Mining Information Interaction Behavior

Academic Papers and Enterprise Emails

In an increasingly digitized world, it is hard to imagine a life without interacting with digital information objects. The internet and mobile devices enable people to access information with ease: be it reading the hottest research paper, or replying to emails from a colleague far away, it is just a matter of a few key strokes, clicks, or swipes on touchscreens. With recent advances in natural language processing and its application in smart devices, people can even get what they want hands-free and through voice commands. As a result, we are witnessing a wealth of user interactions on all kinds of online platforms. Studying these user interactions help us understand users' information needs, their behavior patterns, and difficulties or failures when they interact with the system. Eventually, these observational insights shed light on possible directions to improve the system and the user experience. In this thesis, we have studied user interactions in the domain of academic search and recommender systems, and in enterprise emails. Our findings help predict user behavior and improve retrieval effectiveness.

Mining Information Interaction Behavior : Academic Papers and Enterprise Emails

Xinyi Li

Mining Information Interaction Behavior

Academic Papers and Enterprise Emails

Xinyi Li

mining information interaction behavior academic papers and enterprise emails
0100110
In an increasingly digitized world, it is hard to imagine a life without interacting with digital information objects

Mining Information Interaction Behavior

Academic Papers and Enterprise Emails

Xinyi Li

Mining Information Interaction Behavior

Academic Papers and Enterprise Emails

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. K.I.J. Maex
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in
de Agnietenkapel
op dinsdag 9 oktober 2018, te 10:00 uur

door

Xinyi Li

geboren te Changsha, China

The research was supported by the China Scholarship Council.

Cover by Xiaoxiao Meng
Printed by Off Page, Amsterdam

ISBN: 978-94-6182-909-2

Acknowledgments

I arrived in the Netherlands 4 years ago on a cold Amsterdam morning to start the journey of my PhD. As I stepped out of the customs, the officer's kind words were still whispering in my ears–"good luck with your PhD." But I knew I needed more than luck. It has not been an easy trip to finish the PhD, and I could never have done it without the help of many.

I would like to thank Prof. Maarten de Rijke for being my PhD supervisor. I feel extremely lucky to have him as a guide to navigate my research, and as a role model in many aspects. I thank him for his brilliance in research discussions, his patience with me when I was stuck, and his leadership in uniting us together and making a great team. His professionalism, diligence and dedication to his students are truly things to admire.

I would also like to thank Prof. Evangelos Kanoulas for being my co-supervisor, and providing great ideas during our discussions.

I am honored to have Prof. Natali Helberger, Dr. Christof Monz, Prof. Arjen P. de Vries, Prof. Ting Wang and Prof. Marcel Worring as my committee members.

I thank my collaborators from Elsevier, Bob Schijvenaars and Benj Pettit for their help and useful feedback.

The China Scholarship Council has provided the majority of the funding for my PhD, and the University of Amsterdam has provided supplementary subsidies. I thank them for their financial support.

It is an honor for me to work with the amazing people in ILPS and learn from them. I enjoyed the time we had together in soos talks, reading groups, and social activities like beach volleyball and dumpling parties. It is a long list of names, but everyone matters and I feel obliged to list them here: Adith, Aldo, Aleksandr, Alexey, Ana, Anne, Anya, Arianna, Artem, Bob, Boris, Caroline, Chang, Christof, Christophe, Chuan, Cristina, Daan, Damien, Dan, Dat, David, David, Dilek, Eva, Evgeny, Fei, Hamid, Harrie, Hendra, Hendrik, Hendrike, Hosein, Isaac, Ilya, Ivan, Jiahuan, Jie, Julia, Kaspar, Katja, Katya, Ke, Lars, Manos, Marc, Marlies, Maarten, Maarten, Marzieh, Masrour, Mostafa, Nikos, Pengjie, Petra, Praveen, Richard, Ridho, Rolf, Shangsong, Shaojie, Svitlana, Thorsten, Tobias, Tom, Xiaohui, Xiaojuan, Yaser, Yifan, Zhaochun and Ziming.

It was great to meet and get to know other people in the Faculty of Science at the University of Amsterdam: Dongdong, Guangliang, Hao, Huan, Hui, Jiajia, Jun, Junchao, Lifeng, Muhe, Ninghang, Que, Shuai, Shuo, Songyu, Wei, Xiaolong, Yang, Yongqiang, Yuan, Yunlu, Zhenyang, Zhichao, Zijian and Zhongyu. I thank them for their company.

I have done two industry internships during my PhD. I spent the summer of 2017 at Microsoft AI & Research. I thank Milad Shokouhi, Susan Dumais and Chia-jung Lee for their guidance and useful discussions. As of now I am interning in Amazon A9. I thank my manager Chris Severs for bringing me here, and Michael Shavlovsky for the daily mentorship.

Special thanks go to the US badminton club. For the past 4 years I have been playing badminton there, which gave me a strong body that helped me through paper deadline seasons, and released work stress. I thank the people that frequently play badminton with me: Anne, Anneke, Bas, Chi, Coen, Jens, Joep, Jun, Lintao, Niels, Nils, Rian, Viktor, Xikai, Yin, Zhengzhong and others.

I have met some of my best friends there. I thank Hardy and Sander for all the fun time together and letting me know more about the Dutch culture. I thank Pakho, Chiwai and Illona for all the badminton, basketball and food trips. I also want to thank my friends who are far away from me but have provided support: Jiahua, Jingchao and Yi.

Last but not the least, I thank my parents for their unconditional love, understanding and support.

Xinyi Li
Mountain View, August 11, 2018

Contents

1
Introduction

Digitization has changed the way people interact with information objects. On the one hand it has eased our efforts to acquire the desired information, whatever form this information is presented in: there is no more hustle for browsing through bookshelves and scrolling over pages only to find that one paragraph of text, thanks to modern search engines [20, 71, 186]; no more digging through record crates to listen to that one favorite song, which can be easily accessed via music streaming services [72, 199]; and no need to borrow video tapes to watch movies at home, which can be delivered instantly by video streaming platforms [161, 229]. User interactions with information objects have become increasingly easy and fast, as users save time from performing the physical activities that were once required to access them. The interactions have also become more flexible and less constrained by space, and what was once predicted more than 20 years ago has now become reality [118]: with any kind of smart device (be it a cell phone or a tablet) users are able to access digitized contents on the internet not only whenever, but also wherever they want.

This growing level of convenience gives rise to to an increasing growth of user interactions on these online service platforms [166, 167, 188]. For instance, there are over 3.5 billion searches performed per day on Google [73]. The growing number of user interactions has in turn been the propelling force for the digitization process of information objects, and the development of these online platforms. Take academic publications as an example, there are over 14 million articles indexed on ScienceDirect [187], a major academic search engine. The sheer size of the data makes it increasingly difficult for users to pinpoint what they are looking for. Users can enter textual queries to search for their information objects. These queries, however, may not always generate the search results that they want, and sometimes they even return no relevant results [8]. Besides, user intentions and information needs vary from each other, making it hard for a system to understand the specific information need of a user (e.g., the query "apple" can mean completely different things by two users). There is an ever growing demand from users to obtain the desired information objects in a smooth and personalized experience throughout the interaction process, and correspondingly an ever growing demand for online platforms to deliver a good performance.

The large volume of logged user interactions has attracted much research interest. The benefits brought by mining user interactions are obvious: understanding users may help systems understand their information needs and their behavioral patterns, and to recognize any difficulty or failure that they encounter while interacting with the systems. These insights will in turn lead to better implementations and richer functionalities of the online platforms so as to improve the user experience. For example, in web search scenarios user query logs are utilized in query suggestion [222] and query auto-completion [33] techniques to recommend relevant queries for users. Shopping receipts from online shopping websites can help us understand the consumer shopping patterns, and can be used to make predictions of the next purchase [115]. Users' interaction logs with music streaming services provide hints for their tastes, and can facilitate music recommendations [185].

While many research efforts have been spent on the aforementioned domains, relatively little attention has been paid to uncover the domain of academic search, which involves *researching the researchers*. Academic search is a specialized domain that involves the searching of academic information objects such as papers, authors and journals. While there are existing works that analyze academic search behavior, they are limited in that the study is often performed on a very small group of users [107] or on a single discipline [83, 102], and they only provide high-level statistics without delving into user actions within and across the search sessions. It is also not clear how to use those findings to improve academic search services. Therefore, the findings are not really generalizable and meaningful for system improvements.

As academic search is an important part of scientific activities, so is communication. Out of many communication methods, emails play a very important role [59, 198, 216]. Emails are frequently used when communication is not convenient face-to-face, or when things need to be clearly presented in written form. They can also be used as a way to share electronic files. Recently, through email newsletters, users can also receive academic paper recommendations. Examining user interactions with emails can improve the understanding of the user behavior on email platforms, and also shed light on possible system improvements to make email reading more efficient.

In this dissertation, we address the problem of examining the information interaction behavior in the online setting. We focus on academic search and on enterprise emails. For academic search, we provide a characterization of query and download behavior and also demonstrate application scenarios to improve the system; for email interactions, we study the core action–email reading, which is the only common behavior for different email interaction scenarios.

1.1 Research outline and questions

The theme of this thesis centers around mining information interaction behavior. Specifically, we investigate two scenarios: (1) user interactions in an academic search environment (from Chapter 2 to Chapter 5), and (2) the email reading behavior with enterprise emails (in Chapter 6). Below, we list the research question in each research chapter.

1.1.1 Queries and search failures in academic search

While there has been previous research on academic search queries, it is lacking in two aspects: (1) the observations are not drawn from a large query log to make the findings generalizable, and (2) there is no discussion of user behavior within academic search sessions. In the thesis we approach the queries and failure phenomenon in academic search and try to fill the gaps in our understanding of both. We start with a large-scale observational study based on a transaction log to understand what characteristics they have, and then provide an algorithmic fix for the query failures. Correspondingly, we want to answer the following research question:

RQ1 What are the characteristics of queries and failures in academic search, and how do we remedy failures?

To answer this question, we provide the first characterization of academic search queries that is based on a large-scale transaction log analysis, bringing more generalizable findings than previous small-scale studies [83, 102, 107]. Then, we zoom in on the failure phenomenon–null queries. We characterize null queries and the sessions where they appear. Finally, a query suggestion method is proposed to deal with null queries, which builds on query entities, session information and user preferences.

1.1.2 Topic shift and query reformulation patterns in academic search

Moving on, we extend the study to academic searchers' behavior over a relatively long period. In particular, we are interested in their query reformulation strategies and their topic interests over time. While query reformulations can be seen as explicit user behavior, topic interests are by contrast more implicit. It is unclear how each evolves over time, and how they are correlated. Therefore we investigate the following research question:

RQ2 Do topic shift and query reformulation patterns correlate in academic search?

To answer the question, we define a taxonomy of query reformulations in academic search that includes *revisiting a previous query*, *adding terms*, *dropping terms*, *substituting part of the query*, and *issuing a completely new query*. Different from findings in web search, we find that revisiting and issuing new queries tend to happen more often in academic search. We take a quantitative approach to study topic shift over time by using an LDA model [22]. In this process we identify two types of user: one type, in general increasingly focuses on their topics over time and the other diversifies over time. And finally, we study how query reformulation and topical shift are correlated with each other. To our surprise, we find that user's query reformulation patterns have little correlation with topic shift, i.e., users with distinct reformulation preferences in search could be equally likely to be diversifying or focusing on their topics. However, adding terms and issuing new queries may help predict the immediate topic shift.

1.1.3 Characterizing and predicting downloads in academic search

We study a different type of user behavior in this chapter–downloads. The download behavior in our setting refers to requesting the PDF file of an article. We are motivated to characterize the within session and cross session download behavior. Then using some of the insights gained, we try to predict the user download behavior. We bring up the following research questions:

RQ3 What are the characteristics of user download behavior in academic search and how can we predict that?

We introduce a new dataset for downloads in academic search and characterize user interactions with academic search engines. We study users' actions across sessions, revealing correlations among various behavioral signals and explaining the topical aspects of user downloads. Finally, we build a specialized model for download prediction that utilizes user session history and that is based on user segmentation, which leads to significant improvements over a state-of-the-art baseline.

1.1.4 Reranking paper recommendations using paper content and user behavior

For the research topics listed so far we have been focusing on user interactions with an academic search engine. While most of the user information needs can be met within search sessions, there is a gap to be filled in between search sessions, where a recommender system comes into play. We examine a production paper recommender system that tries to predict user information needs by sending recommendation emails. We address the task of reranking paper recommendations so as to improve the system performance. To this end we aim to answer this research question:

RQ4 How do we utilize both content and behavior to rerank paper recommendations?

First, we propose several content-based measures that are derived from paper aspects, such as word space similarity and author similarity from an embedding space. Next we use joint matrix factorization to learn a mapping from user browsed papers to user clicks on the recommendations, to alleviate the insufficiency of user clicks in the data. We use a pairwise learning model to rerank the paper recommendation candidates that eventually leads to better results in the offline evaluations.

1.1.5 Characterizing reading time on enterprise emails

Finally, we turn to another aspect of information interaction behavior–communication. We examine the reading behavior on enterprise emails. The act of reading is arguably the only activity that is in common in most of the interactions that users have with their emails. We are interested to see how users read their enterprise emails, and how various factors impact reading time. Naturally we bring up the following research question:

RQ5 How do users *read* their emails and how their *reading time* is affected by various contextual factors?

We start with a quantitative analysis of a large sample of enterprise emails from the web and mobile clients of a popular email web service. We uncover the distribution of email reading time and then examine how reading time is affected by various factors, such as email types, user context, cognitive load, re-reading behavior and platforms. To complement the results based on log analyses, we also conduct a user study to look for the causes behind interesting observations from the logs.

1.2 Main contributions

The thesis is devoted to uncovering the interaction behavior of users with information objects in an online setting, and also leveraging insights obtained in this way to provide suggestions to improve both the system performance and user satisfaction. It primarily focuses on user interactions with academic information services, i.e., an academic search engine and an academic recommender system in our research. It also covers user interactions with an email client because the prevalent usage of email communications, which happens quite frequently also during academic cooperations.

1.2.1 Empirical contributions

1. The first large scale characterization of queries in academic search, that highlights the differences between academic search and web search such as query length and query types. (Chapter 2)
2. The first taxonomy of query failures in academic search is defined. (Chapter 2)
3. A taxonomy of query reformulations in academic search is introduced that is different from web search. Characterizations of the topic shifts and query reformulations in academic search over a long term, include types of topic shifts and different query reformulation strategies. The correlations between query reformulation types and topic shifts: there is little correlation in the long term; certain reformulation types are correlated with immediate topic shifts. (Chapter 3)
4. Characterizations of user behavior patterns within and across download sessions in academic search, including actions and action trajectories, time gaps between downloads and temporal patterns. (Chapter 4)
5. The correlations across download sessions, e.g., the number of queries is negatively correlated with the time gap until the next download, and the number of downloads are positively correlated across sessions. (Chapter 4)
6. Behavioral differences of downloads in different topical aspects: users interested in different subjects have different download behavior, reflected by their queries, downloads, and topical coherency. (Chapter 4)
7. Revelations of various user-specific and email-specific factors that correlate with reading time, which can be used to develop better metrics and user models for understanding and improving email interactions. (Chapter 6)

1.2.2 Methodological contributions

8. A session-based query suggestion approach to remedy query failures that is built on a query-entity graph. (Chapter 2)
9. A personalized prediction model for downloads by segmenting users into groups through Dynamic Time Warping, and an LSTM-based model based on user segmentation for predicting download. (Chapter 4)
10. Various measures for comparing paper similarity built on paper metadata. A hybrid reranking model that utilizes the metadata of academic papers and user interactions. (Chapter 5)

11. A method to approximate user email reading time that can be applied on large amounts of user interaction data. (Chapter 6)

1.3 Thesis overview

We provide the overview of the thesis in this section. The thesis consists of seven chapters, including five research chapters. Chapter 1 is the introduction that sets the scene for the research. Chapter 2–5 are the research works on academic search, starting with queries then moving on to download behavior. Chapter 6 is about characterizing reading time on enterprise emails. Chapter 7 contains the conclusion.

- **Chapter 2–Investigating Queries and Search Failures in Academic Search.** In this chapter we reveal important observations about academic search queries, and provide an algorithmic solution to address a type of failure during search sessions: null queries. We start by providing a general characterization of academic search queries, by analyzing a large-scale transaction log of a leading academic search engine. Unlike previous small-scale analyses of academic search queries, we find important differences with query characteristics known from web search. E.g., in academic search there is a substantially bigger proportion of entity queries, and a heavier tail in query length distribution. We then focus on search failures and, in particular, on null queries that lead to an empty search engine result page, on null sessions that contain such null queries, and on users who are prone to issue null queries. In academic search approximately 1 in 10 queries is a null query, and 25% of the sessions contain a null query. They appear in different types of search session, and prevent users from achieving their search goal. To address the high rate of null queries in academic search, we consider the task of providing query suggestions. Specifically, we focus on a highly frequent query type: non-boolean informational queries. To this end we need to overcome query sparsity and make effective use of session information.
  We find that using entities helps to surface more relevant query suggestions in the face of query sparsity. We also find that query suggestions should be conditioned on the type of session in which they are offered to be more effective. After casting the session classification problem as a multi-label classification problem, we generate session-conditional query suggestions based on predicted session type. We find that this session-conditional method leads to significant improvements over a generic query suggestion method. Personalization yields very little further improvements over session-conditional query suggestions.
- **Chapter 3–Topic Shift and Query Reformulation Patterns in Academic Search.** We focus on two aspects: *query reformulation patterns* and *topic shifts in queries*. We first analyze how each of these aspects evolves over time. We identify important query reformulation patterns: revisiting and issuing new queries tend to happen more often over time. We also find that there are two distinct types of user: one type of users becomes increasingly focused on the topics they search for as time goes by, and the other becomes increasingly diversifying. After analyzing these two aspects separately, we investigate whether, and to which degree, there is a correlation between topic shift and query reformulation. Surprisingly, users' preferences of query reformulations correlate little with their topic shift tendency. However, certain reformulations may

help predict the magnitude of the topic shift that happens in the immediate next timespan. Our results shed light on academic searchers' information seeking behavior and may benefit search personalization.

- **Chapter4–Characterizing and Predicting Downloads in Academic Search.**
  We start with a description of a unique dataset of a particular type of conversion in academic search, viz. users' downloads of scientific papers. Then we move to an observational analysis of users' download actions. We first characterize user actions and show their statistics in sessions. Then we focus on behavioral and topical aspects of downloads, revealing behavioral correlations across download sessions. We discover unique properties that differ from other conversion settings such as online shopping. Using insights gained from these observations, we consider the task of predicting the next download. In particular, we focus on predicting the time until the next download session, and on predicting the number of downloads. We cast these as time series prediction problems and model them using LSTMs. We develop a specialized model built on user segmentations that achieves significant improvements over the state-of-the art.
- **Chapter 5–Reranking Paper Recommendations Using Paper Content and User Behavior.**
  We examine an academic paper recommender that sends out paper recommendations in email newsletters, based on the users' browsing history on an academic search engine. Specifically, we address the task of reranking the recommendation candidates that are generated by a production system.
  We propose an approach to reranking the candidates that utilizes both paper content and user behavior. The approach is designed to suit the characteristics unique to the academic recommendation setting, for instance, the inclusion of a knowledge graph derived from paper metadata. We show that the proposed method outperforms the production baseline significantly, providing a relative 13% increase in Mean Average Precision and 28% in Precison@1. Moreover, we provide a detailed analysis on the model components, highlighting where the performance boost comes from. The obtained insights reveal useful components for the reranking process in the academic recommendation setting, such as the usage of graph embedding similarity. Also, the recent papers browsed by users and authors provide strong evidence for recommendation.
- **Chapter 6–Characterizing Reading Time on Enterprise Emails.**
  In this chapter, we characterize how users read their enterprise emails, and reveal the various contextual factors that impact reading time. Our approach starts with a reading time analysis based on the reading events from a major email platform, followed by a user study to provide explanations for some discoveries. We identify multiple contextual factors that are correlated with reading time. For instance, users spend more time reading emails when they have fewer meetings during the day. In addition, we find that users also reread emails across devices: 76% of reread emails are first visited on mobile and then on desktop. Our study is the first to characterize enterprise email reading time on a large scale. The findings enrich the understanding of email reading behavior and provide insights to improve the email system.
- **Chapter 7–Conclusion.**
  We summarize the main findings in this chapter, point out the limitations, and outline

future directions.

1.4 Origins

We list the paper origins of the chapters below as well as related research to the chapters. The main research chapters in the thesis are based on the following papers:

- **Chapter 2** is based on the following paper: X. Li, R. Schijvenaars, and M. de Rijke. Investigating queries and search failures in academic search. *Information Processing & Management*, 53(3):666–683, May 2017.
  Bob Schijvenaars helped with preparing the data and conducted the user study. The methods were proposed by Xinyi Li. The experiments were conducted by Xinyi Li. All authors contributed to the writing.
- **Chapter 3** is based on the following paper: X. Li and M. de Rijke. Do topic shift and query reformulation patterns correlate in academic search? In *ECIR*, pages 146–159, April 2017.
  The research question was proposed by Xinyi Li. The experiments and analyses were conducted by Xinyi Li. Both authors contributed to the writing.
- **Chapter 4** is based on the following paper: X. Li and M. de Rijke. Characterizing and predicting downloads in academic search. *Information Processing & Management*, under review.
  The research question was proposed by Xinyi Li. The method was proposed by Xinyi Li. The experiments and analyses were conducted by Xinyi Li. Both authors contributed to the writing.
- **Chapter 5** is based on the following paper: X. Li, Y. Chen, B. Pettit, and M. de Rijke. Reranking paper recommendations using paper content and user behavior. *ACM Transactions on Information Systems*, under review.
  The research question was proposed by Xinyi Li. Benjamin Pettit helped with preparing the data. Xinyi Li conducted most of the experiments and analyses. Yifan Chen contributed to the behavior matching component in the model. All authors contributed to the writing.
- **Chapter 6** is based on the following paper: X. Li, C.-j. Lee, M. Shokouhi, and S. Dumais. Reranking paper recommendations using paper content and user behavior. *Journal of the Association for Information Science and Technology*, under review.
  This piece of work was done during an internship in Microsoft Artificial Intelligence & Research in 2017. The task was proposed by Milad Shokouhi. Xinyi Li did most of the experiments and analyses. Chia-jung Lee helped to provide the data and contributed to part of the experiments. All authors contributed to the writing.

This thesis also indirectly benefits from the findings and insights of the following research:

- X. Li and M. de Rijke. Academic search in response to major scientific events. In *The 5th International Workshop on Bibliometric-enhanced Information Retrieval*, pages 41–50, 2017.
  The research question was proposed by Xinyi Li. The analyses were conducted by Xinyi Li. Both authors contributed to the writing.
- X. Li, J. Tang, T. Wang, Z. Luo, and M. de Rijke. Automatically assessing Wikipedia

article quality by exploiting article-editor networks. In *ECIR*, pages 574–580, April 2015.
The research question was proposed by Xinyi Li. The experiments were conducted by Xinyi Li. All authors contributed to the writing.

Part I

Academic search

2 Investigating Queries and Search Failures in Academic Search

In Chapter 1 we have set the scene for the research chapters. We start with academic search and examine **RQ1**, which help us understand academic search queries and failures.

2.1 Introduction

Academic search systems existed long before the advent of the World Wide Web [144]. Early systems such as MEDLINE were made for librarian use and supported only preprogrammed queries and a limited number of simultaneous users in a non-interactive way. Modern academic search engines have taken completely different forms, with support of ad hoc queries, which users themselves enter in a search box, made for public use, and with support for many online users in an interactive search environment. In this chapter we look at academic search in this modern form. In recent years there have been numerous studies on a range of aspects of academic search [88, 165, 170, 171]. These studies reveal the information-seeking behavior of researchers by conducting surveys or user studies on a relatively small sample of researchers. They point out that academic search engines have become the primary portal for researchers to gain information. Surprisingly, despite the widespread usage of academic search environments, very little is known about the actual search behavior of users of academic search engines that is based on a large-scale transaction log analysis. Query log analysis is known to be a valuable source for improving search systems [193], whether it concerns query suggestions, learning to rank or personalization. Numerous analyses have been conducted on commercial web search engines, such as Microsoft Bing, Yahoo and AOL [16, 18, 218] as well as on verticals such as blog search [151] or people search [219]. However, little work has been done on academic search engines. A small-scale analysis of a transaction log of ScienceDirect, a leading academic search engine for multiple disciplines, reveals basic statistics such as page views, article downloads, and journal access frequencies [107]; the study was conducted 14 years ago on a district scale, based on only 0.42 million queries, and it does not differentiate

This chapter was published as [138].

between individual users due to limitations of the logging system at the time, and hence it does not inform us about users' behavior during search sessions. In this chapter, we analyze a large, global-scale academic search transaction log containing over 39 million queries in which we are able to track and analyze the behavior of individual users.

Our analysis of user behavior comes in two stages. First, we set the scene by providing a general descriptive analysis of academic queries and highlighting the differences between academic search and web search. Then we zoom in on search failures, and in particular on so-called null queries for which the search engine produces an empty result page as defined in [82]. We study null queries from three angles: queries, sessions and users. Compared to web search, where they make up less than 2% of all queries [8], null queries appear more frequently in academic search. A recent study shows that the null query rate is 15% in PubMed, a popular biomedical academic search engine [55]. As we will see below, in ScienceDirect the null query rate is over 10%. When checking whether planned research is novel, an empty SERP (search engine result page) may be a desirable outcome, but in general it is an outcome that a search engine wants to avoid, thus motivating the development of effective strategies for dealing with null queries and to guide users, either automatically or interactively, to consider alternative queries.

To address the failure phenomena uncovered by our log analysis, we consider a query suggestion task. Query suggestion is a feature in modern search that improves the search experience by providing query recommendations. Previous work on query suggestion comes in several flavors, from suggestions based on syntactic variations to suggestions based on semantic relatedness or behavioral similarity. Techniques based on behavioral similarity perform well for so-called head queries, that occur frequently. The long tail of queries is more challenging for query suggestions, since most of these queries are rare. We contribute an approach to query suggestion for null queries in academic search. We discover that there are different types of null queries and specifically target a highly frequent type: non-boolean informational queries.

In particular, we make use of our analysis of academic search queries by considering entities. This helps us overcome the sparsity issue in academic search queries. Moreover, using automatically predicted types of sessions, we condition our query suggestions on the type of null session by reranking different types of query suggestion candidates. Query suggestion candidates are first generated using graph-based models that incorporate different kinds of relations: links between queries and entities, transitions between queries and transitions between entities. Then they are reranked based on the predicted session types. We also build a personalized model by considering users' preferences of entities.

In this chapter, specifically, we address the following research questions:

RQ1.1 What are academic search queries and how are they different from web search queries?

We perform a transaction log analysis on an academic search engine. We examine the query-level characteristics of academic search, namely query length and query types. The observational results demonstrate clear differences between academic search queries and web search queries.

RQ1.2 What are the characteristics of null queries in academic search?

We first look at null queries, then in null sessions and users to understand how null queries happen. The insights enable us to devise a method to cope with null queries.

RQ1.3 How do we use query, session and user information to make query suggestions for null queries?

We first define the task of using query suggestions to address null queries. Then we use the obtained observational insights (entity-richness, different session types) to improve the ranking of query suggestions.

Our novel contribution of this chapter is three-fold. First is the analysis of academic search queries and their differences from web search. Our findings based on a large-scale transaction log analysis give more insights into query contents and user behavior in sessions than previous small-scale analyses [83, 102, 107]. Second, we drill down on search failures in academic search and thoroughly analyze problems around null queries. Third, we propose a query suggestion method that addresses null queries when they occur. Our query-conditional method uses entities to overcome the severe sparsity in academic search queries. Moreover, our session-conditional query suggestion method results in significant improvements over state-of-the-art query suggestion baselines. We find that a personalized model that infers user preferences of entities further improves query suggestion performance.

Our main findings can be summarized as follows.

1. Academic search differs from web search in query properties, namely in length distribution, query types and noticeable entity richness.
2. Frequent null queries are a unique phenomenon in academic search. They happen more frequently compared to web search, and they happen in various types of sessions, such as refining and exploratory sessions.
3. Query-conditional and session-conditional query suggestion methods improve over methods that do not consider entities in queries and session types, respectively. However, when to apply personalization effectively remains an open problem.

In Section 2.2 we describe the dataset and query properties that form the basis for the remainder of the chapter. In Section 2.3 we focus on failures in academic search: null queries, null sessions and users who frequently issue them. In Section 2.4 we describe our model of query suggestions to address null queries. In Section 2.5 we discuss personalization. In Section 2.6 we discuss related work. We present our conclusions in Section 2.7.

2.2 Dataset

As we will see in the related work section (Section 2.6), previous analyses of academic search logs are limited in scale. We are particularly interested in studying user behavior from large-scale log analysis, which is why we conduct a new transaction log analysis.

We study a 5-month query log from the ScienceDirect search engine collected from September 28, 2014 to March 5, 2015. Due to institution-authorization from ScienceDirect, users in a certain IP range are able to access the search engine from shared devices (e.g., library computers), and they share the same session ID and user ID in the transaction log. Moreover, many institutions have proxies or firewalls whose IP is recorded instead of that of the terminal device. Therefore, it is difficult to differentiate

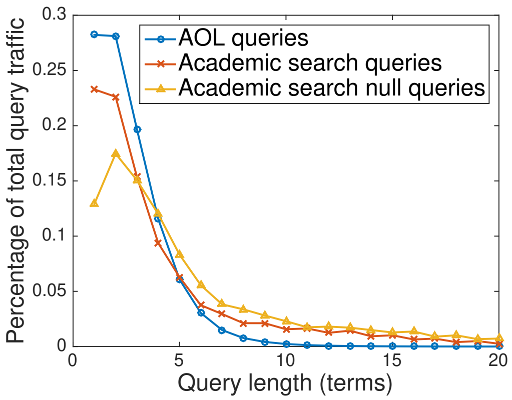


Figure 2.1: Query length distribution of web search queries, academic search queries and academic search null queries

these IP-users in the log. It is only safe to assume a one-to-one mapping between IDs and users when users log in, or when they access the search engine from outside the institution; we refer to such users as *non-IP users*. They contribute about one third of the total query traffic.

2.2.1 Queries

In this section we describe the query characteristics as identified from the transaction log. Table 2.1 contrasts the statistics of academic queries with those of web queries. A marked difference between academic search and web search is the query length, where academic queries are on average 1.4 words longer than web queries. Figure 2.1 shows that the distribution of academic queries follows a power law, which is similar to web search queries [12] but featured with a "bigger tail." The query length statistics in our study differ from the much smaller scale log analysis in [107], e.g., the average length of our queries is 1.5 words longer than their findings.

The verbosity of academic search queries yields multiple challenges for the search engine. One of them is sparsity, which makes it difficult to generate query suggestions for rare queries [25].

Table 2.1: Query length statistics in word count. The AOL query log covers over 650,000 users in a 3-month period. The AOL log statistics come from [10].

Category	**#N**	**min**	**max**	**mean**	**median**
AOL	21M	1	245	2.34	2
Sciencedirect	39M	1	419	3.77	3

Apart from query length, academic queries also differ from web queries in their content and intent. Below we exemplify three typical query intents known from web search [95] in the academic search setting:

Navigational queries A "navigational query" in academic search guides the user to a certain publication (identified by special operators such as DOI, ISBN or a "title" operator). E.g., the query "DOI(10.1111/ jcmm.12096)" seeks to locate a specific publication. These queries are sometimes referred to as known-item queries [168]. Using automatic identification with the special operators, these queries are found to make up 7.6% of all queries.

Transactional queries These are queries that directly aim to retrieve academic information resources, e.g., a PDF file. For instance, "oil paper filter design pdf" and "download journal pressure sensor." The proportion of this type of query is only 0.5‰.

Informational queries Queries that seek, refine or explore research topics act as "informational queries." For example, "vitamin C and cosmetic" aims to retrieve information resources on the topic "vitamin C and cosmetic." We find that the majority of queries belong to this type, making up 92.3% of all the queries. However, we notice that the boundary between query types is not strict. For instance, an "informational query" may first lead to information resources, then a user might not be content with the acquired information and further downloads a resource (e.g., a PDF file), thereby transforming the query into a "transactional query." And sometimes, the user might type in a few key words with the purpose of locating a specific paper. These queries may appear to be informational queries but act as navigational queries in effect. Thus it can be difficult to distinguish these navigational queries accurately, as is also shown by [108].

Besides the three familiar categories just given, which correspond to categories commonly used to classify web queries, we also identify the following types of query:

Entity queries Academic search differs from web search in the proportion of queries that contain named entities. Entities in academic queries are identified using a controlled vocabulary derived from key concepts in papers, author names etc. They are not exactly the same as the entities in web search (people, places or things), but they both represent something that is existent. In web search the percentage of entity queries varies from 43% up to 70% [81, 142]. In academic search, the proportion of queries that contain entities is 92.37%, far more than in web search queries.[1]

Boolean operator queries These queries are often issued by advanced searchers who like to use boolean operators to perform precise match. Specifically, AND, OR and

[1]In web search it is common to apply entity disambiguation to queries, and link entities to, e.g., Wikipedia, DBpedia and Freebase [21, 150]. In academic search, queries are mostly disambiguated (e.g., in most cases technical terms and names are unique), therefore we simplify the entity linking to matching query terms to a thesaurus. To identify entities, we use a human-calibrated thesaurus derived from over 13 million academic papers. The thesaurus is built and maintained by domain experts at Elsevier who review the papers manually. We apply left-most longest matching to extract entities in queries, which provides a lower bound on the number of queries containing entities. There may still be a small portion of them that need disambiguation [204], but elaborating on this task is beyond the scope of this chapter.

NOT operators (in upper case) are applied to address complex search intents. These queries constitute 8.2% of the total traffic.

Now that we have provided the basic descriptive statistics of our dataset, we can zoom in on the phenomenon that is at the center of our interest in this chapter: search failure.

2.3 Search failure

In this section we zoom in on search failures, in particular on null queries, whose high frequency is peculiar for our academic search setting. We analyze null queries and the sessions that contain them (null sessions). We also analyze users who frequently issue null queries.

2.3.1 Null queries

Null queries are queries that lead to an empty search engine result page; they present a severe challenge to the search engine. In academic search, null queries appear four times more often than in web search. Specifically, in our academic search setting the null query rate is 10.3%, much higher than the 2% reported for web search [8]. Altingovde et al. [8] show that null queries in web search contain un-indexed URIs (unique resource identifier), or they are merely meaningless queries. In academic search the situation of null queries is different. For instance, Figure 2.1 shows that null queries tend to be longer than normal queries, which brings in more sparsity in the queries. And at 19.6%, the proportion of boolean operator queries amongst null queries is higher than amongst all queries (8.2%). The vast majority of null queries (80.1%) are non-boolean informational queries.

2.3.2 Null sessions

Null sessions constitute 25.0% of all sessions, indicating the frequent appearances of null queries. Given the high frequency of null queries, we expect that they feature in diverse search contexts. Put differently, if we define a null session to be a session in which a null query occurs, then what types of null sessions occur in the logs? We hypothesize that understanding null sessions should help us to identify a solution to address null queries. To address the question, we annotate null sessions, using 2 professionals with years of experience in academic search engines and a researcher in information retrieval; in total, 300 sessions are annotated.

To begin, null session types were identified through an initial pass through the data by our annotators. This gave rise to a total of 7 null session types; a "stereotypical example" was provided for every null session type. Subsequently, our annotators annotated all null sessions with one or more session types; the annotation is non-trivial as each session may have several types. Below, we provide definitions and examples for the types of session that occur; the null query is marked in ***bold italics***.

(1) Refining sessions. Failure in refining sessions happens when users are searching around one goal. After a failure happens, the user reformulates the query by rephrasing a synonym or correcting spelling mistakes etc, as shown in the following example:

```
04Mar2015:15:42:03.203  Query  pid acel
04Mar2015:15:42:17.041  Query  pid accelerometer
04Mar2015:15:42:38.643  Query  pid accelerometer
04Mar2015:15:42:54.911  Click  shorturl=/scie...pii/S0141029614005793
```

(2) Generalizing sessions. Some queries contain too many entities and cause the search engine to return zero results. Users in this setting may drop terms to obtain some results. We call this a generalizing session as shown in the following example:

```
03Mar2015:23:36:45.022  Query  Leaf blast (Magnaporthe oryzae)
03Mar2015:23:37:35.456  Query  Leaf blast
03Mar2015:23:37:43.372  Click  shorturl=/scie...pii/S1049964411001009
```

(3) Exploring sessions. Users often formulate queries around a pivot entity. They may encounter a failure in their exploration but continue to explore. In the example below, the user encounters a failure when searching around "composite beams," and keeps reformulating the query until finally obtaining a click:

```
01Mar2015:08:55:07.291  Query  Determination of rotation capacity...and composite beams
01Mar2015:08:55:46.910  Click  shorturl=/scie...pii/S0143974X9593900O
01Mar2015:09:03:18.931  Query  a stady on elastic plastic...of composite beams
01Mar2015:09:03:37.627  Query  a study on elastic plastic...of composite beams
01Mar2015:09:04:19.937  Query  behavior of composite beam...dynamic testing
01Mar2015:09:05:29.587  Query  Inelastic analysis of steel frames with composite beams
01Mar2015:09:06:48.446  Click  shorturl=/scie...pii/S0141029613004379
```

(4) Item search session. Users sometimes search with a very specific goal, e.g., looking for an article or a book with a unique ID (e.g., DOI). They issue a navigational or transactional query, and the search engine returns no results if the desired item is not indexed, which may be due to a mal-formed query, e.g. one that occurs with an error in DOI. Or when they query is correct, the failure may be due to the fact that the item is not in the database. For such failures, the search engine is unable to satisfy the user's intent. However, if the item is not indexed online, this may not be a "failure" as the user has confirmation that the item is not online. In the following example the user searches a book using a title identifier and finds no results for the first query.

```
01Mar2015:12:34:14.148  Query  (ttl(Identification of micro-RNA...))
01Mar2015:12:35:01.556  Query  (ttl(end-stage heart failure...))
01Mar2015:12:39:39.638  Query  (DOI(10.1111/jcmm.12096))
```

(5) Expanding sessions. It may happen that users add terms when the current query returns no results. This looks like query specialization but it is not. In query specialization the first query is usually a successful one with results, despite being too general to the user. Here it is a null query. Adding terms might increase the possibility for the search engine to have a matching term in the query. In the example below, the user initially enters a query that causes a failure, and then adds a term to obtain some results.

```
05Mar2015:13:51:45.227  Query  Ulocladium
05Mar2015:13:52:25.827  Query  ulocladium helioanthus
```

(6) Dropout. Dropout sessions are sessions where users abandon search when failure happens, as shown in the example below.

```
04Mar2015:08:52:02.891  Query  ROS1 Rearrangements ... Lung Cancers
```

(7) Anomaly sessions. Some users enter non-ASCII characters that are not supported by the search engine, or other meaningless queries that cause a failure; 20% of the sessions that we annotate belong to this type. It is useless to provide query suggestions for these sessions so we remove them from further consideration in our query suggestion work.

Table 2.2 (top) lists the agreement scores for the null session annotation task. We consider Percentage Agreement [11], Kappa Agreement [67], Observed Disagreement and Expected Disagreement [119]. The Percentage Agreement divides the number of agreements by the item count, while the rest assume the same probability distribution for all raters. Note that this is a multi-label annotation task (one session may have several labels), therefore it is not easy to achieve high agreement scores among the annotators. The annotation agreement is fair by Kappa values [125] for all types of null sessions. We also compute agreement for a subset of the null sessions where the users have subsequent actions after the null query happens, namely those that are not of type dropout (type 6) or anomaly (type 7). For this set of null sessions, the annotation agreement is moderate [125]; see Table 2.2 (bottom).

Certain session labels are prevalent, e.g., refining (39%) and generalizing (26%), as it is common for users to modify or drop a query term upon a null query. Also, 26% of the sessions are item-search oriented whose failures are due to non-existent resources. Exploring (15%) and expanding session labels (13%) are found to be less frequent. And 32% of the sessions ended up as dropout sessions, as the users give up the search goal.

Table 2.2: Annotation agreement; annotators are denoted as 0, 1, and 2.

	0+1	**0+2**	**1+2**	**0+1+2**
All sessions in the sample				
Percentage Agreement	0.52	0.35	0.32	0.40
Kappa Agreement	0.43	0.23	0.20	0.29
ObservedDisagreement	0.48	0.65	0.68	0.60
ExpectedDisagreement	0.85	0.86	0.85	0.86
Leaving out sessions of type dropout or anomaly				
Percentage Agreement	0.57	0.62	0.65	0.61
Kappa Agreement	0.39	0.45	0.53	0.46
ObservedDisagreement	0.43	0.38	0.35	0.39
ExpectedDisagreement	0.71	0.69	0.76	0.72

2.3.3 Users who fail frequently

Next we consider users who frequently fail, that is, who frequently submit null queries: "at least 20% of their queries are null queries." These users account for 20.1% of the users who have submitted at least one null query. We analyze these users and compare them with "normal users," who encounter null queries less frequently.

First, we try to find out if a user's consistency of interests affects their null query rate. To determine a user's consistency of interests, we look at the "self-similarity score" and "cross-session similarity score" of users.

Our notion of self-similarity is defined as follows. We model users as a bag of queries, i.e., profiling every user using all the queries they issued. We measure the similarity score of each user in a pairwise manner by treating each query as a bag of words and compute the mean similarity score between all the query pairs [84]. The similarity of queries q_i and q_j is computed as:

$$\mathrm{sim}(q_i, q_j) = |q_i \cap q_j| / (|q_i| + |q_j| - |q_i \cap q_j|),$$

where $|q_i|$ is the number of terms in query q_i, and $|q_i \cap q_j|$ is the number of common terms between q_i and q_j.

The self-similarity score reflects how consistent a user's interests are, i.e., a higher score means many similar queries are issued by the user, and a lower score means queries are diverse. We find out that frequently failed users tend to have higher similarity scores than our normal users; see Table 2.3.

Similarly, we computed similarity scores between query pairs from different sessions, i.e., cross-session similarity score, and have found exactly the same tendency; see Table 2.4.

Table 2.3: Self-similarity score of users.

	min	**max**	**mean**	**median**
Frequently failed users	0	1	0.290	0.218
Normal users	0	1	0.280	0.212

Table 2.4: Cross-session similarity score of users with at least 3 sessions.

	min	**max**	**mean**	**median**
Frequently failed users	0	1	0.140	0.079
Normal users	0	1	0.132	0.077

Next, we relate the appearance of null queries to the user's characteristics in Figure 2.2, in particular, to the user's self-similarity score, cross-session similarity score, the number of queries, and the number of clicks. We find a medium correlation ($r = 0.46$) between the frequency of null queries and a user's self-similarity score, and a weak correlation ($r = 0.27$) for cross-session similarity score. This suggests that null queries tend to happen to users who have more consistent search interests.

There is a very weak correlation ($r = -0.10$) between the number of queries issued and the null query frequency, meaning that frequent users do not necessarily become

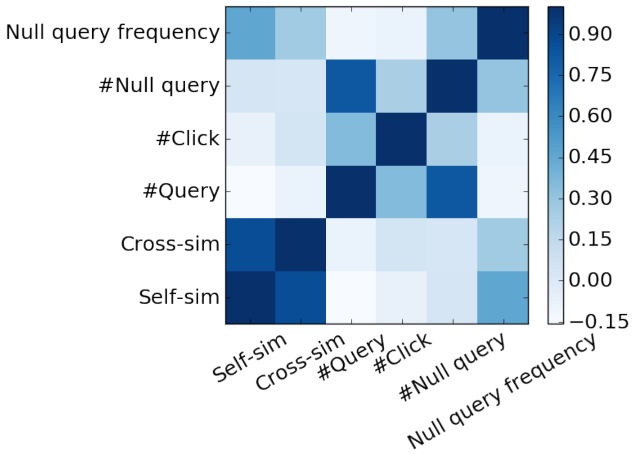


Figure 2.2: Pearson correlation matrix between users' self-similarity score, cross-session similarity score, the number of queries, clicks and null queries, and null query frequency. Correlations with cross-session similarity score are calculated for users with at least 3 sessions.

prone to failures. As to clicks, the correlation with the number of null queries is weak ($r = 0.23$).

2.3.4 Upshot

Our analysis so far yields some suggestions for how to address null queries. For instance, the entity-richness of academic search queries (i.e., the high percentage of queries that contain an entity) might help circumvent the sparsity problem in null queries. And the different null session types might help to tune query suggestions based on session type. Users' entity preferences also provide a hint for personalization.

2.4 Query suggestions for null sessions

In this section we address the task of providing query suggestions for null queries. We combine this task with findings from the log analysis in previous sections and consider two variations: a query suggestion method that uses information about entities found in queries and one that uses information in sessions.

2.4.1 Evaluation and data

In the context of query suggestions, evaluation is aimed at measuring how well the provided suggestion is able to help users continue the session upon entering a failed query. Therefore, it is preferable to use an evaluation scheme that considers the actual user behavior in search sessions, over schemes that do not model user behavior [112]. We adopt a method that is similar to one used in [214], but with some variations to suit our setup. We consider sessions that contain a cascade of queries q_1, q_2, ..., q_n, where q_1 is the first null query. If there is no click after q_1 to q_{n-1}, and there is at least one click after q_n, then we treat q_n as a successful query suggestion. We evaluate whether our query suggestion algorithm can predict q_n, given the null query q_1. In this way, we are simulating the real use case that a query suggestion helps a user upon entering a null query.

Since there is only one relevant query in the evaluation (the one that leads to a click in the log), it is proper to use Success Rate (SR) as a metric [214]. Specifically, given a query q, a ranked list of query suggestions $S(q)$, and the successful query q_n, the success rate is 1 if $q_n \in S(q)$, and 0 otherwise. We use cutoffs at 1, 3, 5, 10 because usually only a limited number of query suggestions can be displayed in a search engine's user interface. We report the mean SR scores averaged over all test cases.

We select test cases from the last 5 days in the log: sessions are chosen that contain a null query and a successful query with a click, both of which have appeared earlier in the log. We do not recommend queries that we have never seen before, instead we look at existent queries that are in the log. Moreover, we focus on sessions that contain non-boolean informational queries, for two reasons: (1) Item search queries' failures are often due to un-indexed items in the database; (2) Boolean queries (with logic operators in uppercase) have been discussed earlier [111] (a simple method to address boolean queries is to relax the logic relations, e.g., changing "AND" to "OR" may surface more search results). In total we have 310 sessions for evaluation.

We test for statistical significance of observed differences in SR using a two-tailed paired t-test and denote significant differences using $^\blacktriangle$ for significance at $\alpha = .01$, or $^\vartriangle$ for $\alpha = .05$.

2.4.2 Query-conditional suggestions

In Section 2.2.1 we mentioned the frequent occurrence of entities in academic queries. Entities contain important information on query intent, which can be utilized for query suggestion. To this end we take inspiration from the *query-entity graph* (QEG) for query suggestions in search sessions. We refer to our proposed method as the *query-conditional* suggestion method, as it bases its suggestions on the query itself and the entities in it.

The original QEG method was used for recommending queries for web pages [28], in which a set of query suggestions are produced given the current web page. There is a very important difference between the original setup and the modification that we consider for academic search: the input for recommendation. In the original setup, the input consists of a web page, but in our case it consists of a query. Importantly, contrary to the web page recommendation setting where a web page contains many seed entities, a query in a search session contains far fewer. To fit the characteristics of search sessions, we tailor the graph structure and the random walk on the QEG, and introduce the *mQEG*: the modified query entity graph model.

The modified query entity graph model is a triple $G_{qe} = (V, E, w)$ that satisfies the following conditions:

1. V, the set of nodes, consists of all unique queries in the query log plus all entities identified in the queries;

2. $E = V \times V$, the set of directed edges, is the union of E_{QQ} (query to query edges), E_{QE} (query to entity edges), and E_{EE} (entity to entity edges).
   (a) Here, E_{QQ} follows the definition in the QFG (Query Flow Graph) [23]. Each edge E_{ij} in E_{QQ} corresponds to a query transition from q_i to q_j in the query log.
   (b) Concerning E_{QE} we deviate from the definitions in [28]. E_{QE} consists of edges connecting queries and the entities extracted from them, which are bidirectional, unlike [28]. Making edges in E_{QE} bidirectional enables the random walker to visit query nodes and entity nodes alternatively, which helps to expand the limited number of entities in a query by using behavioral information (query transitions).
   (c) E_{EE} is defined by connecting entities in q_i to those in q_j if there is a query transition from q_i to q_j in E_{QQ}, the same as in [28].
3. w is the weighting function that assigns a weight to each edge in E_{ij} representing the likelihood of transitions.
   (a) For transitions in E_{QQ} we use the same weighting function as in the QFG [23]:
   $$w(q_i \rightarrow q_j) = |q_i \rightarrow q_j| / |q_i|,$$
   where $|q_i \rightarrow q_j|$ is the number of occurrences of a query transition from q_i to q_j, and $|q_i|$ the total number of query transitions that start from q_i.
   (b) For entity to query transitions in E_{QE} the weight is:
   $$w(e \rightarrow q) = |q| / \sum |e \rightarrow q_i|,$$
   where $|q|$ denotes the frequency of query q that contains e and $\sum |e \rightarrow q_i|$ the number of occurrences of all queries that contain entity e. Since edges in E_{QE} are made bidirectional, unlike [28], we define the weight of query to entity transitions in E_{QE} as:
   $$w(q \rightarrow e) = |q \rightarrow e| / \sum |q \rightarrow e_i|,$$
   where $|q \rightarrow e|$ is the number of mentions of entity e in query q, and $\sum |q \rightarrow e_i|$ is the total number of entity mentions in q. We set the probability of walking away from the query subgraph to the entity subgraph and vice versa as 0.5 by weight normalization.
   (c) For edges in E_{EE}, we define the weight to be proportional to the frequency of the entity transitions:
   $$w(e_i \rightarrow e_j) = |e_i \rightarrow e_j| / |e_i|,$$
   where $|e_i \rightarrow e_j|$ denotes the number of occurrences of an entity transition from e_i to e_j, and $|e_i|$ the number of occurrences of all entity transitions starting from e_i.

We provide an illustration of the mQEG with the QFG [23] and QEG [28] in Figure 2.3, for the query "radiation hazard" that needs query suggestions. The difference between Figure 2.3a and Figure 2.3c shows that the entities in the mQEG have enriched the semantic connections among queries compared to QFG [23]. Furthermore, the mQEG offers stronger connectivity via the bidirectional edges between entities and queries compared to the QEG [28]. For instance, given the query “radiation hazard,” the mQEG is able to provide the query suggestion “nuclear radiation hazard” while the QFG [23] and QEG [28] cannot as the required edges that link these queries are simply missing.

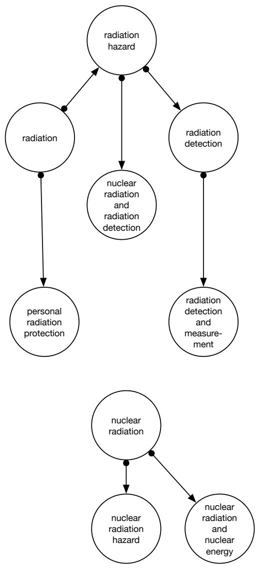
To construct the graph model, we use the entire query log except the last 5 days, which are used as our test set. In total there are 14,774,893 nodes and 100,679,495 edges in the mQEG. We use the Graphchi framework [123] to implement Personalized PageRank. We run 10 iterations for each round of personalized PageRank to achieve an approximation of the power iteration method. The number of walks is set to 100 to achieve a reasonable trade-off between speed and scale, while preventing the inclusion of queries that are only remotely relevant into the candidate sets. Given the input query, we run the algorithm and rank the query suggestion candidates by visiting frequencies in descending order.

To examine the utility of this query-conditional query suggestion method, we compare the performance of the mQEG against the query flow graph (QFG) [23], which is based on a similar graph model but does not consider the queries’ entities. We also compare the mQEG against QEG [28], although the QEG [28] is designed for a different task: recommending queries for web pages. We follow the original steps in [28] by first performing entity set expansion on the entity subgraph in the QEG, which only consists of entity nodes and their links, to expand the entities in a query. And then we perform Personalized PageRank on the full QEG by using a uniform distribution over the expanded entities. We rank the query nodes by their visiting frequencies in descending order. The results (as a percentage) are presented in Table 2.5.

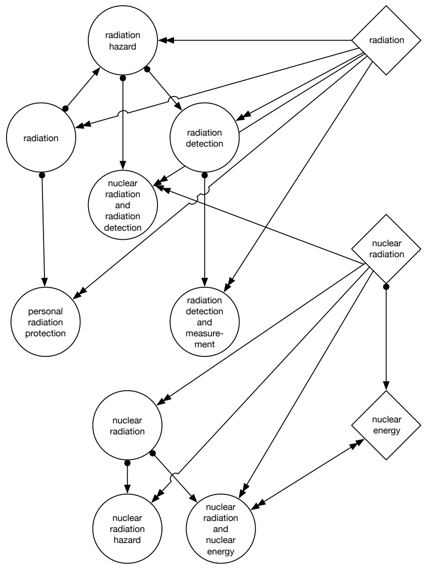
Table 2.5: Automatic evaluation results for the query-conditional method (mQEG) vs. QFG and QEG. (Success rate as a percentage.)

Model	**SR@1**	**SR@3**	**SR@5**	**SR@10**
QFG [23]	0	0	0.32	0.65
QEG [28]	2.90	2.90	2.90	3.23
mQEG	▲3.22	▲3.55	▲4.19	▲6.45

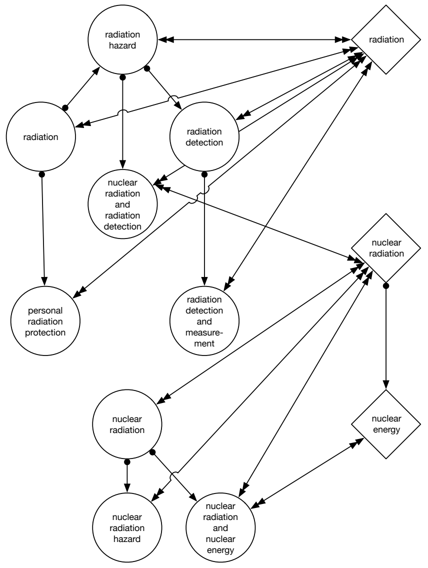
In absolute terms, the SR scores achieved by the mQEG are low because it is difficult to predict the exact same query with a click. Nevertheless, for SR@10 the scores achieved by the mQEG are comparable to the state-of-the-art in a different but related task (recommending orthogonal queries [214]). In Table 2.5, a significant improvement ($\alpha = .01$) in SR is observed in the query-conditional method (mQEG) over the QFG and the QEG. The low scores of the QFG demonstrate its weakness when dealing with rare queries. The performance increase from both models that use entities (mQEG and QEG) shows that utilizing the queries’ entities helps surface more relevant query suggestions. Additionally, it helps tackle the query sparsity problem in academic search (Section 2.2.1), by using entities to relate queries that are not connected in the QFG. The performance increase of the mQEG over the QEG [28] shows that the mQEG is more suitable for this task setting. This may be explained from two perspectives: (1) The bidirectional query-entity edges in the mQEG enhance connectivity and more


radiation hazard
radiation
radiation detection
nuclear radiation and radiation detection
personal radiation protection
radiation detection and measure-ment
nuclear radiation
nuclear radiation hazard
nuclear radiation and nuclear energy

(a) Query Flow Graph [23].


radiation hazard
radiation
radiation
radiation detection
nuclear radiation and radiation detection
nuclear radiation
personal radiation protection
radiation detection and measure-ment
nuclear radiation
nuclear energy
nuclear radiation hazard
nuclear radiation and nuclear energy

(b) Query Entity Graph [28].


radiation hazard
radiation
radiation
radiation detection
nuclear radiation and radiation detection
nuclear radiation
personal radiation protection
radiation detection and measure-ment
nuclear radiation
nuclear energy
nuclear radiation hazard
nuclear radiation and nuclear energy

(c) Modified Query Entity Graph.

Figure 2.3: Illustration of a Query Flow Graph [23], a Query Entity Graph [28], and a Modified Query Entity Graph, all for the query "radiation hazard." Query nodes are represented by circles and entity nodes by squares.

effectively deal with the sparsity of queries. (2) The QEG [28] is designed to make recommendations for web pages that usually have more entities than a query; after performing the entity expansion step, the QEG [28] uses a uniform distribution over the expanded entities to produce query suggestions, which might bring undesirable topic drift to the original query and entities. Since the mQEG and QEG are both entity-based models and the former performs better in this task setting, we will only show the results of the mQEG in the remaining sections. The increase in SR@10 demonstrates the utility of the models as only a limited number of query suggestions can be shown to users.

While the mQEG achieves a significant improvement over the QFG and QEG, there is still considerable room for improvement, especially at SR@1 and SR@3. Since the mQEG only uses query information, we hypothesize that using additional session information may improve the suggestions. Consider the example in Section 2.3.2, where a user is interested in the topic "composite beams" and is exploring. In this case, given that the session is an exploring session, it would be useful to directly provide exploratory suggestions around the entity "composite beams" upon observing a null query, such as the spelling-corrected query "a study on elastic plastic… of composite beams," or "Inelastic analysis of steel frames with composite beams" which is an even better query suggestion because it leads to a click. We hypothesize that predicting session types and biasing suggestions based on those predictions helps surface better query suggestions. We address the issue in the following section, with so-called session-conditional query suggestions.

2.4.3 Session-conditional query suggestions

The assumption underlying our session-conditional method is that we should give certain types of suggestion candidates a higher ranking in the suggestion list, depending on the session type. E.g., for a session that is likely to be exploring, exploratory query suggestions should appear higher up in the ranking. Therefore, we first need to predict the null session type; based on the predicted null session type, we then rerank the suggestion candidates generated by the graph models (QFG, mQEG).

Predicting null session type

We use the annotated sessions described in Section 2.3.2 and assign labels to sessions by considering the majority votes by the annotators as the training set. Recall that each session can be assigned multiple labels; hence, this prediction problem is cast as a multi-label classification problem. As we saw in Section 2.3.2, there is an imbalance of session types in the training set; therefore, the trained classifier is also likely to be biased towards prevalent session types. However, given that the annotated sessions come from random sampling, the distribution of session types is expected to be similar between the training set and the test set.

Our model for predicting null session types uses three feature sets: query features, query transition features, and click features; see Table 2.6.

Query features These features describe basic characteristics of a query, such as length and dwell time.

Table 2.6: Features for prediction of null session type.

Name	**Description**
Query features	
wLength	Average query length in number of words
cLength	Average query length in number of chars.
numPlainSearch	Number of non-boolean informational queries
perPlainSearch	Percentage of non-boolean informational queries
numFail	Number of null search queries
numQuery	Number of queries
dwellTimeQuery	Average dwell time for each query
Query transition features	
numTrans	Number of transitions
avgQuerySim	Average query similarity
addTerms	Number of adding transitions
delTerms	Number of deleting transitions
subTerms	Number of substituting transitions
perAdd	Percentage of adding transitions
perDel	Percentage of deleting transitions
perSub	Percentage of substituting transitions
Click features	
numClick	Number of clicks
dwellTimeClick	Average dwell time for each click

Query transition features These features describe how users reformulate queries during the session. Typical reformulations are adding terms, deleting terms and substituting terms. These transition patterns are likely to help to distinguish types of null sessions. E.g., in exploring sessions there are many substituting transitions, in which the user searches for related entities around a pivot entity. In refining sessions, the user may attempt adding and substituting terms to refine a search goal.

It is important to inspect query similarity within a session as this tells us how diverse the queries are, and also reflects the user's interest changes. Therefore, we inspect the query similarity in a session using the same pairwise similarity metric as we used in Section 2.3.3.

Click features Click features are important feedback from users. They show the satisfaction of a user for a query, as well as the search intents. We look at the number of clicks and the mean dwell time for each click.

Predicting null session type is a multi-label classification problem where the input features and the labels may have very high degrees of dependency. To capture the dependencies, we use a 3-layer deep belief network with Restricted Boltzmann Machines (RBM) [178]. We apply RBMs to learn a compact representation of the underlying patterns of the input features as well as the labels. The final hidden layer contains the output units for each label.

We use 10-fold cross validation and test this method against other methods such as classifier chains [177], binary relevance with SVM and random forests [213]. We only report on the RBM-based method as its performance is better. See Table 2.7.

We choose several metrics to measure the prediction performance:

Hamming score: For each prediction of a session, let T be the true set of labels and S be the predicted set of labels, the hamming score is defined as:

$$\mathrm{H}amming\, score = |T \cap S|/|T \cup S|,$$

The Hamming score is then averaged over all predictions.

Exact match: is the percentage of sessions that have all their labels predicted correctly.

Macro-F1 is the harmonic mean between precision and recall, first averaged per label and then across all labels.

Micro-F1 is the harmonic mean between micro-precision (precision averaged over all the predictions) and micro-recall (recall averaged over all the predictions).

The Hamming score, i.e., the accuracy over all labels in this multi-label setting, reaches a value of 0.891, which indicates a good overall prediction result. If we split out the accuracy per label (in the bottom half of Table 2.7), we see that exploring sessions have the highest accuracy score of 0.974, while refining sessions have the lowest score of 0.768. These differences among the session types shows that certain types are easier to identify automatically.

Table 2.7: Performance of predicting null session type using a Restricted Boltzman Machine.

Performance over all types of null session	
Hamming score	0.891
Exact match score	0.591
F1 micro avg	0.744
F1 macro avg	0.698
Accuracy per null session type	
Refining sessions	0.768
Generalizing sessions	0.892
Exploring sessions	0.974
Expanding sessions	0.923

How well does the prediction of null session type work at different stages in sessions, without knowing later session information? Table 2.8 shows that the performance of predicting the null session type following the first null query achieves a fair performance. Prediction after the next query after initial failure performs worse. This may indicate that the user's initial response upon a failure may be vague at first, which makes it difficult to assign a session type. Then, using two queries after the first failure achieves better performance. Our model for predicting null session type is capable of achieving good performance even with partial information.

Table 2.8: Performance of predicting null session type at different stages.

Time of prediction	**Hamming score**	**F1 micro avg.**	**F1 macro avg.**
After initial null query	0.816	0.578	0.494
. . . 1st subsequent query	0.686	0.427	0.418
. . . 2nd subsequent query	0.885	0.731	0.679
. . . 3rd subsequent query	0.887	0.736	0.684
At session end	0.891	0.744	0.698

Query suggestions

We use our null session type classifier to predict the types of null sessions. Then, the probability of a null session type will be used for reranking suggestion candidates. Here, we proceed as follows.

Given the null query, we generate suggestion candidates from one of the graph models (QFG, mQEG). Then the candidates are classified by an unsupervised multi-label classifier as refining, generalizing, exploring and expanding suggestions. The classification is a simple rule-based approach, defined by syntactic variance and term changes: specifically, generalizing suggestions contain a subset of the terms in the original query. Expanding suggestions contain new terms that are added to the original query. For exploring suggestions, there is term substitution in the original query while at least one entity term remains. For refining queries, we use a character-level edit-distance metric and classify all queries below a distance threshold θ as refining suggestions.

Algorithm 1 details how the ingredients are combined to produce session-conditional query suggestions. At line 1 we obtain query suggestion candidates produced by a baseline method (QFG or mQEG). At line 2 the candidates are divided into different types. In lines 3–7 the candidates are reranked based on the predicted session type probabilities, to form the final suggestion list.
The only variable for tuning is the distance threshold θ; we iterated over possible values and obtained the optimal performance of query suggestion results at $\theta = 0.2$.

Before discussing the experimental results, let us consider an example of Algorithm 1 in action. Consider the following session:

```
04Mar2015:06:49:58.489  Query  supply chain risk management
04Mar2015:06:58:13.402  Query  risk management
04Mar2015:06:58:48.198  Click  shorturl=/scie...pii/S026323730900005X
04Mar2015:06:59:27.762  Click  shorturl=/scie...pii/026323739190056V
04Mar2015:07:00:34.362  Query  AHP TOPSIS
04Mar2015:07:00:43.393  Query  AHP
04Mar2015:07:00:59.431  Click  shorturl=/scie...pii/S0263237312001107
```

First, given the null query "AHP TOPSIS", the mQEG generates a list of query suggestion candidates at line 1. Then the candidates are classified into refining, generalizing, exploring and expanding suggestions at line 2. Using the session information from the input, the session type predictor determines that the current session has a very

Algorithm 1 Session conditional query suggestions

```
Input:
    Session $s$; Null query $q$; $\theta$
    The probability of session type $P(i \mid s)$ for the $i$-th type of session, where
    $i = 1, 2, 3, 4$ which correspond to refining, generalizing, exploring and expanding
    sessions.
Output:
    Fused list of suggestions $R$;
1: Generate query suggestion candidate list $L$ by one of the graphical models (QFG,
   mQEG)
2: Classify suggestions $L$ into sublists $L_i$ of different types, with $\theta$ being the distance
   threshold
3: while $R$ is not full do
4:     select $L_i$ probabilistically according to $P(i \mid s)$
5:     select the top-most unchosen query $q$ on $L_i$
6:     if $q \notin R$ then
7:         append $q$ to $R$
```

high probability of being a generalizing session, as the user has dropped terms "supply chain" in the first query reformulation. Therefore, the algorithm pushes the suggestion candidates that belong to the generalizing type higher up in the ranking among all the candidates at line 3 to line 7, of which the query "AHP" is benefited, and that is the successful query that leads to a click.

Next, we report on the query suggestion results for our session conditional methods in Table 2.9. An increase in SR is observed for both models (QFG and mQEG) after applying our session conditional approach. On top of the mQEG, the session conditional extension leads to significant improvements in SR@3 and SR@5 ($\alpha = .05$).

Table 2.9: Baseline query suggestion methods (QFG, mQEG) vs. session conditional versions of the baseline methods. Prediction of null session type is after the initial null query.

Model	**SR@1**	**SR@3**	**SR@5**	**SR@10**
QFG	0	0	0.32	0.65
SC-QFG	0.65	0.97	0.97	1.61
mQEG	3.23	3.55	4.19	6.45
SC-mQEG	4.52	$^{\triangle}$7.10	$^{\triangle}$7.42	8.39

When we look at the successful recommendations made by the SC-mQEG and mQEG, we find that the SC-mQEG gives a better ranking than the mQEG for the successful query suggestions for 48% of the cases, a draw for 28%, and a lower ranking for 24%.

For the cases where the SC-mQEG and mQEG draw, the successful suggestions rank at the first place for 87.5% of the cases, and rank at the second place for 12.5% of the cases, making it difficult for the SC-mQEG to make further improvements over the mQEG.

For the cases where the SC-mQEG is outperformed by the mQEG, we find that for 57% of the cases the successful suggestions are exploratory ones, but the session prediction's output sees the session as refining or generalizing. For 29% of the cases the session prediction is expanding, which is correct, but query suggestions that belong to other types have been pushed up in the reranking process due to the randomness, thus lowering the rank of the desired query. In the remaining 14% of the cases it should be a refining suggestion but the session prediction is expanding.

For the cases where the SC-mQEG outperforms the mQEG, the improvements come from the session type predictions that are most likely to be refining and generalizing, constituting 29% and 71% of the cases, respectively. Nevertheless, the "less confident" session predictions for exploring and expanding types may still contribute: in the test cases for which the SC-mQEG outperforms the mQEG, the exploratory query suggestions account for 7% of them and expanding suggestions account for 14%, although the most likely session type prediction for these sessions is neither exploring or expanding but refining. This shows that it is not always the most likely prediction that works, instead it can also be a less-likely session type prediction that contributes to the successful query suggestion.

From the analyses we infer that the session type prediction's most confident prediction may not always be correct; however, even a sub-optimal prediction may help push the desired query higher up in reranking. Overall, our findings confirm that in most cases, predicting session type helps make equal or better query suggestions.

2.5 Discussion: user-conditional query suggestions

We have used both queries (or rather: entities in queries) and sessions to improve query suggestions for null queries. It seems natural to consider using user-specific information, i.e., personalization, for query suggestion. However, when we examined the cross-session similarity for each user that has at least 3 sessions, as shown in Table 2.4. The average cross-session similarity score is very low, which indicates shifting interests across sessions, and this, in turn, suggests that the benefit of personalization may be limited. Below, we report on experiments aimed at determining the benefit of personalized query suggestions.

We focus on users' preferences over entities for two reasons: (1) The majority of queries contain at least an entity; (2) Entities reflect the users' topic interests. We aim to see if they help to improve the quality of query suggestions. To this end, we create a personalized query-entity graph (PmQEG) by integrating user preferences into the edge weights (transition probabilities) of the graph.

2.5.1 Inferring user preference

We derive each user's preference for entities by looking at query reformulations. These reformulations fall into two broad categories: query reformulations that have at least a common entity in the queries and those that do not, from which we can infer conditional and unconditional entity preferences, respectively:

1. For the conditional type, we look at three common cases of query reformulation:
   (a) deletion;
   (b) addition;
   (c) substitution.

   We cannot infer a clear user preference based on deletion or addition, as in these cases users try to generalize or refine a query. But in the case of a substitution, we are able to infer a conditional preference. Specifically, assume that two consecutive queries q_i and q_j contain entities E_i and E_j, respectively, and shared entities $E_c = E_i \cap E_j$. Then, the user prefers the new entity set $E_j \setminus E_c$ over the previous entity $E_i \setminus E_c$ given the shared entity set E_c. Put more formally: $P(E_j \setminus E_c \mid E_c) \succ P(E_i \setminus E_c \mid E_c)$.
2. For the second type, where there is no common entity in two consecutive queries, it is possible to infer an unconditional preference. If a user issues consecutive queries q_i and q_j, which contain entity set E_i and E_j respectively, we infer that they have a preference for E_j over E_i ($P(E_j) \succ P(E_i)$).

All inferences of entity preference inherently consider a user's interest shifts over time: since queries arrive sequentially, this pattern favors new entity preferences over old ones.

2.5.2 Preference model

For each user u, we derive preference information from the query log. Let s denote the shared set of entities, which could be $\emptyset$ (in such cases the preference is not conditional), and p and c are two entity sets between which the user prefers p. Then the training data D is made of all preference pairs: $D = \{(u, s, p, c) \mid (p \mid s) \succ (c \mid s)\}$. We use MAP (maximum a posteriori probability) estimation to derive the preferences over entities. For unobserved preferences of entities, we use Laplace smoothing to assign a non-zero probability: $P = (x + \alpha)/(N + \alpha \times \epsilon_{ij})$, where ϵ_{ij} is the number of entity transitions given a source entity e_i, N is the number of observations for e_i and x is the number of the observed entity preference; $x = 0$ for unobserved preferences.

When personalization is applied to all users, an increase of SR at different cutoffs is observed; see Table 2.10. Specifically, the PmQEG has improvements of SR at all cutoffs, and SC-PmQEG improves the SR@3 and SR@5 scores. The improvements show that personalization is better able to put high quality query suggestions into higher rankings, but the differences are not statistically significant.

Table 2.10: Personalized models (PmQEG, SC-PmQEG) vs. Non-personalized models (mQEG, SC-mQEG).

Model	**SR@1**	**SR@3**	**SR@5**	**SR@10**
mQEG	3.23	3.55	4.19	6.45
PmQEG	3.55	4.84	4.84	6.77
SC-mQEG	4.52	7.10	7.42	8.39
SC-PmQEG	4.52	7.42	7.74	8.39

The consistency of a user's interests does not necessarily impact the performance of personalization. For some cases personalization brings performance gains, because users issue queries with exactly the same interest in the history (e.g., a user has a historical preference for “management,” which leads to the query suggestion “customer relationship management” that is desired). But in other cases personalization may not produce much utility for users who have consistent interests in a certain topic, but are constantly exploring around subtopics within it. When a new subtopic comes up, the historical preferences of previous subtopics might lead query suggestions astray and hurt personalization.

In this section we have discussed personalization for users’ entity preferences, as entities are prevalent in academic queries. Looking further, personalization can be achieved by considering topical interests, user behavior, session information, and also specific search patterns such as refinding [208]. It is yet to be seen whether general personalization methods in web search would work in academic search. After all, personalization should be a discreet decision to make for the search engine in order not to hurt user experience. We put the problem of an in-depth analysis of personalization as future work.

2.6 Related work

We consider four areas of related work: academic search, search failure, query suggestions and query auto-completion.

2.6.1 Academic search

Academic search concerns the retrieval and profiling of information objects (papers, journals, authors, reviewers, ...) in the domain of academic research. The first academic search engine MEDLINE came operational in 1971, which supported up to 25 users simultaneously [144]. However, its use was limited to libraries and only pre-programmed search tasks were supported instead of online queries. It was not until the 1990s when the World Wide Web became popular, that online academic search engines started to thrive and became accessible to a larger user base. These online academic search systems include Citeseer [69] and Aminer [206], which focus on citation indexing and metadata extraction as well as academic social network extraction, respectively. Multiple heuristics such as term frequency and citation scores can also be applied to increase the performance of academic search engines [9]. This chapter studies one of the world’s most popular academic search engines, ScienceDirect [186], which is widely used in the physical sciences, engineering and life sciences.

Several transaction log analyses have been conducted on search engines of digital libraries. However they are either focusing on a single discipline [83, 102], or limited in scale [107], thus making them not representative of academic search. Moreover, these analyses focus on revealing basic statistics, and little insight on user behavior in search sessions is given.

2.6.2 Search failure

Lancaster [124] uncovers failure phenomena in an early academic retrieval system MEDLARS in the 1960s. Dwyer et al. [60] examine failures of interlibrary loan-request forms for items in two university libraries from 1989 to 1990. They find 17 types of error such as “in circulation” and “incorrect citation.” However, the “queries” in those obsolete systems are hand-crafted and static requests mostly from librarians, which are completely different from the modern form of queries that we type in the search box. The “failures” in the past are therefore different from what modern users encounter in online and interactive academic search engines.

Singh et al. [194] study search trails of various eBay users and the impact of null queries on purchase rate. They observe a degradation of purchase rate for null search trails compared to the non-null search trails. They find that the purchase rate for both power users as well as novices is lowered when null recall situations are encountered on their trails. They also find the repetition factor to be as low as 1.45 for null queries versus 19.57 for non-null queries. A low repetition factor makes it difficult to use query log-based signals to improve the performance of null queries.

2.6.3 Query suggestions

Query suggestion is a feature in modern web search that improves the search experience by providing recommendations of queries. Most query suggestion techniques exploit a query log in order to give useful suggestions. Zhang and Nasraoui [231] use a similarity measure by exploiting consecutive queries during sessions combined with a content-based method using search frequency and query frequency. Boldi et al. [23] introduce the query-flow graph by examining different reformulation patterns in search sessions, and uses random walk on the graph to obtain suggestions. Guo et al. [80] further use social annotation data to construct a query-URL-tag tripartite graph and use random walks to recommend queries in a structured way. Bordino et al. [27] project the query-flow graph to a lower-dimension space to measure the similarity between queries for diverse query recommendations. Guo et al. [81] use clicks and snippets to identify search intents and provide query recommendations under different intents. Song et al. [197] propose a term-transition graph for query suggestions, using information from queries and documents.

These techniques, however, perform well for queries that come with clicks and other user feedback information. The long tail of the queries is more challenging for query suggestion, since such queries are rare and very little user feedback is available for them. Bonchi et al. [25] use a term-query graph and provide query suggestions at a term level by computing the center-piece subgraph of the terms in queries. Vahabi et al. [214] propose orthogonal query recommendation, which suggests queries that are syntactically different but semantically similar, to address the situation when the original query is ill-posed.

Among the queries in the long tail, there is a specific kind of query that are difficult to deal with: the ones that return very few or no results. In a study by Altingovde et al. [8], these queries are characterized and most of these “failed” queries are found to contain a URI. Apart from the malformed URI queries, one third of them still contain a regular URI that the search engine can not retrieve. In these cases it is difficult to provide query suggestions as the search engine can not understand user intents.

There is an increasing volume of research on providing suggestions around entities. The use of entities utilizes auxiliary resources by linking entities in queries to knowledge bases such as Freebase and Wikipedia. Bordino et al. [28] extend the query-flow graph by introducing entities in queries, and uses personalized PageRank to give query suggestions given a visited page. Hassan Awadallah et al. [85] deal with sessions with complex search tasks. They tag entities in query text and group queries into tasks for recommendation.

2.6.4 Query auto-completion

Query auto-completion (QAC) is a popular function in search engines to help users formulate queries given the prefixes that the user is typing [31]. Contrary to the post-remedy role of query suggestions, QAC has the potential to prevent null queries's appearances. Mitra et al. [154] studies how users engage in QAC and finds that they tend to use it on word boundaries, which is helpful for difficult words. Shokouhi [191] studies personalized QAC by considering a user's previous queries. Cai et al. [32] augment the personalized model by considering time-sensitivity. Recently, Zhang et al. [230] propose an adaptive model that uses implicit negative feedback during user-QAC interaction. Mitra and Craswell [153] design a QAC system for rare query prefixes using a latent semantic model.

Our work differs from previous work on academic search, by studying user behavior at a fine-grained level through a large-scale transaction log analysis. It also differs from algorithmic work on query suggestion by taking into account the unique characteristics of academic search, namely entity queries and, especially, null session types. This work is complementary to query auto-completion, as a post-remedy measure as opposed to being a precaution method.

2.7 Conclusions

In this chapter we have investigated search behavior and failures in academic search. First we have identified the unique features of academic search, some of which provide observational insights for algorithmic improvements, e.g., richness of entity queries, verbosity of queries and unique search session types.

Then we have pointed out the problem of null queries, and motivate the use of query suggestions to address null queries. We generate query suggestion candidates using graph models that utilize entities in queries. Given the various session types, we propose a session-conditional approach. We have subsequently trained a multi-label classifier to predict the type of session in which a null query occurs. We then base the query suggestions on the predicted types of the null sessions. We find that the improvement of the session-conditional method is significant. Furthermore, we investigated personalization and have achieved a slight improvement.

The theoretical implication of this research lies in two main aspects in the query suggestion method. First the query-conditional approach, which effectively uses entities to surface more relevant query suggestions, further proves the importance of utilizing entities in information retrieval tasks. Moreover, the session-conditional approach shows that session information will help reranking query suggestions for null queries. This approach also does not rely much on characteristics of academic search data, which makes it possible to be generalized to web search and other domains, but of course it definitely requires further investigation. The practical implication of this research is the observation of user behavior in a modern academic search engine. The basic characteristics of search queries, their differences with web search, and the failure phenomenon all help to draw the big picture of academic search, and will draw more attention to research in this domain.

As to limitations of our study, we acknowledge that there are many options of personalization techniques for query suggestions, and the one adopted by the authors is only one of them that is motivated by the characteristic of academic queries (entity richness). It is certainly possible to examine whether general personalization methods in web search will achieve comparable performance in academic search, which we leave as future work. It should also be noted that the appearances of null queries may be related to certain search engine design techniques, e.g. whether to use query auto completion, query expansion and semantic matching. Yet it is meaningful to study how users would react upon null queries, especially in our setting where it is prevalent, so that we can deepen our understanding of how to alleviate this problem. In addition, some of those null queries are actually "positive failures" for searchers doing provenance finding: no result for a query confirms the non-existence of novel research ideas. Therefore it is interesting to combine research on null queries with provenance finding so that the search engine can judge whether a null query makes sense, and only provides query suggestions when necessary.

As to future work, we recommend improved profiling of searchers on multiple dimensions, e.g., preferences at the topic level, modeling intent shifts and it would be meaningful to examine when and whether to personalize.

3 Topic Shift and Query Reformulation Patterns in Academic Search

In the previous chapter we have examined the academic search queries and failures in the context of search sessions. Moving on, we examine users' long term behavior that extends beyond sessions. Specifically, we investigate users' query reformulation and topic shift, and reveal their correlations, by answering **RQ2**.

3.1 Introduction

Academic search deals with the retrieval of information resources in the domain of scientific literature. Hemminger et al. [88] point out that academic search engines have become the primary portal for researchers to gain information; see also [165]. In recent years, there have been several publications focused on academic search and academic searchers. However, most are very limited in scale, and rarely reveal insights into the search behavior of academic searchers based on the analysis of large-scale transaction logs [83, 102, 107]. In this study we take a look at academic search through a large-scale log analysis from a major academic search engine.

Academic searchers do have a distinct search pattern that is different from the typical web searchers. For instance, in web search, the search activity becomes the least intensive on Fridays and peaks in the weekends [16]. But, as shown in Fig. 3.1, academic search activity peaks during weekdays, and drops in the weekends.

To study the behavior of academic searchers, we investigate two key aspects: *query reformulations* and *topic shifts*. Both have received much attention in user behavior studies of web search [24, 96, 133], but to the best of our knowledge, there is no previous work on revealing the query reformulation behavior and topic shifts of academic searchers that is based on a large-scale log analysis. In fact, very little is known about these two aspects of academic search.

Through this study, we provide answers to three specific research questions:

RQ2.1 What is the query reformulation behavior of academic searchers?
RQ2.2 Do academic searchers have shifts in topical interests over time?
RQ2.3 Is there a correlation between query reformulation behavior and topic shift?

This chapter was published as [135].

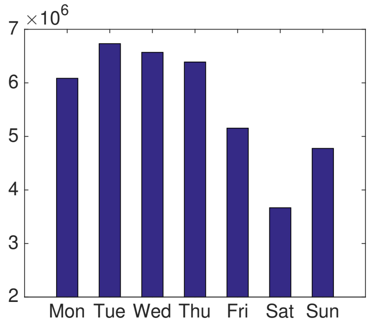


Figure 3.1: Average number of queries per weekday in academic search (based on the dataset described in Section 3.3).

For the first question, we look at query reformulation behavior over time. Query reformulation happens after the user has examined the search engine result page and provides a more explicit type of feedback than clicks, which are implicit and noisy [44]. We look at five frequent types of query reformulation: *revisiting a previous query*, *adding terms*, *dropping terms*, *substituting part of the query*, and *issuing a completely new query*. We study how the type of reformulation behavior changes over time and find that revisiting and issuing new queries tend to happen more often as search goes on.

For the second question, we take a quantitative approach to study topic shift over time. We train an LDA model [22] on all long sessions in the query log that we examine. We segment a user's queries into different timespans, and treat queries in each timespan as a bag of words. We infer a topic vector for each timespan of the user. Topic shift between successive timespans is then calculated using the Euclidean distance between the topic vectors. In this process we identify two types of user: one type increasingly focuses on topics over time and the other diversifies over time.

Finally, we conduct a correlation study to see how these two aspects—query reformulation and topical shift—are correlated with each other. We find that user's query reformulation patterns have little correlation with the tendency of topic shift, meaning that users with distinct reformulation preferences in search could be equally likely to be diversifying or focusing on topics. We also find that certain reformulations (viz. *adding terms* and *issuing new queries*) may help predict the magnitude of the next topic shift.

Contrary to previous work that studies academic searchers through surveys and user studies, this chapter sheds light on the reformulation behavior and topical shifts of academic searchers through a large-scale log analysis. The insights gained help us to understand academic searchers' information seeking patterns from a much larger user base, and may be useful for personalization in academic search.

In Section 3.2 we discuss related work. In Section 3.3 we introduce the dataset characteristics. In Section 3.4 we describe our approach to study query reformulations and topic shifts. In Section 3.5 we show the result and analysis from the correlation studies. We present our conclusions in Section 3.6.

3.2 Related work

3.2.1 Academic search

Academic search involves the indexing and retrieval of information objects (papers, journals, authors, ...) in the domain of academic research. The earliest academic search engine MEDLINE, which began functioning in 1971, allowed a maximum of 25 simultaneous users [144]. It was restricted to library usage and only pre-programmed searches were supported instead of online queries.

When the web became popular in the 1990s, online academic search engines started to flourish and gained popularity. Typical examples are Citeseer [69] and Aminer [206], which focus on metadata retrieval and academic network extraction respectively. There are several surveys and user studies on the search behavior of researchers on modern academic search engines [165, 170, 171], which are based on a relatively small sample of researchers. The few log analyses conducted on search engines of digital libraries are either investigating a single discipline [83, 102], or limited in scale [107], as a result of which they are not representative of academic search. Moreover, they focus on basic usage statistics and lack insights on user behavior in search sessions. Recently, Li et al. [138] studied the user behavior and query failure phenomenon in academic search through a large-scale transaction log analysis.

3.2.2 Query reformulations

Query reformulation is an important aspect of user behavior during search sessions. In recent years, there has been a range of studies that cover patterns and models of query reformulation [24, 30, 96, 126, 181, 190], how they work in a collaborative setting [156], in voice search [98] or in mobile search [192], and their applications [27, 31, 99, 175]. These studies show that query reformulations are the key to understanding user behavior, which will benefit retrieval tasks such as query auto completion [99] as well as topic and intent finding in users' queries [175], and which may help improve retrieval performance [79]. The findings are mostly in the domain of web search and the query reformulation behavior studied is that of the general web users.

Multiple category schemes have been used for query reformulation in the literature [24, 30, 96, 126, 181, 190]. Different category schemes may correspond to (1) search engines of different designs (e.g., whether searches on multiple verticals are supported), (2) whether using search assistance is considered as a reformulation such as query suggestion, or (3) different granularities of query reformulations. Manual categorization may provide fine-grained results [30, 126, 181] but can not easily scale up to large query logs. On the other hand, rule-based [96, 190] or learning-based [24] methods can be applied to a large query log, and are thus more suitable for analyzing long term query reformulations from a large user base.

3.2.3 Topic shift in queries

There has been a whole line of research that investigates topic mining in web search query logs [4, 92, 93, 101], where the emphasis is on how to segment and cluster queries by topic. However, the multi-tasking nature of web searchers, which means searching and switching between multiple topics within and across sessions [149], makes it cumbersome to derive useful insights from users' topic shifts, especially over long periods.

This paper differs from previous work in academic search, by studying a large transaction log from a major academic search engine, with a focus on user behavior in search sessions. The findings are therefore better able to represent academic searchers, compared with earlier small-scale user studies and surveys. It also differs from previous work in query reformulations in web search, by revealing the academic searchers' preferences instead of those of the general web users. The paper differs from work on topic shifts in web search by looking at a different domain: academic search. Compared to the web searchers who have diverse, parallel, and fast-shifting topic interests, academic searchers are more likely to have consistent interests in a general topic. For instance, a researcher in information retrieval is more likely to stay in this general topic than diverting to biology sciences. This makes studying the long term topic shift pattern meaningful. Moreover, this study tries to link query reformulation to topic shift, and provides useful insights into their connections through a series of correlation studies.

3.3 Data

We study a query log from the ScienceDirect search engine,[1] containing over 39 million queries. The query log is collected from September 28, 2014 to March 5, 2015. Table 3.1 shows the length statistics of the query log. Two thirds of the traffic come from institution-authorized access, meaning that users in a certain IP range can access the search engine, and they share the same session ID and user ID in the query log. Besides, many institutions use proxies or firewalls so that their IP is recorded instead of the terminal device. Therefore it is not possible to differentiate these IP-users. We are only confident in an ID-user one-to-one mapping when they log in or access the search engine from outside the institution. And we study these "non-IP" users only, who contribute about one third of the traffic.

Table 3.1: Query length statistics in word count.

Category	#N	min	max	mean	median
Sciencedirect	39M	1	419	3.77	3

With a timeout of thirty minutes as a threshold, there are a total of 4,307,889 sessions for these non-IP users, and 2,833,549 of them contain at least 3 queries which we denote as "long sessions." To obtain enough data of users, we confine the scope of users to those who have a minimum of 30 queries, and whose search behavior lasts over 30 days at least. This leaves us with 29,093 users and 1,918,334 query records.

[1]`http://sciencedirect.com`

3.4 Approach

In this section we describe how we study the behavior and topic change of academic searchers in a series of correlation studies.

First, we highlight the statistics of the prominent types of query reformulations from the query log. Then, we apply a time sequence-based method to make observations of how users progress in search. We break each user's queries into sequences and then align them, so that we can compare how users progress during search even if they start at a different time. Specifically, we put each user's queries into bins separated by a certain length of timespan (to be specified below). Then, we align all searchers' queries by timespan, with the first timespan of a user denoted as 0, the second as 1, in a natural number sequence. We can observe query reformulation and topic shift of users as they move from one timespan to the next. In this case, to gain enough samples from the dataset and also to ensure statistical significance in our later correlation analyses, we sample timespans of 3, 7 and 14 days long. We choose timespans of different lengths to observe whether some changes are more prominent over longer timespans. The length of timespans chosen also corresponds with the usual information seeking cycles of academic searchers, as research suggests that information-seeking happens toward a weekly basis rather than daily basis for faculty and graduate students [39, 165]. Note that users may issue no query in a certain timespan; in such cases the timespan will be neglected for that user.

Query reformulation tendency over time. To uncover the reformulation preferences for the academic searchers as a whole, we examine the query reformulation preference over time for all academic searchers combined. For each timespan, we aggregate the frequency of each reformulation from all users and obtain the proportion of each reformulation type. We hypothesize that certain reformulations might happen more frequently as time goes on, for instance revisiting, because academic searchers tend to have a consistent interest in their field of study [97] and may thus need to submit a previous query repeatedly in search of new information. We try to determine if there is indeed a linear correlation of the proportion of an action over the course of time (represented as a natural number sequence of timespans). To this end, we use Pearson's correlation.

It is common for users to use a combination of the query reformulations listed in the previous section (revisiting, adding a term, dropping a term, substituting a term, new query) in order to reach their search goal. In our analysis, we calculate the proportion of each query reformulation in each time span for every user.

Topic shift. We study the tendency of a user to shift topic over time with a quantitative approach as we aim to measure the magnitude of change in topic. We train an LDA model on long sessions that contain at least 3 queries. Each session is treated as a "document" in training because the queries within a single session mostly likely belong to the same general topic. The number of topics is set to 150, which is a reasonable value in the academic domain [78] and also ensures relatively fast convergence in Gibbs sampling. For each user, we model the queries in each timespan as a bag of words and use the trained LDA model to infer a topic vector. Then, for a given user the magnitude of topic shift between adjacent timespans is calculated using the Euclidean distance between the user's topic vectors for the two timespans.

Correlations. After studying how users' reformulation behavior and topical interest change over time, respectively, we aim to find whether there is a correlation between a user's query reformulation patterns and their topic shift tendency. Specifically, we look at two aspects of the correlations. First, the macroscopic aspect, i.e., whether a user's topic shift tendency is correlated with query reformulation preferences. For instance, suppose a user favors a specific type of reformulation, say substitution; is this user likely to be diversifying in topic shifts? Second, there is the microscopic aspect: in successive timespans, is the proportion of each reformulation type in the first timespan correlated with the topic change that happens during the next timespan? Based on the correlation findings, we consider the task of predicting the magnitude of a user's topic shift during the next timespan.

3.5 Results and analysis

In this section we present the results of our analysis of users' query reformulations and topic shifts. We first analyze these two aspects separately and then perform a series of correlation studies to examine their connections.

3.5.1 Query reformulation types

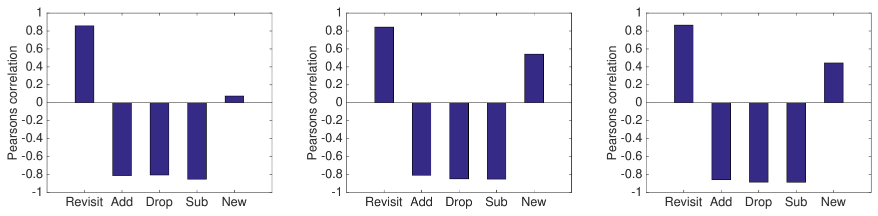
To study users' query reformulation types, we apply a syntactic-based automatic categorization. Our taxonomy does not require human annotations and does not have the fine-granularity of those methods in [24, 96, 190]. However it is fully unsupervised and is scalable to a large query log; it contains five reformulation types that are common to the majority of taxonomies previously used for query reformulations [24, 30, 96, 126, 181, 190]. The main difference is that none of these previous publications considers "revisiting queries" as a reformulation while we do (Bruza and Dennis [30] consider "repeated query" but there is no user identifier in their query log).

Revisiting Revisiting is issuing a query that is already in the user's search history [208]. In academic search, we find that this reformulation type is very prominent, making up 33.8% of all reformulations, which shows that academic searchers tend to have some consistent search intents and will seek information on the same topic repeatedly.

Adding terms This type of reformulation is characterized by adding at least one term to the previous query, and corresponds to the process of refining search. This is typically seen in sessions where users start with a general query on a certain topic, then add terms to examine sub-aspects within the topic [181]. This reformulation type constitutes 8.5% of all reformulations.

Dropping terms This is the opposite process of the adding reformulation type, constituting 5.6% of all reformulations. By dropping at least one term from the previous query, the user aims to retrieve information that is more general than the previous query [181]. This may happen when academic searchers need context information during learning.

Substituting terms Substitution of terms is the second most prominent reformulation type that accounts for 28.0% of all reformulations. Substitution means keeping



(a) Timespan = 3 days. (b) Timespan = 7 days. (c) Timespan = 14 days.

Figure 3.2: The query reformulation preference over time for all the academic searchers, measured in correlation of the proportion of the reformulation actions (revisiting, adding terms, dropping terms, substitution and new query) over time.

certain at least one term in the original query intact, then dropping old terms and adding new terms. Substitution behavior may happen when a user is refining a search, e.g., changing a synonym, or when the user is exploring different aspects about a certain topic [181].

New query This reformulation concerns the situation where the user issues a query that has no overlap of words with the previous query and that does not appear in the user's search history. Submitting a new query that is different often means a change of search intent [24]. It happens when other reformulations will not address the new intent of the users. New queries make up 24.1% of all reformulations.

Compared to web search, where substituting terms accounts for the most popular type (ranging from 22.73% to 37.5% in different datasets [24, 96]), the most prominent type in academic search is revisiting and substituting terms only comes next.

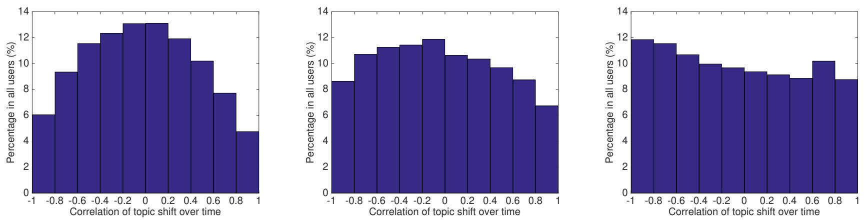
3.5.2 Query reformulation tendency for all academic searchers combined

Fig. 3.2 plots all searchers' query reformulation tendency.

By definition of the correlation strength [64], there is a "very strong" positive correlation of the proportion of revisiting behavior over time, in the analyses of all timespans. This confirms our earlier hypothesis in Section 3.4, that there is an increasing trend of revisiting queries by academic searchers, which shows their consistent interests in certain topics. Interestingly, between timespans of 3 days, the tendency to submit new queries is weak, but at longer timespans (7 or 14 days), we can observe a moderate positive correlation. This suggests that submitting new queries tends to happen not immediately (within a 3 day gap), but within a longer gap. The negative correlation for the other three reformulations (add, drop, and substitute) shows that users perform these reformulations less frequently in the later period of search.

3.5.3 Topic change tendency

Using the approach described in Section 3.4, we study the magnitude of the users' topic shift over time. The tendency is represented by the correlation strength: the larger the correlation, the bigger the topic shift over time for a user. Fig. 3.3 shows the distribution of the correlation of the users, for 3 different timespans.



(a) Timespan = 3 days (b) Timespan = 7 days. (c) Timespan = 14 days.

Figure 3.3: The correlation of user topic shift over time.

The correlation strength of topic shift over time indicates the evolution of user interests over time, namely whether they tend to become more focused or more diversified. In general, we find that nearly half the users tend to have increasing topic shifts over time (diversifying), and the other half have decreasing shifts (focusing). For different timespans, we see from the shape of the distribution, that there are more users showing a stronger tendency of topic shift (either positive or negative) as the timespan increases. This indicates that bigger topic shifts tend to happen when the time gap between searches is longer.

3.5.4 Correlation between reformulation behavior and topic shift

There are users who become more focused over time and those who do not. Correspondingly, we group users by their tendency to shift topics, and study if this tendency has a correlation with query reformulation patterns. Specifically, users are divided into 6 groups by the Pearson correlation strength r of the topic shift tendency over time: moderately diversifying ($0.4 \leq r < 0.6$), strongly diversifying ($0.6 \leq r < 0.8$), very strongly diversifying ($0.8 \leq r \leq 1.0$) and moderately focused ($-0.6 \leq r < -0.4$), strongly focused ($-0.8 \leq r < -0.6$), very strongly focused ($-1.0 \leq r < -0.8$). Then we look at the correlation with the user's different reformulation type's proportions, as shown in Fig. 3.4.

Fig. 3.4 shows that we cannot differentiate diversifying or focused users, purely based on their query reformulation patterns. That is, the user's preference of choosing certain query reformulations is not correlated with their topic shift tendency. This is an interesting finding as it shows that even users with distinct query reformulation preferences, could be equally likely to be focusing or diversifying in search.

Taking a step back, although we cannot determine whether a user is focusing or diversifying based on preference of reformulations, can we predict the magnitude of topic shift to happen in the next timespan given only the user's current reformulation behavior? To answer this question we first examine the individual correlation between the proportion of each reformulation type at a given timespan, with the topic change that happens at the next timespan. See Table 3.2.

Individually, for the majority of users there is only a weak correlation between a query reformulation type and the next topic shift. For users who show a strong

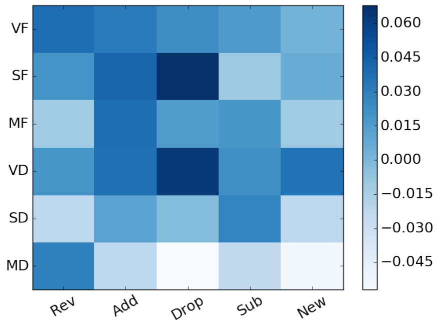


Figure 3.4: The correlation of the topic shift tendency (MD: moderately diversifying, SD: strongly diversifying, VD: very strongly diversifying, MF: moderately focused, SF: strongly focused, VF: very strongly focused), with the proportion of the reformulation actions (revisiting, adding terms, dropping terms, substitution and new query) for each user.

correlation ($-1.00 \leq r < -0.60$ or $0.60 \leq r \leq 1.00$), submitting new queries contributes the least to a decrease in topic shift magnitude and also the most to an increase in topic shift magnitude, respectively, compared with other reformulation types. For longer timespans, there are more users who exhibit a strong correlation. Especially when the timespan is 14 days, 21.0% of the users show a strong or very strong correlation between adding terms and topic change, and the number is even higher at 23.7% for submitting new queries. Interestingly, substituting reformulations tend to correlate the least with topic change. This suggests that users tend to stay in the same general topic, or a subtopic within the general topic, while modifying only part of the original queries.

3.5.5 Predicting the magnitude of the next topic shift

Next, we try to utilize the observational insights that we have just gained for a prediction task: can query reformulation signals help to predict the magnitude of a user's topic shift?

More precisely, we use features from users' reformulations to predict the magnitude of topic shift at the next timespan. The features are the proportions and number of occurrences of query reformulations in a timespan. We cast this task as a regression task. Our training set is comprised of pairs of query reformulations and the topic shift to happen at the next timespan for all users. The test set consists of the second-last query reformulations and the next (final) topic shift for each user.

We use linear regression and three evaluation measures: correlation coefficient, mean absolute error (MAE) and root mean squared error (RMSE). The prediction results are listed in Table 3.4. Prediction is more accurate on shorter timespans, with the 3 day predictions reaching a medium correlation ($r = 0.4530$), while 14 day predictions being at only $r = 0.3225$. The performance difference indicates that topic shift magnitude in a shorter timespan is easier to predict than longer timespans.

Table 3.2: Correlation of reformulation behavior with topic shift at the next timespan. Each column shows the distribution of users (in percentage) who have different correlation strengths between a reformulation type and topic shift, in an interval of 0.2.

Correlation	**Revisit**	**Add**	**Drop**	**Sub**	**New**
Timespan = 3 days					
$[-1.00, -0.80]$	1.6%	1.4%	2.4%	3.7%	0.9%
$[-0.80, -0.60]$	3.7%	2.9%	4.3%	8.1%	2.3%
$[-0.60, -0.40]$	6.9%	6.4%	7.7%	12.9%	4.5%
$[-0.40, -0.20]$	12.7%	11.5%	11.7%	16.9%	7.7%
$[-0.20, 0.00]$	16.5%	15.5%	16.1%	17.2%	11.4%
$[0.00, +0.20]$	17.2%	17.5%	17.6%	14.2%	15.7%
$[+0.20, +0.40]$	15.8%	16.1%	15.1%	11.8%	19.3%
$[+0.40, +0.60]$	13.1%	14.5%	12.9%	7.6%	18.9%
$[+0.60, +0.80]$	8.4%	9.5%	8.0%	4.9%	13.5%
$[+0.80, +1.00]$	4.1%	4.6%	4.2%	2.9%	6.0%
Timespan = 7 days					
$[-1.00, -0.80]$	3.1%	2.4%	3.7%	5.9%	1.7%
$[-0.80, -0.60]$	5.3%	4.5%	6.1%	10.1%	3.4%
$[-0.60, -0.40]$	8.7%	7.4%	8.4%	13.1%	5.7%
$[-0.40, -0.20]$	11.7%	11.7%	11.2%	14.3%	8.4%
$[-0.20, 0.00]$	13.4%	13.9%	13.4%	14.4%	10.9%
$[0.00, +0.20]$	14.7%	14.4%	14.4%	12.7%	13.8%
$[+0.20, +0.40]$	13.7%	13.8%	14.2%	10.6%	16.4%
$[+0.40, +0.60]$	13.1%	14.2%	12.8%	8.6%	17.4%
$[+0.60, +0.80]$	9.9%	10.9%	9.3%	6.4%	13.7%
$[+0.80, +1.00]$	6.3%	6.9%	6.4%	4.0%	8.6%
Timespan = 14 days					
$[-1.00, -0.80]$	4.8%	3.9%	5.5%	8.0%	3.0%
$[-0.80, -0.60]$	7.0%	6.4%	7.1%	11.6%	5.1%
$[-0.60, -0.40]$	8.6%	8.2%	8.9%	12.6%	7.3%
$[-0.40, -0.20]$	11.1%	10.7%	11.1%	12.2%	9.0%
$[-0.20, 0.00]$	11.6%	12.3%	10.9%	11.9%	10.6%
$[0.00, +0.20]$	12.7%	12.3%	11.4%	11.0%	12.1%
$[+0.20, +0.40]$	12.7%	12.5%	12.8%	10.6%	14.5%
$[+0.40, +0.60]$	12.0%	12.8%	12.6%	8.5%	14.7%
$[+0.60, +0.80]$	10.9%	12.2%	11.1%	7.5%	13.5%
$[+0.80, +1.00]$	8.6%	8.8%	8.6%	6.0%	10.2%

3.6 Conclusion

In this study we have examined users' query reformulation behavior and their tendency of topic shift in academic search through a large-scale log analysis. We have found

Table 3.3: Query reformulation features for prediction of the magnitude of topic shift at the next timespan.

Name	**Description**
Reformulation proportions	
Revisiting_Percentage	Percentage of revisiting reformulations
Adding_Percentage	Percentage of adding term reformulations
Dropping_Percentage	Percentage of dropping term reformulations
Substitution_Percentage	Percentage of substituion reformulations
New_Query_Percentage	Percentage of new query reformulations
Reformulation occurrence numbers	
Revisiting_Number	Number of revisiting reformulations
Adding_Number	Number of adding reformulations
Dropping_Number	Number of dropping term reformulations
Substitution_Number	Number of substituion reformulations
New_Query_Number	Number of new query reformulations

Table 3.4: Linear regression results (correlation coefficient, mean absolute error, root mean squared error) for predicting the magnitude of a topic shift in the next timespan given query reformulation features in the current timespan.

	3 days	**7 days**	**14 days**
Correlation Coefficient	0.4530	0.3906	0.3225
MAE	0.0697	0.0755	0.0805
RMSE	0.0931	0.0999	0.1057

that over time, academic searchers as a whole tend to conduct revisiting, as well as submitting completely new queries. This pattern corresponds to the academic searcher's information needs: either seeking previous search results or new results on the same search intents, or simply pursuing new search intents. We have identified two types of topic shift patterns in users, namely the focusing type and the diversifying type.

Through a series of correlation studies, we have found that a user's preference for certain query reformulations does not correlate to their topic shift tendency. Nevertheless, users's current reformulation patterns (adding terms, submitting new queries) may help to predict the magnitude of topic change in the immediate next timespan. We further used features from query reformulations for predicting the magnitude of the next topic shift. The findings of the query reformulation behavior, topic shift type, and their connections help to improve our understanding of the behavior of academic searchers from a large user base. They may provide hints for personalized search, such as whether to provide exploratory or focusing type of search results, and recommendations of queries or papers for users.

In future work we intend to look at query reformulation patterns in the context of different search tasks, e.g., a navigational task for a single document, or a learning task for a certain research topic. And we will examine the utility of using query reformulation features to improve retrieval performance and provide better recommendations in academic search.

4 Characterizing and Predicting Downloads in Academic Search

In the previous chapters we have mainly studied user behavior based on their queries. We have revealed the query failure phenomenon, proposed query suggestion methods to remedy that, and examined the query reformulation and topic shift. In this chapter we investigate a different but also popular behavior–downloads, a conversion behavior on academic search, and answer **RQ3**.

4.1 Introduction

Conversions are critical to websites as they are directly related to revenue, and can indicate the performance of the platform. Research into conversions has received considerable attention in areas such as online shopping and hotel booking [see, e.g., 68, 115, 128], where a conversion is a purchase or booking action. These studies reveal the purchase and booking behavior characteristics of users and provide insights on predicting conversions in those domains.

Academic search concerns the retrieval of information objects in the domain of academic research (papers, journals, authors, etc). Research on conversions has received little attention in the domain of academic search. Conversion in this setting refers to the download action of a paper, which happens when the user finds relevant information and wants to save it for later use. Similar to purchases, downloads of papers generate revenue for the academic search platform, often through a subscription service or pay-per-download. Therefore, it is valuable to study the download behavior of users and understand users' behavioral patterns. E.g., what actions do users perform that lead to a download? Is there a temporal pattern in downloads? Are there behavioral differences among users with different topical interests?

To start, we conduct an observational study to characterize user download behavior in academic search. To the best of our knowledge, this is the first characterization of its kind in the area of academic search. While some of the findings may coincide with the personal experiences of readers of this chapter, many findings provide insights that can only be found from a large user base.

This chapter was published as [136].

Using the insights obtained in this manner, we try to generate predictions of users' download behavior. We are motivated by the information overload problem in online recommendations. Numerous studies have shown that information overload has a negative impact on user reactions [7, 75, 131]. It has been observed in large-scale experiments that showing an excessive number of paper recommendations may not bring benefits to the click through rate, but instead bring harm [15]. In our setting of predicting download behavior, one task is to predict how many downloads the user is going to have next.[1] An application scenario for this prediction task is when the system sends out recommendations in the form of news letters, such as the recommender system on Mendeley.[2] If the system is able to predict the magnitude of the user information need, it can better tailor the length of the list of recommendations, hence avoiding information overload and leading to a better user experience. Moreover, time is an important source of information for understanding user satisfaction [29]: finding the right timing for a recommendation may also improve the performance of recommender systems [50]. Correspondingly, we address the task of predicting the time gap until the next download, aiming to make the system more preemptive and send recommendations when users are in need.

In this chapter, we provide answers to the following research questions:

RQ3.1 What are the user actions that lead to a download in academic search?

RQ3.2 What are behavioral patterns and topical aspects of user download behavior across sessions?

RQ3.3 How do we predict user download behavior?

Our main contributions are:

1. We introduce a new dataset for downloads in academic search and characterize user interactions with academic search engines.
2. We study the users' actions across sessions, revealing correlations among various behavioral signals and explaining the topical aspects of user downloads.
3. We build a specialized model for download prediction that utilizes user session history and that is based on user segmentation, which leads to significant improvements over a state-of-the-art baseline.

4.2 Related work

Related work comes in several kinds: academic search, academic paper recommendation, and online shopping prediction.

4.2.1 Academic search

Academic search concerns the task of indexing and retrieval of entities (papers, journals, ...) in the domain of academic research. Academic search services are commonly provided by academic search engines, such as Google Scholar, Microsoft Academic

[1]Note that we do not consider the "zero download" scenario in our setting, which would be formulated as a different problem: churn prediction. Our focus is on users who regularly use the academic search service.

[2]Mendeley (`https://www.mendeley.com/`) provides personalized paper recommendations through news letters, based on users' interactions with the system.

Search [195], AMiner [206], and CiteSeerX [132]. Several studies have indicated that academic search engines are an essential portal for obtaining research information [88, 164, 165]. Mitra and Awekar [152] study the search results of several academic search engines and find that they have low overlap. Other research concerning user behavior in academic search occurs mostly via surveys [170, 171] or small-scale log analyses on high-level statistics (page views, access frequencies etc.) [107]. They are either restricted to a small group of participants, or to users from a single discipline, which renders the findings less generalizable. Recently, Xiong et al. [225] have proposed to use entity embeddings to improve relevance ranking of papers in academic search, which leads to better performance on a test set of 100 queries from their transaction log.

There are very few studies on academic search that are based on a large-scale transaction log. In recent work, Li et al. [138] and Li and de Rijke [135] study the phenomena of null queries and topic shifts in academic search, respectively, based on large-scale log analyses. Khabsa et al. [108] study the distribution of academic search queries on Microsoft Academic Search and build a classifier for different query types. To the best of our knowledge, no large-scale study has been conducted on download behavior in academic search.

4.2.2 Academic paper recommendation

Paper recommender systems provide users with relevant paper suggestions, preferably personalized to their own interests. Gori and Pucci [74] use random walks on citation graphs to make paper recommendations. Li et al. [141] propose to recommend papers using matrix factorization combined with topic modeling. They find that topic representations for users can help distinguish users with different interests, and surface better suggestions. Nishioka and Scherp [163] use social media streams to profile users, and recommend papers based on the profiles. Sun et al. [203] study research networking sites and leverage social network connections for paper recommendations. Beierle et al. [15] demonstrate how recommendation overload affects click through rate. Through 3.4 million delivered recommendations, they find lower click-through rates for higher numbers of recommendations; users can feel "overloaded" rather quickly.

4.2.3 Online shopping prediction

We introduce related work on online shoppers on e-commerce sites because online purchases share important commonalities with academic downloads: (1) they both represent a conversion after user interactions with the system; (2) both scenarios come with a "budget." Money is the budget factor in online shopping, while in academic search it could be money (subscription service or pay-per-download) and time (assuming that users are aware of the finite amount of time they have to read the chapters). Lee et al. [128] examine the purchase behavior and trajectory of users, and use behavioral features for purchase predictions of items. Kooti et al. [115] extract user purchase histories from emails to analyze their purchase behavior. They find that previous purchase history information helps to predict time and price of the next purchase. Kooti et al. [116] study online shopping behavior in app stores. They discover that 1% of the users account for 59% of the total spending in app purchases, and that they behave very differently from a random user in shopping. For these 1% users, Kooti et al. [116] propose a supervised model to generate shopping predictions. Yeo et al. [227] study purchase prediction for retargeting, by using purchase features extracted from users' browsing history.

In summary, this chapter differs from previous work in academic search because it studies download behavior as opposed to other user behavior. Compared to other high-level log analyses, this chapter provides insights into user actions within sessions and across sessions. It is also based on a large transaction log of a popular academic search engine rather than small-scale user studies or surveys, hence bringing findings that are more generalizable. This work is directly related to paper recommender systems. Whatever the implementations of a paper recommender system may be, they all need to consider information overload and recommendation timing. Therefore, they can benefit both from our characterization of download behavior, and predictions of the download number and time gap.

4.3 User download patterns

In this section we present observations of user download actions in academic search. The definition of a download here is the act of requesting a PDF file for a paper.[3] We study search sessions that include at least one download action. The search sessions are characterized by entering a query as the first interaction, and they end with a cutoff time of 30 minutes inactivity that is commonly used in web session analyses [38, 200].

We first introduce the dataset and the various actions that users perform within a session. We analyze the action statistics as well as the action trajectories that lead to a download action. Then we uncover download patterns across sessions.

4.3.1 Dataset and user action definitions

Dataset

To study users' download behavior, we use a transaction log provided by ScienceDirect,[4] which offers academic search services and primarily covers the domains of health science, life science, physical science and social science. Collected between September 28, 2014 and March 5, 2015, the log contains more than 39 million queries via institution-authorized access as well as personal access. The former access type refers to users in a certain IP range (e.g., from a research institution), who are referred to as *IP-users*. The latter refers to users who log in or access the search engine from outside the institution, i.e., so-called *non-IP users*. Two thirds of the query traffic comes from IP-users.

For the purpose of studying and predicting user downloads, we filter the logs based on two rules: (1) users are uniquely identifiable, so that we can distinguish them from each other, and (2) users are active in terms of issuing queries and requesting downloads, in order to guarantee enough observations for our study. IP-users from the same institution may end up having the same user ID or session ID. Therefore, we look

[3]The dataset used in our study also includes download actions of less importance, such as downloading references, that we include in our study without focusing on.

[4]`http://www.sciencedirect.com/`

at non-IP users only to ensure that each user ID maps to a unique user. The majority of these non-IP users have access via subscription, and the rest through pay-per-download. We then select active users that have a minimum of 30 queries in a timespan of 30 days, and a total of at least 20 download sessions in the period covered by the log. To prevent the inclusion of bots/crawlers, we also remove overly frequent users that have more than 1,000 queries recorded in the log or with more than 100 clicks/downloads on average per day, which account for fewer than 0.1% of the users. We end up with 1,089 users and 30,988 sessions that include at least one download action, referred to as *download sessions*. There are a total of 206,830 download actions, i.e., an average of 190 downloads per user. The above data selection process provides us with enough observations per user to study their download behavior.

User actions

After a user issues a query, a list of papers is shown on the search engine result page (SERP). Then the user can take several subsequent actions: (1) click on a paper title for detailed information, which opens up a new window, (2) directly click to download a paper by requesting a PDF file, or (3) click for other information such as a paper abstract. We summarize the actions of interest in Table 4.1.

Table 4.1: Possible user actions.

	Action	**Explanation**
1.	Query	user issues a query
2.	Download PDF	user requests a PDF version of a paper
3.	Change query source	user changes the source of a previous query, i.e., selecting different sources or subjects for the query.
4.	HTML click	user clicks on a paper on the SERP (search engine result page) for detailed information, which opens a new window
5.	Abrf click	similar to "HTML click" except that the clicked result does not contain full text
6.	Abstract click	user click to see the abstract of a paper on the SERP
7.	Reference download	user downloads the reference of a result

"Abrf click" (5) is an action similar to "HTML click" (4) that leads to a paper page without full text but with a scanned image of the chapter content. "HTML click" (4) and "Abrf click" (5) are triggered when users click on a paper's title on the SERP. Users will notice the difference after the click but not beforehand. "Change query source" (3) should not be confused with query reformulation which refers to typing a new query.

While academic search engines have different user interfaces, most of them provide similar high-level functionalities as the ones we list in Table 4.1: the user actions available on ScienceDirect resemble competing services. Therefore, despite some small differences between functionalities of popular academic search engines, we believe that the insights learned through this study have generalizable implications for academic search engines as a whole.

4.3.2 Download behavior within a session

What actions do users like to perform in a session? We find that the most frequent user

Table 4.2: Statistics of user actions in a session.

		Mean	**Median**
1.	Query	2.82	2
2.	Download PDF	6.45	3
3.	Change query source	0.46	0
4.	HTML click	1.49	0
5.	Abrf click	0.30	0
6.	Abstract click	0.52	0
7.	Reference download	0.08	0
Query dwell time (s)		567	323
Click dwell time (s)		734	397
Session duration (s)		1,336	754

action is "download," followed by "query," shown in Table 4.2. It should be noted that users tend to have multiple downloads within a single session, with the median number being 3 per session. This can be explained by the richness of informational queries in academic search [138], which users issue to search for relevant information around a certain topic. Interestingly, clicks have lower occurrences than queries, and are often absent in sessions. This suggests that clicking results to view detailed information may not be necessary for users to make a download decision, while partial information (title, authors) already provides enough cues of relevance. For sessions that contain a click, users tend to spend relatively much time inspecting detailed information (e.g., glancing over the full text), with the median click dwell time being over 6 minutes.

Table 4.3 lists the frequent action trajectories toward a download in a session. The most frequent trajectory is a single query (1) leading to a download (2), making up 30.3% of all trajectories. Ranking second is a query (1) and a query reformulation (1) leading to a download (2). The trajectories involving clicks are far less frequent. These observations indicate that queries are acting as a more common signal toward downloads than clicks.

Below, we give an example of a download session sampled from the log to illustrate the process:

```
28Nov2014:16:22:13  Query (1)           dynamic friendship network
28Nov2014:16:23:40  Query (1)           dynamic friendship network model
28Nov2014:16:24:34  Abrf click (5)      shorturl=/scie...pii/0378873394002467
28Nov2014:16:25:47  Download PDF (2)    shorturl=/scie...pii/0378873394002467/pdf
```

In this session, the user starts with the query "dynamic friendship network" and proceeds with the query reformulation "dynamic friendship network model." Then, the user clicks

Table 4.3: Top 10 most frequent action trajectories toward the first download in a session. Actions are numbered as in Table 4.1: 1. Query; 2. Download PDF; 3. Change query source; 4. HTML click; 5. Abrf click; 6. Abstract click; 7. Reference download.

Trajectory	**Frequency**
$1 \rightarrow 2$	30.3%
$1 \rightarrow 1 \rightarrow 2$	8.7%
$1 \rightarrow 4 \rightarrow 2$	4.4%
$1 \rightarrow 1 \rightarrow 1 \rightarrow 2$	3.7%
$1 \rightarrow 3 \rightarrow 2$	2.1%
$1 \rightarrow 1 \rightarrow 1 \rightarrow 1 \rightarrow 2$	1.8%
$1 \rightarrow 1 \rightarrow 4 \rightarrow 2$	1.3%
$1 \rightarrow 5 \rightarrow 2$	1.3%
$1 \rightarrow 1 \rightarrow 1 \rightarrow 1 \rightarrow 1 \rightarrow 2$	0.9%
$1 \rightarrow 4 \rightarrow 4 \rightarrow 2$	0.9%

on a result by performing an “Abrf click,” and after examining the result for a while chooses to download the PDF file.

4.3.3 Download behavior across sessions

Next, we go beyond individual patterns and look for temporal patterns and correlations across sessions.

Temporal patterns

Looking at download numbers on different days in a week, we observe a steady trend during weekdays, while the number declines in the weekends; see Figure 4.1. This trend is similar to the e-commerce setting where more purchases happen during weekdays than weekends [115].

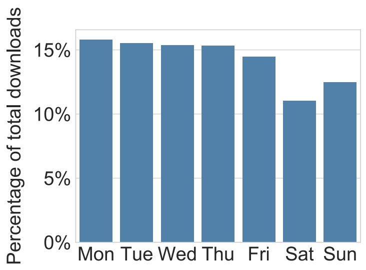


Figure 4.1: Download distribution over the week.

However, we find that academic searchers take longer to perform the next conversion action than online shoppers. This is evident from the time gap between download sessions. Figure 4.2 shows the distribution of the averaged time gap[5] for each user. It

[5]We average the time gap for each user to avoid the bias toward active users that have many sessions.

has a median of 172,915 seconds and a mean of 192,532 seconds (2.00 and 2.23 days respectively), while in online purchases the median time gap is 1 day [115].

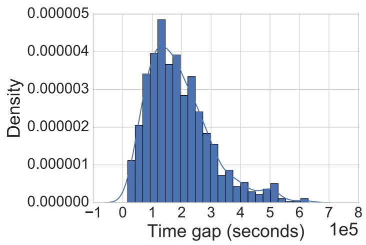


Figure 4.2: Distribution of time gap between consecutive download sessions averaged per user.

Correlations between sessions

We are interested to find out connections among sessions, that is, how one session impacts another. We aim to answer questions such as: if a user has performed many actions (e.g., downloads) in the current session, will the activity intensity sustain in the next session? And will the next download happen in a shorter time gap or a longer one?

We examine two types of correlation: (1) the correlation between the current session and time until the next download session; and (2) the correlation between the current session and the number of downloads in the next download session. We consider several factors in the current session, including user action statistics (Table 4.2) and query statistics (average word/character length of queries). Table 4.4 gives a description of the factors. The correlations are shown in Figure 4.3. In Figure 4.3, factors in the current

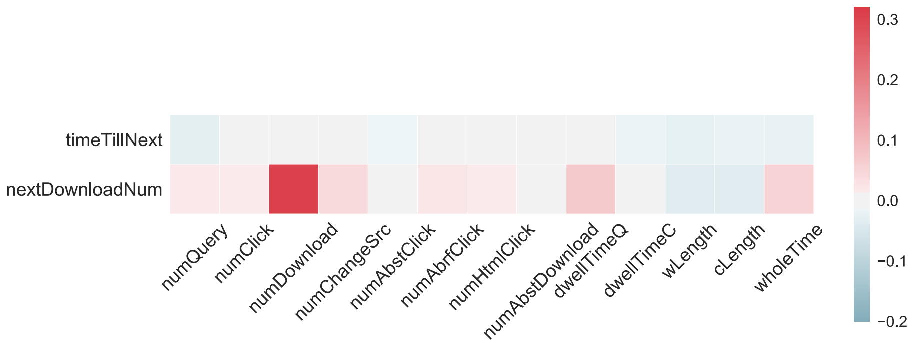


Figure 4.3: Correlations between statistics of the current download session (horizontal) and of the next download session (vertical).

session are all negatively correlated with the time until the next download [125]. Out of all the factors, the number of queries is the most negatively correlated ($p < 0.0001$, two-tailed t-test). This suggests that the more queries the user submitted in the current session, the sooner her next download session might occur.

Table 4.4: Description of factors in Figure 4.3.

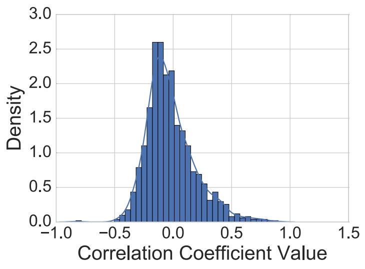
Name	Description
numQuery	Number of queries
numClick	Number of all clicks
numDownload	Number of PDF downloads
numChangeSrc	Number of change query source actions
numAbstClick	Number of abstract clicks
numAbrfClick	Number of Abrf clicks
numHtmlClick	Number of Html clicks
numAbstDownload	Number of abstract downloads
dwellTimeQ	Averaged dwell time on queries
dwellTimeC	Averaged dwell time on clicks
wLength	Average query length in number of words
cLength	Average query length in number of chars
wholeTime	Session duration
timeTillNext	Time gap until the next download session
nextDownloadNum	Number of PDF downloads in the next download session

Conversely, the number of downloads in the next session are positively correlated with most of the current session factors. The prominent factor is the current session's number of downloads, which has a medium positive correlation with that of the next session ($p < 0.0001$, two-tailed t-test). This indicates a certain degree of consistency between the number of downloads across sessions.

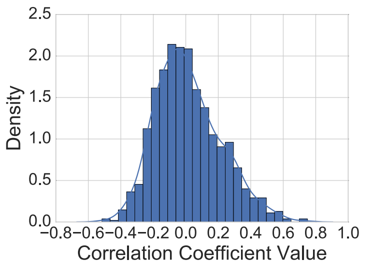
The above correlations are calculated from sessions of all users. Therefore they represent the overall trend from all observations. Next, we examine correlations at the individual level. For the two correlations that we calculate, i.e., time and the number of downloads, we examine the most negative factor "number of queries" for time until next download and the most positive factor "number of downloads" for the next number of downloads, respectively. We examine each user's sessions and obtain the two correlation coefficients. We show the distributions of the correlation values in Figure 4.4. Both correlation values are nearly normally distributed but the means differ. More than half of the users show a negative correlation between time and query in Figure 4.4a, which is in line with the overall trend in Figure 4.3. However, the distribution in Figure 4.4b indicates that nearly half of the users tend to be consistent in the number of downloads between consecutive sessions, and half do not. Bias explains why the overall correlation is positive in Figure 4.3 while the individual correlation distribution disagrees: users with positive correlations have more sessions in the log, thus affecting the overall correlation.

4.3.4 How topics impact downloads

In this section we discover the topical characteristics of the user behavior. Compared to clicks on a result page, we believe that downloads are less noisy, and are stronger signals



(a) Correlations between time until next download and the number of queries in the current session.



(b) Correlations between the number of downloads in the next session and that in the current session.

Figure 4.4: Distribution of correlation coefficients for individual users.

Table 4.5: Statistics of sessions grouped by users with different topical interests. PS: physical sciences, HS: health sciences, LS: life sciences, SS: social sciences.

	Mean				**Median**			
	PS	HS	LS	SS	PS	HS	LS	SS
Query	2.82	2.68	2.95	2.94	2	2	2	2
Click	2.08	2.27	2.06	3.09	0	0	0	1
Download	8.04	5.65	6.31	4.86	3	3	3	2
Duration (s)	1,330	1,282	1,384	1,408	745	718	806	819
Time between download sessions (s)	176,655	180,134	166,943	178,190	60,009	64,984	53,764	56,102

to reflect users' topic interests. Therefore, we represent the topics using downloads. To identify the topics of user downloads, we resort to the Scopus classification of subject areas.[6] Specifically, we look at the journal where a paper is published and use the subject area of the journal to represent the topic. The subject information of journals is manually annotated and publicly available through API access.[7] The subjects fall into 4 broad topics (health sciences, life sciences, physical sciences and social sciences) and a total of 333 specific categories. We use the subject information to represent topics because it is manually annotated and easily interpretable, and is more accurate than topics inferred by topic models such as LDA [22].

Are certain topics more popular than others? Figure 4.5 shows the distribution of the topic of each download record, and the distribution of each user's most popular topic (determined by most frequent download type). While both distributions are heavily imbalanced, physical sciences is the most popular subject and social sciences is the least. This finding is in line with the focus of ScienceDirect on natural science journals. Below, we look at users with different topical interests and examine their behavior and topical differences.

[6] `https://www.elsevier.com/solutions/scopus/content`
[7] `https://dev.elsevier.com/sc_apis.html`

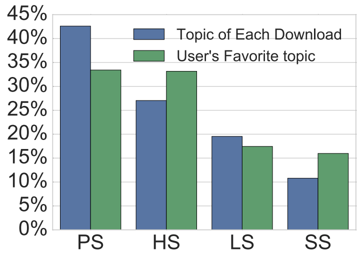


Figure 4.5: Topic distribution of each download and each user's favorite topic respectively. PS: physical sciences, HS: health sciences, LS: life sciences, SS: social sciences.

Behavioral differences

We hypothesize that distinct topical interests may come with different download patterns and examine the behavioral differences among users with different topical interests. To examine behavior, signals such as clicks, downloads, session duration and time until the next download session are investigated, as shown in Table 4.5.

Users interested in the social sciences stand out from the others: they do not perform as many downloads compared to people interested in other subjects; however, they have more clicks and spend more time in sessions. As to the time between download sessions, the health sciences have the longest time gap, while the life sciences have the shortest. And there is a 21% difference between the median values, which is 3 hours. In summary, download behavior indeed varies among user groups interested in different subjects.

Topical profiles of users

First, we examine the interdisciplinary nature of users: how many unique topics out of the 4 general topics do users cover through their downloads? We show the distribution in Figure 4.6.

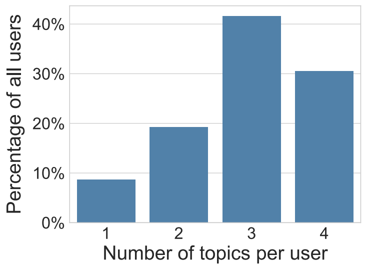


Figure 4.6: Distribution of the number of topics per user.

Most users cover more than one topic in their downloads. This is not surprising, as recent findings [172] suggest that "science is indeed becoming more interdisciplinary." Therefore, users may have information needs for multiple topics when conducting increasingly interdisciplinary research. Besides, some of the users might be research policy designers who need to study several topics.

Furthermore, we try to find out the differences between users that are interested in each of the 4 topics. We first look at the topical diversity of users, i.e., how many unique subtopics users cover through their downloads. Each subtopic corresponds to a specific subject area. Note that users may cover the subtopics of multiple domains. We find that users in the life sciences domain explore the largest number of subtopics with a mean of 24 and a median of 22, while users in the health sciences and in the social sciences explore the fewest, with mean and median values being roughly on par with each other, shown in Table 4.6. This shows that users's topical diversity differs across their disciplines.

Table 4.6: Statistics of unique subtopics covered per user.

Category	**Min**	**Max**	**Mean**	**Median**
Physical sciences users	2	208	24	19
Health sciences users	2	76	21	17
Life sciences users	2	101	24	22
Social sciences users	1	83	20	18

Next, we consider topical coherency in downloads. Consider two users who cover the same number of subtopics: one may have downloaded only a few papers but switches topic whenever possible, while the other may have downloaded many but does not switch subtopics as often. Since they cover the same number of subtopics, we would consider the second user to be topically more coherent due to a smaller number of topical switches. We design a topic coherency metric that measures the likelihood of staying in the same subtopic(s) in downloads as shown below. The higher the score, the more likely the user stays in the subtopic(s).

$$\text{Topical Coherency} = 1 - \#unique\_subtopics/\#downloads$$

Here, #unique_subtopics refers to the unique number of subtopics under one of the 4 general topics. Topic coherency is computed per individual user.

Table 4.7 indicates that users in the physical sciences have the highest topical coherency with a mean score of 0.832 and a median score of 0.846. This might seem surprising because these users exhibit a high level of diversity of downloads (in Table 4.6). However, they also commit the largest number of downloads (in Table 4.5), and they do not switch topics as often as others, and hence they obtain the highest coherency score. On the other hand, users in the life sciences have the lowest topical coherency scores while also being the most diverse in download subtopics (Table 4.6).

4.3.5 Upshot

We have introduced user actions during a download session and investigated their frequencies, trajectories, cross-session correlations, and topical impacts on paper downloads. We have found that certain signals are indicative of downloads, and users with

Table 4.7: Topical coherency per user.

Category	**Mean**	**Median**
Physical sciences users	0.832	0.846
Health sciences users	0.826	0.833
Life sciences users	0.807	0.815
Social sciences users	0.817	0.833

different topical interests tend to behave differently. Next, we use those observations, especially behavioral and topical features, for predicting paper downloads.

4.4 Download prediction models

In this section, we describe our models for paper download prediction. As a baseline, we adopt a state-of-the-art model for predicting online shoppers' behavior [115], since this task bears resemblance to our download prediction task (as explained in Section 4.2).

Then we propose an LSTM-based model to effectively leverage users' historical interactions, as well as a specialized model based on user segmentation.

We consider two prediction tasks: given a user's previous download sessions, (1) predict the time until the next download session, and (2) predict the number of downloads in the next download session. More formally, for each user u, the training sessions ordered by occurrence are denoted as $u_d = \{s_1, s_2, \ldots, s_{n-2}\}$, where n is the total number of sessions the user has. In testing, for each user u and given a session s_{n-1} as input, our models need to predict the number of downloads in s_n and the time gap between the end of s_{n-1} and the start of s_n.

4.4.1 Baseline model

The baseline model [115] considers shopping prediction as a multi-class classification task. It uses a Bayesian network classifier and a set of features derived from the online shopping setting. The features used in [115] include demographics of online shoppers, purchase price history, purchase time history etc. Although it is not possible to directly apply those features in the academic search setting, in some cases it is possible to identify natural counterparts. Specifically, purchase time features can be mapped to download time features, and purchase price features can be mapped to features of the number of downloads. Other features such as queries and other actions (download references etc.) are exclusive to our setting. The baseline Bayesian network model does not allow arbitrary number of inputs,[8] while our LSTM-based models do and are able to take input from the full session history. To make the comparison fairer, we also include aggregated features from the full session history for our baseline. In the end, the following features are used for the baseline Bayesian network predictive model:

1. Current session action features: the number of occurrences of queries, clicks, downloads, change query source, abstract click, HTML click, Abrf click, abstract down-

[8]Users may have different numbers of sessions.

loads.
2. Current session time features: dwell time on queries, dwell time on clicks, session duration.
3. Query features: average word and character length of queries.
4. Historical session features: time gap from the last download session, average/median/standard deviation of time gap between consecutive download sessions; number of downloads in the last download session, average/median/standard deviation of the number of downloads in historical download sessions.

Formally, each session s_i is represented as a feature vector and a label: $s_i = \langle f_i, l_i \rangle$, where s_i is the i-th session of the user, f_i is the feature vector for the session, and label l_i corresponds to the label for prediction, i.e., the time gap until the next session s_{i+1} or the number of downloads in s_{i+1}. In testing, the model is given f_{n-1} to predict l_{n-1}.

4.4.2 LSTM model

The second model we consider is an LSTM (long short-term memory), a recurrent neural network model proposed by Hochreiter and Schmidhuber [89]. Through its memory cells and gate architecture (input, forget and output gates), an LSTM is able to alleviate the vanishing and exploding gradients problem that exists in simple recurrent neural networks. LSTMs are known to perform well for tasks that deal with long sequences. Motivated by the correlations of behavioral statistics across sessions (explained in Section 4.3), we use an LSTM to model the chapter download prediction problem in order to utilize the full session history for prediction. Specifically, sessions of users are modeled as sequences, each consisting of a feature vector and a label (either time or number of downloads). The LSTM model takes the sessions of each user as input and learns to predict the label.

Formally, the training and testing cases are defined similar to the baseline model's setting, except for test sessions. In testing, the model is given $\{s_1, s_2, \ldots, s_{n-2}, f_{n-1}\}$, where f_{n-1} is the feature vector for session s_{n-1}, as input in order to predict label l_{n-1}. In training, we optimize for multi-class cross entropy. We choose Stochastic Gradient Descent with Nesterov momentum as the learning algorithm, and use mini-batches. We initialize the network parameters via Xavier initialization [70], and hyper-parameters such as learning rate are tuned via grid search.

4.4.3 Specialized model based on user segmentation

In mobile shopping prediction, user segmentation has been considered [116] as some users may behave differently than others, and specialized models are built for them. In Figure 4.4 we noticed that individual users may have different download patterns, reflected by the varying correlations of download behavior across sessions. E.g., after a session with many downloads, some users tend to have fewer downloads in the next session, but some may not. In our setting, the LSTM model should in theory learn to distinguish between these different patterns of users. However, it may not work as well on time series data of unequal lengths. E.g., it is known that in terms of classification tasks for unequal-length time series, DTW (dynamic time warping) [19] outperforms LSTMs on some occasions [66, 130] due to its ability to consider warping in time series.

To improve our prediction performance, we segment users and build specialized models for them. Specifically, we segment our users into clusters by behavioral similarity measured via DTW, and then train specialized LSTM models on the user clusters. In this way we are giving special treatments to users that are similar, who share behavioral patterns.

We use DTW because it is able to effectively handle time series of different lengths, which allows for stretching or compressing sequences while comparing similarity. Specifically, DTW is set to find the minimum warping distance between two series P of length n and Q of length m:

$$\begin{aligned} \mathrm{P} &= p_1, p_2, \dots, p_n \\ \mathrm{Q} &= q_1, q_2, \dots, q_m. \end{aligned}$$

An n-by-m matrix is constructed where each element (i, j) corresponds to the squared distance between p_i and q_j. The goal is to find a path W through the matrix that minimizes the accumulated distances

$$\mathrm{DTW}\,(P, Q) = \min\left\{\sum_{i=1}^{K} w_k\right\},$$

where w_k is the k-th element on the warping path W. Then, the warping path can be solved recursively:

$$\gamma(i, j) = d(p_i, q_j) + \min\{\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)\},$$

where $d(p_i, q_j)$ is the distance between p_i and q_j, and $\gamma(i, j)$ is the cumulative distance.

To measure distances between users, we model each user's download behavior as a time series: the number of downloads and time between download sessions. Notice that in our setting users may have different numbers of sessions. We use 1-nearest neighbor DTW to obtain the distance between any pair of users, which ensures good warping accuracy. Then we apply average linkage hierarchical clustering on users based on the distances. The LSTMs are subsequently trained on clusters that represent users with similar download behavior. Each cluster contains a minimum of 10% of the total number of users.

4.5 Experiments and results

In this section we present the experiments and results of download prediction on two tasks: (1) predicting the time until the next paper download session and (2) predicting the number of paper downloads in the next download session.

4.5.1 Experimental setup

Similar to [115], we cast the download prediction task as a multi-class classification task instead of a regression problem, because it is difficult to predict the exact time. For time prediction we divide the time gaps into 5 classes. The time gaps and their distribution in the dataset are described as follows: very short (within 2 hours, 20.6%), short (2 hours to 1 day, 30.4%), median (1 day to 3 days, 18.4%), long (3 days to 7 days, 14.2%) and very long (over 7 days, 16.4%). For predicting the number of downloads, we segment the number of downloads into three classes: 1 download (30.0% of all sessions), 2–4 downloads (36.8%), and ≥ 5 downloads (33.2%).

We use the 1,089 users and 30,988 sessions described in Section 4.3.1. We segment it into training and testing data following the description in the prediction model section.[9] We test the statistical significance of observed differences in predictions using a paired Wilcoxon signed-rank test. We denote significant differences between the baseline and other methods using * for $\alpha = .05$ and ** for significance at $\alpha = .01$. We denote differences between the LSTM with user segmentation and all other methods using $^{+}$ for $\alpha = .05$ and $^{++}$ for significance at $\alpha = .01$.

4.5.2 Experimental results

Baseline

For predicting the time until the next download session, the baseline model yields a prediction accuracy of 0.347; see Table 4.8, first row. This score is comparable to that achieved by the baseline model in the online shopping time prediction task (5 class classification, 0.311 accuracy, [115]).

Table 4.8: Predicting the time until the next paper download with the baseline and LSTMs.

Model	Accuracy
Baseline	0.347
LSTM current session	0.354^{**}
LSTM current session + 1 previous session	0.354^{**}
LSTM current session + 2 previous sessions	0.354^{**}
LSTM full session	0.357^{**}
LSTM full session + user segmentation	0.371^{**++}

For predicting the number of paper downloads in the next session, the baseline achieves an accuracy of 0.441; see Table 4.9, first row.

Time series based models

Next, we present the results of the LSTM models. To determine how historical session information impacts prediction, we control the number of session inputs during testing. We hypothesize that in testing, feeding the network with the full session history will lead to better predictions than feeding only partial session information.

The results of predicting the time until the next download are shown in Table 4.8, rows 2–5. All LSTM models perform significantly better than the baseline, even when

[9] Due to the dependency in the time series, we split the data by time so that the models learn from historical information and predict the future download.

Table 4.9: Predicting the number of paper downloads in the next download session with the baseline and LSTMs.

Model	**Accuracy**
Baseline	0.441
LSTM current session	0.453
LSTM current session + 1 previous session	0.462^{*}
LSTM current session + 2 previous sessions	0.463^{*}
LSTM full session	0.464^{**}
LSTM full session + user segmentation	0.481^{**++}

only using information from the current session as test input. But the performance gap between using different numbers of historical session inputs in testing is small. One explanation is that the LSTM model is already capable of "memorizing" the dependencies across sessions in training. Therefore, it can obtain a good prediction performance even without using the full session history in testing. Compared to the baseline, the LSTM model with full session history gains improvements in predicting very short and short time gaps (time gaps defined in Section 4.5.1), with an increase in accuracy of 18.2% and 20% for the 2 classes respectively, while performing worse in other classes.

The results of predicting the number of downloads (Table 4.9, rows 2–5) show a similar pattern as those for the time prediction tasks. All LSTM models perform better than the baseline, with an increase coming from predicting single download sessions (+23.8%) and 2–4 downloads sessions (+2.6%). However, here historical session information leads to significant improvements over the LSTM models without them. Using full session history significantly improves the performance over models using only part of the session history.

Specialized model based on user segmentation

Both for predicting the time until the next download session (Table 4.8, row 6) and predicting the number of downloads in the next download sessions (Table 4.9, row 6), the LSTM model with user segmentation performs significantly better than the baseline and other LSTM models. This should not come as a surprise as we notice the differences of user behavior separated by clusters, for instance, the median of user download time gaps varies significantly across clusters, ranging from 53,353 to 67,147 seconds. It would be better for the prediction models to train on users that are similar in behavior, rather than on a mixture of different users. This explains the performance increase by applying user segmentation.

Additional topical feature

In Section 4.3.4 we have seen that there are behavioral differences among users with different topical interests. We hypothesize that using the topical interests of users would help download prediction. Next, we examine whether topical features improve performance on our download prediction tasks. We augment the models considered so far with an additional categorical feature indicating which of the 4 general topics the user is most interested in. The results for predicting the time until the next download session are shown in Table 4.10.

Table 4.10: Predicting time until the next download with an additional topical feature. We denote significant differences after using the topic features with ** for significance at $\alpha = .01$.

Model	**Accuracy**
LSTM full session	0.357
LSTM full session + topic feature	0.360**
LSTM full session + user segmentation	0.371
LSTM full session + user segmentation + topic feature	0.376**

The addition of a topical feature leads to significant improvements, both with and without user segmentation. A similar conclusion can be drawn when predicting the number of downloads in the next download session, as shown in Table 4.11.

Table 4.11: Predicting the number of downloads in the next download session with additional topical feature. We denote significant differences after using the topic features with ** for significance at $\alpha = .01$.

Model	**Accuracy**
LSTM full session	0.464
LSTM full session + topic feature	0.466**
LSTM full session + user segmentation	0.481
LSTM full session + user segmentation + topic feature	0.485**

As we discussed in Section 4.3.4, the topical feature in an academic search setting can be a useful indicator of behavioral patterns. Here, the performance boosts show its utility for the two download prediction tasks that we consider. In both tasks, three additions gave us cumulative boosts in performance: (1) switching to LSTMs; (2) employing user segmentation; (3) adding a topical feature, where the most significant improvement comes from the LSTM model with user segmentation.

4.6 Conclusion

We have studied the download behavior of users of an academic search engine, a type of conversion behavior that has not yet been well examined in the literature. We first conducted a thorough observational study. We introduced a new dataset for user download behavior, defined user actions during a session, and showed action trajectories toward a download. Then we examined cross session download behavior, which was our main focus. We identified temporal patterns in users' download behavior. We also discovered multiple correlations of user behavior across sessions. Certain behavioral factors such as the number of downloads are correlated across sessions. The time gap until the next download session is negatively correlated with the number of queries.

We also examined topical aspects of downloads and their impacts on download behavior. We used annotated topical information of journals to classify the topics of users' downloads. We have found a bias in the distribution of topics of downloads, where the natural sciences (physical, life and health sciences) outnumber the social sciences. Furthermore, we identified behavioral variances in terms of download diversity and coherence between users who are interested in different topics. Not only do users download papers across subtopics, but they also download across disciplines, which confirms recent findings that academic research is becoming increasingly interdisciplinary.

Building on the insights gained from our observations, we moved on to two download prediction tasks: predicting the time until the next paper download session and predicting the number of downloads in the next download session. These two tasks help alleviate the information overload problem in academic recommender systems, as well as making the predictions pre-emptive in terms of their timing. We proposed a model based on LSTMs to utilize users' full session history for prediction, which gave rise to significant improvements over a state-of-the-art baseline method developed for a similar problem. Motivated by the observed differences in individual behavioral pattern, and the ability of dynamic time warping (DTW) to measure the similarity between time series, we built specialized models based on user segmentation with DTW. The specialized models showed significant improvements, indicating that user segmentation with DTW is beneficial. Last but not least, we established the usefulness of topic features in download prediction.

As to future work, we would like to pursue three main themes. First, our predictions of the number of paper downloads and time gap can help paper recommender systems handle information overload and recommendation timing; we plan to include such predictions as signals in an online academic paper recommender system. Second, so far we have considered only a "general" topical interests rather than fine grained subtopics in predicting downloads. This is due to sparsity of paper downloads per subtopic in our dataset, e.g., we have no download records for some subtopics. The subtopics, or certain types of journals may have their unique cycle of publishing papers. They can potentially impact user download patterns. We hope to collect data that will allow us to fully explore the impact of subtopics on download behavior. Third, while we have closely studied users' interaction behavior, we have not captured it with click models [44]: downloads can be seen as a special type of click; can we model different actions on an academic search interface to understand the different types of bias or to understand what makes the rich result presentation per item that is customary in academic search (consisting of title, authors, abstract snippet and possibly more) effective?

5 Personalised Ranking of Paper Recommendations Using Paper Content and User Behavior

In the previous chapters, we have been focusing on user behavior on the academic search engine, including queries, topic shift and downloads. In this chapter, we extend our focus beyond the search engine. We examine a different scenario, that is to make academic paper recommendations through email newsletters. Paper recommenders can make it easier for users to access information without the need of an explicit query that is required on the search engine. However, making good recommendations is not a trivial task, especially for new users who have not interacted with the recommender system. We come up with a model that utilizes content and behavior for reranking paper recommendations generated by a production system, and answer **RQ4**.

5.1 Introduction

Along with the digitization of academic resources and the increasing popularity of academic information platforms, access to academic papers online has become ubiquitous for many people. Various online academic service providers have given users access to papers through their search engines, such as Effective Communication, Better Science [62], Aminer [206] and ScienceDirect [186], where users can enter queries to seek relevant papers in their database. In this scenario, users need to have an idea of what they are looking for, and the information needs can be formalized as queries. The system takes a query as input, and returns a ranking of relevant papers for the users to examine and interact with.

While such academic search engines can often fulfill user requests by catering to specific information needs represented as queries, there are cases when users' information needs are not explicitly specified. For instance, users may want to learn about new developments in their domain by looking at emerging papers that are relevant. In this case the user may not have an idea of what queries to enter on the search engine. This

This chapter was published as [139].

is a situation where paper recommender systems can step in and recommend relevant papers without the need of any user query.

A paper recommender system has a role that is complementary to the search engine. The possible recommendation scenarios fall into three categories based on the recommendation timing:

1. displaying paper recommendations before users start a new search session, based on their paper library or previously accessed papers [see, e.g., Google Scholar, 62];
2. during a search session, displaying related recommendations beside the content that the user is currently browsing [see, e.g., ScienceDirect, 186]; and
3. after a search session, sending emails of paper recommendations in the form of a newsletter ScienceDirect [see, e.g., ScienceDirect, 186].

The first and third scenario fill the gap between user search sessions, while the second scenario is related to within-session recommendations.

In this study we focus on the third scenario. We look at the ScienceDirect paper recommender which sends a weekly email of paper recommendations to users. First, we provide a recommendation example from the system in Figure 5.1 to show how it works.[1] The recommender of ScienceDirect generates a ranked list of 5 paper recommendations based on the user's browsed papers. The email newsletter displays the title, venue (journal), authors, and publication date of each recommended paper. On clicking a recommendation, the user is linked to the paper on ScienceDirect. The system then logs which recommendation(s) the user clicks. As a short summary, this system aims to recommend interesting papers to users based on their browsing history. A good recommendation list will place more relevant papers higher in the list.

Since the ScienceDirect paper recommender was released, an increasing number of users have signed up. It is especially challenging to make recommendations for these new users due to the lack of historical interactions with the recommender system. In this chapter, we address the challenge and try to come up with better recommendations for these new users. Specifically, we study the task of reranking the paper candidates generated by the current production system. Ranking is a very common module of the workflow in production recommender systems, which usually include at least a candidate-generation phase and a ranking module [46, 52, 182]. The output of the system is generated by a multi-step process. We address this reranking task so that our model can easily be integrated into paper recommender systems (e.g., the ScienceDirect recommender). A direct application is to use our model to rerank the recommended papers generated by the ScienceDirect recommender system.

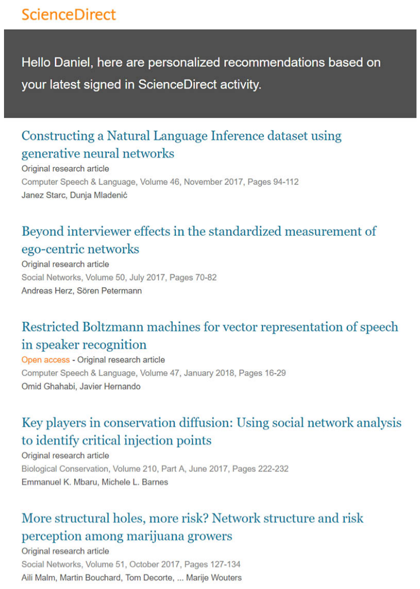
Over 14 million papers are indexed on ScienceDirect [187]. Picking the few papers that may appeal to the user is not a trivial task. Collaborative filtering techniques are often used in recommender systems to generate a candidate pool of papers based on user-paper interactions. Even though there was initially no data on user interaction with the recommender system, there was still a wealth of data on user interactions with papers on ScienceDirect. Apart from this behavior aspect, paper metadata may assist the recommendation task by providing similarity measures that are based on paper contents,

[1] https://www.elsevier.com/connect/suffering-from-information-overload-personalized-recommendations-can-help

e.g., to recommend semantically similar papers, or papers that are authored by the same or similar authors.

In this chapter, we propose a hybrid model that combines content and behavior to rerank the paper recommendation candidates generated by the ScienceDirect recommender. First, we propose several content-based measures that are derived from various paper aspects, such as word space similarity, and author similarity from an embedding space. Next, we use joint matrix factorization to learn a mapping from a user's browsed articles on the search engine to a user's clicks on the recommendations, to alleviate the sparsity of the recommendation click data. We use a pairwise learning model to rerank the paper recommendation candidates that eventually leads to better results in the offline evaluations based on real email click data.

The chapter is structured as follows. We describe the models in Section 5.2, the experimental setup in Section 5.3, and the results and analysis in Section 5.4. We present related work in Section 5.5 and conclude in Section 5.6.


ScienceDirect

Hello Daniel, here are personalized recommendations based on your latest signed in ScienceDirect activity.

Constructing a Natural Language Inference dataset using generative neural networks
Original research article
Computer Speech & Language, Volume 46, November 2017, Pages 94-112
Janez Starc, Dunja Mladenić

Beyond interviewer effects in the standardized measurement of ego-centric networks
Original research article
Social Networks, Volume 50, July 2017, Pages 70-82
Andreas Herz, Sören Petermann

Restricted Boltzmann machines for vector representation of speech in speaker recognition
Open access - Original research article
Computer Speech & Language, Volume 47, January 2018, Pages 16-29
Omid Ghahabi, Javier Hernando

Key players in conservation diffusion: Using social network analysis to identify critical injection points
Original research article
Biological Conservation, Volume 210, Part A, June 2017, Pages 222-232
Emmanuel K. Mbaru, Michele L. Barnes

More structural holes, more risk? Network structure and risk perception among marijuana growers
Original research article
Social Networks, Volume 51, October 2017, Pages 127-134
Aili Malm, Martin Bouchard, Tom Decorte, ... Marije Wouters

Figure 5.1: An excerpt from a sample recommendation email sent to a ScienceDirect user based on his recent activity. The email contains 5 papers linked to ScienceDirect.

5.2 Models

In this section we introduce the models for the paper recommendation task. First, we introduce the production baseline, because it provides the candidates for our proposed reranking model. Then, we introduce the content and behavior components that measure paper similarities based on content and behavior respectively. Based on these, we describe our hybrid reranking model (*Hybrid Reranking Model* (HRM)) that incorporates the behavior and content components and reranks the candidates. Here “hybrid” refers to using both content and behavior.

5.2.1 Production baseline

The production system takes the paper browsing history of a user as input, and produces a ranked list of 5 paper recommendations. While we are not able to elaborate on the exact details of the production system, we can describe the core part of the algorithm: the 5 candidates are generated and ranked by an algorithm that uses an item-item neighborhood-based collaborative filtering method [145, 183], based on usage similarity from ScienceDirect browsing logs. We denote this paper-paper similarity as browsing similarity for later references.

In this study, we apply the reranking model to the top five candidates from the production system, and compare the model’s ranking to the production baseline. The top five candidates were chosen because for these recommendations, there is email click feedback that enables offline evaluation; if successful, the model could be applied to a longer list of candidates.

5.2.2 Proposed model

Our proposed model takes the 5 candidates from the production baseline as input and produces a reranked list. It leverages both user behavior and paper content for reranking. The user behavior component considers both browsing history on the search engine and clicks on the recommendations to alleviate the sparsity issue of the latter part; the content component obtains the similarity of papers derived from paper metadata. Then, a pairwise learning model uses these components for reranking the candidates.

First, we introduce how we consider the paper metadata to measure different types of similarity. Below, we provide formal representations of various paper aspects, of users, and then the similarity functions for them.

Paper representations

Each paper p is represented as a collection of different aspects, and can be categorized as follows:

- metadata from papers: author A, venue V, freshness F, word space W, entity space E;
- metadata from user interactions: impact I and popularity P.

These aspects are available for all papers and users in our scenario and are considered to be potentially useful for the recommender system.

Formally, we can think of every paper p as a tuple $p = \langle A(p), V(p), F(p), W(p), E(p), I(p), P(p)\rangle$, where each aspect is defined as follows:

Authors:

$$A(p) = [a_1, a_2, \ldots, a_n],$$

where a_i is an author of the paper p.

Venue:

$$V(p) = v_i,$$

meaning that paper p is published in venue (journal) v_i.

Freshness score: We also model how "fresh" a paper p is, defined as:

$$F(p) = \frac{1}{e^{t_{current} - t_{publish}(p)}},$$

where $t_{current}$ is the current time, and $t_{publish}(p)$ is the time of the paper p being published online. $F(p) \in (0, 1]$. The more recently a paper has been published, the higher the freshness score is.

Impact score: We use citations as a measure of impact for papers, which is defined as:

$$I(p) = \frac{\log(c(p) + 1)}{\log(c_{max} + 1)},$$

where $c(p)$ is the citation count of the paper p, and c_{max} is the maximum number of citations in the dataset.

Popularity score: The popularity of a paper p reflects how often users interact with the paper. We use the number of downloads to represent popularity:

$$P(p) = \frac{\log(d(p) + 1)}{\log(d_{max} + 1)},$$

where $d(p)$ is the number of downloads of the paper p, and d_{max} is the maximum number of downloads of a paper in the dataset.

Word space: To represent papers in a word space we use tf-idf vectors, with values for words and bigrams in the article title, abstract and keywords. We remove English stop words, very common words and very rare words before calculating the tf-idf values. In the end, each paper is represented as a sparse vector of size of 2^{21}, with hashing to determine token indices in the vector.

Entity space: While word space measures such as tf-idf similarities can be used to directly compare the contents of papers, an entity space is able to provide us with additional information that incorporates both structure and semantics through graph embeddings [26, 143].

We first build a knowledge graph by using important aspects of a paper including keyword, author and venue. The graph contains 4 node types, paper, author, keyword and venue nodes, and 3 relations (predicates) between a paper and an aspect as listed below:

- hasAuthor: the paper has this author;
- hasKeyword: the paper contains this keyword; and
- publishedInVenue: the paper is published in this venue (journal).

Next, to model the entity space we use the state-of-the-art TransE model [26] to derive embeddings based on knowledge graphs. As input the model takes the triplets in the graph; these have the form (h, r, t), with a head entity h, a relation (predicate) r, and a tail entity t. The objective of the model is to learn embeddings so that $h + r$ lies in the proximate neighborhood of t if such a triplet (h, r, t) exists in the training set, and $h + r$ will be far away from t if the triplet is not valid. The model learns the embeddings by minimizing a pairwise hinge loss:

$$\sum_{(h,r,t)\in T} \sum_{(h',r',t')\notin T} [1+ \mid h + r - t \mid - \mid h' + r' - t' \mid]_{+},$$

where T denotes the training set of triples. After training, the cosine distance of the node embeddings reflects their proximity in the knowledge graph.

Due to the relatively high computational costs of working with knowledge graphs [26], we derive the embeddings on a subgraph instead of on the complete graph. We choose a reasonable size for the subgraph so that it is computationally feasible and also alleviates the sparsity problem in the node connections. The subgraph is comprised of the union of the browsed papers and recommended papers from 65,994 users, a superset that is about 15 times the size of the users that we will study in our experiments. In total we have 609,716 paper nodes, 1,650,470 author nodes, 3,961 venue nodes and 808,845 keyword nodes, plus 6,103,728 relation edges.

The graph is then used as the input for the TransE model to derive embeddings of the nodes in the graph. In the end, we obtain embeddings for papers, authors and venues. These embeddings will be used later in the content similarity measures.

User representations

The user representations are straightforward: each user u is represented as a collection of papers in their browsing history:

$$\begin{aligned} u &= [P_{recent}, P_{history}] \\ P_{recent} &= [p_1, p_2, \dots p_k] \\ P_{history} &= [p_{k+1}, p_{k+2}, \dots p_n]. \end{aligned}$$

We segment a user's browsed papers into two sets, the recent ones, P_{recent}, and the historic ones, $P_{history}$. We write p_i to refer to the i-th paper in each of the segmentations, in the order of occurrence in the user's timeline starting from the most recent one. In academic search, users' topic interests may shift over time [135]. We make this segmentation so that it may help us compare the user's recent interests against their historical interests, and see whether and to which extent there is a deviation.

In case of a large deviation, P_{recent} should provide more support to generate paper recommendations.

Specifically, the clicked papers in the most recent session are put into P_{recent} if it contains at least clicks on 2 different papers, and the rest into $P_{history}$. Otherwise, we select the most recent θ papers from u into P_{recent} and put the rest into $P_{history}$. Papers in P_{recent} and $P_{history}$ are deduplicated.

Content similarities

Based on the user and paper representations, in this section we describe the similarity functions to measure different types of content similarity. Specifically, the content component measures the similarity between recommendation candidates and users' browsed papers using information from the paper metadata. The output will consist of similarity scores to feed into the reranking model.

Field-level similarities and attention features. First, we introduce similarity measures for individual fields, which are used to compare paper similarities in each field. When comparing two papers p_i and p_j, the similarity of each field is defined as follows.

For the word space and entity space, we use the cosine similarity of the vectors that represent each paper. The cosine similarity between two vectors v and v' is defined as:

$$\cos(v, v') = \frac{v \cdot v'}{\mid v \mid \cdot \mid v' \mid},$$

where the similarity value $\cos(v, v')$ ranges between -1 and 1.

Then, the similarities for word and entity space are:

$$\begin{aligned} SimW(p_i, p_j) &= \cos(W_{p_i}, W_{p_j}) \\ SimE(p_i, p_j) &= \cos(E_{p_i}, E_{p_j}), \end{aligned}$$

where W_{p_i} is the tf-idf vector and E_{p_i} is the paper entity vector for paper p_i obtained from the output of the TransE model [26].

Similarly, a venue entity vector $E_{v_{p_i}}$ for paper p_i and an author entity vector E_{a_m} for author a_m of p_i are obtained from the output of the TransE model [26]. We apply a "soft match" approach when comparing venue and author similarities. Compared to an "exact match" approach where the similarity ends up being either 1 (same) or 0 (different), the "soft match" approach outputs a continuous similarity score. For instance, "Accident Analysis & Prevention" and "Safety Science" being two different journals (with no overlapping terms in the journal title), they would have a similarity score of 0 in the "exact match" approach. However, in the embedding space they would have a similarity score of 0.48, representing a more precise estimate of the inherent similarity.

Then, venue and author based similarity measures, $SimV(\cdot, \cdot)$ and $SimA(\cdot, \cdot)$ are defined as follows:

$$\begin{aligned} SimV(p_i, p_j) &= \cos(E_{v_{p_i}}, E_{v_{p_j}}) \\ SimA(p_i, p_j) &= \begin{cases} \dfrac{\sum_{a_m \in A_{p_i}} \max_{a_n \in A_{p_j}} \cos(E_{a_m}, E_{a_n})}{|A_{p_i}|}, & \text{if } |A_{p_i}| \leq |A_{p_j}| \\ \dfrac{\sum_{a_n \in A_{p_j}} \max_{a_m \in A_{p_i}} \cos(E_{a_n}, E_{a_m})}{|A_{p_j}|}, & \text{otherwise,} \end{cases} \end{aligned}$$

where v_{p_i} is the venue of paper p_i, $E_{v_{p_i}}$ is the corresponding paper entity vector; A_{p_i} is the set of authors of paper p_i, and E_{a_m} is the entity vector for author a_m. Note that in the author similarity function, we examine each author from the smaller author set and find the most similar one in the other set and then calculate the average of the similarities. This ensures that $SimA(p_i, p_j)$ is symmetrical.

For freshness, impact and popularity, these three measures are single value features. We use $L1$ distance with an adjusted weighting to obtain their similarities:

$$\begin{aligned} SimF(p_i, p_j) &= (1- \mid F(p_i) - F(p_j) \mid_1) \times \max(F(p_i), F(p_j)) \\ SimI(p_i, p_j) &= (1- \mid I(p_i) - I(p_j) \mid_1) \times \max(I(p_i), I(p_j)) \\ SimP(p_i, p_j) &= (1- \mid P(p_i) - P(p_j) \mid_1) \times \max(P(p_i), P(p_j)). \end{aligned}$$

We define the weighting in order to capture the similarities only when two papers both have a high value in this field. In cases where both have a low value, the similarity value will be "down-weighted", representing a weaker level of evidence for similarity. For instance, given 2 paper pairs with low impact values (0.1, 0.2) and high impact values (0.8, 0.9), the similarity score would be 0.09 and 0.81 respectively. Although the absolute difference of impact is the same for both pairs (0.1), the pair with relatively high values has a much larger similarity score.

Field level attention. Now that we can obtain the similarity scores $SimX$ for 7 choices of X (W, E, V, A, F, I, P), we would like to further know which specific fields the user may be focusing on while browsing the papers. This is to tailor the recommendations for those fields, be it the semantic similarity, venues or authors. These "attention features" are implicit. However, we can derive the attention features through past user interactions. In particular, we assume that they can be inferred from P_{recent} (users' recently browsed papers). We hypothesize that for a set of papers, if the average pairwise similarities of certain aspects are higher than other fields, it is probably because users are paying attention to these aspects. For instance, high word space similarity indicates that users are sticking to a specific topic. Likewise, if the venue and freshness similarity scores are high, this could be that the user is mostly checking papers that are both recent and are from a specific journal. We use the averaged pairwise similarities calculated by each field as the field-level attention feature.

The attention feature for field_i is the sum of its pairwise similarities divided by the number of paper pairs in P_{recent}:

$$\alpha_{\mathit{field}_i} = \frac{\sum_{p_i, p_j \in P_{recent}, i \neq j} Sim_{\mathit{field}_i}(p_i, p_j)}{C^2_{|P_{recent}|}},$$

where $C^2_{|P_{recent}|}$ refers to the number of paper pairs.

Recent and history attention. The users' recent and historic paper interactions may both provide evidence to surface good recommendations. We make the distinction between recent and historic papers because users' interests may evolve over time. When the users' recent interests are significantly different from their historical interests, the recommender should be aware of this deviation. Therefore, we define attention features for this situation, where α_{recent} and $\alpha_{history}$ represent the two attention scores on the users' recent and historic papers.

Table 5.1: Notations used in this section.

Notation	**Description**
m	number of users
n	number of papers
k	number of latent dimensions
c_{ui}	click of user u to paper p_i
S	paper-paper browsing similarity matrix
s_{ij}	browsing similarity between paper p_i and p_j
$\mathcal{B}_u$	set of papers browsed by user u
$\mathcal{C}_u^+$	set of papers clicked by user u
$\mathcal{C}_u^-$	set of papers shown but not clicked by user u
$\boldsymbol{q}_i$	latent factor for paper p_i
$\boldsymbol{b_i}$	bias of paper p_i

α_{recent} and $\alpha_{history}$ are calculated using the browsed papers (P_{recent} and $P_{history}$). The more the user's recent interests deviate from the historic interests, the higher the value of α_{recent}, hence providing a bias feature to consider the more recent user activities. It is calculated as follows:

$$\begin{aligned} \alpha_{recent} &= Distance(P_{recent}, P_{history}) \\ \alpha_{history} &= 1 - \alpha_{recent}. \end{aligned}$$

The distance $Distance(P_{recent}, P_{history})$ is calculated by averaging over the distance of each paper in P_{recent} to its closest match in $P_{history}$. The idea of finding each paper's closest match instead of averaging over all papers in $P_{history}$ is because the history may be diversified: a recent paper may be very similar to one paper in the history but different to the rest. In case there is at least one similar paper in $P_{history}$, we consider that the current paper being examined does not deviate far from the history. Formally, the distance is defined as follows:

$$Distance(P_{recent}, P_{history}) = \frac{\sum_{p_i \in P_{recent}} \min_{p_j \in P_{history}} (1 - cos(W_{p_i}, W_{p_j}))}{|P_{recent}|},$$

where $1 - cos(W_{p_i}, W_{p_j})$ is the cosine distance between two papers' tf-idf vectors.

So far, we have explained how we exploit the content aspects for recommendation that are based on the paper metadata. Next we introduce the behavior aspect where user-paper interactions are concerned.

Behavior

The paper metadata has provided evidence for recommendations from the content perspective. User interactions, i.e., users' browsing behavior on the search engine and clicks on recommendation emails also provide signals for generating good recommendations. In our scenario, the users have past browsing behavior but no clicks prior to their first interaction with the recommender system.

Nevertheless, the paper-paper browsing similarities are available to us (as used by the production system, described in Section 5.2.1). They provide a measure of behavior-based similarity based on readership of all users on ScienceDirect.[2] Naturally, we can incorporate this external similarity information into our model.

We devise a behavioral model, that utilizes both browsing and click behavior in the interaction log. The motivations of our model are given below:

- For new users, there are no prior email clicks for predicting their interactions with the recommender. To address the issue, we complement the absence of click ratings by using the browsing history. It is noted that browsing papers on the search engine and clicking a paper in the email are two different user interactions with papers. Thus a mapping function is required to transform browsed papers to email clicks. Although for each user, it is not possible to learn the mapping as there is no click at the time of recommendation, it is possible to utilize the browsed papers and email clicks of other users (and this data quantity will grow over time). Essentially, we try to infer the clicks of the new users from other users' mappings, using supervised learning.
- As paper recommendations are shown in a relatively compact email, we assume that users have noticed all the papers. Therefore, a user's clicks on the 5 shown papers in the email entail implicit pairwise preferences. For instance, given 5 papers p_1, p_2, p_3, p_4 and p_5, if the user clicks paper p_2 and p_3 in the list of 5 papers, then it is reasonable to assume that the user prefers paper p_2 and p_3 over paper p_1, p_4 and p_5.
- The paper-paper similarity based on a user's browsing history is available. It is more accurate than the similarity from user clicks in emails, because it is based on the complete set of ScienceDirect users, which is several orders of magnitudes larger. Moreover, it captures transitive similarities from a global perspective. Therefore, it is necessary for our model to preserve this similarity.

We first present the notation we use in Table 5.1; it is used in the following model descriptions. We propose to learn a mapping function from user browsed papers to user clicks on the email, denoted as:

$$C \sim BM, \tag{5.1}$$

where $B, C \in \mathbb{R}^{m \times n}$ are the matrices for browses and clicks respectively and $M \in \mathbb{R}^{n \times n}$ is a mapping matrix. In practice, n is generally very large so that it could pose a great burden to learn M. Thus we propose to factorize M into the multiplication of a low-dimensional paper factor Q, shown below:

$$M \sim QQ^T, \tag{5.2}$$

where T is the transpose operator of a matrix.

Based on the assumptions given in Eq. (5.1) and (5.2), we can predict the click of user u on paper p_i by the equation below:

$$\tilde{c}_{ui} = b_i + \boldsymbol{q}_i^T \sum_{t \in \mathcal{B}_u} \boldsymbol{q}_t, \tag{5.3}$$

[2]The involved users are larger than the users we study in our experiments by several orders of magnitude.

where b_i is a scalar for the bias of paper p_i. Here we ignore user bias in Eq. (5.3) since it is unknown for new users. To learn Q and $\boldsymbol{b}$, for each user, we draw a pair of papers (p_i, p_j), where $p_i \in \mathcal{C}_u^+$ and $p_j \in \mathcal{C}_u^-$, and optimize a pairwise loss function given by BPR [179]:

$$\mathcal{L}(u,i,j) = -\log \sigma\left(\tilde{c}_{ui} - \tilde{c}_{uj}\right), \tag{5.4}$$

where $\sigma(\cdot)$ stands for the sigmoid function. To preserve paper-paper similarities from browsing history, we follow the assumption that the distance between $\boldsymbol{q}_i$ and $\boldsymbol{q}_2$ is small when s_{ij} is large. Without loss of generality, we adopt the Euclidean distance hereafter, e.g., $\mid \boldsymbol{q}_i - \boldsymbol{q}_j \mid_2^2$. We can then define the following similarity regularization terms:

$$\begin{aligned}\frac{1}{2}\sum_{i,j}^{n} \mid \boldsymbol{q}_i - \boldsymbol{q}_j \mid_2^2 s_{ij} &= \sum_{i=1}^{n} \boldsymbol{q}_i^T \boldsymbol{q}_i d_{ii} - \sum_{i,j}^{n} \boldsymbol{q}_i^T \boldsymbol{q}_j s_{ij} \\ &= \mathrm{Tr}\left(Q^T D Q\right) - \mathrm{Tr}\left(Q^T S Q\right) = \mathrm{Tr}\left(Q^T L Q\right),\end{aligned} \tag{5.5}$$

where Tr $(\cdot)$ is the trace operator of a matrix, D is a diagonal matrix whose entries are the row sums of the browsing similarity matrix S (S is symmetric), i.e., $d_{ii} = \sum_{j=1}^{n} s_{ij}$, and $L = D - S$ is the Laplacian matrix of the graph [45]. Putting Eq. (5.4) and (5.5) together, the model is given as follows:

$$\min_{Q,\boldsymbol{b}} \quad \sum_{u=1}^{m} \sum_{\substack{i\in\mathcal{C}_u^+,\\ j\in\mathcal{C}_u^-}} -\log\sigma\left(\tilde{c}_{ui} - \tilde{c}_{uj}\right) + \alpha \mathrm{Tr}\left(Q^T L Q\right) + \frac{\lambda}{2} \mid Q \mid_F^2 . \tag{5.6}$$

The first term in the objective function captures the pairwise preferences of every user over the papers shown in the emails. The second term preserves the paper-paper similarities in the browsing history through graph regularization. The graph regularization is widely used to preserve similarities, e.g., social regularization [147] and locality regularization [184]. The third term regularizes Q to avoid overfitting. α and λ are hyper-parameters.

Reranking model

In this section we introduce *Hybrid Reranking Model* (HRM). The model scores the paper recommendation candidates generated by the production system, using the aforementioned content and behavior components. Then, the candidates are reranked by the score.

We use a 2-layer feedforward neural network as the scoring function, where the input layer takes features from each candidate paper (the features will be explained shortly afterwards), and the output layer contains one node that gives the score.

An overview of the model is shown in Figure 5.2.

We explain what the input feature representations are in Figure 5.2 from left to right: the attention features on different fields and on recent/history papers are derived from a user's browsed papers; the S_{recent} and $S_{history}$ contain the average similarity scores of each paper aspect by comparing the candidate paper against the recent papers and historic papers, respectively. These functions are described in Section 5.2.2.

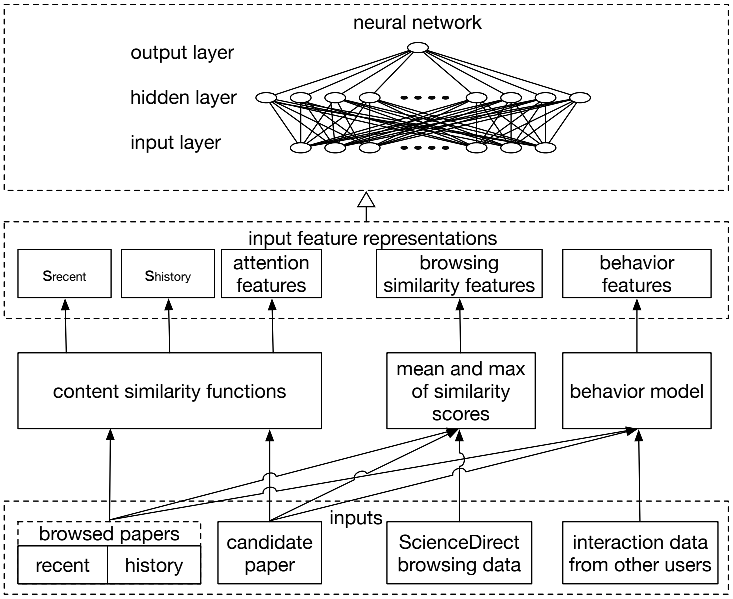

neural network
output layer
hidden layer
input layer
input feature representations
Srecent
Shistory
attention features
browsing similarity features
behavior features
content similarity functions
mean and max of similarity scores
behavior model
inputs
browsed papers
recent
history
candidate paper
ScienceDirect browsing data
interaction data from other users

Figure 5.2: Architecture of the Hybrid Reranking Model (HRM) that shows how a candidate paper recommendation for a user is scored.

The paper-paper browsing similarity features used by the production system are based on ScienceDirect browsing data: we use the mean and maximum similarity scores of each paper recommendation candidate compared against papers in the browsing history; the behavior features are the predicted click scores obtained from the behavior model described in Section 5.2.2. Together these features determine the inputs for Hybrid Reranking Model (HRM).

Training is done by optimizing a pairwise hinge loss from the preferences of clicked papers over the non-clicked papers for each user u, shown below:

$$L(\mathcal{C}_u^+, \mathcal{C}_u^-) = \sum_{p_i \in \mathcal{C}_u^+} \sum_{p_j \in \mathcal{C}_u^-} \left[1 + f(x_{p_i}) - f(x_{p_j})\right]_+ ,$$

where $f(\cdot)$ denotes the scoring function (neural network), $\mathcal{C}_u^+$ are the clicked papers in the email for user u and $\mathcal{C}_u^-$ are the non-clicked ones, x_{p_i} and x_{p_j} denote the feature representations for clicked paper p_i and non-clicked p_j respectively.

We apply rectified linear unit (ReLU) activations on the hidden layer for efficient learning. We apply linear activations on the output layer, because it ensures an unbounded value for the pairwise loss function and also performs the best in our experiments. We use the Adam optimizer [113] and mini batches during training.

5.3 Experiment

In this section we describe the experiments, including research questions, data preparation, experimental setup and the main results.

5.3.1 Research questions

We aim to find out how to utilize both content and behavior to rerank paper recommendations. We are interested in whether HRM which utilizes content and behavior can beat the production baseline, and how useful the different input features are. Specifically, we answer the following research questions.

RQ4.1 Does HRM which utilizes content and behavior, provide improvements in reranking over the production baseline?

RQ4.2 What is the utility of the content features and behavior features in reranking, respectively?

RQ4.3 Within the content features, for paper similarities based on various paper aspects, what paper aspects can result in good reranking performance and what can not?

5.3.2 Dataset

We use a dataset provided by ScienceDirect,[3] a popular academic search engine that offers access to millions of academic papers. Users can gain access either by a subscription service, or by individual purchases of papers. The dataset contains anonymized user activity logs from signed in users. We look at newly signed up users and their interactions on the first paper recommendation email. The paper recommendations emails were sent between December 12, 2017 and January 21, 2018. For each user, browsed papers on ScienceDirect prior to receiving the email were also obtained. A browsing action is characterized by any form of a click on a paper, such as a click on the search engine result page, or a click on related papers shown on the detailed paper content page. For email recommendation data, each email contains 5 recommendation candidates where users' responses to each one of them are logged (clicked or not clicked). To obtain paper metadata, we use the paper metadata from Scopus,[4] which can be obtained by querying paper IDs from papers in the ScienceDirect database.

Since we need to study how *content* contributes to better reranking, we need users that have at least a few papers in their browsing history in order to utilize the content information. The content data of the browsed papers should be clean and complete. Also, we need users who have at least one click on the recommendation email so that we can perform offline evaluations for reranking and calculate the metrics. Correspondingly, we apply the following filtering steps prior to obtaining the data:

1. we filter out cold users with fewer than 5 browsed papers prior to the recommendation;
2. we remove users whose browsed papers have incomplete or corrupt fields of data; and

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

[4]https://www.scopus.com

3. we remove the recommendation emails without any clicks.

In total we have obtained 4,392 recommendation sessions for our experiments. Each session contains one recommendation email with a field that indicates whether each paper has a click, and also the user's browsing history prior to the recommendation's timestamp.

Also readily available are the item-item collaborative filtering scores based on readership of papers from ScienceDirect users. The scores of paper pairs are symmetrical so that $s_{ij} = s_{ji}$.

5.3.3 Experimental setup

The experiments on our dataset are conducted through 5-fold cross validations. For each run, 4 folds are used for training and 1 fold is used for testing. There are one or more clicks on the paper candidates for each email. We code relevance as a binary label, which is 1 for clicked papers and 0 for the rest. We compute the mean average precision (MAP) and Precision@k ($k = 1, 2, 3$, denoted as Prec@k for short) as the evaluation metrics.

Significance tests are applied when comparing the results of different models. Specifically, we apply the two-tailed student t test to MAP and Wilcoxon signed rank test to Prec@k, according to certain underlying assumptions of the significance test.

We select the optimal hyper-parameters for HRM by iterating over possible parameter combinations. For the content component of HRM, we have $\theta = 3$; for the behavior component of HRM, we have $\alpha = \lambda = 0.01$ and $k = 100$; for the scoring function of HRM, the hidden layer contains 32 nodes (more nodes may lead to overfitting and worse performances in the experiments), and the learning rate is set as 0.001.

What are appropriate baselines to consider? The first and obvious baseline is the production system that we seek to improve over; this baseline mainly uses item-item similarity based user browsing data on ScienceDirect. In addition, two families of approaches appear to be natural candidates: *learning to rerank methods* and *collaborative filtering methods*.

As to learning to rerank models, to the best of our knowledge, approaches to learning to rerank a production system published in the literature focus on learning from interaction data (see Section 5.5.5) We, however, focus on similarity-based models. Thus, we consider an (offline) pointwise learning to rerank model based on logistic regression with Adagrad optimization [57], which has recently achieved state-of-the art performance [196]. We also consider an (offline) linear pairwise learning model that is trained using pairwise hinge loss. These two models use the same input as HRM.

As to collaborative filtering methods as possible baselines against which to compare the approaches in this chapter, we would like to consider approaches such as the state-of-the-art neural collaborative filtering [87], PMF [155] and NMF [127]. Testing alternative collaborative filtering approaches would have required access to the full paper browsing data that are used by the production collaborative filtering approach. However, the scope of this paper is only to take the production collaborative filtering system as a baseline and experiment with reranking its output. Therefore, we ignore them in our experiments.[5]

[5]We have also considered applying CF baselines on the 4,392 users that we study. However, the user-paper

Note that when answering the second and third research questions (examining feature utility), certain components' feature size is very small. For instance, the behavior component consists of a feature size of two: one from our proposed behavioral model (Section 5.2.2), one from the browsing similarity (Section 5.2.1). Such a small feature vector is not suitable as input for the neural structure in HRM. Therefore, we use the pairwise linear model in this case.

5.4 Results and Analyses

In this section we present the experimental results, including the results of different models, and break-down analyses on different components of the model.

5.4.1 Overall comparison

To address our first research question, we compare HRM against the production baseline, as well as two other baselines, see Table 5.2.

Table 5.2: Results of reranking paper candidates across models. **W**in/**T**ie/**L**oss are the number of users for which a model performs better than, the same as, or worse than the production baseline.

Model	**MAP**	**Prec@1**	**Prec@2**	**Prec@3**	W/T/L
Production baseline	0.588	0.392	0.350	0.323	-/-/-
Linear pointwise learning to rerank	0.534	0.330	0.296	0.280	1595/753/2044
Linear pairwise learning to rerank	0.620	0.432	0.378	0.343	1822/1254/1316
HRM	0.663	0.502	0.453	0.421	2005/1171/1216

Compared with all three baselines, significant improvements are made in the hybrid reranking model HRM that combines content and behavior ($p < 0.01$). Compared with the production baseline, HRM performs better or the same for 72.3% of the users. There is a relative 13% increase in MAP and a relative 28% increase in Prec@1 for HRM, meaning that users are more likely to click the top candidates in the reranked list. This answers the first research question.

Besides, when given the same input features, HRM also performs better than both the linear pointwise and pairwise learning model. Interestingly, the pointwise learning to rerank method is beaten by the production baseline on all metrics. This shows that learning absolute user preferences of papers based on clicks is not optimal in our scenario. Models based on pairwise learning (HRM and the linear pairwise model) have produced better results by learning relative user preferences.

interactions are so sparse that CF baselines would not generate a rating for over 90% of the papers: the rating density is less than 0.02% even if we consider both browses and clicks as ratings on papers, which is significantly less than common recommendation datasets (Movielens 100K: 6.30%, Movielens 1M: 4.47%, FilmTrust: 1.14%).

5.4.2 Utility of content and behavior features in reranking

To answer the second research question, we analyze the utility of the input features of individual components in HRM, shown in Figure 5.2. Specifically, we look at the reranking performance using the following input features separately.

- Proposed behavior feature (Section 5.2.2).
- All behavior features: browsing similarity features (Section 5.2.1) and proposed behavior feature (Section 5.2.2).
- All content features: S_{recent}, $S_{history}$, attention features (Section 5.2.2).
- All content features without attention features: S_{recent}, $S_{history}$ (Section 5.2.2).
- Only recent content similarity: S_{recent} (Section 5.2.2).
- Only historic content similarity: $S_{history}$ (Section 5.2.2).

The behavior features have small sizes (a single feature from our behavioral model and the production system respectively). Therefore we opt for the linear pairwise model, because the small input feature vector is not suitable for the neural structure in HRM. For other features that have larger sizes we use the neural structure of HRM for reranking; see Table 5.3 for the results.

Table 5.3: The performance of reranking paper candidates using different input features of HRM shown in Figure 5.2.

Model	**MAP**	**Prec@1**	**Prec@2**	**Prec@3**
proposed behavior feature	0.540	0.332	0.302	0.288
all behavior	0.602	0.411	0.358	0.327
all content	0.601	0.402	0.365	0.338
all content without attention	0.598	0.398	0.359	0.337
only recent content	0.590	0.384	0.354	0.333
only historic content	0.582	0.374	0.343	0.327

Using behavior and content separately for reranking, the results (MAP score of 0.602 and 0.601, respectively) already outperform the production baseline (0.588) that mainly uses item-item CF. The proposed behavior feature provides a boost for the behavior component in addition to using the browsing similarity features from the production system ($p < 0.01$). On the other hand, the content component has a performance quite close to the behavior component. The attention features lead to a slight improvement over the model without them. We also find that using the recently browsed papers is better for reranking paper candidates than to using historically browsed papers, and even better is to use both recently and historically browsed papers. This answers the second research question.

5.4.3 Utility of paper aspects in reranking

To answer the third research question, we continue to delve into the content similarity in HRM, which contains similarity measures for different aspects of papers. We are interested to see the reranking performance of features based on a single paper aspect. For each paper aspect, we take the recent/historic similarity and the recent/historic attention scores as the input features for reranking. Similar to Section 5.4.2, we use the pairwise linear model due to the small input feature size. The results are shown in Table 5.4.

Table 5.4: Reranking paper candidates by restricting the pairwise linear learning rerank model to using only one paper aspect.

Field	**MAP**	**Prec@1**	**Prec@2**	**Prec@3**
freshness	0.426	0.153	0.248	0.242
popularity	0.453	0.154	0.276	0.284
venue	0.468	0.203	0.272	0.276
impact	0.489	0.257	0.283	0.267
word	0.526	0.312	0.291	0.284
author	0.549	0.327	0.320	0.311
paper entity	0.550	0.330	0.319	0.311

The reranking performance of the paper candidates differs among the paper aspects. In general, the similarity measures based on semantics or entities perform better than those that do not. The two entity space measures: the author and paper entity similarities perform better than other measures, also beating the word-space similarity. Comparing three entity based measures, the author similarity performs similarly to the paper entity similarity, this is due to the high correlation between them (Pearson correlation coefficient being 0.88); the author similarity performs much better than the venue similarity (0.549 vs 0.468 for MAP scores). This may suggest that users pay attention to the authorship of the paper more than the venue. Using freshness, popularity, or impact similarity alone does not generate good performance, understandably, as these measures do not consider semantic relevance or entity relationships. Combining all paper aspects produces the best performance. The third research question is hence answered by the above comparisons of paper aspects' utility in reranking.

5.5 Related work

In this section we discuss the related work to our study. The related work spans several topics: academic search, paper recommendation, citation recommendation, and top-n recommendation. We introduce them below and explain how they are related to our work.

5.5.1 Academic search

Our work is relevant to academic search because we are examining the recommendation service attached to an academic search engine. Academic search engines [62, 132, 186, 206] have given users convenient access to academic resources such as papers, journals, and authors. Mitra and Awekar [152] found that different academic search engines have their own coverage of literature and ranking strategy, and the overlap among search results is low. Compared to general web search, there is far less research on user behavior in academic search, possibly due to a lack of public datasets. Research on academic search has examined user behavior through surveys [164, 170, 171] and aggregated usage statistics such as query frequencies [107]. Khabsa et al. [108] studied user queries on Microsoft Academic Search and proposed a query classifier.

Recently, more studies have been conducted on user behavior within and across search sessions, based on a large-scale user transaction log. Li et al. [138] have studied the null query phenomenon in academic search and proposed a query suggestion method as a remedy. Li and de Rijke [135] have revealed the correlations of query reformulation and topic shift in academic search. Li and de Rijke [134] have also studied the user queries in academic search following major scientific events. There has also been research aimed to improve the search experience. Tang et al. [207] combined topic modeling with random walks to improve academic search retrieval performance. Khazaei and Hoeber [109] proposed a visual search interface via citation links to help users better navigate through search results. Xiong et al. [225] proposed improving paper rankings in academic search using entity embeddings.

Our work in this chapter differs from previous work in academic search in that we do not directly deal with search. We utilize the browsing history on the academic search engine to make improvements to a paper recommender.

5.5.2 Academic paper recommendation

Our recommendation task falls in the broad category of academic paper recommendations. Generally, based on the system inputs, paper recommendation tasks can be classified into the following scenarios: the system generates a list of paper recommendations given a single paper as input [14, 100, 160]; the system generates a list of paper recommendations given a set of papers as input (without ordering) [121, 189, 212]; the system generates recommendations given a time-ordered set of papers as input [91, 224]. The first and second scenarios include cases where a user is browsing a paper, or a list of relevant papers is available (e.g., through a set of papers selected by the user). The system assumes the input to be representative of a user's interests, then provides related papers as recommendations. These are the most common scenarios that are being studied. The third scenario is rarely studied because: 1) it is relatively difficult to acquire user data that spans a long period, for instance, users' paper browsing history; 2) it is more difficult to model user interests based on a sequence of inputs, compared to static inputs in the first two scenarios.

The common methods involved in making recommendations can be classified as: content-based filtering (CBF), collaborative filtering (CF) and hybrid models that combine the two. CBF involves using various parts of the paper contents, such as titles, abstracts, and keywords, to suggest related papers based on their similarity with input paper(s) [65, 100, 202]. While it is able to expose related papers that are similar by content, CBF models do not take into account user-paper interactions. CF models, on the other hand, utilize the user-paper interactions to generate recommendations, and can result in strong performance [41, 91, 169]. However, a common drawback of CF models is the cold start problem, which is severe in our academic recommendations when using real user-paper interaction data. Finally, there are hybrid models that combine CBF and CF models for paper recommendations [63, 212, 217]. The hybridization process is usually rule-based instead of learned: either the system first runs CBF models and then uses its output as input to run CF models to generate recommendations (cascade hybrid); or it simply mixes results that are separately generated from CBF and CF models (mixed hybrid).

Our work in this chapter differs from previous work on academic paper recommendation in that we study a rarely examined, but real scenario: generating paper recommendations given an ordered sequence as input. Specifically, we make recommendations for new users that sign up for the recommendations based on their browse history on the search engine. Compared to [224], which uses a simulated and artificial recommendation setting, our scenario concerns real user interactions with the recommender system. We have proposed a hybrid model that combines content similarities, that draws distinction between multiple aspects of paper contents, and behavior-based similarities. We have applied pointwise and pairwise learning approach to train the model, unlike the rule based approaches to generate paper recommendation that do not apply learning techniques [63, 212].

5.5.3 Citation recommendation

Citation recommendation is sometimes mixed with paper recommendation. Hence, we draw the distinction between our paper recommendation task and citation recommendation. We consider citation recommendation to be the task of recommending papers to an author who is writing a manuscript. A citation recommender may take a manuscript as input, identifies places where citations are needed, and recommends relevant citations [86, 201]. It may also take a piece of "context" as input, which is represented as a few sentences, and generates relevant citation suggestions [61, 94].

It is obvious that the citation recommendation task is mainly focused on similarity. Even when collaborative filtering is applied, it is using the citation relation matrix as a paper similarity measure [146], instead of using the user-paper rating matrix. The evaluation setup is also confined to predicting the cited papers of an input paper or paragraph.

Our work in this chapter differs from previous work on citation recommendations in terms of the methods we propose, the recommendation goal, and the evaluation setup.

5.5.4 Top-N recommendation

In the context of more general recommendation problems, our scenario is related to top-N recommendation [53]. Top-N recommender systems provide users with a ranked list of items based on predicted scores of individual items, where the relative ranking matters more than the absolute item scores. This is similar to our problem as we aim to produce a ranking of papers according to the predicted scores. However, the candidate set from which we make recommendations is different: we pick the papers from a recommendation email, while a typical top-N recommender selects from all items that have not been rated by users.

Top-N recommenders have been intensively studied [180]. In general, there are approaches that use latent space models [47] and approaches that rely on neighborhood-based models (whether user-based or item-based) [53]. While latent factor models can also generate top-N recommendations, they are originally designed for the rating prediction task. Therefore they are sub-optimal for top-N recommendation. Neighborhood-based methods identify similar users or items, and are shown to be more suitable for the top-N recommendation problem [6, 53, 104, 162]. Item-based methods have been shown to outperform user-based methods for the top-N recommendation task [43]. Similarity models have been recently proposed to improve item-based neighborhood models. They learn a coefficient matrix that is analogous to the item-item similarities [42, 104, 105, 162] directly from the data. A novel similarity model, Sparse Linear Method (SLIM), has been proposed by [162]. Several authors have proposed improvements to SLIM. Low-rankness has been investigated to capture transitive relations [42, 104, 105]. Kabbur et al. [104] proposed the Factored Item Similarity Model (FISM), which factorizes the coefficient matrix into two low-dimensional factor matrices. Cheng et al. [42] proposed the Low-rank Sparse Linear Method (LorSLIM), which introduces a rank regularization to SLIM. Kang and Cheng [105] made improvements over LorSLIM by providing a better proxy to approximate the rank of the coefficient matrix. Instead of estimating a single model for all users, Christakopoulou and Karypis [43] clustered users and estimated several local models. Zhao and Guo [232] minimized a combined heterogeneous loss function, which is a combination of pair-wise ranking loss and point-wise recovery loss. Wu et al. [223] generalized FISM from linear to non-linear by incorporating a denoising auto-encoder.

Our work in this chapter differs from previous work on top-N recommendation in important ways. First, directly applying top-N recommendation models to our task will lead to two problems: new users have no clicks on the recommendation emails, a situation that cannot be handled by existing top-N recommenders. Also, we have two types of interaction between users and items: user browses and user clicks. Existing top-N recommenders focus only on homogeneous interactions.

5.5.5 Reranking the output of a production system

Like us, Lefortier et al. [129], Moon et al. [157], Zoghi et al. [233] use a commercial search engine as their main baseline that they learn to improve. Moon et al. [157] and [129]'s methods directly use click-through rates, with a focus on documents that appear in the first position; both also focus on recency ranking and queries with shifting intent. Zoghi et al. [233] learn from a pairwise signal – out of order clicks in the top 5 produced by the production ranker.

Our work in this chapter differs from previous work on reranking the output of a production search engine or recommender system in that we do not restrict ourselves to recency ranking. Moreover, we do not work in an online setting and we do include content-based signals, not just behavior-based ones.

5.6 Discussions and conclusion

In this chapter, we have examined an interesting recommendation scenario for an academic search engine, namely, to rerank paper recommendations in email newsletters for newly signed up users. We have addressed this challenge by proposing a hybrid recommendation approach that includes a content component and a behavior component. The content component measures similarities of various paper aspects between users' browsed articles and recommendation candidates, and also considers the user's attention on paper aspects and on recent/historic browsing. The behavior component learns a mapping from browsed articles to user clicks in the recommendations. The model combines content and behavior through a pairwise learning approach that is based on user interaction data.

We have found that our hybrid reranking model HRM significantly improves over the production baseline. We have dug into the components of our model to see what works and what does not. In the content component, the graph embeddings work the best, especially the author similarity based on soft matching; users' recently browsed articles can lead to better recommendations compared to historic browsing; on the other hand, popularity and impact similarities are not sufficient to bring up good recommendations alone. In the behavior component, our learned scores combined with browsing similarity scores have led to better performance than the production baseline. The best performance is achieved when combining content and behavior through learning.

Our hybrid reranking model HRM can be seen as a module that can be plugged into a recommender system. Besides, we also have generalizable insights for other paper recommenders. For instance, we have revealed how each paper aspect contributes to the reranking performance.

A limitation of our study is that we have not performed online evaluations, such as A/B testing, to validate the model's effect on user engagement. Another limitation is due to the production dataset: our reranking is limited to the candidate articles generated by the production system. Therefore if the inputs are not of high quality, it will impact our final recommendation performance. In practice, HRM could be used to rerank a longer list of candidate recommendations, so that it effectively chooses the top 5 to be sent in an email. It would be interesting to also explore different methods of paper candidate generation, and examine how they impact the recommendation performance. Besides, if we can obtain user profile information (such as domain interests), can we apply topic modeling to provide more personalized recommendations for users? We leave these interesting questions as future work.

Part II

Enterprise emails

6 Characterizing Reading Time on Enterprise Emails

In the previous chapters we have studied key aspects of user interactions in academic search, including behavior analysis (query and download), download predictions and paper recommendations. While conducting academic searches is important in scientific research, also necessary is the communication that often involves reading emails [59, 198, 216]. Besides, users can read academic paper recommendations in emails. In this chapter, we study user interactions on enterprise emails and answer **RQ5**. We focus on one central aspect: how users read and spend time on enterprise emails. While the strict privacy policy does not allow us to dig into email contents to filter the academic communications and recommendations, we can obtain generic findings from a large set of users and many email reading events.

6.1 Introduction

Emails are one of the most important channels for communication [173]. Over the past two decades, the nature of web emails has significantly evolved and influenced user behavior accordingly [148]. Email usage has become much more diverse including task management, meeting coordination and personal archiving and retrieval. The high demand for intelligent email systems fostered related research in many areas such as email search [3, 35], information organizing with folders or tags [76, 117], and predicting user actions of replying or deleting [49, 54, 114, 226]. Although prior work has provided in-depth analyses and solutions for specific applications, the fundamental understanding of how users interact with email clients remains somewhat unclear. For example, questions such as how and when people read emails, how long they spend doing so, and what factors influence reading are not well understood.

The goal of this study is to characterize and present a comprehensive view on how users *read* their emails and how their *reading time* is affected by various contextual cues.

The reading activity is embedded in most user-email interactions across diverse applications ranging from retrieving information to automatic email prioritization. We

This chapter was published as [140].

argue that understanding email reading time lays the ground work for understanding user satisfaction, as it paves the way to estimating how a user's focus is spent. Capturing user reading time also helps reasoning about how email clients can be improved. Properly characterizing reading time, however, is a very challenging task. In today's environment, email clients are built with rich functionalities and using multi-devices by a single user is common. Even with access to large-scale logs, it requires careful examinations on data selection and interpretations to deliver meaningful analysis.

In this chapter, we provide a quantitative analysis of enterprise emails from the web and mobile clients of a popular email web service. We start by introducing a method to approximate reading time, that can be applied on millions of emails (Section 6.3). Then, we uncover how reading time is affected by various contextual factors in Section 6.4. We delve into temporal factors, user contextual factors and a very common reading behavior – rereading. To complement the results based on the log analysis, we also conduct a user study to look for the causes behind some interesting observations (Section 6.5).

Our findings indicate that reading behavior differs significantly on desktop versus on mobile devices. While the majority of emails are read within 10 seconds on both clients, the distribution of reading time on desktop exhibits a heavier tail than on mobile. Further, we find that desktop and mobile users have different temporal patterns: on desktop the reading time increases through the morning whereas on mobile it increases from the evening till midnight. Email types are also correlated with reading time: e.g. on restaurant and hotel related emails, users spend longer time during weekends compared to weekdays. The average time spent on reading emails is dependent on user status as well. For example, users spend less time reading when their calendar status is "out of office." They also read fewer emails within shorter time when they have more meetings or are busier in a day. We find different reading patterns in cross-device reading events: for instance, when users switch from mobile to desktop, email reading time tends to increase; when they switch vice versa, however, reading time tends to decrease. Last but not the least, our user study sheds light on on why certain behaviors occur.

To the best of our knowledge, this study is the first of its kind to uncover how users spend time reading emails through a large-scale analysis. The findings enrich the understanding of email reading behavior, and benefit research and applications in this field. For instance, correct interpretation of reading time would be essential for determining the importance of an email for email prioritization features[1], or its relevance in information seeking scenarios. Since reading time differs by contexts, the same amount of time spent on a human-authored email and a machine-generated email may mean different degrees of relevance.

6.2 Related work

A rich spectrum of studies have been conducted on users' interactions with email clients. In this section, we provide an overview of the most related work to our study.

Email overload and prioritization. Information overload was identified in the early years as one of the critical issues for email users [48, 221] and still is prominent in current email systems [77]. Beyond spam filtering techniques [51], Yoo et al. [228] focused on

[1]Examples include, Outlook Focused Inbox, or Gmail Priority Inbox.

modeling personal email prioritization to make more critical information surface to the users. Wainer et al. [215] examined how top-level cues such as message importance, subject line specificity, workload and personal utility influence users' attention to emails. Aberdeen et al. [1] introduced a scalable learning system to classify each email as important or not important for Gmail Priority Inbox, where the classification threshold is personalized per user.

Email search. In addition to the above proactive scenarios, users interact with and rely on search to retrieve relevant information or to organize emails. Kruschwitz et al. [120] demonstrate that email search is an essential part of the information seeking behavior in enterprises. Ai et al. [3] have examined the search behavior on email systems. They characterized multiple search strategies and intents, and identified important behavioral differences from web search such as re-finding. Horovitz et al. [90] proposed an auto-completion feature for email search, were suggestions are extracted from personal mailbox content in addition to query logs from similar users. Narang et al. [159] investigated general email activities and search activities. They found that search strategies are correlated with mail box properties as well as organizing strategies. Kim et al. [110] studied email search success by popping up an in-situ survey when a search session is finished to collect feedback. The results showed that generative Markov models can predict the session-level success of email search better than discriminative models. Along the line of searching personal information, Dumais et al. [58] examined in detail users' reusing behavior and established systems that assist users to find items such as emails and documents that users have seen before. Cecchinato et al. [40] investigated different finding strategies on desktop versus mobile devices, and work versus personal accounts via a diary study. Additional efforts [35–37, 122, 176] have also been laid on improving ordering accuracy for better search experiences.

Email interactions. Users tend to perform a variety of actions in email clients. Di Castro et al. [54] conducted large-scale log analysis for predicting users' actions of reading, replying, forwarding and deleting after receiving an email. Yang et al. [226] focused on predicting the reply action and studied the impact of factors such as email metadata, historical interaction features and temporal features. Dabbish et al. [49] studied the decision rules people choose to reply to email messages, or to save or delete them through a survey.

Folders and tags. Email systems not only provide a communication channel but users often manage their personal information by taking actions such as archiving, tagging or foldering. Earlier studies tackled the task of auto-foldering for individuals where the goal is to classify each email into a user defined folder [17, 56, 205]. More recently, Grbovic et al. [76] proposed to address the sparsity problem arising from the earlier personalized approaches by inferring common topics across a large number of users as target folders. Koren et al. [117] associated an appropriate semantic tag with a given email by leveraging user folders. Wendt et al. [220] proposed a hierarchical label propagation model to automatically classify machine generated emails.

Email intelligence. Current email clients aim to help users save time and increase productivity. Kannan et al. [106] investigated an end-to-end method for automatically generating short email responses as an effort to save users' keystrokes. Ailon et al. [5] proposed a method to automatically threading emails for better understanding using causality relationship. Email summarization [34, 158] has been studied as a promising

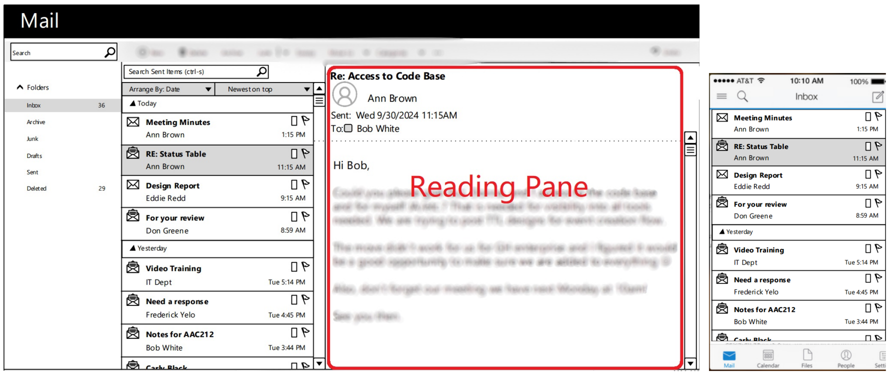


Figure 6.1: The web interface (left) and the mobile app interface (right) for our email clients. The reading time on desktop is computed with respect to the time each message appears in the reading pane (red box). The reading time is computed from the moment an email is clicked on until the the user hits back (available on mobile only), clicks on the next listed email (available desktop only), switches to compose mode by clicking on reply/forward, or closes the app (or browser).

way to solve the problem of accessing an increasing number of emails possibly on small mobile devices.

While prior work studied extensively from different perspectives how users interact with email systems, their focuses were centered around specific scenarios such as search. The goal of this chapter is to present a horizontal, generic view on users' interactions with emails in terms of reading, which is the primary action users take regardless of which application they are currently using. Not only do we study in detail the relations between reading time and a variety of properties, but we contrast the reading behavior on desktop and mobile devices over a large number of real users.

In their highly cited work on *Theory of Reading*, Just and Carpenter [103] argue that reading time depends on text, topic and the user familiarity with both. Almost four decades later, we reassess some aspects of their theory on user interactions with modern emails.

6.3 Measuring reading time

Measuring reading time accurately is challenging. Eye-tracking tools can be used to track the users' gaze, but deploying them over large numbers of users is non-trivial due to privacy concerns, costs and technical limitations around calibration. We rely on user interaction logs of a large commercial email provider to study the reading time indirectly by measuring the time between opening and closing an email. Relying on interaction logs allows us to test our hypotheses over large sets of users at reasonable costs and with minimal intrusion. However, our data-driven approach is limited to what is already captured in the logs, and is not free of issues. For instance, people might be multi-tasking – they might have the email opened but are focusing on a different task in a different window. Furthermore, a logged *open* action on an email followed by a logged *close* action does not always imply that the email is *read* (e.g., the user might be triaging emails quickly, deleting emails as soon as they are displayed on screen).

In our analysis, we use the best possible signals in the logs to get a close approximation of the reading time. We define reading time as the duration between the two paired signals – the start of email reading pane which loads the content of an email into the reading zone and the end of email reading pane which records the closing of that pane, as it forms a consecutive *reading event*. To minimize potential impacts caused by the above issues, we ignore samples with reading time shorter than one second. Reading events on threads (20.5%) are removed since they are more conversational in nature and complex to track.[2] We also only study users who read at least one email per weekday so as to focus on normal traffic and avoid random noises.

Data Our experimental data is sampled from enterprise emails over a two-week period from May 6th to May 20th 2017. We enforce the above filtering rules when collecting the data. Beyond this, we sample the data randomly to minimize potential biases towards specific demographics or enterprises. For simplicity, we refer to this dataset as desktop client dataset. In total, this sample contains 1,065,192 users, 69,625,386 unique emails[3] and 141,013,412 reading events (i.e., an average of 132 reading events per person) from tens of thousands of enterprises. From this set, we further select users who also use the iOS app over the same period and collect their corresponding usage from the mobile logs, which is referred to as the mobile dataset. This gives us 83,002 users with 5,911,107 unique emails and 10,267,188 reading events (an average of 124 reading events per user). By collecting email usage patterns from both desktop and mobile clients, we are able to study in-depth cross-device reading behavior. In addition to the two-week window of data, we also collect another two-week period data prior to this period from the same set of users. This "history" data is used to capture rereading behavior if any.

Desktop (web) client An anonymized version of the user interface of the web email client is shown in Figure 6.1 (left). The interface supports users to manage their emails effectively on web browsers. We find that nearly all the usage data logged from this portal comes from desktop/laptop users, which is why we refer to it as desktop client throughout the chapter. On mobile phones, people tend to use a mobile email client (app), as described later. To read an email on our desktop client, users have to first select it from the email list by clicking on it.

Once an email is selected from the list, its corresponding content will show up instantly on the *reading pane* on the right side of the email list. As mentioned, we use the time gap between when a message appears in the reading pane and when it is replaced with another message to approximate reading time.

As a sanity check, we validate this method by first performing various actions on the client by ourselves and video-record everything, and then checking the corresponding logs recorded by the system. We find that for majority occasions our email reading time

[2]The dwell time on each email of a thread is dependent on the size of the screen, scrolled position of the pane and other factors.

[3]One email that is sent to multiple recipients is counted as one.

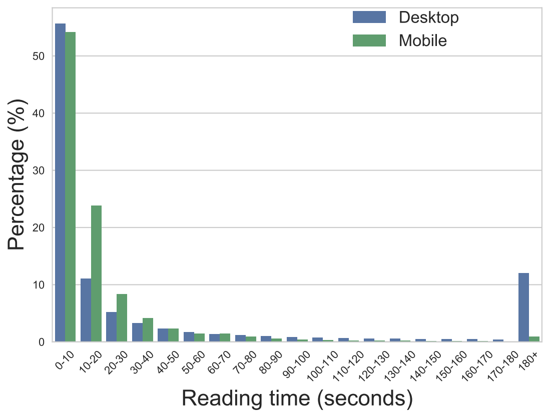


Figure 6.2: Reading time distributions in desktop and mobile platform. Time is binned by every 10 seconds.

can be reflected by the time gap between reading pane's opening and closing. However, when we quickly navigate through emails in the email list by pressing arrow keys or clicking, a very short reading time (such as hundreds of milliseconds, but no more than one second) is recorded by the system. Given the very short time, we assume that it is unlikely for other users to read the email as well. Therefore we set a one second threshold on the reading time in order to filter out these unlikely reading events.

Mobile client (app) The right screen-shot in Figure 6.1 depicts the user interface on the iOS mobile application which is the source of the mobile logs. Users can click into an email by tapping on an email snippet in the list display. A reading pane with the email content will show up that fills the display area of the application. The reading time is the interval between tapping an email and hitting the exit/back button or quitting the app.

Due to data sensitivity, normalized reading time is used in some analyses instead of absolute time. Time is turned to logarithmic form and then min-max normalized to avoid showing absolute time.

6.4 Reading time and contextual factors

In this section we first provide a brief overview of reading time, then delve into various contextual factors that impact reading time.

6.4.1 Reading time overview

The overall distributions of reading time on desktop and mobile are presented in Figure 6.2. In both datasets, more than half of the reading events happen in less than 10 seconds (55.6% on desktop vs. 54.2% on mobile). On mobile, about 25% of emails are read in 10-20 seconds. In comparison on desktop, only about 11% of emails fall into that range.

Interestingly, the reading time distribution on desktop has a much longer tail compared to mobile. On desktop 12.0% of reading events last longer than 180 seconds (3 minutes) which we suspect can cover many cases where the users leave an email opened on the screen while paying attention to something else – potentially not even being at their desk. The longer tail can also be explained by the fact that spending longer time on reading could be relatively easier on larger desktop screens.

Using proprietary email classifiers, we can put email into various categories. Not surprisingly, we find that human emails have longer reading time than robot emails, with promotional and spam emails having the shortest reading time.[4]

6.4.2 Temporal factors

In this section we study how reading time is affected by various temporal factors. To begin with, we investigate how the average time users spend reading emails varies depending on the time of day and the day of the week. Figure 6.3 illustrates the average email reading time in different hours of the day.[5] It can be seen that average time spent reading emails on desktops increases through morning time and peaks around noon, and then decreases through the afternoon and the evening. However, the reading time on mobile is drastically different and increases from around 7PM up until 2AM next day, while it decreases through most of the afternoon.

Moving on, we find that for both desktop and mobile, reading time on weekdays is higher than that on weekends. On both datasets, the weekday pattern is fairly stable with minor changes; for desktop on weekdays, reading time is the lowest on Monday and highest on Friday. On mobile, the reading time slightly peaks on Wednesday and is the lowest on Monday. We omit the visual presentation of details for brevity.

We also compare the type of emails that are typically received and read by users between weekends and weekdays, in order to find out if there is any difference among email types. The green bars in Figure 6.4 represent the magnitude of change in percentages among emails received in each category, and are computed by dividing the number of weekend emails by those received during weekdays. For instance a roughly 100% increase in the number of hotel-delivered emails suggests that people are almost twice as likely to receive such emails over the weekends. We also compare, how often emails from different categories are read on weekends versus weekdays. The blue bars in Figure 6.4 – computed in a similar fashion but based on read statistics – confirm that

[4]Our classifiers follow a semantic taxonomy where emails are first grouped into those sent by human (*human*), those sent by machines (*robot*) and *spam*. Human and robot classifiers are exclusive and exhaustive, while the spam classifier is independently built to output confidence scores indicating how likely an email is deemed as spam. Next, for emails that are classified as robot, ngram-based classifiers are established for identifying different intents from the emails. These include classifiers that identify travel information (*hotel*, *car rental*, *flight*), classifiers that identity reservations for food (*restaurant*) or concerts/festivals (*event*), classifiers that identify your purchases (*receipt*) with tracking information (*parcel*), and finally any coupon codes if available (*promotion*). For human emails, a rule-based classifier that identifies intents asking for meetings (*meeting*) is also included for our analysis. In total, 96.1% of the emails have been classified by the system, which provides us a representative sample for comparing the distribution of email types.

[5]We calibrate the calculation according to users' local time zones.

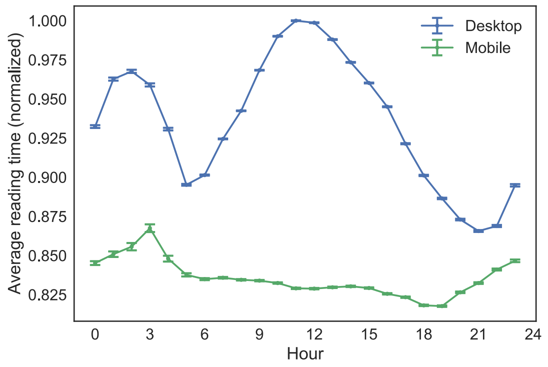


Figure 6.3: Average email reading time across different hours on desktop (top) and mobile (bottom).

the types of emails read by the users over the weekend are mostly consistent with what they receive. However, they also highlight a few exceptions where reading rates deviate from what would be expected based on delivery statistics. For instance, hotel-related emails are almost 3.5 times more likely to be read during weekends, despite the fact the number of hotel-related emails delivered only grows by a factor of 2. By contrast, spam and promotional emails are substantially less likely to be read on weekends versus weekdays.

As a brief summary, through the analyses on temporal factors, we find that the temporal pattern of reading time is not only correlated with the hour and day of the week, but also devices and certain email categories.

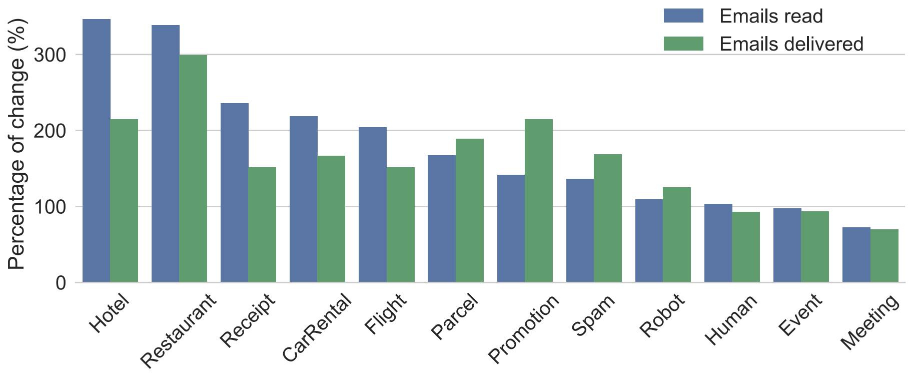


Figure 6.4: The percentage of changes per email type on weekends vs. weekdays. The bars are computed by dividing the frequency of emails received/read in each category over the weekend by the weekday traffic (times 100).

6.4.3 User contextual factors

In this section, we investigate how user contextual factors could potentially affect the reading time. Specifically, we examine calendar status, fatigue and user device.

Calendar status While we do not have direct access to the user status of our users, we can use their calendar status – which can be mined from the calendar app associated with their email client – as proxy to hypothesize about their user status. The status classes include tentative, busy, free, elsewhere (working elsewhere), and OOF (out of office). Note that an empty status means nothing is on the calendar in that period, while "free" is a status that a user explicitly puts on the calendar and hence it suggests "availability". The average reading time provided in Figure 6.5 shows that the reading patterns can be affected by the *calendar pressure* of the user in both platforms. On desktop, reading time is the longest when there is nothing on the calendar and the user is free. On mobile, the peak occurs at working elsewhere,[6], which is the second lowest on desktop. Users tend to spend more time on an email on average, when they have nothing specific on their calendar. They spend the shortest time when they are out of office, perhaps reading emails fast enough mainly to cherry pick the key points.

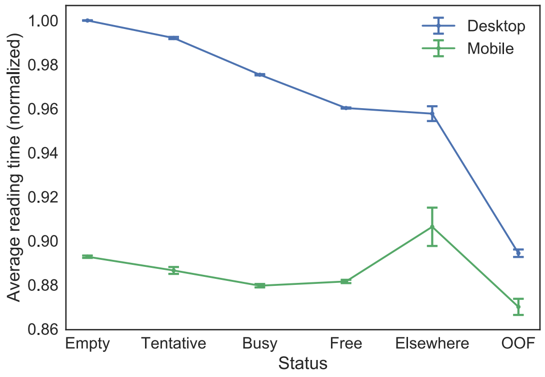


Figure 6.5: Reading time by different calendar status on desktop (top) and mobile (bottom).

In Figure 6.6, we consider the number of daily meetings, and the number of busy hours in the day as proxies for cognitive load of our desktop users. This is inspired by previous work by Barley et al. [13] that reported time spent at meetings as a source of stress at work. It turns out that more frequent meetings, and a larger number of busy hours in the day, are indeed correlated with observing fewer email reading events overall. That is, busy users read fewer emails, and go through those faster than average. We observed similar trends for our mobile users and hence exclude more details for brevity.

[6]We do not have a strong explanation for long reading times on mobile when the user status is *elsewhere*. The high variance (indicated by large error bars) suggests that this is not a frequent/consistent event. We can only conjecture that the spike might have been caused by users that had to read the emails they normally read on desktop at work, on their mobile devices.

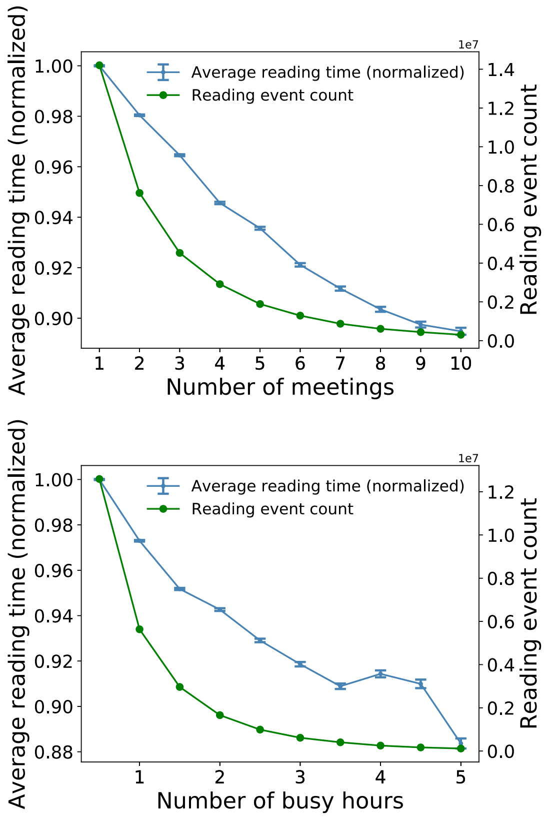


Figure 6.6: Reading time of desktop users by the number of meetings (top) and busy hours (bottom) in a day.

Reading fatigue Psychological research has shown that fatigue after mental work (e.g., proof reading) leads to a performance drop, such as reading speed and reaction time [2]. But how does fatigue impact reading time in the email setting?

We use the accumulated time spent on reading emails in the past two hours of user activities as a proxy for measuring fatigue. The longer the accumulated time, the more we expect the user to be affected by fatigue. For each email reading event, we sum the accumulated reading time of the user in the past two hours prior to that event, and group these sums with a bin size of 10 minutes. The results can be found in Figure 6.7. As accumulated reading time increases up until 60 minutes, the average reading time constantly grows. After that point the reading time does not change much. Although we cannot draw strong conclusions based on these observations in the absence of more information about the users, these trends *may* suggest that fatigue prolongs reading time, and the effect is only up to a certain extent. Overall, our findings are consistent with the reported effect of fatigue in email settings by Ahsberg et al. [2].

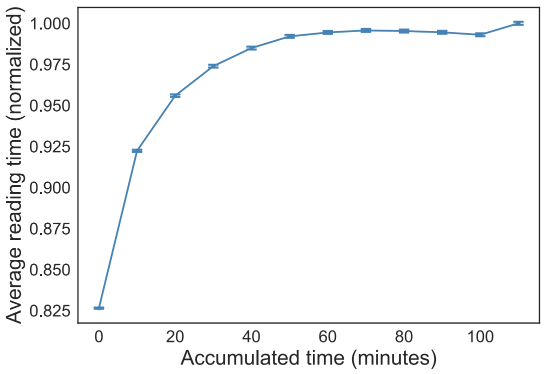


Figure 6.7: Average reading time conditional on the time spent on reading emails during the past 2 hours of user activity (proxy for fatigue).

Table 6.1: Reading time on different devices, ordered by device screen size.

Screen size (inch)	**Average reading time (normalized)**	**Sample device**	**Percentage**
4.0	1	iPhone SE	14.3%
4.7	0.89	iPhone 7	57.2%
5.5	0.89	iPhone 7 Plus	23.9%
7.9	0.90	iPad mini 4	0.9%
9.7	0.82	iPad Pro	3.3%

User device We confine the scope of this part of study to mobile users because the device type information – specifically, screen size details – is only available to us in our mobile logs. Mobile devices have different screen sizes, which we hypothesize could impact the reading experience. In Table 6.1, we group devices by their screen size and present their average reading time per email. Users on the smallest screen devices spend the longest reading time. This may be explained by the limited contents displayed on a small screen, which demands more efforts (scrolling, zooming) to read. The 9.7in iPads which have the largest screen size have the lowest reading time across all devices. Overall, the reading time is negatively correlated with screen size.

6.4.4 Reading and rereading

In this section we investigate the reading time of emails that have been read at least once before. We find that 33% of unique email reading actions are actually *rereads*, a significant portion that may seem surprising at first glance. However, as an interesting reference point, Teevan et al. [209] reported that about 38.8% of all web search queries are re-finding, which further underlines the scope of re-finding activities beyond email. It is worth noting that unlike Section 6.3, here if one email has say three recipients, it is considered as three unique emails for the purpose of computing reread statistics.

Figure 6.8 shows the distribution of reread counts for reread emails. We observe that in 58.4% of the cases emails are reread once (read twice in total), while the majority of reread emails (95.7%) are reread no more than 5 times.

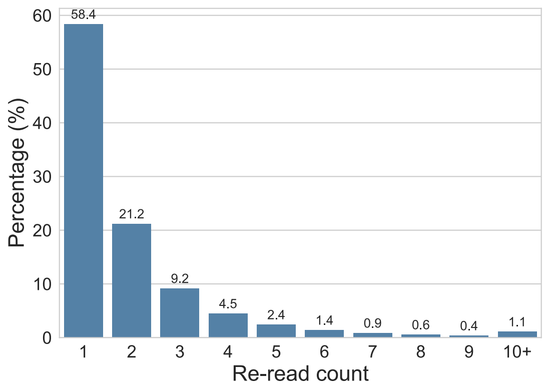


Figure 6.8: The distribution of email reread actions.

Given the high frequency of rereading behavior observed in the logs, we are encouraged to extend our investigation further to compare reread actions for different email types, study the impact of previous reread counts on reading time, and analyze rereading cross-device.

Reread action across email types The results in Figure 6.9 reveal that certain categories of emails (e.g. hotel, car rental and flights) are more likely to be reread. This may be explained by the way users deal with travel related emails, e.g. they may read it upon first delivery, but when they check in at the hotel or catch the flight they need to read it again for information. Human emails have a much higher rereading percentage (37.4%) versus robot emails (26.7%). While spam and promotion have the lowest rereading percentage (19.1% and 15.0%), they represent a noticeable portion in the emails that are read. One possible explanation is that during email triaging some users read spam or promotion, and they may flag or move some important emails to their inbox. Revisiting those later will be seen as rereading.

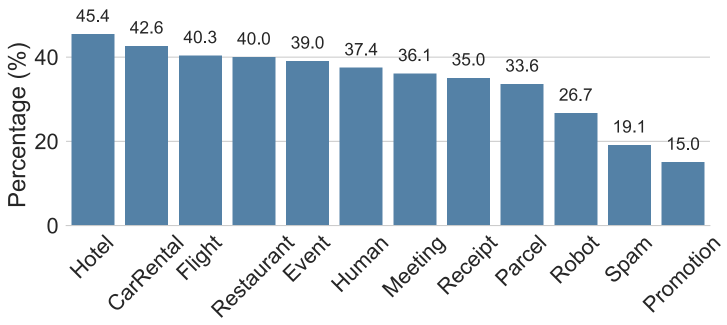


Figure 6.9: The percentage of emails that are reread across different classes of emails.

Reread count vs. reading time Earlier in Figure 6.8 we described that 41.6% of reread emails are reread more than once. How does reading time change, as users read certain emails over and over again? In Figure 6.10, we look at how reading time changes by the reread count. We group the reread emails by the maximum number of times they have been reread from three to five. We observe that as emails are reread more often, users spend less time on reading when they have to go through them again, which probably can be explained by their increasing familiarity with the content. Another interesting finding is that emails that are reread 5 times are read more quickly than those reread 4 times, and those reread 4 times are read more quickly than those reread 3 times.

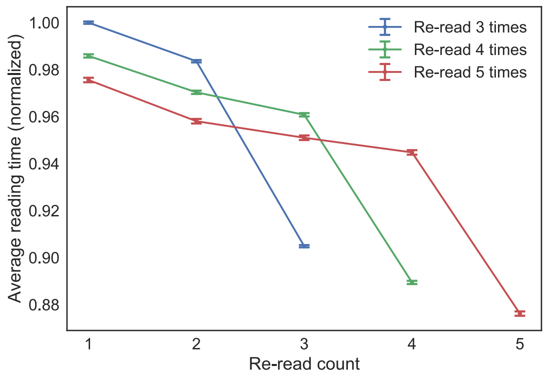


Figure 6.10: Average reading time of reread emails for different number of previous reads.

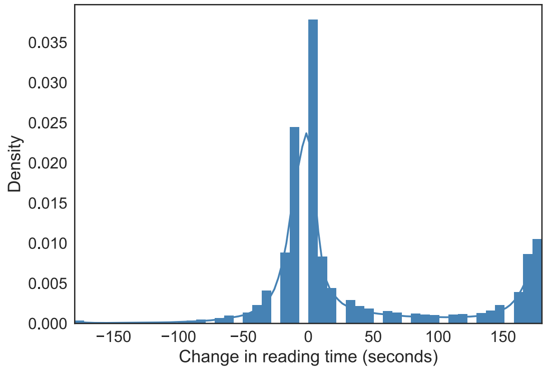
Cross-device rereading With the increasing popularity of smart mobile devices, users can now easily switch to their mobile device to handle emails when they are away from their desktop. In this section we focus on emails that are reread across devices. We only include emails that are received during our sampling period and opened on more than one device. In total we have 587,953 emails and 67,440 users. Specifically, we are interested in cases where users read a new email on mobile first then switch to desktop to read it again, and vice versa.

One prominent characteristic of cross-device reading events is that 75.6% of the emails are first read on mobile before being read on desktop. On the contrary, only 24.4% of emails are first read on desktop, followed by mobile. One reason for this imbalance could be due to the convenient access to mobile devices, that enable users to get to their emails more easily and regularly. Another reason, could be that access to more information and easier typing on desktop encourages people to continue and finish the tasks they initially started on mobile, on desktop.

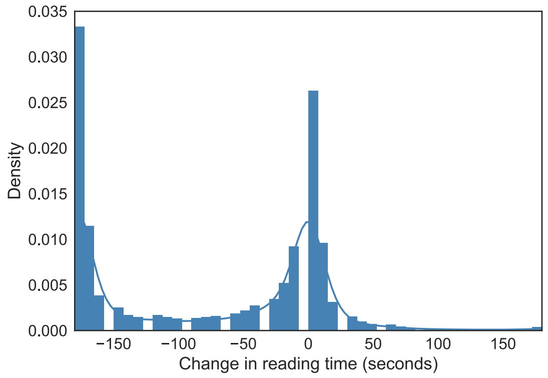
We also notice that when users switch from desktop to mobile, 29.5% of the times the subsequent reading events happen within 30 minutes, while from mobile to desktop the percentage is much lower at 16.7%.

Finally, we explore how reading time changes after a user switches from one client to another. To this end, we measure how the reading time of an email first opened on mobile changes when the user opens it again on desktop and vice versa. Figure 6.11

demonstrates the histogram of changes[7] in reading time for mobile to desktop switches on top, and for desktop to mobile switches at the bottom. On both histograms, there is a large peak around 0, that suggests users spend roughly similar time when reading the same email across different platforms. The right peak on the top histogram represents a large set of emails that are opened for the first time on mobile, and re-opened later on desktop with significantly longer reading time. This set is likely to include emails that the user has glanced through on mobile, but left to fully address on desktop where it is easier to type and access information. The left peak on the bottom histogram includes another interesting set of emails. These are emails that are opened on desktop for the first time and are reread on a mobile device, but with much shorter reading time on average. Perhaps these are emails that the users revisited for quick fact checks and referencing on mobile, when they had no access to their desktop client.



(a) Change in reading time from mobile to desktop.



(b) Change in reading time from desktop to mobile.

Figure 6.11: Change in reading time when the user rereads the same email across platforms.

[7]That is, reading time measured on the second client, minus the reading time recorded on the first client for the same email (in seconds).

6.5 User study

User studies can provide more information about "why" we see certain behaviors in the logs. Through a brief user study, we obtain qualitative explanations for some interesting observations, namely, what lead to long reading time on an email, why users conduct rereading, how and why users read emails across devices.

6.5.1 Methodology

Our user study consists of two phases: 1) screen recording; and 2) interview. We first record a one hour video on the user's computer screen, without interrupting the user's normal activities during work hours. The screen recording is one of the least intrusive ways to help us observe natural behavior on their working computer.

Then we conduct the interview a few hours later. We avoid interviewing immediately after recording so that the user will not hurry their work during the recording time. A one-on-one interview is conducted by playing back the recorded video and asking questions. The video helps the user to remember what he/she was doing during the recording. The interviewer also looks at the screen if permission is granted by the participant. Otherwise the interviewer sits back from a distance to avoid reading the contents on the screen. Specifically, the interview starts with general questions such as user's habits of reading emails, then moves into detailed questions on the user's interaction with emails during the recording. In total, 15 participants from an IT enterprise took part in this user study, including 9 men and 6 women. The demographics also include people at different ages (from 20s to 60s), and at different job positions (3 interns, 4 junior employees, 5 senior employees, and 3 senior managers). Afterwards, participants are awarded a 25 dollar coupon for online shopping.

6.5.2 Findings

Long reading time. In our sample, no one closes the email client during their work hours. All participants either keep it opened or minimized in the background. This can lead to excessive long reading time being recorded for the last opened email in a session, which partially explains the heavy tail for desktop reading time (the reading time longer than 3 minutes) in Figure 6.2. Another reason that may explain the inflated reading time is participants' multi-tasking. Multitasking is frequent for the majority of participants and happens when it is needed to refer to the email contents to complete another task. However, neither of these 2 situations applies to mobile because users would usually close the mobile app after they use it, and they do not multi-task while reading emails on mobiles. This helps explain the much smaller tail of mobile reading time than that of desktop.

Rereading. Rereading is common for all participants. It takes place either after an email search, or simply by browsing through emails that are read. We find two cases that frequently lead to rereading. The first is email triaging, especially for the senior employees and managers who may receive too many emails to finish reading at once. In this case, participants would flag emails or move them to certain folders after a quick skim, then read the emails again some time later. Another case is for difficult or long emails, that participants need to take several reading attempts to digest the content. This also includes the scenario when they first read the email on mobile, and continue processing it on desktop.

Cross-device reading. 40% of the participants report the use of the mobile client for their work email. They report using the mobile client only when they are away from the working environment, for instance morning at home, during transit or away for coffee breaks. For heavier reading tasks, participants would switch from the mobile client to the desktop. This could explain the reading time increase from mobile to desktop. In summary, the mobile client only serves as a complementary platform to the desktop client.

6.6 Discussion and conclusions

This chapter characterized in depth how people read their enterprise emails on desktop and mobile devices. We acknowledge that a limitation of this study is that direct applications are not provided, as the chapter focuses on observational insights. However, the rich findings can open up directions for possible applications in email system design, as well as fostering research in email systems.

Adaptable reading pane. Email types and lengths affect reading time substantially. For instance, we found that people tend to spend more time reading human-authored emails, while ignoring spam or promotions. This may suggest, from users' perspectives, loading an entire promotion email to the reading pane is unnecessary, and the saved space could be utilized to support other "smart" options such as one-click unsubscribe.

Contextual inbox. People tend to be more active reading on desktops during morning and noon hours, whereas on mobile devices reading time increases from evenings to midnights. As expected, our temporal analysis suggests that attention is more focused on work-related communications on weekdays, and on travel activities during the weekends. These findings can help us build a contextual inbox. For instance, reducing pop-up notifications for receipt confirmation emails can potentially help users stay focused in a meeting. Likewise, for important/urgent emails that are delivered at night, auto-replies to senders and reminders to recipients' mobile phones may help reduce tension and response latency.

Email assistant. We find that people with busier calendar schedules may read fewer emails and process those faster than average. Similar to the fatigue effect identified in psychological research conducted by [2], we find that the longer accumulated time users spend reading in the past two hours, the slower they may become in terms of processing new emails. In such cases, email clients can track down things that need to be completed, highlight items that are skipped due to a lack of concentration time, or even auto-complete them (e.g., schedule a follow-up meeting per discussed), which may alleviate users' burdens from busy days.

Cross-platform rereading support. Users also reread emails across platforms, where 76% of cross-platform rereadings happen first on mobile then on desktop, and 24% vice versa. For the former case, users tend to continue heavy tasks on desktop. The system can assist users' rereading activity by remembering the last-read position and help them continue processing the email. For the latter, since users spend significantly less time when rereading on mobile (e.g. fact checking), summarization and highlighting email contents would save user efforts and improve their efficiency.

Our log analysis has painted a rich picture of reading time on emails in general. A user study in an IT enterprise also served as a sanity check for the observations. Further, it would be interesting to investigate how the nature of the business affects the email reading behavior (e.g., a production-based company will possibly be very different to a government organization). Although this was not covered in the log analysis due to privacy protection on user identity (we do not have access to email addresses), conducting pop-up surveys as in [110] can provide large-scale supportive evidence that helps complement our log analysis and user study. We also discussed several ways how these findings could be used. The action “reading” is shared across different email-related scenarios. If we understand reading time for a user query and the corresponding search success, can we infer and adjust our understanding of, for example, user reading time on an auto-generated reminder or meeting invitation? We leave these interesting questions to our future work.

7 Conclusions

In this thesis we have examined user interaction behavior with different types of information objects, specifically with academic papers and enterprise emails. In Chapter 2 we have provided a characterization of academic search queries, analyzed the query failure phenomenon, and proposed a query suggestion approach to remedy the failures. In Chapter 3 we have studied query reformulation behavior and topic shifts in academic search, identified their trends over a long period of time, and revealed their correlations. In Chapter 4 we have studied download behavior in academic search, and addressed the task of predicting the next download. In Chapter 5 we have tackled the task of reranking paper recommendations from a production recommender system, using both content and behavioral information. And finally, in Chapter 6 we have studied user reading behavior on enterprise emails, with a focus on how users spend time on reading.

In this chapter, we list the main findings in our research, discuss their implications and limitations, and point out future directions. Section 7.1 summarizes the answers to our research questions. Section 7.2 describes limitations of our work and suggests directions for future work.

7.1 Main findings

7.1.1 Academic search and null queries

Our research started with a study on academic search. We proposed our research question aimed at understanding how users conduct academic search:

RQ1 What are the characteristics of queries and failures in academic search, and how do we remedy failures?

Answering this question has given rise to a characterization of academic search queries. We have studied a transaction log from ScienceDirect that contains the search log of many users during a long period of time. As a result we have obtained results that are representative for academic search. At a first glance, academic search queries are found to be more verbose than general web search queries, and their query length distribution features a heavier tail. The relatively long queries can be challenging for a search engine to address the information needs embedded in them due to the query complexity. Besides, long queries lead to the query sparsity problem, i.e., queries do not repeat often in the transaction log. This makes it difficult for a search engine to learn from previous user interactions with the same queries.

We have also identified counterparts of certain query types in general web search, including navigational queries, transactional queries and informational queries. We notice that the boundaries do not appear to be very strict in academic search, for example, between navigational and informational queries: users may enter a few keywords in a query with the sole purpose of downloading a specific paper (navigational), or the same query may be used for searching information around a topic (informational). The implication is that to make an accurate classification of query types in academic search, it is necessary to consider not only the query itself, but also the search contexts such as query reformulations and clicks. We have also found that the richness of entity queries in the log, which may contain key concepts in papers, author names etc., can help to address retrieval tasks.

After obtaining a characterization of academic search queries, we have zoomed in on search failures, namely on so-called null queries. We have found a surprisingly high occurrence rate of null queries in our academic search scenario compared to general web search. We have studied the null query phenomenon from three angles: queries, sessions and users. We have found that null queries are even longer than normal queries in academic search, and there are more boolean operator queries in them. This indicates that query complexity correlates with the occurrences of null queries. We have examined the contexts where null queries happen, namely null sessions. We have discovered that null queries can occur in seven types of null sessions such as refining, generalizing and exploring sessions, where users conduct searches in different contexts or with different goals. We have also examined users who frequently fail and those who do not. We have found a medium correlation between a user's self-similarity score and the null query frequency, which suggests that users who have more consistent search interests tend to issue more null queries.

After having conducted our observational study on queries and failures in academic search, we have proposed an algorithmic fix for the null query problem. We have used the obtained insights to create a query suggestion method that makes query recommendations when null queries occur. We have proposed to use entity information in our modified query entity graph (mQEG). The mQEG is constructed using entities in queries and query transitions in sessions to surface relevant query suggestion candidates. On top of this, we have proposed a session-conditional approach that adjusts the ranking of query suggestion candidates generated by mQEG to be based on the predicted null session type. We have proposed to use a set of query features, query transition features and click features for the prediction, and cast the query suggestion problem as a multi-label classification problem. We have then used the output of different probabilities of the session labels in our algorithm to produce a reranked list, which achieves significant improvements over the baseline query suggestion method without using session information. We have also discussed a personalized query suggestion approach that is based on user preferences of entities in queries. There is a slight improvement in performance but it is not statistically significant.

The implications from the experiments aimed at answering RQ3 are that entity and session information can help bring up quality query suggestions for null queries. In contrast, personalization based on entity interests may not always help, e.g., when users shift interests away from their historical interests.

7.1.2 Query reformulation and topic shift

The above findings are mostly focused on user behavior in search sessions. Moving on, we have also studied long term user interaction behavior in academic search, namely their query reformulation and topic shift. Query reformulations concern ways of composing queries, a type of explicit behavior that can be measured, while topic shifts are on the implicit side of user behavior. We have answered the following research question:

RQ2 Do topic shift and query reformulation patterns correlate in academic search?

We have provided the first query reformulation taxonomy in academic search, that is built on query logs collected from many users over a long period of time. Our taxonomy includes adding terms, dropping terms, substituting terms, revisiting and new queries. We have found the most frequent reformulation type to be revisiting, i.e., entering a query that is already in the user's search history. This reformulation type accounts for nearly one third of all reformulation types. Comparing different query reformulation tendencies over the course of time, we have also found that revisiting queries tend to happen more often than others. The prevalent revisiting occurrences indicate a certain degree of recurrent information need of academic searchers, e.g., constantly looking for new papers on the same topic or simply revisiting old papers.

We already know that in general web search, revisiting is an important aspect of user behavior [209]. Through our study in the domain of academic search, we have found a major portion of the query traffic to be revisiting queries. The implication for modern academic search engines is that assistance on these revisiting queries should be provided, which can either be realized by helping users refind old information or by helping them to discover new information around recurring topic interests, depending on the user intent.

Besides query reformulation, we have examined users' topical interest over time, and how this is correlated with query reformulation. Our findings reveal topic shift trends in general: we have found that nearly half the users tend to have increasing topic shifts over time, meaning that their interests are diversifying to a certain extent; the other half have decreasing topic shifts, i.e., their interests are gradually focusing. It is hard to find out why without conducting interviews with users, which is costly. However, it is reasonable to speculate that users may be at different stages of their research and this phenomenon affects their information needs: new academic searchers may be exploring different topics and have very diversified interests; while senior searchers may be more focused on their domain. We have also found that bigger topic shifts tend to happen over longer gaps between searches than shorter gaps.

Finally, we have tried to correlate topic shift trends with query reformulation preferences of users. To our surprise, we have found very little correlation over a long period, meaning that regardless of their query reformulation strategies, users could be equally likely to be focusing or diversifying during their searches. Looking at a shorter period, we have found that certain query reformulations may be correlated with immediate topic shifts, such as submitting new queries. The implication is that features derived from query reformulations can help predict the magnitude of immediate topic shifts.

7.1.3 Download prediction

Besides queries, we have studied download behavior in academic search. We consider a download to be the retrieval of a PDF paper file. We started with a characterization of download behavior, and then conducted predictions on download behavior. We have answered the following research question:

RQ3 What are the characteristics of user download behavior in academic search and how can we predict that?

We have identified the characteristics of user actions in a download session, and the action trajectories that lead to a download. We have found that the most frequent trajectory is a query leading to a download, and that clicks are much less frequent in download sessions. Looking beyond sessions, we have found that certain behavioral factors are correlated across sessions. For instance, a positive correlation between download numbers of successive sessions shows a consistency of download intensity. On the other hand, the time gap until the next download session is negatively correlated with the number of queries in the current session: the more queries in the current session, the sooner the next download session might occur. We have also proposed an approach to use downloads to represent user's topical interests. We have found a bias in the topics of the downloads. Moreover, we classify users into different groups based on their topical interests, and have found differences among users with different topical interests, such as behavioral variances in terms of download diversity and coherence. Another interesting finding is that many users download papers across disciplines, which is a reflection of recent findings on the increasingly interdisciplinary nature of scientific research.

Furthermore, we have utilized some of the observations of user downloads for two prediction tasks. We have addressed two prediction tasks: predicting the time gap until the next paper download session and predicting the number of downloads in the next download session. These tasks help to make a paper recommender system pre-emptive in terms of their timing of sending recommendations, and alleviate the information overload problem for users, respectively. We have proposed a model based on Long Short-Term Memory (LSTM) that utilizes user action history, and that is based on user segmentations by Dynamic Time Warping (DTW), which has produced reasonably good prediction performances. Additionally, we find the topic feature beneficial for predictions. Our prediction tasks of the time gap and the number of downloads are general, so that they are applicable to any paper recommender system.

7.1.4 Reranking paper recommendations

We have examined another recommendation scenario, where users on an academic search engine sign up for an email newsletter service. The emails contain paper recommendations based on user browsing on the academic search engine. We have dealt with the task of making better paper recommendations for new users by reranking paper candidates generated from the production recommender system. In doing so, we have answered the following research question:

RQ4 How do we utilize both content and behavior to rerank paper recommendations?

The primary challenge in our setting is the absence of user interactions with the recommender system, when users sign up for the email newsletters. Therefore, we have addressed this challenge by proposing a hybrid model that utilizes both paper content and user behavior. For the paper content component, we have modeled various aspects from the paper metadata, and have proposed similarity measures for these aspects. We have found certain aspects to be useful for making better recommendations, such as entity embeddings of papers, and hence the embedding similarities between papers browsed and recommendation candidates. For the behavioral component, we have utilized user browsing on the search engine, which are less sparse, and user clicks, by learning a mapping function from browsing to clicks on recommendations.

Our hybrid reranking model combines both behavior and content and is trained using a pairwise learning approach based on real user interaction data. The outputs are reranked paper recommendations that are significantly better than the production system in terms of getting user clicks. We have found that both content and behavior contribute to better recommendations. We have also detailed individual contributions from different paper aspects and components of the model. Our reranking model can be applied to production recommender systems as a separate module.

7.1.5 Email reading time

Finally, we have examined another type of user interactions – email reading. Emails are an essential part of modern communications, for instance, in academic cooperations emails are universally used to discuss research [59, 198, 216]. Along with the popularity of recommender systems, emails are also used as the carrier of recommendations to users. The core of email interaction behavior, namely email reading, has not been well studied in the literature. Therefore, we have answered the following research question:

RQ5 How do users *read* their emails and how their *reading time* is affected by various contextual factors?

We have proposed a methodology to approximate reading time based on user interactions with email clients. Then we have studied the email reading time based on a very large enterprise email dataset. We have uncovered many contextual factors that are correlated with reading time, including temporal factors, user contextual factors, and re-reading behavior. We have obtained interesting findings in each category, for example: reading time distribution on desktop devices has a much heavier tail than that on mobile, which can be explained by possible multitasking in the desktop environment; users have different temporal patterns of reading emails depending on the devices; users spend more time on certain emails during weekends compared to weekdays, such as restaurant and hotel related emails; users spend less time reading emails when they are busy; the majority of cross-device reading events happen when switching from mobile to desktop, where reading time tends to increase after switching devices.

A user study has also revealed the reasons behind certain findings. Overall, we have characterized email reading time, and found its connections with various factors. The findings have enriched our understanding of email reading time, and may help us to come up with ways to optimize email reading, for instance through adaptive reading panes, contextual inboxes and better cross-device reading assistance.

7.2 Future work

In this thesis, we have examined user interaction behavior with academic papers and emails. Specifically, we have answered our research questions while inspecting users' queries, downloads, interactions with a recommender system, and email reading behavior. While diverse, our study is not free of limitations. In this section, we identify some of those limitations and propose ways to address them in future work.

7.2.1 Interpreting user intent

It is crucial for a search engine or a recommender system to properly interpret user intent, in order to give users what they want. From the system's perspective, user intent can be inferred from user inputs such as textual queries, or actions such as clicks and downloads. However, these signals may be noisy and sometimes hard to interpret. For instance, users may have placed a wrong click on an item, or issue queries that can be ambiguous [210]. In academic search, the interface layout in our setting is more complex than that of a general web search engine, where users can perform different types of actions. This could make it more difficult to interpret and denoise user signals. Another tricky case is when users perform no actions. In Chapter 3, we have examined how to help users who encounter null queries by using query suggestions. There are certain cases when users may actually not need that help, and abandon the current search because the null result is exactly what they want. In Chapter 4, we assume that user topic interests are reflected in their queries. Still, without explicit feedback from users, we cannot be 100% certain that this case is true. In Chapter 5 and In Chapter 6, we make predictions and recommendations based on users' downloads and browses, which may not reveal all of the user interests. Although our methods are designed to be applied to large volumes of data, automatically and without human intervention, they may not be precise enough to always capture user intent.

To better interpret user intent, we propose a number of future directions. In the context of complex search interfaces such as those in academic search, it is possible to use carefully designed click models [44] to interpret user behavior. User signals such as mouse hover, dwell time, clicks on results and other parts of the search result page (title, authors and abstract etc.) may encode user intent. In academic search, different types of clicks may encode different levels of relevance. For example, download is a special click action that is a stronger relevance signal than a mere click on a paper abstract. However, current click models do not sufficiently consider the differences and therefore should be improved in this aspect. Search contexts should also be considered, for instance, previous queries and clicks, in order to better understand the state of the user.

Another interesting direction is to introduce conversational search [174] into the system, which could be handy when the user has an ambiguous intent and the system may not interpret it with high confidence. The system can "talk" to the user by asking questions about ambiguous queries and let the user choose the correct intents. When a null query occurs, the system may also ask the user whether assistance is necessary.

7.2.2 Exploitation versus exploration

In Chapter 3 and Chapter 6 we have examined how to recommend personalized queries and papers to users, based on their historical interactions with the systems. Our models rely heavily on exploitation, that is, we try to make recommendations based on the "known user interests," represented in their historical behavior. In the domain of academic recommendation, however, sometimes users may wish to break out of the "filter bubble" and see something completely new. Recommending something new to the user that is not based on their interests is highly risky, as it may undermine user satisfaction of the system. Still, if used properly, this strategy brings benefits to users. For instance, recommending new algorithms in the domain of machine translation to a researcher specialized in information retrieval may inspire inventions of new models. We have not explicitly modeled this strategy in our work.

We argue that in academic recommendation tasks, exploitation should be complemented with exploration, which involves recommending contents that are not entirely based on user interests. We refer to this type of recommendation as "exploratory recommendations." The task of making exploratory recommendations involves deciding a good timing, and selecting useful contents for the user. First and foremost, exploratory recommendations should come in at the right moment to maximize their utility to users, instead of disturbing users. Therefore, the system needs to understand the user status. Query reformulations can be used to infer the user status (Chapter 3 and Chapter 4) – the system can speculate, based on query reformulations, that a user is looking for a specific paper; in that case recommending a paper about a very different topic may not be ideal; what would be better, is when the user is actively exploring around some subtopics in search, to recommend something different. Automatically inferring user status may be less intrusive to users, but it may not always be accurate; therefore, it could be complemented by explicitly asking users whether they would like to see exploratory recommendations.

7.2.3 Explainability of recommendations

In Chapter 3 and Chapter 6, we have discussed making recommendations of queries and academic papers. What we have not discussed, on the user side, is how to interpret the trustworthiness of recommendations, especially for the personalized recommendations. A survey in the domain of news recommendation has shown that the majority of users want to see explanations for personalized recommendations [211].

It is yet to be seen whether this holds for other domains, such as academic recommendations. Nevertheless, the system could offer such an option to let users choose, whether to see the explanations behind personalized recommendations. This brings the following benefits: (1) Explanations may help users' decision making process: for instance, when a paper recommendation is made primarily based on author similarity, the system also makes the user aware of the fact that author similarity is used (e.g., displaying a short label "based on author similarity"). If the user is actually looking for papers written by similar authors, then this explanation may grab the user's attention and make it easier to decide whether to click on it. (2) The explanations serve as reminders to let users understand what part of their personal information is utilized. Being transparent to users about the recommendation process may improve trustworthiness of the system and that of the recommendations. As future work, it is interesting to examine methods for generating explanations of academic recommendations. Meanwhile, A/B tests can reveal how the explanations affect the recommendation performance, and the user satisfaction of the explanations.

7.2.4 Applications based on predicting email reading time

In the age of information overload, e.g., facing an excessive number of emails, it is meaningful to make email reading easier. While we have obtained a characterization of user reading time on enterprise emails, we have not applied the findings to build new email applications. We propose to make several applications based on automatic prediction of email reading time as future work, which can help users be more efficient in reading emails. As we have pointed out in Chapter 6, based on predictions of email reading time, the system can use an adaptable reading pane to display emails; the system can also make the inbox contextual based on user status, assist users through busy schedules, and provide cross-platform support when users reread emails.

Bibliography

[1] D. Aberdeen, O. Pacovsky, and A. Slater. The learning behind Gmail Priority Inbox. In *LCCC: NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds*, 2010. (Cited on page 97.)

[2] E. Ahsberg, F. Gamberale, and K. Gustafsson. Perceived fatigue after mental work: an experimental evaluation of a fatigue inventory. *Ergonomics*, 43(2):252–268, 2000. (Cited on pages 104 and 110.)

[3] Q. Ai, S. T. Dumais, N. Craswell, and D. Liebling. Characterizing email search using large-scale behavioral logs and surveys. In *WWW*, pages 1511–1520, 2017. (Cited on pages 95 and 97.)

[4] L. M. Aiello, D. Donato, U. Ozertem, and F. Menczer. Behavior-driven clustering of queries into topics. In *CIKM*, pages 1373–1382, 2011. (Cited on page 41.)

[5] N. Ailon, Z. S. Karnin, E. Liberty, and Y. Maarek. Threading machine generated email. In *WSDM*, pages 405–414, 2013. (Cited on page 97.)

[6] F. Aiolli. A preliminary study on a recommender system for the million songs dataset challenge. In *The 4th Italian Information Retrieval Workshop*, pages 73–83, 2013. (Cited on page 90.)

[7] M. Aljukhadar, S. Senecal, and C.-E. Daoust. Information overload and usage of recommendations. In *UCERST*, pages 26–33, 2010. (Cited on page 52.)

[8] I. S. Altingovde, R. Blanco, B. B. Cambazoglu, R. Ozcan, E. Sarigil, and O. Ulusoy. Characterizing web search queries that match very few or no results. In *CIKM*, pages 2000–2004, 2012. (Cited on pages 1, 14, 18, and 35.)

[9] E. Amolochitis, I. T. Christou, Z.-H. Tan, and R. Prasad. A heuristic hierarchical scheme for academic search and retrieval. *Information Processing & Management*, 49(6):1326–1343, 2013. (Cited on page 34.)

[10] A. Arampatzis and J. Kamps. A study of query length. In *SIGIR*, pages 811–812, 2008. (Cited on page 16.)

[11] R. Artstein and M. Poesio. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596, 2008. (Cited on page 20.)

[12] P. Bailey, R. W. White, H. Liu, and G. Kumaran. Mining historic query trails to label long and rare search engine queries. *ACM Transactions on the Web*, 4(4):15:1–15:27, 2010. (Cited on page 16.)

[13] S. R. Barley, D. E. Meyerson, and S. Grodal. E-mail as a source and symbol of stress. *Organization Science*, 22(4):887–906, 2011. (Cited on page 103.)

[14] J. Beel, A. Aizawa, C. Breitinger, and B. Gipp. Mr. dlib: Recommendations-as-a-service (raas) for academia. In *JCDL*, pages 1–2, 2017. (Cited on page 88.)

[15] F. Beierle, A. Aizawa, and J. Beel. Exploring choice overload in related-article recommendations in digital libraries. *arXiv preprint arXiv:1704.00393*, 2017. (Cited on pages 52 and 53.)

[16] S. M. Beitzel, E. C. Jensen, A. Chowdhury, O. Frieder, and D. Grossman. Temporal analysis of a very large topically categorized web query log. *Journal of the Association for Information Science and Technology*, 58(2):166–178, 2007. (Cited on pages 13 and 39.)

[17] R. Bekkerman, A. Mccallum, and G. Huang. Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora. In *Technical Report, Computer Science Department, University of Massachusetts, IR-418*, pages 1–23, 2004. (Cited on page 97.)

[18] M. Bendersky and W. B. Croft. Analysis of long queries in a large scale search log. In *WSCD*, pages 8–14, 2009. (Cited on page 13.)

[19] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD*, pages 359–370, 1994. (Cited on page 64.)

[20] Bing. `https://www.bing.com/`, 2018. Last accessed April 3, 2018. (Cited on page 1.)

[21] R. Blanco, G. Ottaviano, and E. Meij. Fast and space-efficient entity linking for queries. In *WSDM*, pages 179–188, 2015. (Cited on page 17.)

[22] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003. (Cited on pages 3, 40, and 60.)

[23] P. Boldi, F. Bonchi, C. Castillo, D. Donato, and S. Vigna. Query suggestions using query-flow graphs. In *WSCD*, pages 56–63, 2009. (Cited on pages 24, 25, 26, and 35.)

[24] P. Boldi, F. Bonchi, C. Castillo, and S. Vigna. Query reformulation mining: models, patterns, and applications. *Information Retrieval Journal*, 14(3):257–289, 2011. (Cited on pages 39, 41, 44, and 45.)

[25] F. Bonchi, R. Perego, F. Silvestri, H. Vahabi, and R. Venturini. Efficient query recommendations in the long tail via center-piece subgraphs. In *SIGIR*, pages 345–354, 2012. (Cited on pages 16 and 35.)

[26] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013. (Cited on pages 75, 76, and 77.)

[27] I. Bordino, C. Castillo, D. Donato, and A. Gionis. Query similarity by projecting the query-flow graph.

In *SIGIR*, pages 515–522, 2010. (Cited on pages 35 and 41.)

[28] I. Bordino, G. De Francisci Morales, I. Weber, and F. Bonchi. From Machu_Picchu to "rafting the urubamba river": Anticipating information needs via the Entity-Query Graph. In *WSDM*, pages 275–284, 2013. (Cited on pages 23, 24, 25, 26, 27, and 36.)

[29] A. Borisov, I. Markov, M. de Rijke, and P. Serdyukov. A context-aware time model for web search. In *SIGIR*, pages 205–214, 2016. (Cited on page 52.)

[30] P. D. Bruza and S. Dennis. Query reformulation on the internet: Empirical data and the hyperindex search engine. In *RIAO*, 1997. (Cited on pages 41 and 44.)

[31] F. Cai and M. de Rijke. A survey of query auto completion in information retrieval. *Foundations and Trends in Information Retrieval*, 10(4):273–363, 2016. (Cited on pages 36 and 41.)

[32] F. Cai, S. Liang, and M. de Rijke. Time-sensitive personalized query auto-completion. In *CIKM*, pages 1599–1608, 2014. (Cited on page 36.)

[33] F. Cai, W. Chen, and X. Ou. Learning search popularity for personalized query completion in information retrieval. *Journal of Intelligent & Fuzzy Systems*, 33(4):2427–2435, 2017. (Cited on page 2.)

[34] G. Carenini, R. T. Ng, and X. Zhou. Summarizing email conversations with clue words. In *WWW*, pages 91–100, 2007. (Cited on page 97.)

[35] D. Carmel, G. Halawi, L. Lewin-Eytan, Y. Maarek, and A. Raviv. Rank by time or by relevance?: Revisiting email search. In *CIKM*, pages 283–292, 2015. (Cited on pages 95 and 97.)

[36] D. Carmel, L. Lewin-Eytan, A. Libov, Y. Maarek, and A. Raviv. The demographics of mail search and their application to query suggestion. In *WWW*, pages 1541–1549, 2017.

[37] D. Carmel, L. Lewin-Eytan, A. Libov, Y. Maarek, and A. Raviv. Promoting relevant results in time-ranked mail search. In *WWW*, pages 1551–1559, 2017. (Cited on page 97.)

[38] L. D. Catledge and J. E. Pitkow. Characterizing browsing strategies in the world-wide web. *Computer Networks and ISDN systems*, 27(6):1065–1073, 1995. (Cited on page 54.)

[39] J. Catlow, M. Górny, and R. Lewandowski. Students as users of digital libraries. *Qualitative and Quantitative Methods in Libraries*, 65:60–1, 2015. (Cited on page 43.)

[40] M. E. Cecchinato, A. Sellen, M. Shokouhi, and G. Smyth. Finding email in a multi-account, multi-device world. In *CHI*, pages 1200–1210, 2016. (Cited on page 97.)

[41] L. Charlin, R. S. Zemel, and H. Larochelle. Leveraging user libraries to bootstrap collaborative filtering. In *KDD*, pages 173–182, 2014. (Cited on page 88.)

[42] Y. Cheng, L. Yin, and Y. Yu. Lorslim: Low rank sparse linear methods for top-n recommendations. In *ICDM*, pages 90–99, 2014. (Cited on page 90.)

[43] E. Christakopoulou and G. Karypis. Local item-item models for top-n recommendation. In *RecSys*, pages 67–74, 2016. (Cited on page 90.)

[44] A. Chuklin, I. Markov, and M. de Rijke. *Click Models for Web Search*. Morgan & Claypool, 2015. (Cited on pages 40, 69, and 118.)

[45] F. R. Chung. *Spectral graph theory*. American Mathematical Society, 1997. (Cited on page 81.)

[46] P. Covington, J. Adams, and E. Sargin. Deep neural networks for YouTube recommendations. In *RecSys*, pages 191–198, 2016. (Cited on page 72.)

[47] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *RecSys*, pages 39–46, 2010. (Cited on page 89.)

[48] L. A. Dabbish and R. E. Kraut. Email overload at work: An analysis of factors associated with email strain. In *CSCW*, pages 431–440, 2006. (Cited on page 96.)

[49] L. A. Dabbish, R. E. Kraut, S. Fussell, and S. Kiesler. Understanding email use: Predicting action on a message. In *CHI*, pages 691–700, 2005. (Cited on pages 95 and 97.)

[50] N. Dali Betzalel, B. Shapira, and L. Rokach. Please, not now!: A model for timing recommendations. In *RecSys*, pages 297–300, 2015. (Cited on page 52.)

[51] A. Dasgupta, M. Gurevich, and K. Punera. Enhanced email spam filtering through combining similarity graphs. In *WSDM*, pages 785–794, 2011. (Cited on page 96.)

[52] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston, et al. The YouTube video recommendation system. In *RecSys*, pages 293–296, 2010. (Cited on page 72.)

[53] M. Deshpande and G. Karypis. Item-based top-N recommendation algorithms. *ACM Transactions on Information Systems*, 22(1):143–177, 2004. (Cited on pages 89 and 90.)

[54] D. Di Castro, Z. Karnin, L. Lewin-Eytan, and Y. Maarek. You've got mail, and here is what you could do with it!: Analyzing and predicting actions on email messages. In *WSDM*, pages 307–316, 2016. (Cited on pages 95 and 97.)

[55] R. I. Dogan, G. C. Murray, A. Névéol, and Z. Lu. Understanding PubMed® user search behavior through log analysis. *Database*, 2009, 2009. (Cited on page 14.)

[56] M. Dredze, H. M. Wallach, D. Puller, and F. Pereira. Generating summary keywords for emails using topics. In *IUI*, pages 199–206, 2008. (Cited on page 97.)

[57] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011. (Cited on page 84.)

[58] S. Dumais, E. Cutrell, J. J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins. Stuff I've seen: a system for personal information retrieval and re-use. In *SIGIR*, pages 72–79, 2003. (Cited on page 97.)

[59] R. B. Duque, M. Ynalvez, R. Sooryamoorthy, P. Mbatia, D.-B. S. Dzorgbo, and W. Shrum. Collaboration paradox: Scientific productivity, the internet, and problems of research in developing areas. *Social Studies of Science*, 35(5):755–785, 2005. (Cited on pages 2, 95, and 117.)

[60] C. M. Dwyer, E. A. Gossen, and L. M. Martin. Known-item search failure in an OPAC. *RQ*, pages 228–236, 1991. (Cited on page 35.)

[61] T. Ebesu and Y. Fang. Neural citation network for context-aware citation recommendation. In *SIGIR*, pages 1093–1096, 2017. (Cited on page 89.)

[62] Effective Communication, Better Science. `https://blogs.scientificamerican.com/guest-blog/effective-communication-better-science/`, 2018. Last accessed May 3, 2018. (Cited on pages 71, 72, and 87.)

[63] M. D. Ekstrand, P. Kannan, J. A. Stemper, J. T. Butler, J. A. Konstan, and J. T. Riedl. Automatically building research reading lists. In *RecSys*, pages 159–166, 2010. (Cited on pages 88 and 89.)

[64] J. S. B. T. Evans and D. E. Over. *Reasoning and Rationality*. Psychology Press, 1996. (Cited on page 45.)

[65] F. Ferrara, N. Pudota, and C. Tasso. A keyphrase-based paper recommender system. In *Italian Research Conference on Digital Libraries*, pages 14–25, 2011. (Cited on page 88.)

[66] M. Fiterau, S. Bhooshan, J. Fries, C. Bournhonesque, J. Hicks, E. Halilaj, C. Ré, and S. Delp. Shortfuse: Biomedical time series representations in the presence of structured information. *arXiv preprint arXiv:1705.04790*, 2017. (Cited on page 64.)

[67] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5): 378, 1971. (Cited on page 20.)

[68] A. Ghose, P. G. Ipeirotis, and B. Li. Examining the impact of ranking on consumer behavior and search engine revenue. *Management Science*, 60(7):1632–1654, 2014. (Cited on page 51.)

[69] C. L. Giles, K. D. Bollacker, and S. Lawrence. Citeseer: An automatic citation indexing system. In *DL*, pages 89–98, 1998. (Cited on pages 34 and 41.)

[70] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256, 2010. (Cited on page 64.)

[71] Google. `https://www.google.com/`, 2018. Last accessed April 3, 2018. (Cited on page 1.)

[72] Google Music. `https://play.google.com/music/`, 2018. Last accessed April 3, 2018. (Cited on page 1.)

[73] Google search stastics. `http://www.internetlivestats.com/google-search-statistics/`, 2018. Last accessed May 3, 2018. (Cited on page 1.)

[74] M. Gori and A. Pucci. Research paper recommender systems: A random-walk based approach. In *WI*, pages 778–781, 2006. (Cited on page 53.)

[75] S. Goswami. Analysing effects of information overload on decision quality in an online environment. *SAMVAD International Journal of Management*, 9:65–69, 2015. (Cited on page 52.)

[76] M. Grbovic, G. Halawi, Z. Karnin, and Y. Maarek. How many folders do you really need?: Classifying email into a handful of categories. In *CIKM*, pages 869–878, 2014. (Cited on pages 95 and 97.)

[77] C. Grevet, D. Choi, D. Kumar, and E. Gilbert. Overload is overloaded: email in the age of Gmail. In *CHI*, pages 793–802, 2014. (Cited on page 96.)

[78] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(suppl 1):5228–5235, 2004. (Cited on page 43.)

[79] D. Guan, S. Zhang, and H. Yang. Utilizing query change for session search. In *SIGIR*, pages 453–462, 2013. (Cited on page 41.)

[80] J. Guo, X. Cheng, G. Xu, and H. Shen. A structured approach to query recommendation with social annotation data. In *CIKM*, pages 619–628, 2010. (Cited on page 35.)

[81] J. Guo, X. Cheng, G. Xu, and X. Zhu. Intent-aware query similarity. In *CIKM*, pages 259–268, 2011. (Cited on pages 17 and 35.)

[82] M. Gupta and M. Bendersky. Information retrieval with verbose queries. *Foundations and Trends in Information Retrieval*, 9(3–4):209–354, 2015. (Cited on page 14.)

[83] H. Han, W. Jeong, and D. Wolfram. Log analysis of academic digital library: user query patterns. *iConference 2014 Proceedings*, pages 1002–1008, 2014. (Cited on pages 2, 3, 15, 34, 39, and 41.)

[84] A. Hassan, R. W. White, S. T. Dumais, and Y.-M. Wang. Struggling or exploring?: Disambiguating long search sessions. In *WSDM*, pages 53–62, 2014. (Cited on page 21.)

[85] A. Hassan Awadallah, R. W. White, P. Pantel, S. T. Dumais, and Y.-M. Wang. Supporting complex search tasks. In *CIKM*, pages 829–838, 2014. (Cited on page 36.)

[86] Q. He, D. Kifer, J. Pei, P. Mitra, and C. L. Giles. Citation recommendation without author supervision. In *WSDM*, pages 755–764, 2011. (Cited on page 89.)

[87] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *WWW*, pages 173–182, 2017. (Cited on page 84.)

[88] B. M. Hemminger, D. Lu, K. Vaughan, and S. J. Adams. Information seeking behavior of academic scientists. *Journal of the American Society for Information Science and Technology*, 58(14):2205–2225, 2007. (Cited on pages 13, 39, and 53.)

[89] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. (Cited on page 64.)

[90] M. Horovitz, L. Lewin-Eytan, A. Libov, Y. Maarek, and A. Raviv. Mailbox-based vs. log-based query completion for mail search. In *SIGIR*, pages 937–940, 2017. (Cited on page 97.)

[91] M. Hristakeva, D. Kershaw, M. Rossetti, P. Knoth, B. Pettit, S. Vargas, and K. Jack. Building recommender systems for scholarly information. In *The 1st Workshop on Scholarly Web Mining*, pages 25–32, 2017. (Cited on page 88.)

[92] Y. Hu, Y. Qian, H. Li, D. Jiang, J. Pei, and Q. Zheng. Mining query subtopics from search log data. In *SIGIR*, pages 305–314, 2012. (Cited on page 41.)

[93] W. Hua, Y. Song, H. Wang, and X. Zhou. Identifying users' topical tasks in web search. In *WSDM*, pages 93–102, 2013. (Cited on page 41.)

[94] W. Huang, Z. Wu, L. Chen, P. Mitra, and C. L. Giles. A neural probabilistic model for context based citation recommendation. In *AAAI*, pages 2404–2410, 2015. (Cited on page 89.)

[95] B. J. Jansen, D. L. Booth, and A. Spink. Determining the informational, navigational, and transactional intent of web queries. *Information Processing & Management*, 44(3):1251–1266, 2008. (Cited on page 16.)

[96] B. J. Jansen, D. L. Booth, and A. Spink. Patterns of query reformulation during web searching. *Journal of the Association for Information Science and Technology*, 60(7):1358–1371, 2009. (Cited on pages 39, 41, 44, and 45.)

[97] W. Jeng, D. He, and J. Jiang. User participation in an academic social networking service: A survey of open group users on mendeley. *Journal of the Association for Information Science and Technology*, pages 890–904, 2015. (Cited on page 43.)

[98] W. Jeng, J. Jiang, and D. He. Users' perceived difficulties and corresponding reformulation strategies in google voice search. *Journal of Library and Information Studies*, 14(1):1–14, 2016. (Cited on page 41.)

[99] J.-Y. Jiang, Y.-Y. Ke, P.-Y. Chien, and P.-J. Cheng. Learning user reformulation behavior for query auto-completion. In *SIGIR*, pages 445–454, 2014. (Cited on page 41.)

[100] Y. Jiang, A. Jia, Y. Feng, and D. Zhao. Recommending academic papers via users' reading purposes. In *RecSys*, pages 241–244, 2012. (Cited on page 88.)

[101] R. Jones and K. L. Klinkner. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *CIKM*, pages 699–708, 2008. (Cited on page 41.)

[102] S. Jones, S. J. Cunningham, R. McNab, and S. Boddie. A transaction log analysis of a digital library. *International Journal on Digital Libraries*, 3(2):152–169, 2000. (Cited on pages 2, 3, 15, 34, 39, and 41.)

[103] M. A. Just and P. A. Carpenter. A theory of reading: From eye fixations to comprehension. *Psychological Review*, 87(4):329, 1980. (Cited on page 98.)

[104] S. Kabbur, X. Ning, and G. Karypis. FISM: factored item similarity models for top-n recommender systems. In *KDD*, pages 659–667, 2013. (Cited on page 90.)

[105] Z. Kang and Q. Cheng. Top-n recommendation with novel rank approximation. In *ICDM*, pages 126–134, 2016. (Cited on page 90.)

[106] A. Kannan, K. Kurach, S. Ravi, T. Kaufmann, A. Tomkins, B. Miklos, G. Corrado, L. Lukacs, M. Ganea, P. Young, and V. Ramavajjala. Smart reply: Automated response suggestion for email. In *KDD*, pages 955–964, 2016. (Cited on page 97.)

[107] H.-R. Ke, R. Kwakkelaar, Y.-M. Tai, and L.-C. Chen. Exploring behavior of e-journal users in science and technology: Transaction log analysis of Elsevier's ScienceDirect OnSite in Taiwan. *Library &*

Information Science Research, 24(3):265–291, 2002. (Cited on pages 2, 3, 13, 15, 16, 34, 39, 41, 53, and 88.)

[108] M. Khabsa, Z. Wu, and C. L. Giles. Towards better understanding of academic search. In *JCDL*, pages 111–114, 2016. (Cited on pages 17, 53, and 88.)

[109] T. Khazaei and O. Hoeber. Supporting academic search tasks through citation visualization and exploration. *International Journal on Digital Libraries*, 18(1):59–72, 2017. (Cited on page 88.)

[110] J. Y. Kim, N. Craswell, S. Dumais, F. Radlinski, and F. Liu. Understanding and modeling success in email search. In *SIGIR*, pages 265–274, 2017. (Cited on pages 97 and 111.)

[111] Y. Kim, J. Seo, and W. B. Croft. Automatic boolean query suggestion for professional search. In *SIGIR*, pages 825–834, 2011. (Cited on page 23.)

[112] Y. Kim, J. Seo, W. B. Croft, and D. A. Smith. Automatic suggestion of phrasal-concept queries for literature search. *Information Processing & Management*, 50(4):568–583, 2014. (Cited on page 22.)

[113] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (Cited on page 82.)

[114] F. Kooti, L. M. Aiello, M. Grbovic, K. Lerman, and A. Mantrach. Evolution of conversations in the age of email overload. In *WWW*, pages 603–613, 2015. (Cited on page 95.)

[115] 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 *WSDM*, pages 205–214, 2016. (Cited on pages 2, 51, 53, 57, 58, 63, 65, and 66.)

[116] F. Kooti, M. Grbovic, L. M. Aiello, E. Bax, and K. Lerman. iPhone's digital marketplace: Characterizing the big spenders. In *WSDM*, pages 13–21, 2017. (Cited on pages 53, 54, and 64.)

[117] Y. Koren, E. Liberty, Y. Maarek, and R. Sandler. Automatically tagging email by leveraging other users' folders. In *KDD*, pages 913–921, 2011. (Cited on pages 95 and 97.)

[118] D. Kotz and R. S. Gray. Mobile agents and the future of the internet. *Operating Systems Review*, 33 (3):7–13, 1999. (Cited on page 1.)

[119] K. Krippendorff. *Content analysis: An introduction to its methodology*. SAGE Publications, 2004. (Cited on page 20.)

[120] U. Kruschwitz, C. Hull, et al. Searching the enterprise. *Foundations and Trends® in Information Retrieval*, 11(1):1–142, 2017. (Cited on page 97.)

[121] O. Küçüktunç, E. Saule, K. Kaya, and Ü. V. Çatalyürek. Recommendation on academic networks using direction aware citation analysis. *arXiv preprint arXiv:1205.1143*, 2012. (Cited on page 88.)

[122] S. Kuzi, D. Carmel, A. Libov, and A. Raviv. Query expansion for email search. In *SIGIR*, pages 849–852, 2017. (Cited on page 97.)

[123] A. Kyrola. DrunkardMob: Billions of random walks on just a PC. In *RecSys*, pages 257–264, 2013. (Cited on page 25.)

[124] F. W. Lancaster. *Evaluation of the MEDLARS Demand Search Service*. National Library of Medicine (U.S.), 1968. (Cited on page 35.)

[125] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977. (Cited on pages 20 and 58.)

[126] T. Lau and E. Horvitz. Patterns of search: analyzing and modeling web query refinement. In *UM*, pages 119–128, 1999. (Cited on pages 41 and 44.)

[127] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562, 2001. (Cited on page 84.)

[128] M. Lee, T. Ha, J. Han, J.-Y. Rha, and T. T. Kwon. Online footsteps to purchase: Exploring consumer behaviors on online shopping sites. In *WebSci*, pages 5:1–15:10, 2015. (Cited on pages 51 and 53.)

[129] D. Lefortier, P. Serdyukov, and M. de Rijke. Online exploration for detecting shifts in fresh intent. In *CIKM*, 2014. (Cited on page 90.)

[130] Q. Lei, J. Yi, R. Vaculin, L. Wu, and I. S. Dhillon. Similarity preserving representation learning for time series analysis. *arXiv preprint arXiv:1702.03584*, 2017. (Cited on page 64.)

[131] C.-Y. Li. Why do online consumers experience information overload? an extension of communication theory. *Journal of Information Science*, 43(6):835–851, 2016. (Cited on page 52.)

[132] H. Li, I. Councill, W.-C. Lee, and C. L. Giles. Citeseerx: an architecture and web service design for an academic document search engine. In *WWW*, pages 883–884, 2006. (Cited on pages 53 and 87.)

[133] L. Li, H. Deng, Y. He, A. Dong, Y. Chang, and H. Zha. Behavior driven topic transition for search task identification. In *WWW*, pages 555–565, 2016. (Cited on page 39.)

[134] X. Li and M. de Rijke. Academic search in response to major scientific events. In *The 5th International Workshop on Bibliometric-enhanced Information Retrieval*, pages 41–50, 2017. (Cited on page 88.)

[135] X. Li and M. de Rijke. Do topic shift and query reformulation patterns correlate in academic search?

In *ECIR*, pages 146–159, April 2017. (Cited on pages 39, 53, 76, and 88.)

[136] X. Li and M. de Rijke. Characterizing and predicting downloads in academic search. *Information Processing & Management*, under review. (Cited on page 51.)

[137] X. Li, J. Tang, T. Wang, Z. Luo, and M. de Rijke. Automatically assessing Wikipedia article quality by exploiting article-editor networks. In *ECIR*, pages 574–580, April 2015.

[138] X. Li, R. Schijvenaars, and M. de Rijke. Investigating queries and search failures in academic search. *Information Processing & Management*, 53(3):666–683, May 2017. (Cited on pages 13, 41, 53, 56, and 88.)

[139] X. Li, Y. Chen, B. Pettit, and M. de Rijke. Reranking paper recommendations using paper content and user behavior. *ACM Transactions on Information Systems*, under review. (Cited on page 71.)

[140] X. Li, C.-j. Lee, M. Shokouhi, and S. Dumais. Reranking paper recommendations using paper content and user behavior. *Journal of the Association for Information Science and Technology*, under review. (Cited on page 95.)

[141] Y. Li, M. Yang, and Z. M. Zhang. Scientific articles recommendation. In *CIKM*, pages 1147–1156, 2013. (Cited on page 53.)

[142] T. Lin, P. Pantel, M. Gamon, A. Kannan, and A. Fuxman. Active objects: Actions for entity-centric search. In *WWW*, pages 589–598, 2012. (Cited on page 17.)

[143] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015. (Cited on page 75.)

[144] D. Lindberg. Internet access to the National Library of Medicine. *Effective Clinical Practice*, 3(5): 256–260, 2000. (Cited on pages 13, 34, and 41.)

[145] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003. (Cited on page 74.)

[146] H. Liu, X. Kong, X. Bai, W. Wang, T. M. Bekele, and F. Xia. Context-based collaborative filtering for citation recommendation. *IEEE Access*, 3:1695–1703, 2015. (Cited on page 89.)

[147] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *WSDM*, pages 287–296, 2011. (Cited on page 81.)

[148] Y. Maarek. Is mail the next frontier in search and data mining? In *WSDM*, pages 203–203, 2016. (Cited on page 95.)

[149] R. Mehrotra, P. Bhattacharya, and E. Yilmaz. Uncovering task based behavioral heterogeneities in online search behavior. In *SIGIR*, pages 1049–1052, 2016. (Cited on page 42.)

[150] E. Meij, M. Bron, L. Hollink, B. Huurnink, and M. de Rijke. Mapping queries to the linking open data cloud: A case study using dbpedia. *Web Semantics*, 9(4):418–433, Dec. 2011. (Cited on page 17.)

[151] G. Mishne and M. de Rijke. A study of blog search. In *ECIR*, pages 289–301, 2006. (Cited on page 13.)

[152] A. Mitra and A. Awekar. On low overlap among search results of academic search engines. In *WWW*, pages 823–824, 2017. (Cited on pages 53 and 87.)

[153] B. Mitra and N. Craswell. Query auto-completion for rare prefixes. In *CIKM*, pages 1755–1758, 2015. (Cited on page 36.)

[154] B. Mitra, M. Shokouhi, F. Radlinski, and K. Hofmann. On user interactions with query auto-completion. In *SIGIR*, pages 1055–1058, 2014. (Cited on page 36.)

[155] A. Mnih and R. R. Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2008. (Cited on page 84.)

[156] A. S. Mohammad Arif, J. T. Du, and I. Lee. Examining collaborative query reformulation: a case of travel information searching. In *SIGIR*, pages 875–878, 2014. (Cited on page 41.)

[157] T. Moon, W. Chu, L. Li, Z. Zheng, and Y. Chang. An online learning framework for refining recency search results with user click feedback. *ACM Transactions on Information Systems*, 30(4):20:1–20:28, 2012. (Cited on page 90.)

[158] S. Muresan, E. Tzoukermann, and J. L. Klavans. Combining linguistic and machine learning techniques for email summarization. In *CoNLL*, pages 19:1–19:8, 2001. (Cited on page 97.)

[159] K. Narang, S. T. Dumais, N. Craswell, D. Liebling, and Q. Ai. Large-scale analysis of email search and organizational strategies. In *CHIR*, pages 215–223, 2017. (Cited on page 97.)

[160] C. Nascimento, A. H. Laender, A. S. da Silva, and M. A. Gonçalves. A source independent framework for research paper recommendation. In *JCDL*, pages 297–306, 2011. (Cited on page 88.)

[161] Netflix. `https://www.netflix.com/`, 2018. Last accessed April 3, 2018. (Cited on page 1.)

[162] X. Ning and G. Karypis. SLIM: sparse linear methods for top-n recommender systems. In *ICDM*, pages 497–506, 2011. (Cited on page 90.)

[163] C. Nishioka and A. Scherp. Profiling vs. time vs. content: What does matter for top-k publication

recommendation based on twitter profiles? In *JCDL*, pages 171–180, 2016. (Cited on page 53.)

[164] X. Niu and B. M. Hemminger. A study of factors that affect the information-seeking behavior of academic scientists. *Journal of the American Society for Information Science and Technology*, 63(2): 336–353, 2012. (Cited on pages 53 and 88.)

[165] X. Niu, B. M. Hemminger, C. Lown, S. Adams, C. Brown, A. Level, M. McLure, A. Powers, M. R. Tennant, and T. Cataldo. National study of information seeking behavior of academic researchers in the United States. *Journal of the American Society for Information Science and Technology*, 61(5): 869–890, 2010. (Cited on pages 13, 39, 41, 43, and 53.)

[166] Number of households subscribing to Netflix worldwide from 2010 to 2020. `https://www.statista.com/statistics/273885/quarterly-subscriber-numbers-of-netflix/`, 2018. Last accessed May 3, 2018. (Cited on page 1.)

[167] Number of paying Spotify subscribers worldwide from July 2010 to January 2018. `https://www.statista.com/statistics/244995/number-of-paying-spotify-subscribers/`, 2018. Last accessed May 3, 2018. (Cited on page 1.)

[168] P. Ogilvie and J. Callan. Combining document representations for known-item search. In *SIGIR*, pages 143–150, 2003. (Cited on page 17.)

[169] D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach. In *UAI*, pages 473–480, 2000. (Cited on page 88.)

[170] S. Pontis and A. Blandford. Understanding "influence:" an exploratory study of academics' processes of knowledge construction through iterative and interactive information seeking. *Journal of the Association for Information Science and Technology*, 66(8):1576–1593, 2015. (Cited on pages 13, 41, 53, and 88.)

[171] S. Pontis, A. Blandford, E. Greifeneder, H. Attalla, and D. Neal. Keeping up to date: An academic researcher's information journey. *Journal of the American Society for Information Science and Technology*, 68(1):22–35, 2015. (Cited on pages 13, 41, 53, and 88.)

[172] A. Porter and I. Rafols. Is science becoming more interdisciplinary? Measuring and mapping six research fields over time. *Scientometrics*, 81(3):719–745, 2009. (Cited on page 61.)

[173] K. Purcell and L. Raine. Email and the internet are the dominant technological tools in american workplaces. Technical report, Pew Research Center, 2014. (Cited on page 95.)

[174] F. Radlinski and N. Craswell. A theoretical framework for conversational search. In *CHIIR*, pages 117–126, 2017. (Cited on page 118.)

[175] F. Radlinski, M. Szummer, and N. Craswell. Inferring query intent from reformulations and clicks. In *WWW*, pages 1171–1172, 2010. (Cited on page 41.)

[176] P. Ramarao, S. Iyengar, P. Chitnis, R. Udupa, and B. Ashok. Inlook: Revisiting email search experience. In *SIGIR*, pages 1117–1120, 2016. (Cited on page 97.)

[177] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine learning*, 85(3):333–359, 2011. (Cited on page 29.)

[178] J. Read, F. Perez-Cruz, and A. Bifet. Deep learning in partially-labeled data streams. In *SAC*, pages 954–959, 2015. (Cited on page 28.)

[179] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009. (Cited on page 81.)

[180] F. Ricci, L. Rokach, B. Shapira, and P. Kantor. *Recommender Systems Handbook*. Springer, 2015. (Cited on page 89.)

[181] S. Y. Rieh et al. Analysis of multiple query reformulations on the web: The interactive information retrieval context. *Information Processing & Management*, 42(3):751–768, 2006. (Cited on pages 41, 44, and 45.)

[182] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *EC*, pages 158–167, 2000. (Cited on page 72.)

[183] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295, 2001. (Cited on page 74.)

[184] M. Saveski and A. Mantrach. Item cold-start recommendations: learning local collective embeddings. In *RecSys*, pages 89–96, 2014. (Cited on page 81.)

[185] M. Schedl, P. Knees, B. McFee, D. Bogdanov, and M. Kaminskas. Music recommender systems. In *Recommender Systems Handbook*, pages 453–492. Springer, 2015. (Cited on page 2.)

[186] ScienceDirect. `https://sciencedirect.com`, 2015. Last accessed September 14, 2015. (Cited on pages 1, 34, 71, 72, and 87.)

[187] ScienceDirect statistics. `https://www.elsevier.com/solutions/sciencedirect/`

features/, 2016. Last accessed April 26, 2018. (Cited on pages 1 and 72.)

[188] Search Engine Statistics 2018. https://www.smartinsights.com/search-engine-marketing/search-engine-statistics/, 2018. Last accessed May 3, 2018. (Cited on page 1.)

[189] A. Sesagiri Raamkumar, S. Foo, and N. Pang. Can I have more of these please? Assisting researchers in finding similar research papers from a seed basket of papers. *The Electronic Library*, 2018. (Cited on page 88.)

[190] A. Shiri. Query reformulation strategies in an interdisciplinary digital library: The case of nanoscience and technology. In *ICDIM*, pages 200–206, 2010. (Cited on pages 41 and 44.)

[191] M. Shokouhi. Learning to personalize query auto-completion. In *SIGIR*, pages 103–112, 2013. (Cited on page 36.)

[192] M. Shokouhi, R. Jones, U. Ozertem, K. Raghunathan, and F. Diaz. Mobile query reformulations. In *SIGIR*, pages 1011–1014, 2014. (Cited on page 41.)

[193] F. Silvestri. Mining query logs: Turning search usage data into knowledge. *Foundations and Trends in Information Retrieval*, 4(1–2):1–174, 2010. (Cited on page 13.)

[194] G. Singh, N. Parikh, and N. Sundaresan. Rewriting null e-commerce queries to recommend products. In *WWW*, pages 73–82, 2012. (Cited on page 35.)

[195] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-j. P. Hsu, and K. Wang. An overview of Microsoft Academic Service (MAS) and applications. In *WWW*, pages 243–246, 2015. (Cited on page 53.)

[196] G. Song. Point-wise approach for Yandex personalized web search challenge. In *WSCD*, 2014. (Cited on page 84.)

[197] Y. Song, D. Zhou, and L.-w. He. Query suggestion by constructing term-transition graphs. In *WSDM*, pages 353–362, 2012. (Cited on page 35.)

[198] R. Sooryamoorthy. Producing information: communication and collaboration in the South African scientific community. *Information, Communication & Society*, 19(2):141–159, 2016. (Cited on pages 2, 95, and 117.)

[199] Spotify. https://www.spotify.com/, 2018. Last accessed April 3, 2018. (Cited on page 1.)

[200] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: Discovery and applications of usage patterns from web data. *ACM SIGKDD Explorations Newsletter*, 1(2):12–23, 2000. (Cited on page 54.)

[201] T. Strohman, W. B. Croft, and D. Jensen. Recommending citations for academic papers. In *SIGIR*, pages 705–706, 2007. (Cited on page 89.)

[202] K. Sugiyama and M.-Y. Kan. Scholarly paper recommendation via user's recent research interests. In *JCDL*, pages 29–38, 2010. (Cited on page 88.)

[203] J. Sun, Y. Jiang, X. Cheng, W. Du, Y. Liu, and J. Ma. A hybrid approach for article recommendation in research social networks. *Journal of Information Science*, 2017. (Cited on page 53.)

[204] X. Sun, J. Kaur, L. Possamai, and F. Menczer. Ambiguous author query detection using crowdsourced digital library annotations. *Information Processing & Management*, 49(2):454–464, 2013. (Cited on page 17.)

[205] T. Tam, A. Ferreira, and A. Lourenço. Automatic Foldering of Email Messages: A Combination Approach. In *ECIR*, pages 232–243, 2012. (Cited on page 97.)

[206] J. Tang. AMiner: Toward understanding big scholar data. In *WSDM*, pages 467–467, 2016. (Cited on pages 34, 41, 53, 71, and 87.)

[207] J. Tang, R. Jin, and J. Zhang. A topic modeling approach and its integration into the random walk framework for academic search. In *ICDM*, pages 1055–1060, 2008. (Cited on page 88.)

[208] J. Teevan. The re:search engine: Simultaneous support for finding and re-finding. In *UIST*, pages 23–32, 2007. (Cited on pages 34 and 44.)

[209] J. Teevan, E. Adar, R. Jones, and M. A. Potts. Information re-retrieval: repeat queries in Yahoo's logs. In *SIGIR*, pages 151–158, 2007. (Cited on pages 105 and 115.)

[210] J. Teevan, S. T. Dumais, and D. J. Liebling. To personalize or not to personalize: Modeling queries with variation in user intent. In *SIGIR*, New York, NY, USA, 2008. (Cited on page 118.)

[211] M. ter Hoeve, M. Heruer, D. Odijk, A. Schuth, and M. de Rijke. Do news consumers want explanations for personalized news rankings? In *FATREC Workshop on Responsible Recommendation Proceedings*, 2017. (Cited on page 119.)

[212] R. Torres, S. M. McNee, M. Abel, J. A. Konstan, and J. Riedl. Enhancing digital libraries with TechLens+. In *JCDL*, pages 228–236, 2004. (Cited on pages 88 and 89.)

[213] G. Tsoumakas, I. Katakis, and I. Vlahavas. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer, 2010. (Cited on page 29.)

[214] H. Vahabi, M. Ackerman, D. Loker, R. Baeza-Yates, and A. Lopez-Ortiz. Orthogonal query recommendation. In *RecSys*, pages 33–40, 2013. (Cited on pages 22, 23, 25, and 35.)

[215] J. Wainer, L. Dabbish, and R. Kraut. Should I open this email?: Inbox-level cues, curiosity and attention to email. In *CHI*, pages 3439–3448, 2011. (Cited on page 97.)

[216] J. P. Walsh and N. G. Maloney. Collaboration structure, communication media, and problems in scientific work teams. *Journal of Computer-Mediated Communication*, 12(2):712–732, 2007. (Cited on pages 2, 95, and 117.)

[217] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *KDD*, pages 448–456, 2011. (Cited on page 88.)

[218] S. Wedig and O. Madani. A large-scale analysis of query logs for assessing personalization opportunities. In *KDD*, pages 742–747, 2006. (Cited on page 13.)

[219] 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 *SIGIR*, pages 45–54, 2011. (Cited on page 13.)

[220] J. B. Wendt, M. Bendersky, L. Garcia-Pueyo, V. Josifovski, B. Miklos, I. Krka, A. Saikia, J. Yang, M.-A. Cartright, and S. Ravi. Hierarchical label propagation and discovery for machine generated email. In *WSDM*, pages 317–326, 2016. (Cited on page 97.)

[221] S. Whittaker and C. Sidner. Email overload: Exploring personal information management of email. In *CHI*, pages 276–283, 1996. (Cited on page 96.)

[222] B. Wu, C. Xiong, M. Sun, and Z. Liu. Query suggestion with feedback memory network. In *WWW*, pages 1563–1571, 2018. (Cited on page 2.)

[223] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *WSDM*, pages 153–162, 2016. (Cited on page 90.)

[224] Z. Xiao, F. Che, E. Miao, and M. Lu. Increasing serendipity of recommender system with ranking topic model. *Applied Mathematics & Information Sciences*, 8(4):2041, 2014. (Cited on pages 88 and 89.)

[225] C. Xiong, R. Power, and J. Callan. Explicit semantic ranking for academic search via knowledge graph embedding. In *WWW*, pages 1271–1279, 2017. (Cited on pages 53 and 88.)

[226] L. Yang, S. T. Dumais, P. N. Bennett, and A. H. Awadallah. Characterizing and predicting enterprise email reply behavior. In *SIGIR*, pages 235–244, 2017. (Cited on pages 95 and 97.)

[227] J. Yeo, S. Kim, E. Koh, S.-w. Hwang, and N. Lipka. Predicting online purchase conversion for retargeting. In *WSDM*, pages 591–600, 2017. (Cited on page 54.)

[228] S. Yoo, Y. Yang, and J. Carbonell. Modeling personalized email prioritization: Classification-based and regression-based approaches. In *CIKM*, pages 729–738, 2011. (Cited on page 96.)

[229] YouTube. `https://www.youtube.com/`, 2018. Last accessed April 3, 2018. (Cited on page 1.)

[230] A. Zhang, A. Goyal, W. Kong, H. Deng, A. Dong, Y. Chang, C. A. Gunter, and J. Han. adaQAC: Adaptive query auto-completion via implicit negative feedback. In *SIGIR*, pages 143–152, 2015. (Cited on page 36.)

[231] Z. Zhang and O. Nasraoui. Mining search engine query logs for query recommendation. In *WWW*, pages 1039–1040, 2006. (Cited on page 35.)

[232] F. Zhao and Y. Guo. Improving top-n recommendation with heterogeneous loss. In *IJCAI*, pages 2378–2384, 2016. (Cited on page 90.)

[233] M. Zoghi, T. Tunys, L. Li, D. Jose, J. Chen, C. M. Chin, and M. de Rijke. Click-based hot fixes for underperforming torso queries. In *SIGIR*, pages 195–204, July 2016. (Cited on page 90.)

Samenvatting

In een steeds meer gedigitaliseerde wereld is het moeilijk om ons een leven voor te stellen zonder interactie met digitale informatieobjecten. Het internet en mobiele apparaten stellen mensen in staat om eenvoudig toegang te krijgen tot informatie: of het nu gaat om het lezen van het populairste onderzoeksartikel of het beantwoorden van e-mails van een collega ver weg, het is slechts een kwestie van een paar toetsaanslagen, klikken of vegen op *touch screens*. Met de recente vooruitgang in natuurlijke taalverwerking en de toepassing ervan in slimme apparaten zoals Alexa en Google Home, kunnen mensen krijgen wat ze willen, *handsfree* en via spraakopdrachten. Als gevolg hiervan zijn we getuige van een schat aan gebruikersinteracties op allerlei online platforms. Door deze gebruikersinteracties te bestuderen, kunnen we de informatiebehoeften van gebruikers, hun gedragspatronen en problemen of fouten begrijpen wanneer ze met een informatiesysteem omgaan. Uiteindelijk werpen deze observationele inzichten licht op mogelijke richtingen om het systeem en de gebruikerservaring te verbeteren. In dit proefschrift hebben we gebruikersinteracties bestudeerd in het domein van academische zoek- en aanbevelingssystemen en in zakelijke e-mails.

In het eerste deel van het proefschrift concentreren we ons op het blootleggen van hoe academische zoekers omgaan met informatieobjecten. We beginnen met het karakteriseren van academische zoekopdrachten en ontdekken dat deze verschillen van algemene zoekopdrachten op het web. Academische zoekopdrachten zijn over het algemeen complexer en bevatten meer entiteiten. In onze context vinden we ook gevallen waarin gebruikers aanlopen tegen falen van een zoekmachine, waardoor geen zoekresultaat terug gegeven wordt. Gebruikmakend van de kenmerken van academische zoekopdrachten en sessie-informatie wanneer gebruikers zoekopdrachten uitvoeren, vinden we dat het mogelijk is om goede aanbevelingen voor zoekvragen te suggereren om gebruikers in nood te helpen. Verder gaan we na hoe gebruikersgedrag over een langere periode verloopt. We bekijken met name de herformulering van de vraag en de verschuiving van onderwerpen. We identificeren meerdere herformuleringsstrategieën voor zoekopdrachten en vinden dat het opnieuw bezoeken van vragen heel gangbaar is. We zoeken naar correlaties tussen herformulering van vragen en verschuiving van onderwerpen, ervan uitgaande dat bepaalde herformuleringen kunnen aangeven hoe gebruikers van onderwerp veranderen. Tot onze verbazing vinden we op de lange termijn weinig correlatie. Verschuiving van onderwerp op korte termijn is gecorreleerd aan bepaalde herformuleringstypes, zoals het indienen van nieuwe vragen. Na het onderzoeken van het zoekgedrag van gebruikers bestuderen we ook hoe gebruikers artikelen downloaden. We karakteriseren hun downloadgedrag zowel binnen sessies als tussen sessies, en observeren ook verschillende patronen tussen disciplines. Met behulp van de observationele inzichten stellen we de taak voor om *downloads* van gebruikers te voorspellen, gebruikmakend van LSTM-gebaseerde modellen in combinatie met gebruikerssegmentatie. Een ander interessant scenario dat we bestuderen, betreft gebruikersinteracties met aanbevelingen voor papier. We bestuderen een aanbevelingssysteem dat artikelen aanbeveelt via nieuwsbrieven en bestuderen in het bijzonder de taak om de aanbevelingen opnieuw te rangschikken, met behulp van een hybride *ranking* model dat inhoud en gedrag in overweging neemt.

In het tweede deel van het proefschrift concentreren we ons op hoe gebruikers hun zakelijke e-mails lezen en hoeveel tijd ze hieraan besteden. E-mails zijn belangrijk in academisch onderzoek en communicatie. Zoals bestudeerd in het eerste deel van het proefschrift, kunnen gebruikers aanbevelingen van artikelen lezen over e-mails. Onze studie is de eerste om de leestijd van de gebruiker op grote schaal te karakteriseren. We vinden dat de leestijd gecorreleerd is met veel contextuele factoren. De resultaten verbeteren ons begrip van gebruikersgedrag op e-mailplatforms en werpen ook licht op systeemverbeteringen om het lezen van e-mail efficiënter te maken.

Summary

In an increasingly digitized world, it is hard to imagine a life without interacting with digital information objects. The internet and mobile devices enable people to access information with ease: be it reading the hottest research paper, or replying to emails from a colleague far away, it is just a matter of a few key strokes, clicks, or swipes on touchscreens. With recent advances in natural language processing and its application in smart devices such as Alexa and Google Home, people can even get what they want hands-free and through voice commands. As a result, we are witnessing a wealth of user interactions on all kinds of online platforms. Studying these user interactions help us understand users' information needs, their behavior patterns and difficulties or failures when they interact with the system. Eventually, these observational insights shed light on possible directions to improve the system and the user experience. In this thesis, we have studied user interactions in the domain of academic search and recommender systems, and in enterprise emails.

In the first part of the thesis, we focus on uncovering how academic searchers interact with information objects. We start by characterizing academic search queries, and find that they are different from general web search queries. Academic search queries tend to be more complex, and contain more entities. In our setting, we also find cases when users encounter query failures that lead to no search result. Utilizing the characteristics of academic search queries, and session information when users conduct searches, we find that it is possible to suggest good query recommendations to help users in need. Moving on, we examine user behavior observed over a longer period. In particular, we look at query reformulation and topic shift. We identify multiple query reformulation strategies, and find that revisiting queries is especially common. We look for correlations between query reformulation and topic shift, assuming that certain reformulations may indicate how users change their topic. To our surprise, we find little correlation in the long term. Topic shift in the short term is correlated with certain reformulation types, such as submitting new queries. After examining users' query behavior, we also study how users download papers. We characterize their download behavior both within sessions and across sessions, and also observe different patterns among disciplines. Using the observational insights, we propose the task of predicting user downloads, using LSTM-based models in combination with user segmentations. Another interesting scenario that we study concerns user interactions with paper recommendations. We study a recommender that sends out paper recommendations through newsletters and propose the task of reranking the recommendations, using a hybrid reranking model that considers both content and behavior.

In the second part of the thesis, we focus on how users read their enterprise emails and how much time they spend doing so. Emails are important in academic research and communication. Users can also read paper recommendations on emails. Our study is the first to characterize user reading time at a large scale. We find that reading time is correlated with many contextual factors. The results improve our understanding of user behavior on email platforms, and also shed light on system improvements to make email reading more efficient.