. CIK-P-POP is less sensitive to the number of sessions than to the number of companies, and its performance sharply degrades when the number of sessions exceeds 50, which makes it more difficult for CIK-P-POP to select the target company.

Summary. We have proposed a simple filing recommendation model based on our insights from our user behavior analysis. Our proposed model significantly outperforms

a state-of-the-art baseline, and is especially more effective for users that are focused on a small number of companies in their history.

4.7 Conclusion

Financial company filings are a primary source for investors, analysts, advisors, and regulators to acquire information about a company and to support their decision making. We answer **RQ3** by studying the EDGAR Log File Dataset, a publicly available dataset containing the log of accesses to company filings on the US Securities and Exchange Commission (SEC) website. Through a user behavior analysis, we provide the first study on this dataset from an information access perspective. We find that sessions on EDGAR are focused on filings from a small number of companies and individual users are interested in a limited number of companies during their life cycle on EDGAR.

The goal of our work is to provide a stepping stone for the academic community to tackle retrieval and recommendation tasks for the finance domain. We identify two filing recommendation tasks that correspond to different usage patterns in the dataset. To address **RQ3**, we propose two-stage filing recommendation models that first predict the company of the next filing(s), and then rank the filings of the predicted company. Our proposed models perform significantly better than the state-of-the-art direct filing recommendation models. The contents of the filings are available through EDGAR, and can be used to better understand the users. In future work, we aim to design content-aware recommendation models. EDGAR-LFD contains the data for a rather long period. While we study some temporal aspects of the user behavior in this chapter, many dimensions remain unexplored. For example, it will be worthwhile to see how the co-accesses to filings of different companies shift over time, and whether such a shift is a reflection of a change in companies' business lines.

In the next chapter we switch back to retail. We study the repurchase behavior of users in grocery shopping and introduce a repeat consumption-aware neural network for next basket recommendation.

5 Modeling Repeat Behavior in Retail

In this chapter, we switch domain from finance to retail, and continue with designing user-oriented recommender methods. This chapter addresses the following research question: **RQ4**: What are the characteristics of repeat consumption behavior in the grocery shopping domain and how can they be used in desiging next basket recommender systems? We study the repeat behavior across six grocery shopping datasets, and propose a neural next basket recommendation model that explicitly considers the repeat behavior signals.

5.1 Introduction

Recommender systems in retail help users to find the items that they need from large inventories. Different retail industries such as fashion [37], general e-commerce [88], and grocery shopping [108] utilize recommender systems to facilitate the shopping experience for their users. Different types of recommender systems, such as top-n [38], sequential [137], and session-based [61] systems, are deployed for different scenarios. In grocery shopping in particular, the Next Basket Recommendation (NBR) formulation is considered to be the most relevant and has gained increasing attention in the literature recently [49, 66, 117].

Next basket recommendation for grocery shopping. The NBR task is defined as recommending a group of items to a user based on their shopping history, where the history is a time-ordered sequence of baskets that they have purchased in the past. Each basket is a set of items with no particular order [165]. This formulation fits the grocery shopping setting well, where a user's purchase history occurs naturally in the form of such baskets. Two main characteristics of the grocery shopping scenario make it distinct from other retail domains: 1) users shop for grocery items repeatedly and on a regular basis, and 2) grocery items have a short life time and are repurchased frequently by the same user [97].

Previous work has studied the NBR problem within different algorithmic frameworks. Specifically, a number of neural network-based methods have been proposed

This chapter was published as 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 '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11–15, 2022*, pages 1240–1250. ACM, 2022.

to learn basket representations [65, 117, 138, 165], while nearest neighbor-based methods are dominant in modeling the personal history of users [49, 66]. In the grocery shopping domain, recent studies of NBR models has shown that nearest neighbor-based models that consider the characteristics of this domain and explicitly model the personal history of users significantly outperform neural network-based models that are mostly focused on basket representation and consider the previously purchased items only implicitly [66, 92]. The superiority of nearest neighbor-based models is further shown to be caused by their ability to correctly recommend items that have previously been purchased by a user, so-called *repeat items* [92].

Understanding repeat-consumption behavior. Inspired by previous work highlighting the importance of repeat items in grocery shopping [12, 49, 66, 92], we study the repeat consumption behavior across six public and proprietary datasets in the grocery shopping domain (Section 5.4). We find that different users exhibit different levels of repeat consumption in all datasets. As time goes by and more baskets are added to a user's history, the repeat consumption increases. In all datasets, most of the items are subject to repurchasing with a high probability. We further demonstrate what these findings mean in terms of predictive performance for NBR. Specifically, we discover that averaged over all datasets, over 54% of the performance in terms of recall comes from such repeat items: items that the users have already purchased in their history, which only account for 1% of the total items on average. This high performance indicates that a NBR model that is focused on previously purchased items can potentially achieve a high performance, while considering only a small number of candidate items per user. Section 5.4.3 shows that most of the performance of recommender systems in NBR stems from the minority of repeat items in the dataset.

Existing nearest neighbor-based models [49, 66] consider three main factors in recommending repeat items for the next basket: 1) the frequency of items bought in a users' history, 2) the recency of items bought in the history, and 3) similarities between users. Repurchasing patterns of items are *not* reflected in these models. For example, one user might buy milk in every basket, and buy apples every two-to-three baskets only. Moreover, similarities between items are also not taken into account. Considering item similarities has the potential to help the generalization power of a recommendation model, in cases where the consumption data is scarce [20].

ReCANet. We propose ReCANet, a repeat consumption-aware neural network that addresses the shortcomings of existing NBR models (Section 5.5). In addition to frequency and recency of items purchased by a user, and user similarities, ReCANet further considers item similarities and personal repeat consumption patterns. The intuition behind ReCANet is that the consumption pattern of an item by a user indicates whether the item will appear in the next basket or not. Additionally, the consumption pattern of that particular item by other users can also help the prediction. To this end, we use embedding layers to represent items and users, and model the consumption patterns with LSTM [62] layers. To evaluate the performance of ReCANet for NBR, we compare it with several state-of-the-art models on six datasets in terms of multiple standard ranking metrics (Section 5.6). The experimental results confirm the effectiveness of ReCANet (Section 5.7); it significantly outperforms the state-of-the-art models in terms of recall and nDCG. The improvements over state-of-the-art baselines are consistent

across all datasets.

To summarize, our contributions are as follows:

- We provide an empirical study on the characteristics of repeat consumption behavior across six grocery shopping datasets (Section 5.4). To the best of our knowledge, this is the first study of this kind.
- We propose ReCANet, a novel neural architecture for NBR in grocery shopping, focused on repeat items (Section 5.5).
- We demonstrate the effectiveness of ReCANet in comparison with state-of-the-art NBR models through experiments on three public and three proprietary datasets, where ReCANet is shown to significantly outperform the baselines (Section 5.7).
- We validate the importance of different components of ReCANet through an ablation study and conduct parameter and user sensitivity experiments to demonstrate its robustness for different hyper-parameters and user groups (Section 5.7).

5.2 Problem Formulation

The goal of the NBR task is to recommend a full basket composed of a list of items to the user for their next basket, based on the history of the items that they have purchased in the past. Such recommendations reduce the burden on users to proactively find items of interest every time they need to shop.

Formally, a basket is a set of items defined as $B = \{x_1, x_2, \ldots, x_t\}$, where $x_i \in I$ denotes an item from a set of items I. Given the history of purchases for user $u \in U$ as the sequence $B^u = [B^u_1, B^u_2, \ldots, B^u_n]$, where B^u_i is the i-th basket in the purchase history of user u and U denotes the set of users, the goal is to predict the items in the next basket of the user, i.e., B^u_{n+1}. For the basket history B^u, the recommendation model assigns a score to all items $x_i \in X$, and the top-k items are returned as the candidate items for the next basket.

One of the main use cases for the NBR task is grocery shopping, which is also the focus of this chapter. Users tend to purchase multiple items at a time, which corresponds to the concept of a shopping basket. Grocery shopping is repetitive; it is usually done on a regular basis. Moreover, many grocery items have a short life time, which leads to repeated purchases of these items [97].

The next basket of a user consists of both *repeat items*, i.e., the items that they have already purchased in the past, and completely new items that they have not purchased previously, i.e., *explore items*. A recent study on state-of-the-art methods in NBR on grocery shopping datasets [92] has shown that NBR methods that are biased towards repetition, i.e., that generate a predicted basket mostly from repeat items, significantly outperform explore-biased approaches. Explore-biased approaches methods are even repeatedly outperformed by the simplest model for NBR, which recommends the most frequently purchased items in a user's history. This calls for further studies aimed at understanding repeat consumption behavior in grocery shopping, as well as designing personalized NBR models that are able to effectively model this behavior.

In this chapter, we focus on the following two research questions as a step towards this direction:

RQ4.1: What are the characteristics of repeat consumption behavior in the grocery shopping domain?

RQ4.2: How can we design a NBR model that takes the personal repeat behavior of users into account?

In Section 5.4, we address RQ4.1 through an empirical study of six grocery shopping datasets. To answer RQ4.2, we propose ReCANet in Section 5.5, a personalized next basket recommender that models the repeat consumption behavior effectively.

5.3 Related Work

Next basket recommendation. There is growing interest in research into e-commerce in the information retrieval (IR) community [142]. As one of the core recommendation tasks in e-commerce, the NBR task has gained considerable attention in the recent years, where many studies focus on learning representations for sequence of baskets with neural networks. Yu et al. [165] use recurrent layers and inspired by word2vec in natural language processing, Wan et al. [148] introduce triple2vec for NBR. Correlations between items in baskets are used to recommend more coherent baskets in [85]. An encoder-decoder architecture using recurrent layers is proposed in [65]. Inspired by the transformer architecture, Sun et al. [138] leverage multi-head attention to learn a representation for a sequence of sets. Yu et al. [166] build a co-occurrence graph for items and use graph convolutions to learn item relationships in baskets. A contrastive learning framework is introduced in [117] to denoise basket generation by considering only the relevant items in the history. The focus in these works is on modeling item relations in baskets; personal preferences of users are considered only implicitly.

In another line of work, Hu et al. [66] propose TIFU-KNN, a nearest neighbor-based model for NBR that directly models the personal history of users. The model is shown to perform superior to strong neural network-based baselines designed for the task, demonstrating the importance of repeat behavior, i.e., recommending items that a user has purchased before. Similarly, Faggioli et al. [49] propose UP-CF@r, a simple model that combines the personal popularity with collaborative filtering, leading to strong performance for NBR. A recent survey on NBR models demonstrates the effectiveness of TIFU-KNN and UP-CF@r compared to more complex neural models [92]. While these models consider the frequency and the recency of items bought in the history, repurchasing patterns of items are not reflected in these models.

In this chapter, we propose a neural architecture that learns from the personal consumption patterns of users and recommends from the previously purchased items.

Understanding repeat behavior. People are creatures of habit [6, 110]. The tendency to repeat behavior has been uncovered in many different domains, from revisiting places to re-listening to the same songs [6, 23]. Reiter-Haas et al. [121] adopt a psychological theory of human cognition that models the operations of human memory for music re-listening prediction. Chen et al. [35] derive four generic features that influence people's

short-term reconsumption behaviors independent of domain and predict whether or not a user will perform a reconsumption at a specific time.

Repeat behavior has also been studied in the context of recommendation. Rappaz et al. [120] show that carefully modeling repeat consumption plays a significant role in achieving state-of-the-art recommendation performance on a video live-streaming platform. Repeat purchase recommendations on the Amazon.com website lead to an over 7% increase in product click through rate on its personalized recommendations page [24]. Ren et al. [122] propose a model with an encoder-decoder architecture for session-based recommendation, where the encoder has a separate component for modeling repeat consumption. Finally, Wang et al. [149] combine collaborative filtering with temporal point processes to recommend novel items and consumed items in a sequential recommendation setting.

What we add to this literature is a next basket recommendation model that focuses on recommending items from the users' past consumption history in the grocery shopping domain.

5.4 Empirical Study

In this section, we study the characteristics of repeat consumption behavior in grocery shopping. We describe six datasets that contain transaction data from online and physical grocery stores. We then analyze the repeat behavior across users, baskets, and items. We conclude this section by establishing an upper bound on the predictive performance for repeat-focused recommendation.

5.4.1 Dataset description

We use six grocery shopping datasets in this chapter. The datasets are described below and their statistics are summarized in Table 5.1. We leverage three types of datasets: "online" and "offline" refer to online shops and physical stores, respectively. "Multi-channel" is the case where users purchase goods from both online and offline shopping channels. Among them, three are publicly available: Dunnhumby,[1] ValuedShopper,[2] and Instacart.[3] We further use three proprietary datasets from a large food retailer in Europe, namely X-online, X-offline, and X-multi-channel.

All datasets contain transactions: which items are bought by which user at which time. All items bought in the same transaction are treated as a basket. In all datasets, we remove users who have less than three baskets and items that occur in less than five baskets in total. For every user, we sort the baskets by time. The last basket of every user is reserved for testing or validation, while the rest are treated as the training data. Table 5.1 shows the statistics for the training data after preprocessing.

"Avg. item per user" shows the number of unique items that a user has interacted with, averaged over all users. "Personal item ratio (%)" is the average number of items per user divided by the total number of items in the dataset, in percentage. "Avg. repeat

[1] https://www.dunnhumby.com/careers/engineering/sourcefiles
[2] https://www.kaggle.com/c/acquire-valued-shoppers-challenge/overview
[3] https://www.kaggle.com/c/instacart-market-basket-analysis

Table 5.1: Dataset statistics after preprocessing.

Type	Dataset	Users	Items	Baskets	Avg. item per basket	Avg. basket per user	Avg. item per user	Personal item ratio (%)	Avg. repeat ratio	Avg. explore ratio
Online	Instacart	197,523	47,389	3,122,453	10.08	16	65	0.14	0.60	0.40
	X-online	55,528	18,570	360,672	41.84	6	161	0.87	0.58	0.42
Offline	Dunnhumby	2,483	36,963	270,416	9.15	109	524	1.42	0.49	0.51
	ValuedShopper	9,601	13,271	1,068,285	8.87	111	371	2.80	0.71	0.29
	X-offline	62,789	23,379	791,971	11.15	13	96	0.41	0.36	0.64
Multi-channel	X-multi-channel	94,912	24,693	1,286,689	23.13	14	197	0.80	0.46	0.54

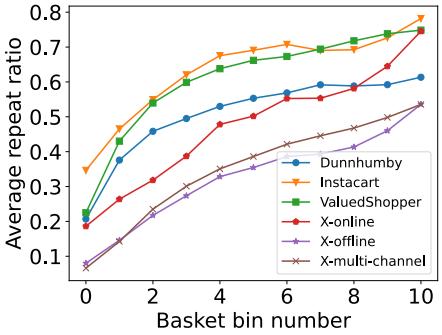


Figure 5.1: Distribution of repeat ratio in the baskets of users for all datasets.

ratio" shows the fraction of items in the last training basket of the users that have already appeared in their past, averaged over users. "Avg. explore ratio" shows the opposite, the fraction of new items.

The datasets vary in different aspects. In terms of history per user, Dunnhumby and ValuedShopper are very rich, with more than a hundred baskets per user on average. Instacart, X-offline and X-multi-channel have much less history for users, ranging from 13 to 16 basket per user on average, and X-online is scarce with only 6 basket per user on average. The number of users varies across datasets as well, with Instacart being the largest with 197K users and Dunnhumby being the smallest with 2K users. The basket size is similar for all datasets, except X-online and X-multi-channel which have larger basket sizes than the rest. The total number of items does not vary a lot across datasets, ranging from 13K to 47K.

The average repeat ratio differs between datasets, with ValuedShopper having the highest and X-offline having the lowest values. However, even the lowest repeat ratio is over 36%. Personal item ratio is less than 3% for all datasets, which shows that from all of the items that are present in the inventory, every individual user is only concerned with a very small fraction of them. Interestingly, this is also the case for the datasets with a rich history for users, such as Dunnhumby and ValuedShopper: even when a user's purchase history covers a long period, the number of unique items they interact with does not grow that much.

5.4.2 Characteristics of repeat consumption

In this section, we study the repeat consumption behavior in the grocery shopping datasets introduced above. We aim to gain insights that show what a recommendation model should focus on when using user and item data. In this section, we only consider the training baskets.

Repeat ratio across baskets. Repeat ratio in a basket is defined as the fraction of items in that basket that have already been purchased by the same user in the previous baskets. We study the repeat ratio in users' baskets to understand the stability of repeat consumption across time. Specifically, we are interested in analyzing how the repeat

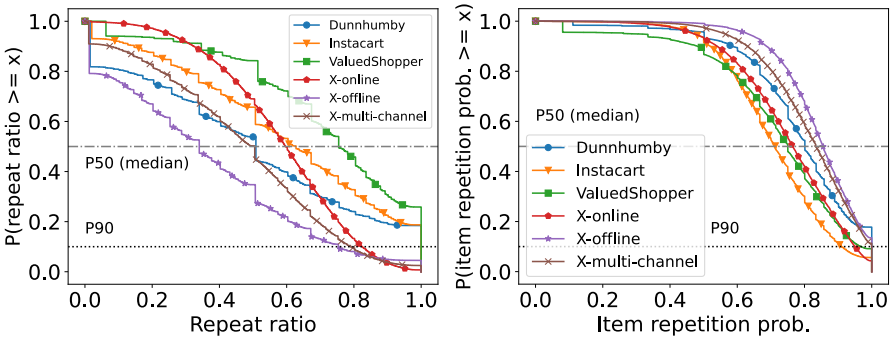


Figure 5.2: Cumulative distribution of repeat ratio in the last basket of users (Left) and repetition probability of items (Right) in all datasets.

ratio changes as more baskets are added to the history of a user. For each user, we distribute their baskets uniformly in 10 bins, where the smaller bin numbers correspond to the earliest baskets in the user's history. Given B_t^u as a basket in user u's history, where t indicates the index in the history sequence and n is the total number of u's baskets, the bin number for B_t^u is defined as $\lceil \frac{t}{bl_u} \rceil$, where bl_u corresponds to the bin length for u, defined as $\lceil \frac{n}{10} \rceil$.

Fig. 5.1 shows the distribution of the averaged repeat ratio for each bin across datasets. As expected, the repeat ratio grows as more baskets are added to the history. For the Instacart, Dunnhumby and ValuedShopper datasets, we see a jump at the start of the history. The jump at the start shows that after the first baskets of the user, when sufficient history has been gathered, the repeat ratio stabilizes to some extent. That is, when users have purchased goods for some time, their history is rich enough for a repeat-based recommendation. On the X-datasets, the repeat ratio is still increasing and has not reached the plateau point yet. This might be the result of the minimum preprocessing and preselection of users that we performed when creating these datasets, in contrast to the ones that are made public by others.

The repeat ratio reaches its highest value in the final baskets of the users. This means that the last baskets of the users in the history contain more repeat items than the earlier ones. This shows the importance of considering the recency: the next basket we aim to predict is closer to the last baskets of the users, in terms of repeat ratio, than to the early baskets.

Repeat ratio across users. To analyze the repeat ratio across users, we consider the repeat ratio in their last basket in the training data, which is the temporally closest one to the future basket we aim to predict. The cumulative distribution of repeat ratio in the last basket of training data is shown in Fig. 5.2 (Left). The distributions differ across datasets. On the Instacart, Dunnhumby and ValuedShopper datasets, 20 to 25 percent of users have a repeat ratio of 1: their last basket is entirely composed of items that they have already purchased before. On the ValuedShopper dataset, half of the users have a repeat ratio of more than 0.75, and for only 20 percent of the users the repeat ratio is less than 0.50. Instacart follows the same trend, with slightly lower values for the repeat

Table 5.2: Performance of the oracle personal NBR model.

Type	Dataset	Recall@$\|B\|$	nDCG@$\|B\|$
Online	Instacart	0.6087	0.6945
	X-online	0.6214	0.7236
Offline	Dunnhumby	0.4488	0.5331
	ValuedShopper	0.7168	0.7815
	X-offline	0.3811	0.4843
Multi-channel	X-multi-channel	0.4959	0.6055

ratio. On the other hand, in the X-offline and Dunnhumby datasets, around 20 percent of the users have a repeat ratio of zero: these are the users who will not benefit from repeat-based recommendations. This percentage is less than 10 for the other datasets. X-online has the smoothest distribution, with no long tail on either side of the spectrum.

Overall, we observe that not all users benefit to the same degree from a personalized recommendation, which is expected: users differ in terms of amount of history that we have for them, and they differ in their tendency for repeat consumption. However, even for the X-offline dataset as the most challenging dataset, over 30% of the last basket can be predicted based on the history, for more than 50% of the users.

Repeat ratio across items. So far, we have analyzed the repeat behavior from the users' perspective and across baskets. Another aspect to consider for repeat behavior is the item dimension. Items differ in their tendency to be repurchased. We define the *item repetition probability* as the number of users who have repurchased the item (a.k.a. purchased the item more than once) divided by the number of users who have purchased the item only once.

Fig. 5.2 (Right) shows the cumulative distribution of the probability of an item being repurchased for all items across datasets. Unlike the analysis on baskets and users, where the repeat ratio varied a lot across datasets, we observe very similar distributions for items. On all datasets, less than 10% of the items have a repetition probability of less than 0.5. The median ranges between 0.7 to 0.9, which means that most of the items in the grocery shopping datasets will be repurchased with a high probability.

5.4.3 Upper bound for next basket recommendation performance

Our study in Section 5.4.2 shows that the users' repeat behavior becomes more dominant as time passes. Furthermore, while being different for different users, repeat items account for a considerable fraction of items in baskets even when the average repeat ratio is low. Moreover, most of the items in grocery shopping datasets are repurchased frequently. In this section, we analyze how the repeat consumption behavior relates to the next basket recommendation problem. To this end, we consider a personal NBR model that only focuses on items that a user has purchased before. That means that instead of ranking all of the items in the dataset, the model only assigns scores to the users' previously purchased items.

What is the maximum performance of a personal NBR model? We design an oracle personal NBR model that achieves this performance as follows: given the items purchased in the history $B^u = [B_1^u, B_2^u, \ldots, B_n^u]$ of user u, the oracle recommends the ones that occur in B_{n+1}^u. Table 5.2 shows the performance of the oracle model on different datasets, in terms of Recall@$|B|$ and nDCG@$|B|$, where $|B|$ is the size of the target basket B_{n+1}^u. Recall@$|B|$ measures the fraction of items in the target basket that are present in the top $|B|$ items, and nDCG measures how high in the top $|B|$ list the relevant items are ranked. Repeat items, which account for less than 3% of all items in all datasets, are responsible for 38 to 62% of the oracle performance in NBR. The remaining 97% of items contribute to the oracle performance in NBR similarly: they account for the remaining 38 to 62% of performance. Averaged over all datasets, 54% of the oracle performance in NBR in terms of Recall@$|B|$ comes from only 1% of items! This indicates the importance of the repeat items, and the potential of a personalized recommendation model that is focused on those.

In answer to RQ4.1, we find that 1) users tend to buy more of the items they have purchased before as time passes, 2) repeat items remain a significant fraction of items in baskets on average, and 3) focusing on repeat items for next basket recommendation offers a great opportunity to improve the overall performance of recommender systems. Next, we introduce our proposed recommendation model that builds on these insights.

5.5 A Personalized Next Basket Recommendation Model

Based on insights from our empirical study, we design a network that is focused on repeat items and that models the personal consumption patterns in a user's history. We introduce ReCANet, a repeat consumption-aware neural network.

5.5.1 Architecture overview

The goal of the personalized NBR task is to decide which of the items that a user consumed in the past will appear in the next basket. We model it as a binary classification task, where the labels "positive" and "negative" correspond to an item being present in the target basket or not, respectively.

We design a neural model for this problem, whose architecture is shown in Fig. 5.3. The intuition is that the repurchase of an item is predictable from the past consumption pattern of the item by this user, other users, and other items. To model this, the network takes three inputs: item id x, user id u, and history vector with the length of window size w, corresponding to the consumption pattern of x by u. Formally, given user u, for every $x \in X_u$ where X_u is the set of all items user u has purchased in their history, ReCANet produces p_x^u, which is the probability of item x being in the next basket of u. At the first layer, item id and user id are embedded separately, concatenated, and fed to a feed-forward layer to create a user-item representation. The combination of the user-item representation and the history vector is then fed to a stack of LSTM layers, which models the temporal consumption pattern of item x by user u. The output of the last LSTM layer is the input for a stack of feed-forward layers, with a sigmoid activation at the last layer which generates the probability of item x appearing in the target basket.

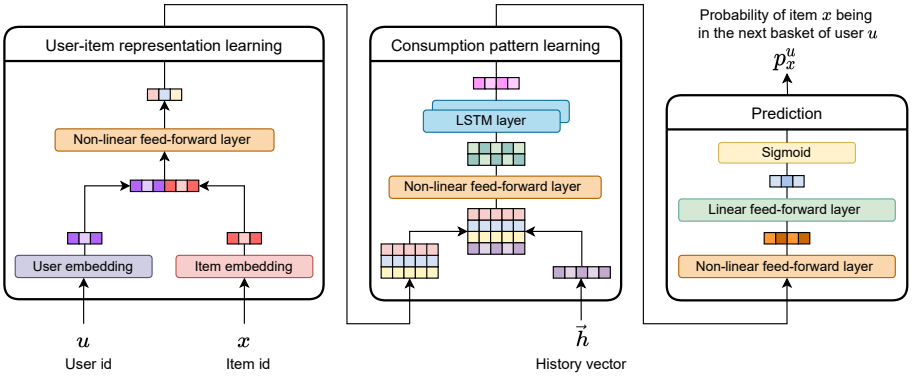


Figure 5.3: Overview of ReCANet's architecture.

5.5.2 Architecture details

The goal of ReCANet is to predict the probability of an item appearing in the next basket of a user. As explained above, to model this, ReCANet utilizes different components, which we will now discuss in detail.

Inputs. The network takes three inputs: 1) An item id x, which is a unique identifier for each item $x \in X$; 2) A user id u, similarly a unique identifier for each user $u \in U$; 3) A history vector $\vec{h} \in \mathbb{R}^w$, where window size w is a hyper-parameter of our model.

The history vector $\vec{h}$ is defined in a way to reflect the consumption history of item x by user u. As we aim to predict the occurrence of x in basket B_{n+1}^u, we consider the last w baskets in $[B_1, \ldots, B_n]$ that contain x. $[b_1, b_2, \ldots, b_w]$ denote the indices in the basket history sequence, that is $1 \leq b_i \leq n$. For $0 \leq i \leq w-1$, the i-th element in $\vec{h}$ is equal to $b_{i+1} - b_i$. For $i = w$, the i-th element in $\vec{h}$ is equal to $(n+1) - b_w$. If the number of baskets that contain x are less than w, the corresponding elements in $\vec{h}$ are filled with zero.

As an example, assume that user u has 99 baskets in their history, and we want to predict whether item x appears in the 100th basket. We know that x has appeared in baskets {15,29,45,78,95,97,99}. Given a window size of $w = 5$, vector $\vec{h}$ is equal to $[78-45, 95-78, 97-95, 99-97, 100-99] = [33, 17, 2, 2, 1]$.

The history vector $\vec{h}$ has three important properties: 1) it contains the frequency of consuming the item, as the number of nonzero elements in the vector are the representative of the purchase frequency; 2) it contains the recency information, since it is a sequence and the index in the sequence indicates how far in the history the item has been consumed; and 3) it contains information about the consumption pattern, as the values in the vector demonstrate the gaps between consecutive purchases of the item.

User item representation learning. Our goal is to predict the appearance of item x in the next basket of user u. As a result, we aim to model the user-item relationship. We separately embed users and items, and then combine the embeddings. Assume $W_i \in \mathbb{R}^{|I| \times d_i}$ is the trainable embedding matrix for items and $W_u \in \mathbb{R}^{|U| \times d_u}$ is the trainable embedding matrix for users, where d_i and d_u are the embedding dimensions. Given x

and u as item and user identifiers, the corresponding rows in the embedding matrices denote the item embedding $e_x \in \mathbb{R}^{d_i}$ and user embedding $e_u \in \mathbb{R}^{d_u}$, respectively. These embeddings allow the network to learn item and user representations that keep similar items and similar users close in the embedding space. The final user-item representation $e_{ui} \in \mathbb{R}^m$ is obtained as follows:

$$e_{ui} = \text{ReLU}((e_u \oplus e_i) \cdot W_1 + b_1),$$

where $W_1 \in \mathbb{R}^{(d_i+d_u)\times m}$ and $b_1 \in \mathbb{R}^m$ are trainable parameters, ReLU is the activation function, and m is a hyper-parameter of our model that denotes the hidden layer size, i.e., the model size. $\oplus$ denotes concatenation.

Consumption pattern learning. The history vector $\vec{h} \in \mathbb{R}^w$ contains the consumption pattern of item x by user u. In order to predict the repurchase probability, we first need to combine the user-item representation with the history vector. To this end, we replicate e_{ui} w times to obtain $e^w_{ui} \in \mathbb{R}^{w\times m}$. The combination representation $c_{ui} \in \mathbb{R}^{w\times m}$ is the computed as follows:

$$c_{ui} = \text{ReLU}((e^w_{ui} \oplus_r \vec{h}) \cdot W_2 + b_2), \tag{5.1}$$

where $\oplus_r$ denotes row-wise concatenation; that is, each element of $\vec{h}$ is concatenated with the corresponding row in e^w_{ui} (i.e., $(e^w_{ui} \oplus_r \vec{h}) \in \mathbb{R}^{w\times(m+1)}$). $W_2 \in \mathbb{R}^{(m+1)\times m}$ and $b_2 \in \mathbb{R}^m$ are trainable parameters. c_{ui} is a sequence of length w that contains information from both the user-item representation and the consumption history vector. To model sequential information, we use a stack of two LSTM layers. Formally:

$$\begin{aligned}
f_{j_t} &= \text{Sigmoid}(W_{f_j} x_t + U_{f_j} h_{t-1_j} + b_{f_j}) \\
i_{j_t} &= \text{Sigmoid}(W_{i_j} x_t + U_{i_j} h_{t-1_j} + b_{i_j}) \\
o_{j_t} &= \text{Sigmoid}(W_{o_j} x_t + U_{o_j} h_{t-1_j} + b_{o_j}) \\
\hat{c}_{j_t} &= \text{Tanh}(W_{c_j} x_t + U_{c_j} h_{t-1_j} + b_{c_j}) \\
c_{j_t} &= f_{j_t} \cdot c_{j_{t-1}} + i_{j_t} \cdot \hat{c}_{j_t} \\
h_{j_t} &= o_{j_t} \cdot \text{Tanh}(c_{j_t}),
\end{aligned} \tag{5.2}$$

where $j \in \{1, 2\}$ denotes the first and the second LSTM layer, respectively; $1 \le t \le w$ indexes the time step; $x_t \in \mathbb{R}^m$ denotes the input of the LSTM, which is equal to the corresponding time stamp in c_{ui} for the first layer, and the corresponding time stamp in cell state vector of the first layer (i.e., c_{1_t}) for the second layer. W_{f_j}, W_{u_j}, W_{o_j}, U_{f_j}, U_{u_j}, U_{o_j}, b_{f_j}, b_{u_j}, b_{o_j} are weight matrices and bias vector parameters that will be learned during training. The hidden state of the second layer, i.e., $h_{w_2} \in \mathbb{R}^m$, is used as the input for the prediction layers.

Prediction. We apply a two-layer feed-forward network with ReLU activation in between and a sigmoid activation at the end to produce the probability of item x appearing in the next basket of user u, with trainable parameters $W_{o_1}, W_{o_2} \in \mathbb{R}^{m\times m}$ and $b_{o_1}, b_{o_2} \in \mathbb{R}^m$:

$$p^u_x = \text{Sigmoid}(\text{ReLU}(h_{w_2} \cdot W_{o_1} + b_{o_1}) \cdot W_{o_2} + b_{o_2}). \tag{5.3}$$

5.5.3 Learning process

Training. We use the training baskets for creating training samples. For each user, we consider their last L baskets, where L is a parameter than can be set per user or per dataset. We set it per dataset. The intuition behind truncating the baskets is two-fold: 1) as shown in our empirical study, the last baskets of users are more similar to their upcoming basket in terms of repeat consumption behavior than their earlier baskets; and 2) the number of baskets per user varies, and this may result in unfair presence of users in training data. Limiting the number of baskets per user helps to mitigate this problem. For each basket B_t in the last L baskets of a user, the items that have been purchased in the previous $t-1$ baskets are the candidates for recommendation. Each item along with the user and its consumption history make up one training sample. To create labels for the binary classification task, the items that occur in B_t are the positive samples; the rest are negative ones. We use the binary cross-entropy loss for training the network.

Testing. The last basket of each user is used for testing or validation. The inputs of the network are generated with the same procedure as the training phase, where all items in the training baskets of the user are recommendation candidates. For each user, the items are then ranked based on the output of the network, and the top-k items are returned as the predictions for the next basket.

5.6 Experimental Setup

We introduce our baselines, data split scheme, evaluation metrics, parameter setting and implementation details.

Baselines. In addition to two simple baselines (P-POP, GP-POP), we compare ReCANet with two neighborhood-based models (TIFU-KNN, UP-CF@r), and one neural network (DNNTSP). These models have been shown to outperform other methods in NBR for grocery shopping and achieve state-of-the-art performance [92]:[4]

- P-POP: Recommends the most popular items in the user history, sorted by frequency of purchases. In NBR, P-POP is considered one of the strongest baselines.
- GP-POP: First uses P-POP to fill the predicted basket. If there are remaining empty slots (a.k.a., the basket size is larger than the number of items in the user's history), they will be filled with the most popular items in total.
- TIFU-KNN [66]: A nearest neighbor-based model that has been shown to outperform deep recurrent neural networks. The model relies on the similarity of the target user with other users and the history of the target user.
- UP-CF@r [49]: A collaborative filtering-based approach based on user-wise item popularity which also considers the recency of purchases.

[4]While a large number of NBR methods have been proposed in recent years, these are all outperformed by the methods that we have selected as baselines; see [66, 92].

- DNNTSP [166]: A deep neural network model that learns item relationships by constructing a co-occurrence graph and performs graph convolutions on the dynamic relationship graphs.

Data split. We follow the same procedure as Faggioli et al. [49]. The training data is composed of all baskets but the last one of all users. We randomly select 50% of the last baskets for the validation set, and the remaining ones for the test set. The validation set is used to select the best hyper-parameters of the methods, and the final results are reported on the test set.

Evaluation metrics. Following [66, 166], we use Recall@k and nDCG@k for evaluation. Recall is widely used in NBR, measuring how many of the items in the target basket are present in the predicted items for the next basket. nDCG is a ranking-based measure that takes into account the order of elements. Both measures are calculated across the predicted next basket for all test users. We report the metrics for $k \in \{10, |B|\}$, where $|B|$ stands for the length of the test basket B. We use 10 because there are limits for showing recommendations to users. We use $|B|$ to be able to compare performance across varying basket sizes and different datasets.

Parameter settings. We perform a grid search to find the hyper parameters that result in the best performance on the validation set, and use those for testing. We set the model size m to 64, sweep the embedding sizes d_u and d_i in $\{16, 32, 64, 128\}$ and the window size w in $\{5, 15, 25, 35, 45\}$. The hyper-parameters of the baselines are either tuned or set according to instructions in the papers if available. For TIFU-KNN we use the best parameters reported in the original paper [66] for the Instacart and ValuedShopper datasets. For UP-CF@r, we use the reported parameters on the Instacart and Dunnhumby dataset [49]. We tune these models on the rest of the datasets. DNNTSP is used with the parameter setting stated in [166] for all datasets, as there are no instructions for tuning.

Implementation details. ReCANet is implemented using the Keras framework with TensorFlow as backend. We use mini-batch stochastic gradient descent (SGD) together with the Adam optimizer to train the models. We set the batch size to 2048, and stop training when the loss on the validation set converges. To remove the random initialization effect, we run our model 10 times and report the average results. It is worth mentioning that the standard deviation is less than 0.001 in all cases. For users who have more than 100 baskets in the training data, we only keep their last 100 baskets for creating the training samples ($L = 100$), except on the ValuedShopper dataset, where we set the threshold to 50. We make our code public to facilitate reproducibility and follow up research.[5]

5.7 Results and Discussion

5.7.1 Overall performance

Table 5.3 shows the results of ReCANet alongside state-of-the-art baselines on all datasets. We observe that the performance of P-Pop and GP-Pop is identical in most

[5] https://github.com/mzhariann/recanet

Table 5.3: Results of ReCANet compared against the baselines. Boldface and underline indicate the best and the second best performing model among NBR models, respectively. Significant improvements of ReCANet over the best baseline results are marked with † (paired t-test, $p < 0.05$). ▲% shows the improvements of ReCANet against the best baseline.

Dataset	Metric	Baselines					ReCANet (▲%)
		P-Pop	GP-Pop	TIFU-KNN	UP-CF@r	DNNTSP	
Instacart	Recall@10	0.3222	0.3230	<u>0.3434</u>	0.3400	0.3427	**0.3592**† (4.6)
	nDCG@10	0.3131	0.3134	0.3332	0.3319	<u>0.3336</u>	**0.3502**† (5.0)
	Recall@$\|B\|$	0.2901	0.2904	<u>0.3115</u>	0.3077	0.3079	**0.3295**† (5.8)
	nDCG@$\|B\|$	0.3299	0.3301	<u>0.3521</u>	0.3479	0.3480	**0.3717**† (5.6)
X-online	Recall@10	0.1603	0.1603	0.1443	<u>0.1619</u>	0.1601	**0.1645**† (1.6)
	nDCG@10	0.6283	0.6283	0.5771	<u>0.6361</u>	0.6334	**0.6462**† (1.6)
	Recall@$\|B\|$	0.3613	0.3613	0.3009	<u>0.3680</u>	0.3679	**0.3735**† (1.5)
	nDCG@$\|B\|$	0.4323	0.4323	0.3731	<u>0.4396</u>	0.4391	**0.4467**† (1.6)
Dunnhumby	Recall@10	0.1315	0.1315	<u>0.1485</u>	0.1421	0.0892	**0.1621**† (9.2)
	nDCG@10	0.1267	0.1267	<u>0.1379</u>	0.1364	0.0825	**0.1489**† (8.0
	Recall@$\|B\|$	0.1048	0.1048	<u>0.1244</u>	0.1173	0.0576	**0.1310**† (5.3)
	nDCG@$\|B\|$	0.1216	0.1216	<u>0.1423</u>	0.1346	0.0685	**0.1503**† (5.6)
Valued-Shopper	Recall@10	0.2030	0.2030	<u>0.2194</u>	0.2161	0.1927	**0.2311**† (5.3)
	nDCG@10	0.2006	0.2006	<u>0.2142</u>	0.2096	0.1954	**0.2228**† (4.0)
	Recall@$\|B\|$	0.1735	0.1735	<u>0.1853</u>	0.1822	0.1647	**0.1958**† (5.7)
	nDCG@$\|B\|$	0.2032	0.2032	<u>0.2174</u>	0.2134	0.1929	**0.2282**† (5.0)
X-offline	Recall@10	0.1585	0.1587	0.1136	<u>0.1644</u>	0.1298	**0.1704**† (3.6)
	nDCG@10	0.2006	0.2007	0.1345	<u>0.2055</u>	0.1964	**0.2120**† (3.2)
	Recall@$\|B\|$	0.1569	0.1570	0.1090	<u>0.1619</u>	0.1593	**0.1675**† (3.5)
	nDCG@$\|B\|$	0.1882	0.1883	0.1282	<u>0.1933</u>	0.1899	**0.1998**† (3.4)
X-multi-channel	Recall@10	0.1170	0.1170	0.0797	<u>0.1181</u>	0.1064	**0.1203**† (1.9)
	nDCG@10	0.3628	0.3628	0.2375	**0.3673**	0.3607	<u>0.3648</u> (-0.7)
	Recall@$\|B\|$	0.2132	0.2133	0.1634	<u>0.2172</u>	0.2152	**0.2214**† (1.9)
	nDCG@$\|B\|$	0.2574	0.2575	0.1848	<u>0.2616</u>	0.2582	**0.2642**† (1.0)

cases, and in cases where a difference is observed (all metrics on Instacart and X-offline, Recall@$|B|$ and ndCG@$|B|$ on X-multi-channel) GP-Pop outperforms P-Pop with a very small margin that is not significant. This means that given 10 and $|B|$ as cut-off points for the recommendation list, users have enough items in their history that will surpass the threshold. However, the differences will likely be more substantial if larger thresholds are used, as suggested in [92].

The performance of P-Pop is the lowest in most cases, but it is in the same range as state-of-the-art methods. This result once again confirms the importance of personal

history in next basket recommendation. P-Pop as the simplest method for NBR should always serve as a baseline when a new model is proposed. This has not always been the case; in [117], for example, the authors do not compare their proposed method with P-Pop, and it has later been shown that P-Pop outperforms it in most cases [92].

DNNTSP is not the best baseline for any of the datasets and metrics, except for nDCG@10 on Instacart. The performance in all cases is in the same range with other methods, except on Dunnhumby, where there is a large gap between DNNTSP's performance and the others. We suspect that this is a result of the extremely low number of users in Dunnhumby, compared to the other datasets. DNNTSP has been tested on datasets with 10k to 100k users in the original paper [166], and additional measures such as hyper-parameter tuning might be required for a dataset with small number of users.

TIFU-KNN and UP-CF@r are the best performing baselines on all datasets and metrics, except for one case (nDCG@10 on Instacart). Both methods are nearest neighbor-based, and show superior performance to the state-of-the-art neural approach DNNTSP. These results are in line with findings in previous work [66, 92]; modeling the personal history is at least as effective as basket representation learning, if not more. The performance of UP-CF@r is more stable across datasets than TIFU-KNN; in cases where it is not the best baseline (Instacart, Dunnhumby, ValuedShopper) its performance does not differ significantly from TIFU-KNN. On the other hand, TIFU-KNN's performance drops on X datasets and even falls short of P-Pop. We suspect that this a result of non-optimal hyper-parameters on these datasets. The ranges for tuning the parameters given in the original paper are for Dunnhumby (but another version of the dataset), Instacart and ValuedShopper [66].

ReCANet significantly outperforms the best baseline in all datasets and across all metrics, except for one case (nDCG@10 on X-multi-channel), where it performs the same as UP-CF@r. The improvements range from 1.0% to 9.2%. X-online and X-multi-channel are the datasets with the lowest performance gains compared to the best baseline. These are the datasets with a large average basket size (42 and 23, compared to around 10 on other datasets). This indicates that predicting larger baskets is more challenging for ReCANet. Improving the ranking component of the network could mitigate this problem, which is left as future work.

5.7.2 Ablation study

We design an ablation study to investigate the effect of different components of ReCANet and compare it with four variations: 1) w/o user: the user embedding layer is omitted, and the item embedding alone is combined with history vector; 2) w/o item: the item embedding layer is omitted, and the user embedding alone is combined with history vector; 3) w/o user and item: the user-item representation learning module is omitted; and 4) w/o history: the consumption pattern learning module is omitted.

Table 5.4 shows the results. ReCANet w/o history has the lowest performance compared to the other versions, across all metrics and datasets. Omitting the consumption pattern learning module results in drops in performance ranging from 12.7% to 31.3%. This indicates that the history vector contains valuable information for next basket predication, and the designed module is effective in utilizing this information.

Table 5.4: Results of ablation study. Boldface and underline indicate the best and the second best performing model. ▼% shows the drop in performance compared to ReCANet.

Dataset	Metric	ReCANet	Variations			
			w/o user (▼%)	w/o item (▼%)	w/o user and item (▼%)	w/o history (▼%)
Instacart	Recall@10	**0.3592**	0.3585 (0.2)	0.3507 (2.4)	0.3506 (2.4)	0.3135 (12.7)
	nDCG@10	**0.3502**	0.3496 (0.2)	0.3406 (2.7)	0.3407 (2.7)	0.2976 (15.0)
	Recall@$\|B\|$	**0.3295**	0.3291 (0.1)	0.3203 (2.8)	0.3202 (2.8)	0.2807 (14.8)
	nDCG@$\|B\|$	**0.3717**	0.3712 (0.1)	0.3613 (2.8)	0.3612 (2.8)	0.3172 (14.7)
X-online	Recall@10	**0.1645**	0.1632 (0.8)	0.1609 (2.2)	0.1598 (2.9)	0.1155 (29.8)
	nDCG@10	**0.6462**	0.6409 (0.8)	0.6285 (2.7)	0.6238 (3.5)	0.4604 (28.8)
	Recall@$\|B\|$	**0.3735**	0.3712 (0.6)	0.3637 (2.6)	0.3626 (2.9)	0.2997 (19.8)
	nDCG@$\|B\|$	**0.4467**	0.4436 (0.7)	0.4342 (2.8)	0.4323 (3.2)	0.3441 (23.0)
Dunnhumby	Recall@10	**0.1621**	0.1615 (0.4)	0.1544 (4.8)	0.1526 (5.9)	0.1330 (18.0)
	nDCG@10	**0.1489**	0.1488 (0.1)	0.1447 (2.8)	0.1417 (4.8)	0.1206 (19.0)
	Recall@$\|B\|$	**0.1310**	0.1337 (-2.1)	0.1226 (6.4)	0.1250 (4.6)	0.1067 (18.5)
	nDCG@$\|B\|$	**0.1503**	0.1529 (-1.7)	0.1412 (6.1)	0.1428 (5.0)	0.1230 (18.2)
Valued-Shopper	Recall@10	**0.2311**	0.2309 (0.1)	0.2264 (2.0)	0.2257 (2.3)	0.2007 (13.2)
	nDCG@10	**0.2228**	0.2221 (0.3)	0.2209 (0.9)	0.2193 (1.6)	0.1949 (12.5)
	Recall@$\|B\|$	**0.1958**	0.1966 (-0.4)	0.1933 (1.3)	0.1925 (1.7)	0.1650 (15.7)
	nDCG@$\|B\|$	**0.2282**	0.2287 (-0.2)	0.2264 (0.8)	0.2244 (1.7)	0.1958 (14.2)
X-offline	Recall@10	**0.1704**	0.1692 (0.7)	0.1597 (6.3)	0.1598 (6.2)	0.1342 (21.2)
	nDCG@10	**0.2120**	0.2113 (0.3)	0.2009 (5.2)	0.2006 (5.4)	0.1575 (25.7)
	Recall@$\|B\|$	**0.1675**	0.1669 (0.4)	0.1579 (5.7)	0.1582 (5.6)	0.1280 (23.6)
	nDCG@$\|B\|$	**0.1998**	0.1990 (0.4)	0.1895 (5.2)	0.1893 (5.3)	0.1496 (25.1)
X-multi-channel	Recall@10	**0.1203**	0.1203 (0.0)	0.1166 (3.1)	0.1174 (2.4)	0.0826 (31.3)
	nDCG@10	<u>0.3648</u>	**0.3683** (-1.0)	0.3509 (3.8)	0.3548 (2.7)	0.2587 (29.1)
	Recall@$\|B\|$	<u>0.2214</u>	**0.2218** (-0.2)	0.2128 (3.9)	0.2144 (3.2)	0.1685 (23.9)
	nDCG@$\|B\|$	**0.2642**	0.2656 (-0.5)	0.2541 (3.8)	0.2564 (3.0)	0.1954 (26.0)

The smallest effect on performance comes from the user embedding layer, as illustrated in the results of ReCANet w/o user. The drops in performance are small, ranging from 0% to 0.8%. In some cases ReCANet w/o user even outperforms ReCANet, albeit with small margins. The user information is implicitly encoded in the model, in the way that we create the training and test data. That is, the positive and negative samples come from the personal history of users in the training data, and the personal items are ranked during testing. This results in a reduced need for explicitly modeling users via the user embedding layer. Item embeddings have a positive effect on performance across all metrics and dataset; ReCANet w/o item results in 0.9% to 6.4% drops in performance. Hence, modeling item information is important for the NBR task. Omitting the user-

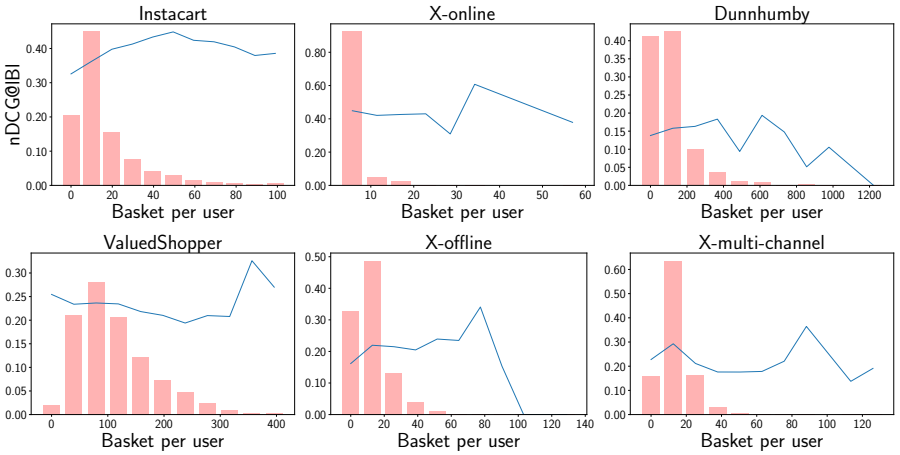


Figure 5.4: Performance of ReCANet in terms of nDCG@$|B|$ based on the number of baskets per user. Bars show the distribution of users.

item representation learning module all together has a similar effect on performance. As expected, in cases where user information is not helpful, the effect is lower than omitting the item embedding alone. In other cases where user information helps the performance, the effect of the user-item representation learning is more apparent.

In summary, all components of ReCANet contribute to its performance, and are essential to achieve the best overall performance.

5.7.3 User-level analysis

Table 5.3 shows the performance of ReCANet averaged over all users. In this section, we analyze the treatment effect for different groups of users [129]. We group users in three ways: 1) by the number of baskets per user, 2) by the number of unique items per user, and 3) by the average repeat ratio per user. Do different groups of users benefit from ReCANet in different degrees?

Fig. 5.4 shows the performance of ReCANet in terms of nDCG@$|B|$ w.r.t. the number of baskets per user for each dataset. The distribution of users based on their number of baskets is also shown with bars. All datasets are right-skewed; most of the users have relatively small number of baskets in their history. The performance shows no correlation with the number of baskets, and is robust across users. Sudden drops in performance on the Dunnhumby and X-offline datasets only appear at the far right side of the distributions, where the number of users is very low.

Fig. 5.5 shows the performance of ReCANet in terms of nDCG@$|B|$ w.r.t. to the number of unique items per user for each dataset. Similar to the number of baskets, the distribution of users w.r.t. to the number of unique items has a right-skewed shape, but it is closer to the normal distribution on X-online, ValuedShopper and X-multi-channel. The performance is robust w.r.t. the number of items on Instacart and X-offline, but there is a drop on Instacart for a small number of users in the tail. On Dunnhumby,

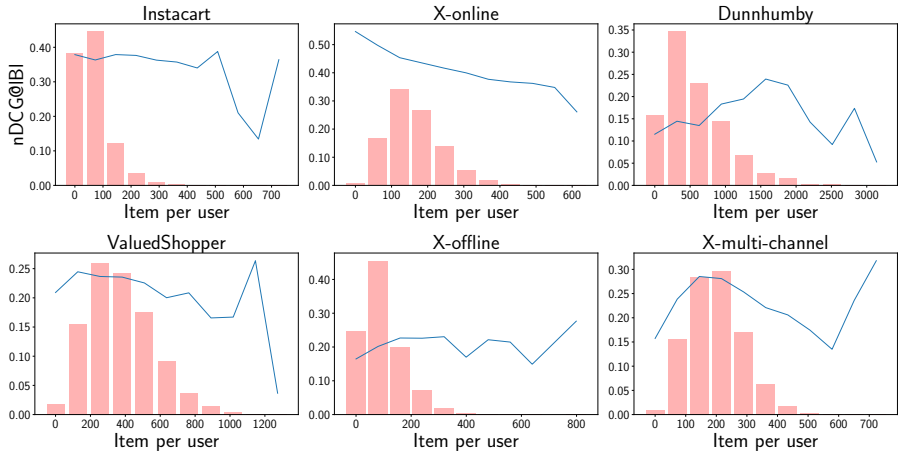


Figure 5.5: Performance of ReCANet in terms of nDCG@$|B|$ based on (the number of unique items per user. Bars show the distribution of users.

ValuedShopper and X-multi-channel, the performance increases with the increase in the number of items and decreases after a pick around 1500, 200, and 200 items, respectively. When the number of unique items per user is very small, the performance is low due to the fact that the user is still exploring; their history is not rich enough for personalized recommendation. When the number of items passes the peak point, the performance decreases; for users with many unique items, the final ranking is more subject to noise. ReCANet performs better for users with a higher number of unique items, as it results in a more accurate estimation of the users' taste.

Fig. 5.6 shows the performance of ReCANet in terms of nDCG@$|B|$ w.r.t. to the average repeat ratio in the training baskets per user for each dataset. The distribution of users is rather close to the normal distribution in all datasets; most of the users have a medium repeat ratio, with smaller numbers of users with a very high or very low repeat ratio. In all datasets, performance of ReCANet has a clear correlation with the repeat ratio; the more a user tends to repurchase, the bigger benefit they will get from the personalized recommendations. This also means that the average repeat ratio can serve as predictor for the expected performance of ReCANet. In that case, one can decide to explore non-personal items for recommendation to users with low repeat ratios.

5.7.4 Parameter sensitivity

We analyze the performance of ReCANet w.r.t. to three hyper-parameters: window size w, user embedding size d_u and item embedding size d_i. On all datasets, the performance is most sensitive to the window size. Small and large window sizes both degrade the performance; 15 and 25 result in the highest performance. This means that too little history makes it hard for the model the learn the consumption patterns, and too much history adds noise to the data that matters most. In contrast, the performance is not sensitive to the user and embedding size; the performance only changes marginally (less

Figure 5.6: Performance of ReCANet in terms of nDCG@$|B|$ based on the average repeat ratio in users' history. Bars show the distribution of users. Performance correlates with the average repeat ratio.

than 0.1%) when increasing the dimension size from 16 to 128. This indicates that we can use a small embedding size, which results in a smaller model size and less training time without loss of performance.

5.8 Conclusion

In this chapter, we have analyzed the repeat consumption behavior in grocery shopping. We addressed **RQ4** with two sub-questions. In answer to RQ4.1, we have found that repeat items, i.e., items that a user has previously purchased, have a high chance of reappearing in users' future baskets. We have focused on the next basket recommendation (NBR) problem, and proposed ReCANet to answer RQ4.2, a neural model for the task that learns from users' personal consumption patterns of items. Our experiments show that ReCANet, while focused on the repeat items that make up a small percentage of the total items in the inventory, is able to outperform state-of-the-art NBR models. This means that the repeat consumption behavior in grocery shopping is a strong indicator for future purchases, and explicitly modeling it leads to improvements in the recommendation performance.

Our experiments further show that not all users benefit to the same degree from the repurchase recommendations; the recommendation performance correlates with the average repeat ratio in the previous baskets of the users. In future work, we aim to extend ReCANet to help such users in discovering new items, while modeling the consumption behavior of repeat items at the same time. Moreover, while ReCANet significantly outperforms the best baselines, there is still a gap between ReCANet's performance and the performance of an oracle personalized NBR model. This indicates that the repeat-focused recommendation task is far from solved and there is still room

for improvement. Currently, item relations in baskets are only considered implicitly and through the training sample generation procedure in ReCANet. In future work, we aim to add a basket representation learning component, in the history baskets modeling phase as well as the final basket generation phase.

Next, in the final research chapter, we continue with the retail domain and study a slightly different recommendation scenario, that is within-basket recommendation.

6 Personalized Within-basket Recommendation

This chapter follows the previous chapter in that it also focuses on recommendation in retail, but it differs by tackling basket *completion* as a task, instead of basket *recommendation*. We address **RQ5**: How can we design an effective and scalable within-basket recommender system that explicitly considers users' personal preferences? We propose a neighborhood-based model and detail an efficient implementation, which allows for real-time inference in large scale environments.

6.1 Introduction

Recommender systems in retail help users to find the items that they need from large inventories in different domains [148]. In many shopping scenarios, such as grocery shopping, users purchase multiple items in a single transaction, which corresponds to a shopping *basket*. Within-basket recommendation is defined as recommending items for an incomplete shopping basket, which can reduce the burden on users to explore the inventory proactively, and result in shorter shopping times [89, 99]. This problem is relevant in both online and offline shopping. For an online platform, items can be recommended based on user activity and items that have previously been added to a basket. In brick-and-mortar grocery stores, RFID-tagged items and smart shopping carts allow real-time recommendation of items based on a user's smart cart [84].

The characteristics of recommendation in grocery shopping have been studied extensively in a closely related line of work, namely next basket recommendation [13, 66, 92]. These studies reveal the importance of the personal purchase history in grocery shopping. In particular, users shop for grocery items repeatedly and on a regular basis; grocery items have a short life-time and are repurchased frequently by the same user. Previous work on within-basket recommendation mostly focuses on learning a representation for the incomplete basket using factorization machines [84, 89], product

This chapter was published as M. Ariannezhad, M. Li, S. Schelter, and M. de Rijke. A personalized neighborhood-based model for within-basket recommendation in grocery shopping. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023, Singapore, 27 February 2023 – 3 March 2023*, pages 87–95. ACM, 2023.

embeddings [148], or graph neural networks [98, 99]. However, no prior work considers explicitly modeling personal preferences.

Recent studies show that nearest neighbor methods provide state-of-the-art performance in different recommendation settings, such as session-based recommendation [103, 105], next basket recommendation [66, 92], and session-aware recommendation [83]. Moreover, neighborhood-based recommendations are transparent and explainable [105]. Recent work on real-world recommender systems show that nearest neighborhood-based models scale to industry workloads [75]; the training time required for neural approaches has been found to be at least an order of magnitude larger than neighbor-based approaches [74].

We propose PerNIR, a *personalized nearest neighbor-based model for within-basket recommendation* that explicitly considers users' personal preferences (Section 6.4). PerNIR has two main components: 1) a personal component that explicitly models the personal preference of a user, and 2) a collaborative component, which leverages the neighboring users for scoring candidate items. In both components, items in the current incomplete basket are used as signals to find candidate items for recommendation. The main novelty of PerNIR is that it models the long-term and short-term interests of users at the same time in the personal component. The personal component uses the historical purchases to assign scores to items that the user has previously purchased, and leverages the items in the current basket as signals for the short-term interests. The collaborative component calculates how neighboring users would score the candidate items, given the items in the user's current basket. The personal and collaborative item scores are aggregated to make up the final recommendation scores. Additionally, we provide an optimized vectorized implementation of our proposed method in Section 6.5, which precomputes static parts of our models on a per user-basis and thereby provides low-latency inference performance.

In Sections 6.6 and 6.7, we conduct experiments on two public and private grocery shopping datasets to evaluate PerNIR. We compare PerNIR against 10 state-of-the-art baselines and find that it significantly outperforms all, with gains of over 12% in terms of hit rate over the second best performing approach. We conduct an extensive ablation study with nine model variations as well as a hyper-parameter sensitivity experiment. We analyze the performance along different dimensions, i.e., characteristics of the users' historical purchase behavior and the current incomplete basket. Moreover, we study the inference efficiency of PerNIR to demonstrate its effectiveness under real-world latency requirements.

6.2 Problem Formulation

The goal of the within-basket recommendation task is to recommend the next item that a user would add to their current incomplete basket, based on the items that are currently in the basket and the history of the items that they have purchased in the past [89, 98, 99]. One of the main use cases for the within-basket recommendation task is grocery shopping, which is the focus of this chapter. In grocery shopping, users tend to purchase multiple items at a time, which corresponds to a shopping basket [13, 66].

A basket is a list of items defined as $\mathbf{b} = [x_0, x_1, \ldots, x_t]$, where $x_i \in I$ denotes

an item from a set of items I, and x_i is the i-th item that the user has added to the basket $\mathbf{b}$. The sequence $\mathbf{B}^u = [\mathbf{b}_0^u, \mathbf{b}_1^u, \ldots, \mathbf{b}_n^u]$ denotes the history of baskets for a user $u \in U$. $\mathbf{b}_i^u$ is the i-th basket in the history of user u, and U is the set of all users. The goal is to predict the next item x_{t+1} to be added to the current incomplete basket $\mathbf{b}_{n+1}^u = [x_0, x_1, \ldots, x_t]$. For the basket history $\mathbf{B}^u$ and the new basket $\mathbf{b}_{n+1}^u$, the recommendation model assigns a score to all items $x_i \in X$, and the top-n items are returned as the candidates for the next item to be added to the basket.

6.3 Related Work

Our work is related to several lines of work in the recommender systems literature.

Within-basket recommendation. Although less studied than other forms and definitions of the recommendation task, there are a few papers that address the exact same problem as ours in the literature. As some of the first authors active in this area, Le et al. [84] propose a basket-sensitive factorization machine, which models the recommendation as a function of four associations between a user, a target item, and the items in a basket. Focused on the grocery shopping domain, Wan et al. [148] learn product embeddings using the co-occurrence of items in baskets and further use the learned embeddings for in-basket and next basket recommendation. Graph-based neural networks have also been studied in work associated with Walmart Labs, again focused on grocery [98, 99]. More recently, Li et al. [89] have proposed a deep learning-based model (DBFM, deep basket-sensitive factorization machine) to address the task. However, the order of items in a basket is ignored in this work.

Our work resembles the papers listed in terms of problem definition, domain, and datasets used. What we add is a highly effective, low-latency neighborhood-based model with a focus on personal history for within-basket recommendation in grocery shopping.

Session-based recommendation. A well-studied recommendation scenario is session-based recommendation (SBR), which refers to the task of recommending items for the next interactions in a given ongoing anonymous session. Various neural models have been proposed for the task [58, 88, 95, 111, 157, 160], as well as neighborhood-based models [56, 70]. Recent studies have revealed that neighborhood-based models, despite their sometimes simple nature, often perform equally well as, or even outperform, conceptually and computationally more complex deep neural models [103–105].

We build on successful ideas from neighborhood-based models in SBR in our work. However, in contrast to SBR, the identity of the user with an ongoing basket is assumed to be known for within-basket recommendation, which makes our problem space and solution different from existing work.

Personalized session-based recommendation. Analogous to within-basket recommendation, the goal in this scenario is to recommend items for an ongoing session, while the recommender system is aware of the user identifier (and the corresponding user history). Neural recommendation models are the dominant approach in this area, where recurrent neural networks [67, 113, 118, 126], attention networks [163], and graph neural networks [112, 168] have been proposed. A surprising recent study on

personalized session-based recommender systems has revealed that such neural models are not better than approaches that do not use users' personal history, and extensions of neighborhood-based models proposed for SBR consistently outperform recent neural techniques [83].

None of these publications consider the grocery shopping domain for recommendation, which has been shown to have specific characteristics, such as the importance of a user's personal history [13, 66]. We focus on this domain, and propose a neighborhood-based model that is designed for within-basket recommendation.

Next basket recommendation. In this scenario, the goal is to recommend a list of items to a user, based on their history of previously purchased shopping baskets. This definition is similar to within-basket recommendation, as the history is in the form of baskets, and is mostly studied in the grocery shopping domain. Similar to SBR, neighborhood-based methods [49, 66] and neural methods [13, 85, 117, 165] have both been examined and show strong performance in next basket recommendation (NBR) [92]. In within-basket recommendation, however, the goal is to recommend items to *complete* an incomplete basket, which already contains some items that can be utilized for recommendation.

In this work, we make use of ideas that have been shown to be effective in NBR for grocery shopping and adapt them to our problem setting by using both the personal history and the current basket for computing the recommendation scores.

6.4 Model

Next, we introduce PerNIR, our personalized neighborhood-based model for within-basket recommendation in grocery shopping. We introduce the personal scoring function, the neighborhood-based scoring, as well as our final scoring function, which combines them.

6.4.1 Personal scoring function

Recent studies have shown the importance of users' personal history in grocery shopping [13, 66, 92]. However, they consider the next basket recommendation task as their main goal, where there is no information about a current basket $\mathbf{b}^u_{n+1}$. For simplicity, we use the notation $\mathbf{b}_c$ as the current basket that we aim to recommend items for as an alias for $\mathbf{b}^u_{n+1}$. We propose the following personal scoring function $PSF(x, u, \mathbf{b}_c)$ to compute the score of a candidate item x for the current incomplete basket $\mathbf{b}_c$ of user u:

$$PSF(x, u, \mathbf{b_c}) = \alpha \cdot HSF(x, u) + (1 - \alpha) \cdot BSF(x, u, \mathbf{b_c}), \tag{6.1}$$

where $HSF(x, u)$ is a history-based scoring function that scores an item x based on how the user u has consumed it in the past, regardless of the current basket. $BSF(x, u, \mathbf{b}_c)$ is a basket-based scoring function that considers both the history and the items in the current basket for assigning a score to x. Finally, α is a hyper-parameter to balance the two parts.

History-based scoring. $HSF(x, u)$, the history-based scoring function, accounts for the loyalty of a user to an item. In other words, it considers how frequently a user has

purchased the item in the past. In addition to frequency, the recency of occurrences of an item is also important in predicting the next occurrence, as shown in [13, 49]. We define $HSF(x, u)$ as follows:

$$HSF(x, u) = \sum_{t=0}^{|\mathbf{B}_u|-1} \frac{1_{\{x \in \mathbf{b}_t\}}}{|\mathbf{B}_u| - t}, \tag{6.2}$$

where $1_{\{x \in \mathbf{b}_t\}}$ is equal to one if x occurs in $\mathbf{b}_t$ and zero otherwise. In this formulation, more recent occurrences of an item in the user history are given more weight.

Basket-based scoring. To compute the score of a candidate item x given the items that are already in the current basket, we define the basket-based scoring function $BSF(x, u, \mathbf{b}_c)$. It scores an item x based on its past co-occurrences with the items in the user's past baskets. It takes into account three factors while scoring an item: 1) recent baskets are more informative than older baskets, 2) recent items in the current basket are more important than older items, and 3) the distance between the candidate item x and the current item x_i in the past baskets is important. Formally:

$$BSF(x, u, \mathbf{b}_c) = \sum_{i=0}^{|\mathbf{b}_c|-1} \frac{1}{|\mathbf{b}_c| - i} \cdot \sum_{t=0}^{|\mathbf{B}_u|-1} \frac{1_{\{x \in \mathbf{b}_t\}}}{|\mathbf{B}_u| - t} \cdot \frac{1_{\{x_i \in \mathbf{b}_t\}}}{|I(x, \mathbf{b}_t) - I(x_i, \mathbf{b}_t)|}, \tag{6.3}$$

where $I(x, \mathbf{b}_t)$ indicates the index at which x has occurred in basket $\mathbf{b}_t$. In other words, more weight is given to a candidate item that has appeared in close proximity of current items in the recent baskets of the user. The first two factors have already been shown to be effective for session-based recommendation in [56], where they are used to compute the similarity of the current session with all existing sessions. Here, however, we compute the similarity with the personal baskets of the user for scoring candidate items.

6.4.2 Neighbor-based scoring function

In addition to the personal scoring function, our model has a collaborative component that assigns scores to candidate items according to the neighboring users of the target user u. Specifically, the neighbor-based scoring function $NSF(x, u, \mathbf{b}_c)$ is defined as follows:

$$NSF(x, u, \mathbf{b}_c) = \frac{1}{|N_u|} \sum_{v \in N_u} sim(u, v) PSF(x, v, \mathbf{b}_c), \tag{6.4}$$

where N_u is the set of neighbors of u of size k and $sim(u, v)$ is the similarity between users u and v. Note that the number of neighbors k is a hyper-parameter in our model. The neighbor-based scoring function $NSF(x, u, \mathbf{b}_c)$ calculates how an item x would be scored by the neighbors of user u, given the items already in the basket. The similarity between a neighbor and the target user is also considered.

User similarity. To compute the similarity $sim(u, v)$ between two users u and v, we consider the history-based scoring from Eq. 6.2. We represent each user u as a vector over the item space I, where each element in the vector, corresponding to an item $x \in I$, is the history-based score of the item for the user. The similarity is then defined as the

cosine similarity between two user vectors. Formally:

$$sim(u,v) = \frac{\sum_{x \in I} HSF(x,u) HSF(x,v)}{\sqrt{\sum_{x \in I} HSF(x,u)^2} \sqrt{\sum_{x \in I} HSF(x,v)^2}}. \tag{6.5}$$

The set of k nearest neighbors of a user u, containing the most similar users in U to u, is then defined as N_u.

6.4.3 Final scoring function

The final score of a candidate item x for an incomplete basket $\mathbf{b}_c$ belonging to user u, is calculated as follows:

$$score(x,u,\mathbf{b}_c) = \beta \cdot PSF(x,u,\mathbf{b}_c) + (1-\beta) \cdot NSF(x,u,\mathbf{b}_c), \tag{6.6}$$

where β is a hyper-parameter of our model, balancing the contributions of the personal history component and the collaborative component. The model assigns a score to all items in $I \setminus \mathbf{b}_c$, and the top n scored items are returned as the recommendation list.

6.5 Vectorized Implementation

Analogous to session-based recommendation, within-basket recommendation models are challenging to deploy in real-world scenarios, as they need to respond online to users filling their baskets. E-commerce applications typically need to respond with a latency of less than 50 milliseconds in at least 90 percent of requests [75], and it has been observed that low prediction latency contributes to the acceptance of recommendations by users [7, 74].

The design of PerNIR supports such low-latency prediction scenarios. Major parts of PerNIR depend on the static user history $\mathbf{B}_u$ (which only changes after purchases), while only the incomplete basket $\mathbf{b}_c$ changes dynamically during the shopping session. In the following, we describe how to exploit this structure to precompute the parts of the model that only depend on the static user history $\mathbf{B}_u$ and how to vectorize the underlying computations to score multiple candidate items at once. Note that we model a basket $\mathbf{b} \in \{0,1\}^{|I|}$ as a binary vector in item space for these purposes.

6.5.1 Offline precomputation

For each user u, we can precompute the outputs of HSF (Eq. 6.2) into a user-specific history vector $\mathbf{h}_u$:

$$\mathbf{h}_u = \sum_{t=0}^{|\mathbf{B}_u|-1} \frac{1}{|\mathbf{B}_u| - t} \mathbf{b}_t. \tag{6.7}$$

We analogously precompute the item co-occurrences (the second sum from Eq. 6.3) from their basket history matrix into a user-specific basket co-occurrence matrix $\mathbf{C}_u$.

Each entry (i, j) of this matrix denotes the weight assigned to the co-occurrence of two items i and j in the basket history $\mathbf{B}_u$:

$$\mathbf{C}_u = \left[\sum_{t=0}^{|\mathbf{B}_u|-1} \frac{1_{\{x_i \in \mathbf{b}_t\}} \, 1_{\{x_j \in \mathbf{b}_t\}}}{(|\mathbf{B}_u| - t) \; |I(x_i, \mathbf{b}_t) - I(x_j, \mathbf{b}_t)|} \right]_{ij}. \tag{6.8}$$

Note that $\mathbf{C}_u \in \mathbb{R}^{|I| \times |I|}$ is high-dimensional but extremely sparse, as its number of non-zeros is at most the sum of the squares of the number of distinct items per basket in the user history. Next, we precompute the top-k similar users N_u for a user u (which only depend on the users' static history vectors). Based on the corresponding similarities, we precompute the combination $\mathbf{h}_{s_u}$ of the user's history vector $\mathbf{h}_u$ with the history vectors of their neighbors $v \in N_u$, and analogously precompute the combination $\mathbf{C}_{s_u}$ of their co-occurrence matrices:

$$\mathbf{h}_{s_u} = \beta \, \mathbf{h}_u + \frac{1-\beta}{|N_u|} \sum_{v \in N_u} sim(u, v) \, \mathbf{h}_v \tag{6.9}$$

$$\mathbf{C}_{s_u} = \beta \, \mathbf{C}_u + \frac{1-\beta}{|N_u|} \sum_{v \in N_u} sim(u, v) \, \mathbf{C}_v. \tag{6.10}$$

6.5.2 Online inference

At inference time, we compute a "summation and selection" vector $\phi(\mathbf{b}_c)$ from the dynamically changing incomplete basket $\mathbf{b}_c$, which contains the associated weights from the first sum of Eq. 6.3:

$$\phi(\mathbf{b}_c) = \left[\frac{1_{\{x_i \in \mathbf{b}_c\}}}{|\mathbf{b}_c| - I(x_i, \mathbf{b}_c)} \right]_i. \tag{6.11}$$

We can now efficiently compute item scores based on $\phi(\mathbf{b}_c)$ and the precomputed "personal" model $\mathbf{h}_{s_u}$ and $\mathbf{C}_{s_u}$ for user u:

$$score(u, \mathbf{b}_c) = \alpha \, \mathbf{h}_{s_u} + (1 - \alpha) \, \mathbf{C}_{s_u} \, \phi(\mathbf{b}_c). \tag{6.12}$$

Note that the scoring computation only requires a single sparse matrix vector multiplication and a single sparse vector addition, for which we can leverage highly optimized implementations from SparseBLAS [47].

6.6 Experimental Setup

We address the following research questions (RQs):

RQ5.1: How does PerNIR perform compared with existing state-of-the-art models for within-basket recommendation in grocery shopping?

RQ5.2: What are the effects of the different components HSF, BFS, and NFS of PerNIR and hyper-parameters α, β, and the number of neighbors k on the overall performance?

Table 6.1: Dataset statistics after preprocessing.

Dataset	Users	Items	Baskets	Avg. item per basket	Avg. basket per user
Instacart	30,000	43,936	413,860	11.9	13.8
X-online	10,000	22,486	141,342	38.2	14.1

RQ5.3: How sensitive is PerNIR's performance to characteristics such as the number of baskets per user or the number of items in the current basket?

RQ5.4: How efficient is PerNIR in terms of inference latency?

To answer our research questions we consider two experimental setups.

6.6.1 Setup for effectiveness experiments

To answer RQ5.1–RQ5.3 we design a set of contrastive experiments with a diverse set of state-of-the-art baselines.

Datasets. We use two datasets in our experiments. While other grocery shopping datasets do exist [92], they do not contain the add-to-basket order data which is crucial for within-basket recommendation [98, 99]. The datasets are described below and their statistics are summarized in Table 6.1. Instacart[1] is publicly available, and X-online comes from a large food retailer in Europe. We randomly sample 30,000 users from Instacart and 10,000 users from X-online and retrieve all baskets of the users. In both datasets, we remove users with less than three baskets, items occuring in less than five baskets in total, and baskets with less than four items.

Baselines. In addition to the simple baseline P-POP, we compare PerNIR with within-basket recommendation models (BasConv, MITGNN), session-based recommendation models (VSKNN, STAN), a personalized session-based recommendation model (HG-GNN), and next basket recommendation models (TIFU-KNN, ReCANet):

- P-POP: Recommends the most popular items in the user history, sorted by their frequency of purchases. P-POP is considered one of the strongest baselines in NBR for grocery shopping [13], and we consider it as a baseline for our task as well.
- BasConv [99]: Defines a basket entity to represent the basket intent and models the recommendation task as a basket-item link prediction task in the user-basket-item graph. It utilizes graph convolutional networks to learn representations.
- MITGNN [98]: An approach based on graph neural networks to model multiple intents in the baskets.
- VSKNN [103]: A neighborhood-based model proposed that puts a strong emphasis on more recent events of a session when computing session similarities.

[1] `https://www.kaggle.com/c/instacart-market-basket-analysis`

- STAN [56]: A neighborhood-based model that considers the position of an item in the current session, the recency of a past session w.r.t. to the current session, and the position of a recommendable item in a neighboring session when computing session similarities.
- HG-GNN [112]: A graph-augmented hybrid encoder that consists of a heterogeneous graph neural network and a personalized session encoder to generate a session preference embedding for personalized session-based recommendation.
- TIFU-KNN [66]: A nearest neighbor-based model that outperforms deep recurrent neural networks in NBR. The model relies on the similarity of the target user with other users and the purchase history of the target user.
- ReCANet [13]: A repeat consumption-aware neural network that explicitly models the repeat consumption behavior of users in order to predict their next basket.

In the case of NBR baselines, we considered two setups: discarding the current basket and treating it as the final basket in the history. The former gave the best performance results and we report those. In case of session-based and personalized session-based baselines, each session is considered as a basket. We further use two modified session-based models, namely Personalized SKNN (PSKNN) and Personalized STAN (PSTAN). In these two models, the neighboring baskets are the personal baskets of the user.

Data split. We follow the same procedure as Latifi et al. [83]. For every user, we sort the baskets by purchase time and use the last basket of each user as test data. The second-to-last basket is used as validation data to tune the parameters. The remaining baskets are considered as training data. Table 6.1 shows the statistics for the training data after preprocessing. Each test or validation basket produces several test samples. Specifically, the first three items in the basket are used as the seed for the current basket, and the rest of the items are added iteratively to the current basket and the next item is used as the ground truth item. Evaluation is performed for each non-seed item in the basket. For example, given a test or validation basket $b = [x_1, x_2, \ldots, x_6]$, the evaluation is performed on the following samples: $\{X = [x_1, x_2, x_3], Y = [x_4]\}$, $\{X = [x_1, x_2, x_3, x_4], Y = [x_5]\}$,$\{X = [x_1, x_2, x_3, x_4, x_5], Y = [x_6]\}$.

Evaluation metrics. We use Hit Rate (HR) @n and Mean Reciprocal Rank (MRR) @n as evaluation metrics. HR measures if the relevant item appears in the list of recommendations, and MRR measure how high the relevant item is ranked. Both measures are calculated across the predicted next item for all test users and all test samples. We report the metrics for $n \in \{10, 20\}$.

Parameter settings. We perform a grid search to find the hyper parameters that result in the best performance on the validation set, and use those for testing. The hyper-parameters of the baselines are either tuned or set according to instructions in the original papers if available.

6.6.2 Setup for efficiency experiments

To answer RQ5.4, we use an alternative experimental design. Our goal will be to showcase that PerNIR is able to handle inference workloads with a latency low enough

for production workloads, where users browse a site and fill their baskets in response to online recommendations. We choose the Instacart dataset for this experiment as it is larger than the X-Online dataset in terms of users, items and baskets, and therefore more challenging for inference workloads.

We run this experiment in a single thread on a `StandardD8v4` instance in the Microsoft Azure cloud, with an Intel Xeon Platinum 8272CL CPU@2.60GHz and 32gb of RAM, using Ubuntu 20.04, Python 3.9 and scipy 1.9.0.

We pick 1,000 users at random from the Instacart dataset, and ask PerNIR to score items for five randomly chosen incomplete test baskets per user. We base the prediction on the precomputed "personal" model for each user (as discussed in Section 6.5). We measure the response time in milliseconds, and repeat this experiment for increasing numbers of neighbors k, ranging from 20 to 500.

Reproducibility. To facilitate reproducibility and follow-up research, we share our code.[2]

6.7 Results and Analysis

We conduct extensive experiments to answer our research questions. In this section, we describe the results of our experiments.

6.7.1 Performance comparison (RQ5.1)

We compare the performance of PerNIR with several state-of-the-art baselines, and list the results in Table 6.2. First, we observe that HG-GNN is the baseline with the weakest performance. This model is originally proposed for personalized session-based recommendation, which is identical to within-basket recommendation in terms of problem formulation. However, the grocery shopping domain has special characteristics, such as repeat behavior and larger baskets compared to online sessions, which are not considered by HG-GNN and other personalized session-based recommendation models.

The GNN-based within-basket recommendation models BasConv and MITGNN perform poorly. One possible reason for this is that these models are trained on large baskets only (more than 40 items) in the original papers, which might limit their ability to generalize to our setting, where we set the lower bound of the basket size to four items. Moreover, these models are trained for predicting the rest of the basket where 80% of the items in the current basket are already given, which is again different from our setting. We add items to the baskets one by one and use each item as a label during evaluation, which is closer to the real-world setting where users receive recommendations after each item is added to the basket.

The session-based recommendation models VSKNN and STAN in their original formulation perform better than the previous models, but fall short of their modified personalized versions. This shows that the user personal history is an important indicator in within-basket recommendation. VSKNN and STAN focus on finding items from baskets that are similar to the current incomplete basket, and cannot utilize the personal

[2] `https://github.com/mzhariann/pernir`

Table 6.2: Results of PerNIR compared against the baselines. Boldface and underline indicate the best and second best performing model, respectively. Significant improvements of PerNIR over the best baseline are marked with † (paired t-test, $p < 0.05$).

		Instacart				**X-Online**			
Recommendation model type	**Model**	HR@10	MRR@10	HR@20	MRR@20	HR@10	MRR@10	HR@20	MRR@20
Within-basket	BasConv	0.0699	0.0268	0.1070	0.0293	0.0385	0.0151	0.0602	0.0166
	MITGNN	0.0672	0.0258	0.1026	0.0283	0.0379	0.0151	0.0606	0.0166
Personalized session-based	HG-GNN	0.0353	0.0125	0.0551	0.0139	0.0499	0.0222	0.0726	0.0237
Session-based	VSKNN	0.1105	0.0426	0.1637	0.0462	0.0829	0.0312	0.1271	0.0342
	STAN	0.0914	0.0385	0.1280	0.0410	0.0975	0.0477	0.1282	0.0498
	PVSKNN	0.2372	0.0915	0.3354	0.0983	0.1839	0.0742	0.2621	0.0796
	PSTAN	0.2142	0.0871	0.3115	0.0938	<u>0.2009</u>	<u>0.1049</u>	0.2607	<u>0.1090</u>
Next basket	PPOP	0.2224	0.0846	0.3210	0.0914	0.1765	0.0695	0.2557	0.0750
	TIFUKNN	0.2069	0.0799	0.2843	0.0853	0.1555	0.0600	0.2267	0.0649
	ReCANet	<u>0.2550</u>	<u>0.0977</u>	<u>0.3585</u>	<u>0.1049</u>	0.1835	0.0734	<u>0.2665</u>	0.0791
	PerNIR	**0.2592**†	**0.1044**†	**0.3625**†	**0.1115**†	**0.2258**†	**0.1069**†	**0.3006**†	**0.1120**†

preferences of users. By limiting the search for similar baskets to the personal baskets, PVSKNN and PSTAN get a boost in performance. Specifically, PSTAN is the best performing baseline on X-Online based on three out of four metrics, and PVSKNN is the second-best performing one on Instacart. This shows that the components in VSKNN and PSTAN are able to utilize the items in the current basket for scoring candidate items.

Next basket recommendation models provide the best baseline performance on both datasets. Among them, ReCANet is the strongest competitor, followed by PPOP and TIFUKNN. This is in line with the results observed for NBR [13]. NBR models focus on utilizing personal purchasing history, but are unable to use the signals in the current incomplete basket. These models demonstrate higher performance compared to the performance of session-based recommendation models that use only the incomplete basket but not the personal history. This result once again confirms the importance of the personal history in grocery shopping.

PerNIR outperforms all the baselines on both datasets, and the improvements over the best performing baseline are statistically significant on all metrics, using a paired t-test. The improvements on Instacart are 1.6% and 1.1% for HR@10 and HR@20, respectively. Larger improvements are obtained on MRR@10 and MRR@20, namely 6.8% and 6.2%. Improvements on X-Online are even more substantial, especially in terms of HR. We observe 1.9% and 2.7% increases in terms of MRR@10 and MRR@20 with respect to PSTAN, as well as 12.3% and 12.7% improvements over PSTAN and ReCANet in terms of HR@10 and HR@20. Overall, the results show that PerNIR is effective for within-basket recommendation.

6.7.2 Ablation study (RQ5.2)

We conduct an extensive ablation study with nine different variations of PerNIR to analyze the effect of different components in the model. Table 6.3 shows the results. First, we observe that the final model is superior to all variations on both datasets for all metrics, with the exception of one case (HR@20 on variation 6, where there is a 0.1% performance degradation). This indicates that all of the components in PerNIR contribute to the final performance and are necessary to achieve the best performance.

The first two variations concern the effect of HSF and BSF in the personal scoring function. By setting α to zero, we remove the HSF and with setting it to one, we remove BSF. Both cases result in a drop in performance for all metrics on both datasets. However, removing BSF has a more substantial effect on performance ranging from 3.0% to 8.5% on Instacart and 16.4% to 35.1% on X-Online. The effect is considerably higher on X-Online; one reason for this could be the larger basket sizes on X-Online, which translates to more signals being lost when the current basket is ignored.

In variations 3 and 4, we remove the personal scoring function PSF and neighboring scoring function NSF by setting β to zero and one, respectively. In both cases the performance degrades for all metrics and datasets. The effect of removing PSF is substantially larger than of removing NSF, where the performance drop ranges from 43.1% to 48.7% on Instacart and 58.4% to 63.6% on X-Online. This indicates the importance of modeling personal history in PerNIR. Removing NSF results in 1.6% to 2.3% and 3.5% to 5.4% drops in performance on Instacart and X-Online, respectively.

Table 6.3: Results of the ablation study. Different variations of PerNIR are compared with each other and the final model.

		Instacart				**X-Online**			
#	**Model variation**	HR@10	MRR@10	HR@20	MRR@20	HR@10	MRR@10	HR@20	MRR@20
1	w/o history-based scoring ($\alpha = 0$)	0.2525	0.1011	0.3544	0.1081	0.2138	0.1021	0.2867	0.1072
2	w/o basket-based scoring ($\alpha = 1$)	0.2465	0.0955	0.3513	0.1028	0.1764	0.0693	0.2513	0.0745
3	w/o personal history ($\beta = 0$)	0.1363	0.0594	0.1859	0.0628	0.0880	0.0389	0.1249	0.0414
4	w/o neighbor scoring ($\beta = 1$)	0.2548	0.1020	0.3542	0.1089	0.2145	0.1031	0.2841	0.1079
5	w/o user similarity weighting	0.2564	0.1032	0.3569	0.1101	0.2047	0.0936	0.2778	0.0987
6	w/ binary user similarity scoring	0.2586	0.1035	0.3628	0.1107	0.2166	0.1042	0.2873	0.1091
7	w/o basket recency	0.2481	0.0986	0.3496	0.1056	0.2120	0.1011	0.2801	0.0989
8	w/o item recency	0.2519	0.0969	0.3587	0.1043	0.1851	0.0751	0.2643	0.0805
9	w/o item distance	0.2527	0.0996	0.3569	0.1068	0.1835	0.0744	0.2576	0.0795
	PerNIR	0.2592	0.1044	0.3625	0.1115	0.2258	0.1069	0.3006	0.1120

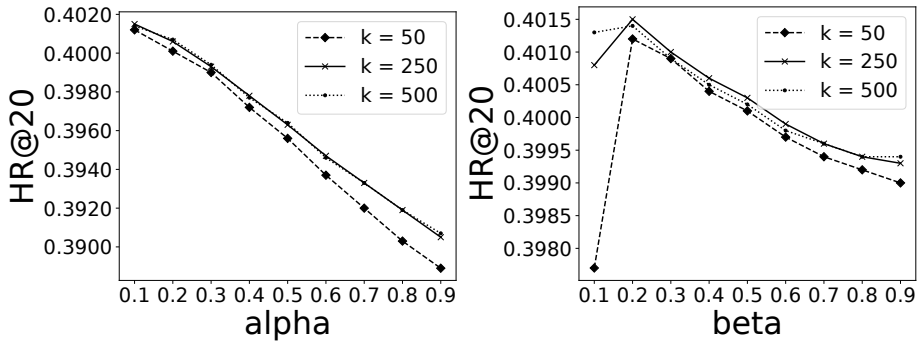


Figure 6.1: Sensitivity of the performance in terms of HR@20 to hyper-parameters α and β for different values of k on Instacart.

Therefore, the neighbor-based scoring function is essential and contributes to the final performance.

Another component in PerNIR is the user similarity computation, which is analyzed by variations 5 and 6. In variation 5, we remove the user similarity scores altogether and set $sim(u, v) = 1$ in Eq. 6.4, which means that all neighbors have the same weight in the neighbor-based scoring function NSF. This results in a small drop in performance on Instacart, ranging from 1% to 1.5%. The effect is considerable on X-Online, where the degradation ranges from 7.5% to 12.4%. One explanation is the larger number of users on Instacart, which could lead to more similar users in the neighborhood, which lowers the need for the similarity scores. In variation 6, we use another similarity scoring function to compute $sim(u, v)$, which is essentially removing HSF from Eq. 6.5, and representing each user by a binary vector with the length of the item space, where each element in the vector indicates the occurrence of the corresponding item in the purchasing history of the user. This results in the lowest performance drop on both datasets for all metrics compared to the other variations, ranging from -0.1% to 4.4%. Hence, using HSF in computing user similarity is effective, but the effect is not critical.

In variations 7, 8, and 9, we study the effect of different components in computing the basket-based scoring function BSF. In variation 7, we remove the effect of basket recency by using the same weight for all baskets in the user history in Eq. 6.3. We remove the effect of item recency in the current basket in variation 8 by setting the weights of all items in the current basket to one. We further remove the effect of the distance between a candidate item and the item in the current basket in historical baskets in variation 9, by using the same weight of one for all candidate items. In all cases, the performance degrades; all of these components contribute to the performance. Interestingly, for variations 8 and 9, the performance drop for X-Online is more than for Instacart, ranging from 12.0% to 30.4%. This could be a result of larger basket sizes in X-Online; since baskets are larger on X-Online, it becomes more important to consider the position of items in the baskets, in both incomplete baskets and in the historical baskets.

We further study the effect of the three hyper-parameters α, β, and k on the performance. We perform a grid search where α and β are swiped in $[0.1, 0.2, 0.3, \cdots, 0.8, 0.9]$

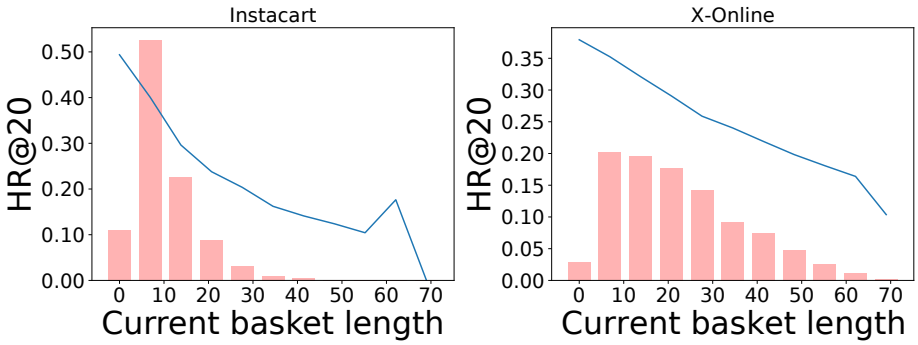


Figure 6.2: Performance of PerNIR in terms of HR@20 based on the number of items in the current basket. Bars show the distribution of basket lengths.

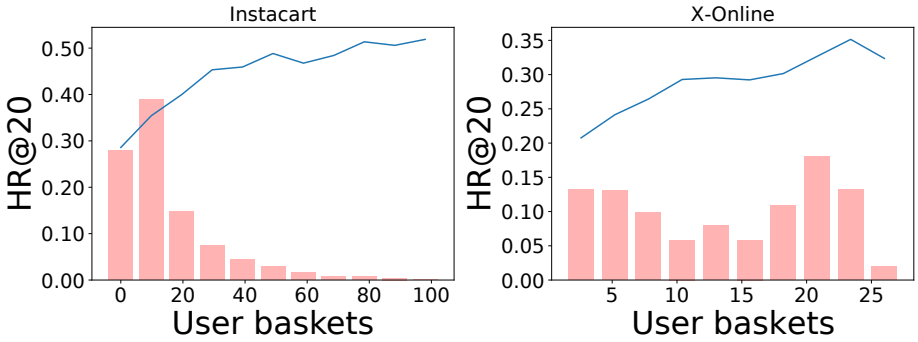


Figure 6.3: Performance of PerNIR in terms of HR@20 based on the number of baskets in users' history. Bars show the distribution of number of baskets per user.

and k in $\{50, 250, 500\}$. We find that on Instacart, $k = 250$, $\alpha = 0.1$, and $\beta = 0.2$ result in the best performance on the validation set in terms of HR@20. A similar observation is made on X-Online, with the difference of $k = 50$ giving the best performance.

Fig. 6.1 (left) shows the performance of PerNIR for different values of α and k. Increasing α leads to lower performance for all k values. Hence, in the personal scoring function PSF, items in the current basket should be considered more effectively; relying heavily on HSF, which ignores the current basket hurts, performance.

Fig. 6.1 (right) shows the performance of PerNIR for different values of β and k. For all values of k, setting β to 0.2 results in the best performance. β balances the contribution of personal scoring function and neighbor-based scoring function to the final score of items. A high value for β results in a lower effect for NSF, which degrades the performance. The optimal value of k is 250 on Instacart, however, it has a lower effect on the performance compared to α and β. Increasing the number of neighbors from 250 to 500 has a minimal effect on performance, while the difference is more apparent between $k = 50$ and $k = 250$, which indicates that a too low value for k hurts the performance.

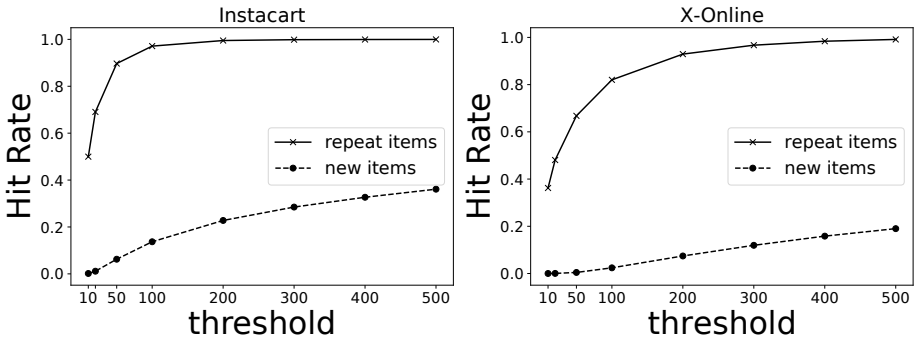


Figure 6.4: Performance of PerNIR in terms of hit rate at different thresholds, for two cases: when the target item is a previously purchased item and when it is a new item.

6.7.3 Sensitivity analysis (RQ5.3)

In this section, we study the sensitivity of PerNIR's performance to characteristics of users and baskets. Specifically, we analyze the recommendation performance for incomplete baskets of varying lengths and users with varying numbers of baskets in their history. We also consider the performance for cases where the target item is a previously purchased item vs. cases where it is a new item.

Fig. 6.2 shows the distribution of basket lengths, i.e., number of items in the current incomplete basket, in the test samples for Instacart and X-Online (bars) as well as the performance in terms of HR@20 (line). We observe that in both datasets the distribution is right-skewed; most of the baskets contain a small number of items. The performance correlates negatively with the basket size in both cases: the smaller the size of the current basket, the higher the performance of PerNIR. This means that recommending the last items in the basket is more difficult for PerNIR. One reason for this could be that users add more obvious, regular items earlier to their basket, which are easier for the model to predict.

Fig. 6.3 shows the distribution of the number of baskets in the users' history (bars) as well as the performance in terms of HR@20 (line). In Instacart, the distribution is right-skewed; most users have a small number of baskets in their history. In X-Online, the distribution is closer to uniform, with users with a small, medium, or large number of baskets in their history. In both datasets, the performance has a positive correlation with the number of baskets in the users' history. As users purchase more, PerNIR gets better at learning their shopping behavior and recommends more accurately.

We further analyze the performance of PerNIR for cases where the target item has previously been purchased by the user (i.e., a repeat item) vs. the cases where it is a new item. Fig. 6.4 shows the performance in terms of hit rate at different thresholds, for repeat and new items on both datasets. There is a significant difference in performance: predicting a repeat item is much easier than predicting a new item. While the performance reaches it maximum for repeat items at a threshold of 100 and 500 items for Instacart and X-Online, respectively, recommending accurate new items remains challenging even at high thresholds. The new items are recommended based on

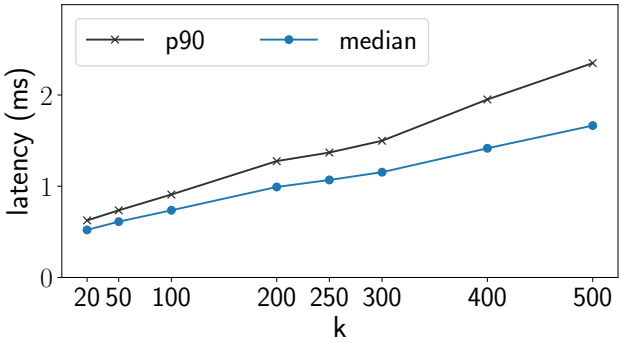


Figure 6.5: Median and 90-th percentile of the prediction latency of PerNIR for an increasing number of neighbors on the Instacart dataset (in milliseconds). PerNIR is able to respond with a p90 latency of less than 1.4 milliseconds for $k = 250$, which scored high in our evaluation.

Table 6.4: Average number of non-zero entries in the sparse history vectors (`nnz-h`) and cooccurrence matrices (`nnz-c`) of the precomputed personal models for various values of k.

k	100	200	250	300	400	500
nnz-h	3,173	5,027	5,789	6,468	7,654	8,700
nnz-c	153,236	307,630	384,850	463,615	614,432	761,671

the items that neighboring users have purchased before. This makes the search space larger and finding the correct target item more challenging. Additionally, not all new items can be found in the neighbors' item space, which further limits the achievable performance. We conclude that repeat item recommendation is an easier task for PerNIR in the within-basket recommendation scenario, in line with previous findings in NBR for grocery shopping [92].

6.7.4 Prediction latency (RQ5.4)

Next, to answer RQ4, we evaluate the prediction latency of our precomputation approach from Section 6.5.

We plot the median and 90th percentile (p90) of the response times of PerNIR in Fig. 6.5. Both the median and p90 prediction latency scale linearly with the number of neighbors k. Our precomputation approach is able to conduct inference with low prediction latency even for higher numbers of k, e.g., with a p90 latency of less 1.4 milliseconds for $k = 250$, which scored high in our evaluation. These results give a strong indication that PerNIR handles real-world deployments where low-latency online responses for users browsing a site and filling their baskets are required.

In addition, we list the average model sizes (in terms of the number of non-zero en-

tries in the sparse history vectors $\mathbf{h}_{s_u}$ (`nnz-h`) and cooccurrence matrices $\mathbf{C}_{s_u}$ (`nnz-c`)) of the precomputed personal models for various values of k in Table 6.4. The model size also scales linearly with k and the personal models only require a small number of megabytes in terms of storage, e.g., the average model size for $k = 250$ would be less than ten megabytes with 64bit floating point numbers. This small size would, for example, make it feasible to ship these models to user devices such as mobile phones.

6.8 Conclusion

In this chapter, we have proposed PerNIR, a personalized neighborhood-based model for within-basket recommendation in grocery shopping to answer **RQ5**, and analyzed its performance from different aspects through answering RQ5.1 – RQ5.4. PerNIR has different components for explicitly modeling personal preferences of users and utilizing the purchasing behavior of neighboring users.

Through extensive experiments, we have demonstrated the effectiveness of PerNIR compared to state-of-the-art baselines in terms of different performance metrics. We have further studied the contribution of different components in our model in an ablation study, which has revealed the necessity of all of them for achieving the best performance. Moreover, we have provided a vectorized implementation of PerNIR which allows for fast inference. As a result, PerNIR can provide low-latency predictions that are required in real-world recommendation systems.

A broader implication of our work is that PerNIR is applicable to the personalized session-based recommendation scenario, where sessions can be treated as baskets.

In terms of limitations, we have found that the performance of PerNIR degrades as the number of items in an incomplete basket increases. In future work, we aim to further study and address this phenomenon. One possible solution for this could be learning better weights for items in a given basket, which is currently calculated solely based on the recency of the addition of the item to the basket. Our experimental results have revealed a substantial difference in performance of PerNIR between the cases when the target item is a previously purchased item or a new item. In future work, we aim to improve the collaborative component in PerNIR to better capture signals for recommending items unseen in users' personal history.

In the next chapter, we conclude this thesis, discuss our main findings and future research directions.

7 Conclusions

This chapter concludes the thesis by summarizing our main findings and pointing out future directions. In Section 7.1, based on the results obtained in the research chapters we reflect on the research questions we asked in Section 1.1. In Section 7.2, we discuss possible future research work that builds on the work in this thesis.

7.1 Main Findings

We first revisit the research questions we posed in Section 1.1. The first two focus on multi-channel retail.

RQ1 How can we effectively utilize data from multiple shopping channels to improve the accuracy of demand forcasting models?

We focused on forecasting the demand for individual products at a given target sale channel. To answer **RQ1**, we modeled sales from other channels as privileged information, i.e., information that is available at training time but not at prediction time. We proposed a neural framework consisting of two different architectures, where the two combined can leverage the privileged information accurately. We evaluated our approach on two real-world forecasting datasets, and found that it outperforms state-of-the-art competitors in terms of mean absolute error and symmetric mean absolute percentage error metrics. We further provided visualizations and conducted experiments to validate the contribution of different components in our proposed architecture.

RQ2 What are the characteristics of user behavior in multi-channel retail settings?

To answer **RQ2**, we analyzed a large sample of 2.8 million transactions originating from 300,000 customers of a food retailer in Europe. We grouped the customers into three groups, namely online-only, offline-only, and multi-channel customers, based on their choice of shopping channels. Our analysis revealed significant differences in customer behavior across online and offline channels, for example with respect to the repeat ratio of item purchases and basket size. Based on these findings, we investigated the performance of a next basket recommendation model under multi-channel settings. We found that the recommendation performance

differs significantly for customers based on their choice of shopping channel, which strongly indicates that future research on recommenders in this area should take into account the particular characteristics of multi-channel retail shopping.

The next three questions concern the design of recommender systems that are informed by insights from user behavior understanding studies, in two different domains: finance and retail.

RQ3 What are the characteristics of financial information seeking behavior in user interactions with company filings and how can they be used in designing user-oriented filing recommender systems?

To answer **RQ3**, we analyzed 14 years of logs of users accessing company filings of more than 600K distinct companies on the U.S. Securities and Exchange Commission's (SEC) Electronic Data Gathering, Analysis, and Retrieval (EDGAR) system. We provided a comprehensive picture of the information-seeking behavior for this high-impact domain. We found that little behavioral history is available for the majority of users, while frequent users have rich histories. Most sessions focus on filings belonging to a small number of companies, and individual users are interested in a limited number of companies. Based on our observations, we defined two filing recommendation tasks that correspond to either less or more frequent users, i.e., within-session and next-session recommendation. We designed recommendation models that build on insights from our user behavior analysis. The models are two-staged: we first predict the company that the user will be interested in, limiting the search space to the filings of that company. In the next stage we rank the filings based on relevance signals, such as being timely for the user. Our experiments showed the effectiveness of our proposed models.

RQ4 What are the characteristics of repeat consumption behavior in the grocery shopping domain and how can they be used in desiging next basket recommender systems?

To answer **RQ4**, we conducted an empirical study on six public and proprietary grocery shopping transaction datasets. We discovered that averaged over all datasets, over 54% of next basket recommendation (NBR) performance in terms of recall comes from repeat items: items that users have already purchased in their history, which constitute only 1% of the total collection of items on average. This implies that a Next Basket Recommendation (NBR) model with strong focus on previously purchased items can potentially achieve a high overall performance. We introduced *ReCANet*, a repeat consumption-aware neural network that explicitly models the repeat consumption behavior of users in order to predict their next basket. ReCANet significantly outperformed state-of-the-art models for the NBR task, in terms of recall and nDCG. We performed an ablation study and showed that all of the components of ReCANet contribute to its performance, and demonstrated that a user's repetition ratio has a direct influence on the treatment effect of ReCANet.

RQ5 How can we design an effective and scalable within-basket recommender system that explicitly considers users' personal preferences?

> To answer **RQ5**, we proposed PerNIR, a *personalized nearest neighbor-based model for within-basket recommendation* that explicitly models the personal history of users for within-basket recommendation in grocery shopping. PerNIR has two components to model both short-term and long-term interests of users, which are represented by the current basket and the purchasing history, respectively. In addition to the personal preferences, user neighbors are modeled in PerNIR to capture the collaborative purchase behavior. We evaluated PerNIR on two public and proprietary datasets. The experimental results showed that it outperforms 10 state-of-the-art competitors with a significant margin, i.e., with gains of more than 12% in terms of hit rate over the second best performing approach. Additionally, we showcased an optimized implementation of PerNIR, which computes recommendations fast enough for real-world production scenarios.

The main takeaway from our results is that insights from understanding how users interact with a service provider can guide the design of recommender systems, such that they address the specific needs of the users.

7.2 Future Work

In this section, we discus the limitations of the work in this thesis and possible directions for extending it.

The first part of the thesis studies multi-channel retail settings. Our user behavior study in Chapter 3 is based on transaction logs only, which is not the only source of data available. Click behavior data is another important resource which we did not consider, and could be used to answer questions like *how do the customers of a multi-channel retail organization make use of its online platforms?* Analyzing the click data in this context can help in detecting customer intention given the interactions in an online session. Depending on the intention, let it be an online purchase, an offline purchase, or simply browsing, appropriate online features can be shown to the user. As an example, if an offline purchase is predicted, pointing the customer to the location of a product in a store is considered more useful than displaying the "add to basket" button.

In Chapters 3 and 5, we studied the recommendation performance in online-only, offline-only, and multi-channel settings. Our experiments revealed that the recommendation performance is not consistent across channels, when using a model that is agnostic to the shopping channel. Given the differences between online and offline baskets, a single model might not be optimal for representing users' history. Moreover, online click behavior data can also be useful in providing signals for recommendation. An interesting research question to answer here is *how can we design a recommendation algorithm for the multi-channel setting?* A modular system could be the solution, with a component to predict or suggest the channel for the next basket, together with separate components for modeling online baskets, offline baskets, and click behavior data.

In Chapters 3 to 6, we focused mostly on repeat recommendation – recommending items that a user has previously interacted with. Helping users discover new items is another important goal of recommender systems, which can be more challenging: the number of items not explored by a user is often much larger than the number of items they have previously explored. This makes the search space for *unseen* item

recommendation much larger, consequently making the task of finding relevant items harder. Understanding users' exploration behavior is a promising first step in tackling this challenge: *how does a user decide to add a new item to their personal item space?* Identifying signals of exploration and designing recommender systems that can leverage such signals remains an interesting scope for future research.

We mostly focused on accuracy-based metrics in the evaluation of recommender systems throughout this thesis. Understanding the performance in other dimensions such as diversity, novelty, serendipity, and coverage remains for future work. We went beyond reporting only the average performance in Chapters 5 and 6: we analyzed the performance for different users. Our preliminary analysis already shows there is a large room to study user-oriented fairness criteria in this domain: *is it fair that users with a certain pattern in their interaction with the recommender system receive better recommendations?* See [48] for a recent publication that argues in favor of more fine-grained analysis of the impact of recommender systems for different populations. The same question can be asked from an item perspective to understand supplier-oriented fairness as well.

In all our experiments, the recommender systems are trained and evaluated on a static snapshot of data. As a result, temporal changes in user behavior that might affect the accuracy of recommendations are not considered. As an example, the sample we used in our user behavior analysis in Chapter 3 was carefully selected to only include data before Covid-19. Studies have shown the effect of the pandemic on online shopping behavior [133], and we had to exclude this manually by adjusting our sample period. Ideally, recommender systems should be able to pick up on such changes automatically, instantly, reliably, and potentially alter their predictions based on this. Similar to our experimental designs, most of the existing literature on recommendation considers static datasets only, providing an opportunity for future research in this area. See [44] for a recent study on shortcomings of evaluating recommendation performance on static, historical data.

Finally, we mostly focused on user-oriented recommendation in retail, and additionally in finance. We believe that understanding user behavior and designing recommender systems should go hand-in-hand, regardless of the domain. We hope this thesis serves as an example for this research methodology and that is encourages future research in filling the gap between understanding user behavior and designing recommender systems.

Bibliography

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2–4, 2016*, pages 265–283, 2016. (Cited on page 23.)

[2] E. Acquila-Natale and S. Iglesias-Pradas. A matter of value? Predicting channel preference and multichannel behaviors in retail. *Technological Forecasting and Social Change*, 162:120401, 2021. (Cited on pages 2, 29, and 30.)

[3] B. Adhikari, X. Xu, N. Ramakrishnan, and B. A. Prakash. EpiDeep: Exploiting embeddings for epidemic forecasting. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019*, pages 577–586, 2019. (Cited on page 15.)

[4] Y. Agrawal, V. Anand, S. Arunachalam, and V. Varma. Hierarchical model for goal guided summarization of annual financial reports. In *Companion of The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19–23, 2021*, pages 247–254. ACM / IW3C2, 2021. (Cited on page 44.)

[5] Q. Ai, S. T. Dumais, N. Craswell, and D. J. Liebling. Characterizing email search using large-scale behavioral logs and surveys. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3–7, 2017*, pages 1511–1520. ACM, 2017. (Cited on pages 2 and 44.)

[6] A. Anderson, R. Kumar, A. Tomkins, and S. Vassilvitskii. The dynamics of repeat consumption. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7–11, 2014*, pages 419–430. ACM, 2014. (Cited on pages 1 and 68.)

[7] I. Arapakis, X. Bai, and B. B. Cambazoglu. Impact of response latency on user behavior in web search. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast , QLD, Australia, July 6–11, 2014*, pages 103–112. ACM, 2014. (Cited on page 92.)

[8] M. Ariannezhad. Understanding and learning from user behavior for recommendation in multi-channel retail. In *Advances in Information Retrieval – 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022*, pages 455–462. Springer, 2022. (Cited on page 2.)

[9] M. Ariannezhad, A. Montazeralghaem, H. Zamani, and A. Shakery. Improving retrieval performance for verbose queries via axiomatic analysis of term discrimination heuristic. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7–11, 2017*, pages 1201–1204. ACM, 2017.

[10] M. Ariannezhad, A. Montazeralghaem, H. Zamani, and A. Shakery. Iterative estimation of document relevance score for pseudo-relevance feedback. In *Advances in Information Retrieval – 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8–13, 2017*, pages 676–683, 2017.

[11] M. Ariannezhad, S. Schelter, and M. de Rijke. Demand forecasting in the presence of privileged information. In *Advanced Analytics and Learning on Temporal Data - 5th ECML PKDD Workshop, AALTD 2020, Ghent, Belgium, September 18, 2020*, pages 46–62. Springer, 2020. (Cited on pages 1 and 2.)

[12] M. Ariannezhad, S. Jullien, P. Nauts, M. Fang, S. Schelter, and M. de Rijke. Understanding multi-channel customer behavior in retail. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1–5, 2021*, pages 2867–2871. ACM, 2021. (Cited on pages 1 and 66.)

[13] 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 '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11–15, 2022*, pages 1240–1250. ACM, 2022. (Cited on pages 2, 87, 88, 90, 91, 94, 95, and 98.)

[14] M. Ariannezhad, M. Yahya, E. Meij, S. Schelter, and M. de Rijke. Understanding financial information seeking behavior from user interactions with company filings. In *Companion of The Web Conference 2022, Virtual Event / Lyon, France, April 25–29, 2022*, pages 586–594. ACM, 2022.

[15] M. Ariannezhad, M. Li, S. Jullien, and M. de Rijke. Complex item set recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23–27, 2023*, pages 3444–3447. ACM, 2023.

[16] M. Ariannezhad, M. Li, S. Schelter, and M. de Rijke. A personalized neighborhood-based model for within-basket recommendation in grocery shopping. In *Proceedings of the Sixteenth ACM International*

Conference on Web Search and Data Mining, WSDM 2023, Singapore, 27 February 2023 – 3 March 2023, pages 87–95. ACM, 2023. (Cited on page 2.)

[17] S. Bai, J. Z. Kolter, and V. Koltun. Convolutional sequence modeling revisited. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30–May 3, 2018*. OpenReview.net, 2018. (Cited on page 19.)

[18] T. Bai, J. Nie, W. X. Zhao, Y. Zhu, P. Du, and J. Wen. An attribute-aware neural attentive model for next basket recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 8–12, 2018*, pages 1201–1204. ACM, 2018. (Cited on page 45.)

[19] W. Bao, J. Yue, and Y. Rao. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS one*, 12(7):e0180944, 2017. (Cited on page 15.)

[20] O. Barkan and N. Koenigstein. Item2vec: Neural item embedding for collaborative filtering. In *Proceedings of the Poster Track of the 10th ACM Conference on Recommender Systems (RecSys 2016), Boston, USA, September 17, 2016*. CEUR-WS.org, 2016. (Cited on page 66.)

[21] B. Barreau and L. Carlier. History-augmented collaborative filtering for financial recommendations. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22–26, 2020*, pages 492–497. ACM, 2020. (Cited on pages 2 and 41.)

[22] J. Bennett, S. Lanning, et al. The Netflix prize. In *KDD Cup Workshop 2007*, pages 3–6, 2007. (Cited on page 1.)

[23] A. R. Benson, R. Kumar, and A. Tomkins. Modeling user consumption sequences. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11–15, 2016*, pages 519–529. ACM, 2016. (Cited on page 68.)

[24] 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, KDD 2018, London, UK, August 19–23, 2018*, pages 62–70. ACM, 2018. (Cited on page 69.)

[25] J. Boese, V. Flunkert, J. Gasthaus, T. Januschowski, D. Lange, D. Salinas, S. Schelter, M. W. Seeger, and B. Wang. Probabilistic demand forecasting at scale. *Proc. VLDB Endow.*, 10(12):1694–1705, 2017. (Cited on page 13.)

[26] H. Bota, A. Fourney, S. T. Dumais, T. L. Religa, and R. Rounthwaite. Characterizing search behavior in productivity software. In *Proceedings of the 2018 Conference on Human Information Interaction and Retrieval, CHIIR 2018, New Brunswick, NJ, USA, March 11–15, 2018*, pages 160–169. ACM, 2018. (Cited on pages 2 and 44.)

[27] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 2015. (Cited on page 14.)

[28] R. Carbonneau, K. Laframboise, and R. M. Vahidov. Application of machine learning techniques for supply chain demand forecasting. *European Journal of Operational Research*, 184(3):1140–1154, 2008. (Cited on page 13.)

[29] D. Ceccarelli, F. Nidito, and M. Osborne. Ranking financial tweets. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17–21, 2016*, pages 527–528. ACM, 2016. (Cited on page 44.)

[30] J. Chang, W. Tu, C. Yu, and C. Qin. Assessing dynamic qualities of investor sentiments for stock recommendation. *Inf. Process. Manag.*, 58(2):102452, 2021. (Cited on pages 2 and 41.)

[31] P. Chatterjee. Multiple-channel and cross-channel shopping behavior: Role of consumer shopping orientations. *Marketing Intelligence & Planning*, 28, 2010. (Cited on pages 2, 29, and 30.)

[32] C. Chen, S. Kim, H. Bui, R. A. Rossi, E. Koh, B. Kveton, and R. C. Bunescu. Predictive analysis by leveraging temporal user behavior and user embeddings. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22–26, 2018*, pages 2175–2182. ACM, 2018. (Cited on pages 2 and 29.)

[33] C. Chen, P. Lamere, M. Schedl, and H. Zamani. RecSys challenge 2018: Automatic music playlist continuation. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2–7, 2018*, pages 527–528. ACM, 2018. (Cited on page 1.)

[34] C. Chen, L. Zhao, J. Bian, C. Xing, and T. Liu. Investment behaviors can tell what inside: Exploring stock intrinsic properties for stock trend prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019*, pages 2376–2384. ACM, 2019. (Cited on pages 2 and 41.)

[35] J. Chen, C. Wang, and J. Wang. Will you "reconsume" the near past? Fast prediction on short-term reconsumption behaviors. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial*

Intelligence, January 25–30, 2015, Austin, Texas, USA, pages 23–29. AAAI Press, 2015. (Cited on page 68.)

[36] T. Chen, H. Yin, H. Chen, L. Wu, H. Wang, X. Zhou, and X. Li. TADA: Trend alignment with dual-attention multi-task recurrent neural networks for sales prediction. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17–20, 2018*, pages 49–58. IEEE Computer Society, 2018. (Cited on pages 1, 13, 15, 16, 21, 22, and 26.)

[37] X. Chen, H. Chen, H. Xu, Y. Zhang, Y. Cao, Z. Qin, and H. Zha. Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21–25, 2019*, pages 765–774. ACM, 2019. (Cited on page 65.)

[38] Y. Chen and M. de Rijke. A collective variational autoencoder for top-n recommendation with side information. In *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2018, Vancouver, BC, Canada, October 6, 2018*, pages 3–9. ACM, 2018. (Cited on page 65.)

[39] Y. Chen, X. Jin, J. Feng, and S. Yan. Training group orthogonal neural networks with privileged information. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19–25, 2017*, pages 1532–1538. ijcai.org, 2017. (Cited on page 15.)

[40] D. Cheng, F. Yang, X. Wang, Y. Zhang, and L. Zhang. Knowledge graph-based event embedding framework for financial quantitative investments. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25–30, 2020*, pages 2221–2230. ACM, 2020. (Cited on page 41.)

[41] F. Chollet et al. Keras. `https://keras.io`, 2015. (Cited on page 23.)

[42] 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, RecSys 2019, Copenhagen, Denmark, September 16–20, 2019*, pages 101–109. ACM, 2019. (Cited on page 58.)

[43] 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 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26–30, 2010*, pages 293–296. ACM, 2010. (Cited on page 58.)

[44] R. Deffayet, T. Thonet, J. Renders, and M. de Rijke. Offline evaluation for reinforcement learning-based recommendation: A critical issue and some alternatives. *SIGIR Forum*, 56(2):3:1–3:14, 2022. (Cited on page 108.)

[45] M. S. Drake, D. T. Roulstone, and J. R. Thornock. The determinants and consequences of information acquisition via EDGAR. *Contemporary Accounting Research*, 32(3):1128–1161, 2015. (Cited on pages 43 and 44.)

[46] M. S. Drake, B. A. Johnson, D. T. Roulstone, and J. R. Thornock. Is there information content in information acquisition? *The Accounting Review*, 95(2):113–139, 2020. (Cited on pages 43, 44, and 53.)

[47] I. S. Duff, M. A. Heroux, and R. Pozo. An overview of the sparse basic linear algebra subprograms: The new standard from the BLAS technical forum. *ACM Transactions on Mathematical Software (TOMS)*, 28(2):239–267, 2002. (Cited on page 93.)

[48] M. D. Ekstrand, B. Carterette, and F. Diaz. Distributionally-informed recommender system evaluation. *ACM Trans. Recomm. Syst.*, 2023. (Cited on page 108.)

[49] 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, UMAP 2020, Genoa, Italy, July 12–18, 2020*, pages 80–87. ACM, 2020. (Cited on pages 65, 66, 68, 77, 78, 90, and 91.)

[50] C. Fan, Y. Zhang, Y. Pan, X. Li, C. Zhang, R. Yuan, D. Wu, W. Wang, J. Pei, and H. Huang. Multi-horizon time series forecasting with temporal attention learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019*, pages 2527–2535. ACM, 2019. (Cited on pages 13, 15, and 21.)

[51] M. Fan, D. Cheng, F. Yang, S. Luo, Y. Luo, W. Qian, and A. Zhou. Fusing global domain information and local semantic information to classify financial documents. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19–23, 2020*, pages 2413–2420. ACM, 2020. (Cited on page 44.)

[52] H. Fang, D. Zhang, Y. Shu, and G. Guo. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *ACM Trans. Inf. Syst.*, 39(1):10:1–10:42, 2020. (Cited on page 1.)

[53] F. Feng, C. Luo, X. He, Y. Liu, and T. Chua. FinIR 2020: The first workshop on information retrieval in finance. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25–30, 2020*, pages 2451–2454. ACM, 2020. (Cited on pages 2, 41, and 44.)

[54] F. Feng, M. Li, C. Luo, R. Ng, and T. Chua. Hybrid learning to rank for financial event ranking. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11–15, 2021*, pages 233–243. ACM, 2021. (Cited on page 41.)

[55] F. Gao, V. V. Agrawal, and S. Cui. The effect of multichannel and omnichannel retailing on physical stores. *Management Science*, 68(2):809–826, 2022. (Cited on pages 2 and 30.)

[56] D. Garg, P. Gupta, P. Malhotra, L. Vig, and G. Shroff. Sequence and time aware neighborhood for session-based recommendations: STAN. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21–25, 2019*, pages 1069–1072. ACM, 2019. (Cited on pages 89, 91, and 95.)

[57] A. W. Gatian. Is user satisfaction a valid measure of system effectiveness? *Inf. Manag.*, 26(3):119–131, 1994. (Cited on page 1.)

[58] J. Guo, Y. Yang, X. Song, Y. Zhang, Y. Wang, J. Bai, and Y. Zhang. Learning multi-granularity consecutive user intent unit for session-based recommendation. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21–25, 2022*, pages 343–352. ACM, 2022. (Cited on page 89.)

[59] A. Halfaker, O. Keyes, D. Kluver, J. Thebault-Spieker, T. T. Nguyen, K. Shores, A. Uduwage, and M. Warncke-Wang. User session identification based on strong regularities in inter-activity time. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18–22, 2015*, pages 410–418. ACM, 2015. (Cited on page 50.)

[60] M. Hendriksen, E. Kuiper, P. Nauts, S. Schelter, and M. de Rijke. Analyzing and predicting purchase intent in e-commerce: Anonymous vs. identified customers. In *eCOM 2020: The 2020 SIGIR Workshop on eCommerce*. ACM, July 2020. (Cited on pages 1, 2, and 29.)

[61] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016, Conference Track Proceedings*, 2016. (Cited on page 65.)

[62] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997. (Cited on page 66.)

[63] J. A. Hodson and J. Y. Zhang. Entity extraction and disambiguation in finance. In *ERD'14, Proceedings of the First ACM International Workshop on Entity Recognition & Disambiguation, July 11, 2014, Gold Coast, Queensland, Australia*, pages 1–2. ACM, 2014. (Cited on page 44.)

[64] J. Hoffman, S. Gupta, and T. Darrell. Learning with side information through modality hallucination. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27–30, 2016*, pages 826–834. IEEE Computer Society, 2016. (Cited on page 15.)

[65] 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 & Data Mining, KDD 2019, Anchorage, AK, USA, August 4–8, 2019*, pages 1491–1499. ACM, 2019. (Cited on pages 43, 45, 66, and 68.)

[66] H. Hu, X. He, J. Gao, and Z. 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, SIGIR 2020, Virtual Event, China, July 25–30, 2020*, pages 1071–1080. ACM, 2020. (Cited on pages 30, 34, 35, 36, 45, 61, 65, 66, 68, 77, 78, 80, 87, 88, 90, and 95.)

[67] L. Hu, Q. Chen, H. Zhao, S. Jian, L. Cao, and J. Cao. Neural cross-session filtering: Next-item prediction under intra- and inter-session context. *IEEE Intell. Syst.*, 33(6):57–67, 2018. (Cited on page 89.)

[68] G. T. M. Hult, P. N. Sharma, F. V. Morgeson, and Y. Zhang. Antecedents and consequences of customer satisfaction: Do they differ across online and offline purchases? *Journal of Retailing*, 95(1):10–23, 2019. (Cited on pages 2 and 30.)

[69] R. S. Hussein and A. Kais. Multichannel behaviour in the retail industry: Evidence from an emerging market. *International Journal of Logistics Research and Applications*, 24(3):242–260, 2021. (Cited on pages 2 and 30.)

[70] 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, RecSys 2017, Como, Italy, August 27–31, 2017*, pages 306–310. ACM, 2017. (Cited on pages 43, 45, 57, 58, and 89.)

[71] S. Jullien, M. Ariannezhad, P. Groth, and M. de Rijke. GLDQN: Explicitly parameterized quantile reinforcement learning for waste reduction. *CoRR*, abs/2205.15455, 2022.

[72] S. Jullien, M. Ariannezhad, P. Groth, and M. de Rijke. A simulation environment and reinforcement learning method for waste reduction. *Trans. Mach. Learn. Res.*, 2023, 2023.

[73] M. Karimi, D. Jannach, and M. Jugovac. News recommender systems - Survey and roads ahead. *Inf. Process. Manag.*, 54(6):1203–1227, 2018. (Cited on page 1.)

[74] B. Kersbergen and S. Schelter. Learnings from a retail recommendation system on billions of interactions at bol.com. In *37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19–22, 2021*, pages 2447–2452. IEEE, 2021. (Cited on pages 88 and 92.)

[75] B. Kersbergen, O. Sprangers, and S. Schelter. Serenade - Low-latency session-based recommendation in e-commerce at scale. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12–17, 2022*, pages 150–159. ACM, 2022. (Cited on pages 88 and 92.)

[76] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings*, 2015. (Cited on page 23.)

[77] F. Kooti, K. Lerman, L. M. Aiello, M. Grbovic, N. Djuric, and V. Radosavljevic. Portrait of an online shopper: Understanding and predicting consumer behavior. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, San Francisco, CA, USA, February 22–25, 2016*, pages 205–214. ACM, 2016. (Cited on pages 2 and 29.)

[78] I. Koprinska, D. Wu, and Z. Wang. Convolutional neural networks for energy time series forecasting. In *2018 International Joint Conference on Neural Networks, IJCNN 2018, Rio de Janeiro, Brazil, July 8–13, 2018*, pages 1–8. IEEE, 2018. (Cited on page 15.)

[79] L. Korkeamäki and S. Kumpulainen. Interacting with digital documents: A real life study of historians' task processes, actions and goals. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval, CHIIR 2019, Glasgow, Scotland, UK, March 10–14, 2019*, pages 35–43. ACM, 2019. (Cited on page 44.)

[80] P. Ladyzynski, K. Zbikowski, and P. Grzegorzewski. Stock trading with random forests, trend detection tests and force index volume indicators. In *Artificial Intelligence and Soft Computing - 12th International Conference, ICAISC 2013, Zakopane, Poland, June 9–13, 2013*, pages 441–452. Springer, 2013. (Cited on page 16.)

[81] G. Lai, W. Chang, Y. Yang, and H. Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 8–12, 2018*, pages 95–104. ACM, 2018. (Cited on pages 13, 15, and 26.)

[82] J. Lambert, O. Sener, and S. Savarese. Deep learning under privileged information using heteroscedastic dropout. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18–22, 2018*, pages 8886–8895. Computer Vision Foundation / IEEE Computer Society, 2018. (Cited on page 15.)

[83] S. Latifi, N. Mauro, and D. Jannach. Session-aware recommendation: A surprising quest for the state-of-the-art. *Inf. Sci.*, 573:291–315, 2021. (Cited on pages 88, 90, and 95.)

[84] D. Le, H. W. Lauw, and Y. Fang. Basket-sensitive personalized item recommendation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19–25, 2017*, pages 2060–2066. ijcai.org, 2017. (Cited on pages 87 and 89.)

[85] D. Le, H. W. Lauw, and Y. Fang. Correlation-sensitive next-basket recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10–16, 2019*, pages 2808–2814. ijcai.org, 2019. (Cited on pages 68 and 90.)

[86] C. M. Lee, P. Ma, and C. C. Wang. Search-based peer firms: Aggregating investor perceptions through internet co-searches. *Journal of Financial Economics*, 116(2):410–431, 2015. (Cited on pages 41, 44, and 56.)

[87] F. Li and C. Sun. Information acquisition and expected returns: Evidence from EDGAR search traffic. *Social Science Research Network*, page 1, 2018. (Cited on pages 43 and 44.)

[88] 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. ACM, 2017. (Cited on pages 45, 58, 65, and 89.)

[89] M. Li, X. Bao, L. Chang, and T. Gu. Modeling personalized representation for within-basket recommendation based on deep learning. *Expert Syst. Appl.*, 192:116383, 2022. (Cited on pages 87, 88, and 89.)

[90] M. Li, M. Ariannezhad, A. Yates, and M. de Rijke. Who will purchase this item next? Reverse next period recommendation in grocery shopping. *ACM Trans. Recomm. Syst.*, 1(2), 2023.

[91] 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*. ACM, 2023.

[92] M. Li, S. Jullien, M. Ariannezhad, and M. de Rijke. A next basket recommendation reality check. *ACM Trans. Inf. Syst.*, 41(4), 2023. (Cited on pages 45, 61, 66, 67, 68, 77, 79, 80, 87, 88, 90, 94, and 103.)

[93] R. Li, X. W. Wang, Z. Yan, and Y. Zhao. Sophisticated investor attention and market reaction to earnings announcements: Evidence from the SEC's EDGAR log files. *Journal of Financial Economics*, 2019. (Cited on pages 43 and 44.)

[94] J. Liu, M. Mitsui, N. J. Belkin, and C. Shah. Task, information seeking intentions, and user behavior: Toward a multi-level understanding of web search. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval, CHIIR 2019, Glasgow, Scotland, UK, March 10–14, 2019*, pages 123–132. ACM, 2019. (Cited on page 44.)

[95] 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, KDD 2018, London, UK, August 19–23, 2018*, pages 1831–1839. ACM, 2018. (Cited on pages 45 and 89.)

[96] Y. Liu, L. Liu, C. Wang, and M. Tsai. FIN10K: A web-based information system for financial report analysis and visualization. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24–28, 2016*, pages 2441–2444. ACM, 2016. (Cited on page 44.)

[97] Y. Liu, H. Lee, P. Achananuparp, E. Lim, T. Cheng, and S. Lin. Characterizing and predicting repeat food consumption behavior for just-in-time interventions. In *Proceedings of the 9th International Conference on Digital Public Health, PDH 2019, Marseille, France, November 20–23, 2019*, pages 11–20. ACM, 2019. (Cited on pages 2, 65, and 67.)

[98] Z. Liu, X. Li, Z. Fan, S. Guo, K. Achan, and P. S. Yu. Basket recommendation with multi-intent translation graph neural network. In *2020 IEEE International Conference on Big Data (IEEE BigData 2020), Atlanta, GA, USA, December 10–13, 2020*, pages 728–737. IEEE, 2020. (Cited on pages 88, 89, and 94.)

[99] Z. Liu, M. Wan, S. Guo, K. Achan, and P. S. Yu. BasConv: Aggregating heterogeneous interactions for basket recommendation with graph convolutional neural network. In *Proceedings of the 2020 SIAM International Conference on Data Mining, SDM 2020, Cincinnati, Ohio, USA, May 7–9, 2020*, pages 64–72. SIAM, 2020. (Cited on pages 87, 88, 89, and 94.)

[100] C. Lo, D. Frankowski, and J. Leskovec. Understanding behaviors that lead to purchasing: A case study of Pinterest. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13–17, 2016*, pages 531–540. ACM, 2016. (Cited on pages 1, 2, and 29.)

[101] D. Lopez-Paz, L. Bottou, B. Schölkopf, and V. Vapnik. Unifying distillation and privileged information. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016*, 2016. (Cited on pages 14 and 15.)

[102] T. Loughran and B. McDonald. The use of EDGAR filings by investors. *Journal of Behavioral Finance*, 18(2):231–248, 2017. (Cited on pages 43, 44, and 50.)

[103] M. Ludewig and D. Jannach. Evaluation of session-based recommendation algorithms. *User Model. User Adapt. Interact.*, 28(4-5):331–390, 2018. (Cited on pages 45, 88, 89, and 94.)

[104] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach. Performance comparison of neural and non-neural approaches to session-based recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16–20, 2019*, pages 462–466. ACM, 2019.

[105] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach. Empirical analysis of session-based recommendation algorithms. *User Model. User Adapt. Interact.*, 31(1):149–181, 2021. (Cited on pages 88 and 89.)

[106] P. Luo, S. Yan, Z. Liu, Z. Shen, S. Yang, and Q. He. From online behaviors to offline retailing. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13–17, 2016*, pages 175–184. ACM, 2016. (Cited on pages 2

and 29.)

[107] Z. Ma, S. Pomerville, M. Di, and A. Nourbakhsh. SPot: A tool for identifying operating segments in financial tables. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 2157–2160. ACM, 2020. (Cited on page 41.)

[108] Z. Meng, R. McCreadie, C. Macdonald, and I. Ounis. Variational Bayesian representation learning for grocery recommendation. *Inf. Retr. J.*, 24(4-5):347–369, 2021. (Cited on page 65.)

[109] G. Mishne and M. de Rijke. A study of blog search. In *Advances in Information Retrieval, 28th European Conference on IR Research, ECIR 2006, London, UK, April 10–12, 2006*, pages 289–301. Springer, 2006. (Cited on page 50.)

[110] T. Neifer, D. Lawo, G. Stevens, A. Boden, and A. Gadatsch. Recommender systems in food retail: Modeling repeat purchase decisions on transaction data of a stationary food retailer. In *Proceedings of the 18th International Conference on e-Business, ICE-B 2021, Online Streaming, July 7–9, 2021*, pages 25–36. SciTePress, 2021. (Cited on page 68.)

[111] Z. Pan, F. Cai, W. Chen, H. Chen, and M. de Rijke. Star graph neural networks for session-based recommendation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19–23, 2020*, pages 1195–1204. ACM, 2020. (Cited on page 89.)

[112] Y. Pang, L. Wu, Q. Shen, Y. Zhang, Z. Wei, F. Xu, E. Chang, B. Long, and J. Pei. Heterogeneous global graph neural networks for personalized session-based recommendation. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21–25, 2022*, pages 775–783. ACM, 2022. (Cited on pages 89 and 95.)

[113] T. M. Phuong, T. C. Thanh, and N. X. Bach. Neural session-aware recommendation. *IEEE Access*, 7: 86884–86896, 2019. (Cited on page 89.)

[114] V. Plachouras, C. Smiley, H. Bretz, O. Taylor, J. L. Leidner, D. Song, and F. Schilder. Interacting with financial data using natural language. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17–21, 2016*, pages 1121–1124. ACM, 2016. (Cited on page 44.)

[115] Y. Qi, C. Li, H. Deng, M. Cai, Y. Qi, and Y. Deng. A deep neural framework for sales forecasting in e-commerce. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3–7, 2019*, pages 299–308. ACM, 2019. (Cited on pages 13 and 15.)

[116] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. W. Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19–25, 2017*, pages 2627–2633. ijcai.org, 2017. (Cited on pages 13, 15, and 26.)

[117] Y. Qin, P. Wang, and C. Li. The world is binary: Contrastive learning for denoising next basket recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11–15, 2021*, pages 859–868. ACM, 2021. (Cited on pages 65, 66, 68, 80, and 90.)

[118] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27–31, 2017*, pages 130–137. ACM, 2017. (Cited on page 89.)

[119] R. Rahimi, A. Shakery, J. Dadashkarimi, M. Ariannezhad, M. Dehghani, and H. N. Esfahani. Building a multi-domain comparable corpus using a learning to rank method. *Nat. Lang. Eng.*, 22(4):627–653, 2016.

[120] J. Rappaz, J. J. McAuley, and K. Aberer. Recommendation on live-streaming platforms: Dynamic availability and repeat consumption. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021–1 October 2021*, pages 390–399. ACM, 2021. (Cited on page 69.)

[121] M. Reiter-Haas, E. Parada-Cabaleiro, M. Schedl, E. Motamedi, M. Tkalcic, and E. Lex. Predicting music relistening behavior using the ACT-R framework. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021–1 October 2021*, pages 702–707. ACM, 2021. (Cited on pages 1 and 68.)

[122] 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 *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference,*

IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27–February 1, 2019, pages 4806–4813. AAAI Press, 2019. (Cited on pages 45, 57, 58, 59, and 69.)

[123] 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, WWW 2010, Raleigh, North Carolina, USA, April 26–30, 2010*, pages 811–820. ACM, 2010. (Cited on page 60.)

[124] F. Ricci, L. Rokach, and B. Shapira. Recommender systems: Introduction and challenges. In *Recommender Systems Handbook*, pages 1–34. Springer, 2015. (Cited on page 1.)

[125] G. Ristanoski, W. Liu, and J. Bailey. Time series forecasting using distribution enhanced linear regression. In *Advances in Knowledge Discovery and Data Mining, 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14–17, 2013*, pages 484–495. Springer, 2013. (Cited on page 14.)

[126] M. Ruocco, O. S. L. Skrede, and H. Langseth. Inter-session modeling for session-based recommendation. In *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2017, Como, Italy, August 27, 2017*, pages 24–31. ACM, 2017. (Cited on page 89.)

[127] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2019. (Cited on pages 13, 15, and 21.)

[128] S. Schelter, M. Ariannezhad, and M. de Rijke. Forget me now: Fast and exact unlearning in neighborhood-based recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23–27, 2023*, pages 2011–2015. ACM, 2023.

[129] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, and T. Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19–24, 2016*, pages 1670–1679. JMLR.org, 2016. (Cited on page 82.)

[130] Z. Shen, Y. Zhang, J. Lu, J. Xu, and G. Xiao. SeriesNet: A generative time series forecasting model. In *2018 International Joint Conference on Neural Networks, IJCNN 2018, Rio de Janeiro, Brazil, July 8–13, 2018*, pages 1–8. IEEE, 2018. (Cited on page 15.)

[131] S. Shih, F. Sun, and H. Lee. Temporal pattern attention for multivariate time series forecasting. *Machine Learning*, 108(8-9):1421–1441, 2019. (Cited on page 15.)

[132] C. Siro, M. Aliannejadi, and M. de Rijke. Understanding user satisfaction with task-oriented dialogue systems. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11–15, 2022*, pages 2018–2023. ACM, 2022. (Cited on page 1.)

[133] J. C. Soares, R. Limongi, J. H. De Sousa Júnior, W. S. Santos, M. Raasch, and L. Hoeckesfeld. Assessing the effects of COVID-19-related risk on online shopping behavior. *Journal of Marketing Analytics*, 11(1):82–94, 2023. (Cited on page 108.)

[134] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014. (Cited on page 20.)

[135] R. M. Stein. Benchmarking default prediction models: Pitfalls and remedies in model validation. *Moody's KMV, New York*, 20305, 2002. (Cited on pages 16 and 22.)

[136] N. Su, J. He, Y. Liu, M. Zhang, and S. Ma. User intent, behaviour, and perceived satisfaction in product search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5–9, 2018*, pages 547–555. ACM, 2018. (Cited on page 30.)

[137] 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, CIKM 2019, Beijing, China, November 3–7, 2019*, pages 1441–1450. ACM, 2019. (Cited on page 65.)

[138] 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, SIGIR 2020, Virtual Event, China, July 25–30, 2020*, pages 1439–1448. ACM, 2020. (Cited on pages 66 and 68.)

[139] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information*

Processing Systems 2014, December 8–13 2014, Montreal, Quebec, Canada, pages 3104–3112, 2014. (Cited on page 13.)

[140] J. Y. L. Thong and C. Yap. Information systems effectiveness: A user satisfaction approach. *Inf. Process. Manag.*, 32(5):601–610, 1996. (Cited on page 1.)

[141] A. Toth, L. Tan, G. D. Fabbrizio, and A. Datta. Predicting shopping behavior with mixture of RNNs. In *Proceedings of the SIGIR 2017 Workshop On eCommerce co-located with the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, eCOM@SIGIR 2017, Tokyo, Japan, August 11, 2017*. CEUR-WS.org, 2017. (Cited on pages 1, 2, and 29.)

[142] M. Tsagkias, T. H. King, S. Kallumadi, V. Murdock, and M. de Rijke. Challenges and research opportunities in ecommerce search and recommendations. *SIGIR Forum*, 54(1), 2020. (Cited on page 68.)

[143] M. Tsai and C. Wang. Risk ranking from financial reports. In *Advances in Information Retrieval - 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, March 24–27, 2013*, pages 804–807. Springer, 2013. (Cited on page 44.)

[144] N. Vairagade, D. Logofatu, F. Leon, and F. Muharemi. Demand forecasting using random forest and artificial neural network for supply chain management. In *Computational Collective Intelligence - 11th International Conference, ICCCI 2019, Hendaye, France, September 4–6, 2019*, pages 328–339. Springer, 2019. (Cited on page 13.)

[145] T. van de Looij and M. Ariannezhad. Cluster-based forecasting for intermittent and non-intermittent time series. In *Advanced Analytics and Learning on Temporal Data – 6th ECML PKDD Workshop, AALTD 2021, Bilbao, Spain, September 13, 2021*, pages 139–154. Springer, 2021.

[146] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu. WaveNet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, September 13–15, 2016*, page 125. ISCA, 2016. (Cited on pages 19 and 20.)

[147] V. Vapnik and A. Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5-6):544–557, 2009. (Cited on page 15.)

[148] M. Wan, D. Wang, J. Liu, P. Bennett, and J. 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, CIKM 2018, Torino, Italy, October 22–26, 2018*, pages 1133–1142. ACM, 2018. (Cited on pages 30, 34, 68, 87, 88, and 89.)

[149] C. Wang, M. Zhang, W. Ma, Y. Liu, and S. Ma. Modeling item-specific temporal dynamics of repeat consumption for recommender systems. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13–17, 2019*, pages 1977–1987. ACM, 2019. (Cited on pages 1 and 69.)

[150] P. Wang. Demand for information and stock returns: Evidence from EDGAR. *Available at SSRN 3348513*, 2019. (Cited on pages 43 and 44.)

[151] P. Wang, J. Guo, and Y. Lan. Modeling retail transaction data for personalized shopping recommendation. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3–7, 2014*, pages 1979–1982. ACM, 2014. (Cited on pages 2 and 29.)

[152] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, and D. Lian. A survey on session-based recommender systems. *ACM Comput. Surv.*, 54(7):154:1–154:38, 2022. (Cited on page 1.)

[153] X. Wang, N. Su, Z. He, Y. Liu, and S. Ma. A large-scale study of mobile search examination behavior. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 8–12, 2018*, pages 1129–1132. ACM, 2018. (Cited on pages 2 and 44.)

[154] W. Weerkamp, R. Berendsen, B. Kovachev, E. Meij, K. Balog, and M. de Rijke. People searching for people: Analysis of a people search engine log. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25–29, 2011*, pages 45–54. ACM, 2011. (Cited on page 50.)

[155] R. Wen, K. Torkkola, B. Narayanaswamy, and D. Madeka. A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053*, 2017. (Cited on pages 13 and 21.)

[156] Y. T. Wen, P. Yeh, T. Tsai, W. Peng, and H. Shuai. Customer purchase behavior prediction from payment datasets. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5–9, 2018*, pages 628–636. ACM, 2018. (Cited on pages 2 and 29.)

[157] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan. Session-based recommendation with graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The*

Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27–February 1, 2019, pages 346–353. AAAI Press, 2019. (Cited on page 89.)

[158] X. Wu, B. Shi, Y. Dong, C. Huang, L. Faust, and N. V. Chawla. RESTFul: Resolution-aware forecasting of behavioral time series data. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22–26, 2018*, pages 1073–1082. ACM, 2018. (Cited on page 15.)

[159] Q. Xia, P. Jiang, F. Sun, Y. Zhang, X. Wang, and Z. Sui. Modeling consumer buying decision for recommendation based on multi-task deep learning. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22–26, 2018*, pages 1703–1706. ACM, 2018. (Cited on pages 2 and 29.)

[160] X. Xia, H. Yin, J. Yu, Q. Wang, L. Cui, and X. Zhang. Self-supervised hypergraph convolutional networks for session-based recommendation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2–9, 2021*, pages 4503–4511. AAAI Press, 2021. (Cited on page 89.)

[161] Y. Yang, Z. Wei, Q. Chen, and L. Wu. Using external knowledge for financial event prediction based on graph neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3–7, 2019*, pages 2161–2164. ACM, 2019. (Cited on page 41.)

[162] J. Yeo, S. Kim, E. Koh, S. Hwang, and N. Lipka. Browsing2purchase: Online customer model for sales forecasting in an e-commerce site. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11–15, 2016*, pages 133–134. ACM, 2016. (Cited on pages 1, 2, and 29.)

[163] H. Ying, F. Zhuang, F. Zhang, Y. Liu, G. Xu, X. Xie, H. Xiong, and J. Wu. Sequential recommender system based on hierarchical attention networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden, July 13–19, 2018*, pages 3926–3932. ijcai.org, 2018. (Cited on page 89.)

[164] X. Ying, C. Xu, J. Gao, J. Wang, and Z. Li. Time-aware graph relational attention network for stock recommendation. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19–23, 2020*, pages 2281–2284. ACM, 2020. (Cited on pages 2 and 41.)

[165] 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, SIGIR 2016, Pisa, Italy, July 17–21, 2016*, pages 729–732. ACM, 2016. (Cited on pages 30, 34, 45, 65, 66, 68, and 90.)

[166] L. Yu, L. Sun, B. Du, C. Liu, H. Xiong, and W. Lv. Predicting temporal sets with deep neural networks. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23–27, 2020*, pages 1083–1091. ACM, 2020. (Cited on pages 35, 36, 68, 78, and 80.)

[167] H. R. Zagheli, M. Ariannezhad, and A. Shakery. Negative feedback in the language modeling framework for text recommendation. In *Advances in Information Retrieval – 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8–13, 2017*, pages 662–668, 2017.

[168] M. Zhang, S. Wu, M. Gao, X. Jiang, K. Xu, and L. Wang. Personalized graph neural networks with attention mechanism for session-aware recommendation. *IEEE Trans. Knowl. Data Eng.*, 34(8): 3946–3957, 2022. (Cited on page 89.)

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

[170] M. Zhou, Z. Ding, J. Tang, and D. Yin. Micro behaviors: A new perspective in e-commerce recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5–9, 2018*, pages 727–735. ACM, 2018. (Cited on pages 2 and 29.)

Summary

User satisfaction is considered a key objective for all service provider platforms, regardless of the nature of the service, encompassing domains such as media, entertainment, retail, and information. While the goal of satisfying users is the same across different domains and services, considering domain-specific characteristics is of paramount importance to ensure users have a positive experience with a given system. User interaction data with a system is one of the main sources of data that facilitates achieving this goal. In this thesis, we investigate how to learn from domain-specific user interactions. We focus on recommendation as our main task, and retail as our main domain. We further explore the finance domain and the demand forecasting task as additional directions to understand whether our methodology and findings generalize to other tasks and domains. The research in this thesis is organized around the following dimensions: 1) Characteristics of multi-channel retail: we consider a retail setting where interaction data comes from both digital (i.e., online) and in-store (i.e., offline) shopping. 2) From user behavior to recommendation: we conduct extensive descriptive studies on user interaction log datasets that inform the design of recommender systems in two domains, retail and finance.

Our key contributions in characterizing multi-channel retail are two-fold. First, we propose a neural model that makes use of sales in multiple shopping channels in order to improve the performance of demand forecasting in a target channel. Second, we provide the first study of user behavior in a multi-channel retail setting, which results in insights about the channel-specific properties of user behavior, and their effects on the performance of recommender systems. We make three main contributions in designing user-oriented recommender systems. First, we provide a large-scale user behavior study in the finance domain, targeted at understanding financial information seeking behavior in user interactions with company filings. We then propose domain-specific user-oriented filing recommender systems that are informed by the findings of the user behavior analysis. Second, we analyze repurchasing behavior in retail, specifically in the grocery shopping domain. We then propose a repeat consumption-aware neural recommender for this domain. Third, we focus on scalable recommendation in retail and propose an efficient recommender system that explicitly models users' personal preferences that are reflected in their purchasing history.

Overall, this thesis contributes to the research literature by providing novel insights in user behavior understanding and introducing state-of-the-art recommendation models. For future work, we call for research in the different directions that build on our findings. Our user behavior study in multi-channel retail settings is based on transaction logs only. Click behavior data is another important resource which we did not consider. Analyzing click data can help in detecting customer intentions for downstream tasks. Our experiments further show that the recommendation performance is not consistent across online-only, offline-only, and multi-channel settings. Designing a recommendation algorithm specifically for the multi-channel setting remains an unexplored direction. Finally, we focused on repeat recommendation across the thesis. However, helping users discover new items is another important goal of recommender systems. Understanding users' exploration behavior with previously unseen items as a stepping stone towards achieving this goal is another promising direction for future work.

Samenvatting

Gebruikerstevredenheid wordt beschouwd als een hoofddoelstelling voor alle platforms van dienstverleners, ongeacht de aard van de dienst, en omvat domeinen als media, entertainment, detailhandel en informatie. Hoewel het doel om gebruikers tevreden te stellen hetzelfde is voor verschillende domeinen en services, is het in aanmerking nemen van domeinspecifieke kenmerken van het allergrootste belang om ervoor te zorgen dat gebruikers een positieve ervaring hebben met een bepaald systeem. Gegevens over gebruikersinteractie met een systeem zijn één van de belangrijkste gegevensbronnen die het bereiken van dit doel mogelijk maken. In dit proefschrift onderzoeken we hoe we kunnen leren van domeinspecifieke gebruikersinteracties. Wij richten ons op aanbevelingen als onze hoofdtaak, en retail als ons hoofddomein. We onderzoeken verder het financiële domein en de het voorspellen van de toekomstige vraag naar een product of dienst als aanvullende richtingen om te begrijpen of onze methodologie en bevindingen generaliseren naar andere taken en domeinen. Het onderzoek in dit proefschrift is georganiseerd rond de volgende dimensies: 1) Kenmerken van multi-channel retail: we beschouwen een winkelomgeving waarin interactiegegevens afkomstig zijn van zowel digitaal (dat wil zeggen online) als in-store (dat wil zeggen offline) winkelen. 2) Van gebruikersgedrag tot aanbevelingen: we voeren uitgebreide beschrijvende onderzoeken uit naar datasets van gelogde gebruikersinteracties die het ontwerp van aanbevelingssystemen in twee domeinen, retail en financiën, informeren.

Onze belangrijkste bijdragen aan het karakteriseren van multi-channel retail zijn tweeledig. Om te beginnen stellen we een neuraal model voor dat gebruik maakt van gegevens over verkopen in meerdere winkelkanalen om de prestaties van de vraagvoorspelling in een doelkanaal te verbeteren. Ten tweede bieden we de eerste studie van gebruikersgedrag in een multi-channel retailomgeving, die resulteert in inzichten over de kanaalspecifieke eigenschappen van gebruikersgedrag en hun effecten op de prestaties van aanbevelingssystemen. We leveren drie belangrijke bijdragen aan het ontwerpen van gebruikersgerichte aanbevelingssystemen. Ten eerste bieden we een grootschalig onderzoek naar gebruikersgedrag in het financiële domein, gericht op het begrijpen van het zoekgedrag naar financiële informatie bij gebruikersinteracties met bedrijfsdocumenten. Vervolgens stellen we domeinspecifieke, gebruikersgerichte aanbevelingssystemen voor zogeheten *filings* voor die gebaseerd zijn op de bevindingen van onze analyse van gebruikersgedrag. Ten tweede analyseren we herhaal-aankopen in de detailhandel, met name in de boodschappenbranche. Vervolgens stellen we een neurale methode voor aanbevelingen met herhaald consumptiegedrag in dit domein voor. Ten derde richten we ons op schaalbare aanbevelingen in de detailhandel en stellen we een efficiënt aanbevelingssysteem voor dat expliciet de persoonlijke voorkeuren van gebruikers modelleert, die worden weerspiegeld in hun aankoopgeschiedenis.

Over het geheel genomen draagt dit proefschrift bij aan de onderzoeksliteratuur door nieuwe inzichten te verschaffen in gebruikersgedrag en door de introductie van *state-of-the-art* aanbevelingsmodellen. Voor toekomstig werk roepen we op tot onderzoek in de verschillende richtingen die voortbouwen op onze bevindingen. Ons onderzoek naar gebruikersgedrag in multi-channel retailomgevingen is uitsluitend gebaseerd op gelogde transacties. Gegevens over klikgedrag zijn een andere belangrijke bron waar we geen rekening mee hebben gehouden. Het analyseren van klikgegevens kan helpen bij het de-

tecteren van klantintenties voor downstream-taken. Uit onze experimenten blijkt verder dat de aanbevelingsprestaties niet consistent zijn in instellingen die alleen online, alleen offline, of via meerdere kanalen werken. Het ontwerpen van een aanbevelingsalgoritme specifiek voor de meerkanaalsomgeving blijft een onontgonnen richting. Ten slotte hebben we ons in het hele proefschrift geconcentreerd op herhaalde aanbevelingen. Het helpen van gebruikers bij het ontdekken van nieuwe items is echter een ander belangrijk doel van aanbevelingssystemen. Het begrijpen van het verkenningsgedrag van gebruikers met voorheen ongeziene items als opstap naar het bereiken van dit doel is een andere veelbelovende richting voor toekomstig werk.