SCUOLA DI DOTTORATO
UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA

Department of

Informatics, Systems and Communication

Ph.D. program in Computer Science Cycle XXXIV

Neural Approaches to Personalized Search

Author: Elias Bassani

Registration number: 74803

Tutor: Prof. Giuseppe Vizzari

Supervisor: Prof. Gabriella Pasi

Coordinator: Prof. Leonardo Mariani

ACADEMIC YEAR 2021/2022

To my parents.

Well, they say it right when they flood the house, and they tear it to shreds that, like, uh, destruction is a form of creation. So the fact that they burn the money is ironic. They just want to see what happens when they tear the world apart. They want to change things.

— Richard Kelly, *Donnie Darko*

People sometimes say "There must be more than just this world, than just this life". But how much more do you want? We are going to die, and that makes us the lucky ones. Most people are never going to die because they're never going to be born. The number of people who could be here, in my place, outnumber the sand grains of Sahara. If you think about all the different ways in which our genes could be permuted, you and I are quite grotesquely lucky to be here. The number of events that had to happen in order for you to exist, in order for me to exist... we are privileged to be alive and we should make the most of our time on this world.

— Richard Dawkins, *The God Delusion*

ABSTRACT

NEURAL APPROACHES TO PERSONALIZED SEARCH

OCTOBER 2022

ELIAS BASSANI

B.Sc., UNIVERSITY OF MILANO-BICOCCA

M.Sc., UNIVERSITY OF MILANO-BICOCCA

Ph.D., UNIVERSITY OF MILANO-BICOCCA

Supervised by: Professor Gabriella Pasi

The recent advancements in Neural Networks research have pushed forward the state-of-the-art in many language-related tasks, including Information Retrieval, bringing new opportunities for representing and leveraging user-related information during personalization. However, their application in the context of Personalized Search is still an open research area, with many issues and challenges to be addressed and tackled. In this dissertation, we focus on representing the user preferences from multiple perspectives, managing and selecting the user information to personalize the current search, and improving query representations with user-specific data by proposing new approaches based on Neural Networks. Moreover, we address the lack of publicly available large-scale datasets suited for training and evaluating Neural Networks-based approaches for Personalized Search. We first study the problem

of leveraging the user preferences represented from multiple perspectives by proposing a multi-representation re-ranking model. We show that our proposed approach achieves competitive performance while being fast, scalable, and extended to include additional representations and features. We then conduct an in-depth analysis of a Neural Networks mechanism, the Attention, when employed for user modeling, highlighting some shortcomings due to one of its internal components, the Softmax normalization function. We address those shortcomings by introducing a novel Attention variant, the Denoising Attention, that adopts a more robust normalization scheme and employs a filtering mechanism. Experimental evaluations clearly show the benefits of our proposed approach over other Attention variants. Furthermore, we address the enhancement of query representations with user-specific data by proposing a novel Personalized Query Expansion approach designed for contextualized word embeddings, which leverages an offline clustering-based procedure to identify the user-related terms that better represent the user interests. We show it improves in terms of retrieval effectiveness over word embedding-based Query Expansion methods at the state-of-the-art while also achieving sub-millisecond expansion time thanks to an approximation we propose. Finally, we discuss the state of Personalized Information Retrieval evaluation and the available publicly available datasets and propose and share a novel large-scale benchmark across four domains, with more than 18 million documents and 1.9 million queries. We present a detailed description of the benchmark construction procedure, highlighting its characteristics and challenges, and provide baselines for future works. The solutions and findings presented in this dissertation show that Personalized Search is still an open research area. Moreover, the new opportunities brought to the table by the recent advancements in Neural Networks also introduce new challenges that need to be correctly addressed to both take full advantage of their potential and make them valuable for real-world Personalized Search applications.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Information Retrieval is a scientific discipline concerned with finding information, primarily in the form of unstructured textual documents, in response to the information needs of the users. Usually, users express their information needs as textual queries, which they issue to an Information Retrieval System, *i.e.*, a search engine. In response, the system provides lists of documents ranked by their estimated relevance w.r.t. the user queries, and their underlying information needs. Every day, Information Retrieval technologies enable billions of users to find the information they need through Web search engines and domain-specific retrieval systems. The concept of the relevance of a document w.r.t. to a user query is often inherently subjective as user-specific preferences play a crucial role in the users' perception of relevance. Multiple users issuing the same query can find different documents more appropriate to answer their specific information needs. For example, users looking for events happening in the city they live could search for "events in Milan today" regardless of whether they are interested in live music shows or visual art events.

In the past two decades, academia and industry have put much effort into tailoring search results to specific user preferences by leveraging previously-gathered and heterogeneous user-specific information during the information retrieval process. However, despite the conspicuous number of previous studies, Personalization in Information Retrieval is far from being a solved task. Novel techniques for representing user-related information and documents have recently brought new opportunities

and challenges to the field. In particular, the recent advancements in Deep Learning and its Representation Learning sub-field have paved the way for designing novel approaches for Personalized Search. Deep Learning techniques allow building rich representations of textual information by projecting text into latent vector spaces that embed the language semantics. These representations of text are commonly called word embeddings. Word embedding techniques have pushed forward the state-of-the-art in many language-related tasks, including Information Retrieval. Their adoption allows information systems to leverage the semantics of the documents' contents and exploit it during retrieval, overcoming some limitations of traditional approaches. However, their usage in the context of Personalization is still an open research area. For example, how to represent the user preferences inferred from multiple perspectives and signals and how to leverage them for retrieval purposes is still an open issue. Similarly, how to correctly manage and select the user information to be used for personalizing the current search conducted by the user is an almost entirely unexplored research area. Another open issue is how to effectively and efficiently enhance query representations based on word embeddings with user-specific information. Finally, the data-hungry nature of Deep Learning techniques and the lack of publicly available large-scale benchmark datasets suited for evaluating Personalized Search Systems poses additional challenges for evaluating Deep Learning-based personalization approaches.

In this dissertation, we deal with the aforementioned open challenges of Personalization in Information Retrieval. In the following sections, we review the topics, challenges, and tasks studied in this dissertation and briefly introduce our contributions, outlining the content of the subsequent chapters.

1.1 Multi-Representation User Modeling

In real-world applications, user-related information of multiple natures is often available. We can leverage this information to infer both document properties and related users' interests. For example, users frequently generate textual content in the form of comments on news articles or reviews describing their experience with some product. This content usually includes information regarding both the characteristics of the object under consideration and the related users' opinions. Moreover, as documents are often classified in topical categories, we could infer the users' categorical interests by mining the user interactions with them. The mined categorical interests can contribute to delineating user profiles. Finally, the similarity between a user's behavior and those of other users can provide further information to personalize the user's subsequent interactions with the system. Making full use of these different kinds of information, accounting for multiple perspectives of the user's preferences, could allow a more informed representation of the user model. During the retrieval process, a system can rely on this representation to deliver personalized search results, thus enhancing its retrieval capabilities.

In this dissertation, we study the problem of leveraging multiple personalized relevance signals in the context of Product Search (*i.e.*, search on e-commerce websites). In particular, we propose a multi-representation re-ranking model based on the fusion of scores computed by comparing distinct user and product representations, which support multiple perspectives. More specifically, the representations we propose account for both content-based information extracted from user reviews and item categories and collaborative information extracted from user-item interactions. The proposed approach is fast and scalable, can be added to the top of any search engine, and can be extended to include additional representations and features.

1.2 Query-Aware User Modeling

Two main challenges of Personalized Search are *how* and *when* to personalize the user search results. As users have multiple and diverse interests, not all the information we can collect about their preferences is equally related to all their searches. Thus, it is natural that some of the data gathered about a specific user will be helpful to personalize some of her queries and not others. Moreover, it could happen that by relying on user-related information unrelated to her current search, a personalization procedure could negatively impact the overall retrieval process. For example, this situation could occur when the user's preferences towards a specific domain are unknown. In this case, if the system tries to personalize the search results, it could potentially decrease the retrieval effectiveness.

With the recent advances in the research related to Neural Networks, a new trend in Personalized Search emerged. Commonly called query-aware user modeling, it consists in building a representation of the user preferences, *i.e.*, the user model, at query time. In such a user modeling approach, the sources of user interests are weighed w.r.t. the current search by assigning more importance to those most related to the query. This procedure, usually carried out by employing the neural Attention mechanism, allows to automatically discern between beneficial and noisy user-related information on a query basis.

In this dissertation, we conduct an in-depth analysis of the Attention mechanism when employed for query-aware user modeling, highlighting some shortcomings due to one of its internal components, the Softmax normalization function. In user modeling, Softmax can cause the model to be excessively noisy or skewed towards a single source of user interest. This component also causes personalization to be performed even when those sources are not related to the current search conducted by the user, potentially harming the retrieval process. We address the previous shortcomings by introducing a novel Attention variant, the Denoising Attention, which

adopts a more robust normalization scheme and employs a filtering mechanism. We specifically design the Denoising Attention's components to finely filter out noisy user-related information and produce a balanced representation of the user interests w.r.t. the current search.

1.3 Personalized Query Expansion

Users often rely on short keyword queries to express their information needs when interacting with Information Retrieval systems. While formulating simple queries allows users to retrieve documents rapidly, a system may struggle to provide satisfactory results when the users provide only a broad description of their information needs. One well-known solution to this problem is Query Expansion, the task of reformulating the initial user query with additional terms to improve the retrieval effectiveness of the system. This process can be performed on a user basis, taking into account the user-related information to extract the terms to add the query. In this case, we talk about Personalized Query Expansion.

Recent approaches to this task rely on word embeddings to select the additional terms from the user-related information. Although delivering promising results with former word embedding techniques, we argue that these methods are not suited for use with the more recent and powerful contextual word embedding approaches, which produce a unique vector representation for each term occurrence in texts by accounting for its context. Specifically, current methods have a high probability of selecting redundant expansion terms as, with contextual word embeddings, we deal with many potentially similar vector representations. Moreover, as those approaches rely on computing a similarity score between the query and each user-related term embedding to select those most appropriate for expanding the query, they can introduce an overhead directly proportional to the number of candidate expansion

terms. If not carefully handled, these issues could slow down the retrieval process, impacting the system's scalability.

In this dissertation, we address these issues by introducing a novel Personalized Query Expansion method designed to take advantage of contextual word embeddings in this task. Specifically, by leveraging an offline clustering-based procedure to group the embeddings of the user terms and identify those that better represent the user interests, it avoids selecting multiple expansion terms of similar meanings. In addition, by employing an approximation procedure based on the user term clusters during the expansion terms selection, our approach introduces very low latency in the retrieval process, even in very data-rich scenarios. Finally, we introduce a novel metric to evaluate the query expansion terms diversity and empirically show the unsuitability of previous Personalized Query Expansion approaches based on word embeddings when employed along with contextual word embeddings, which cause the selection of multiple semantically overlapping terms for expanding the query.

1.4 Personalized Search Evaluation

Personalization in Information Retrieval has been a hot topic in both academia and industry for the past two decades. However, there is still a lack of high-quality standard benchmark datasets for conducting offline comparative evaluations in this context. In the past, the Text Retrieval Conference (TREC) promoted some evaluation campaigns targeting Personalized Search. However, those initiatives are limited in the amount of user-related information made available, thus reducing their appeal for evaluating modern Deep Learning-based Personalized Search methods, which require large amounts of data to be trained. For this reason, the problem of defining a standard approach to the evaluation of Personalized Search is still a hot research topic. Recently, some efforts have been made to define large-scale datasets suitable for

evaluating Personalized Search approaches. Unfortunately, all these datasets come with specific issues, from content availability and privacy concern to anonymized texts, which make personalized semantic retrieval approaches not usable. In the past few years, researchers have proposed several methodologies to overcome those issues and derive synthetic datasets for evaluating Personalized Search models. Unfortunately, due to data-quality issues, concerns have been raised about adopting as standard benchmarks for Personalized Search the synthetic datasets obtained by employing those methods.

In this dissertation, we first discuss the methodologies previously proposed for deriving synthetic datasets suitable for evaluating Personalized Information Retrieval approaches, highlighting some problems that affect them. Then, we revisit and extend the most promising of those methodologies, overcoming their limitations while preserving their benefits, and propose and share a novel large-scale benchmark across four domains, with more than 18 million documents and 1.9 million queries, designed for the evaluation of Personalized Search approaches. We present a detailed description of the benchmark construction procedure by highlighting its characteristics and challenges. We also provide baselines for future works, opening room for the evaluation of Personalized Search approaches, as well as Domain Adaptation and Transfer Learning methods in the context of Personalization.

1.5 Other Works

Besides the works already introduced, this dissertation also covers additional works regarding the Semantic Labeling of Information Retrieval queries and the work undertaken in the implementation of an open-source tool for the evaluation, comparison, and fusion of the ranked lists produced by Information Retrieval systems in response to a query.

1.5.1 Semantic Query Labeling

Searching in a domain-specific corpus of structured documents (*e.g.*, e-commerce, media streaming services, job-seeking platforms) is often managed as a traditional retrieval task. Semantic Query Labeling is the task of locating the constituent parts of a query and assigning domain-specific predefined semantic labels to each of them. It allows unfolding the relations between the query terms and the documents' structure while leaving unaltered the keyword-based query formulation typical of the Information Retrieval Systems. Due to both the lack of a publicly available dataset and the high cost of producing one, there have been few published works in this regard.

In this dissertation, we first introduce a novel large-scale dataset of manually annotated queries in the movie domain that are suitable for studying Semantic Query Labeling. Then, based on the assumption that a corpus already contains the information the users search for, we propose a method for the automatic generation of semantically labeled queries and show that a semantic tagger trained on our synthetic queries achieves results comparable to those obtained by the same model trained with real-world data. We also investigate whether pre-training the model with synthetic queries can improve the performance of the model trained only with real-world data. Lastly, by simulating a dynamic environment, we evaluate the consistency of performance improvements brought by pre-training as real-world training data becomes available.

1.5.2 `ranx`

Offline evaluation and comparison of different Information Retrieval systems is a fundamental step in developing innovative solutions. A few years ago, the introduction of `trec_eval`[1] by the Text Retrieval Conference (TREC) allowed standardizing evaluation metrics in Information Retrieval. This handy tool comes as a standalone

[1] https://github.com/usnistgov/trec_eval

C executable that researchers and practitioners must compile and run through a command-line interface. Unfortunately, it does not provide additional functionalities, such as comparing results from different Information Retrieval systems or exporting the evaluation response to specific formats (*e.g.*, LaTeX). Nowadays, the large majority of Information Retrieval researchers rely on Python as their primary coding language. Because of that, many recent tools provide experimentation and evaluation utilities in Python, such as evaluation metrics. Nevertheless, we think there is still the need for a user-friendly Python library following a truly *Plug & Play* paradigm, which can also be helpful for young researchers with different backgrounds.

For this reason, in this dissertation, we present `ranx`[2]: a library of fast ranking evaluation metrics implemented in Python with modern technologies that allows for high-speed vector operations and automatic parallelization. `ranx` offers a user-friendly interface to compute multiple evaluation metrics, run statistical tests, and visualize comparison summaries, all in a few lines of code. Furthermore, it offers a convenient way of managing the evaluation results, allowing the user to export them in LaTeX format for scientific publications. We also recently extended `ranx` with several Metasearch algorithms and dedicated functionalities previously unavailable to the research community.

[2] https://github.com/AmenRa/ranx

1.6 Outline and Contributions

The following chapters of this dissertation are organized as follows:

1. In Chapter 2, we introduce the fundamental concepts of Personalization in Information Retrieval and discuss how it is usually carried out.

2. In Chapter 3, we introduce a novel personalized results re-ranking approach for Product Search. The proposed model is based on the fusion of the relevance score deriving from multiple user/item representations, accounting for both content-based information (*i.e.*, reviews, categorical information) and collaborative information (*i.e.*, representations extracted from the user-item interaction graph). Furthermore, the approach we introduce is fast and scalable, can be easily added on top of any search engine and it can be extended to include additional user/item representations. The performed comparative evaluations show that our model outperforms modern Neural Network-based personalized retrieval models for Product Search in the great majority of cases.

3. In Chapter 4, we analyze the effects of the Attention mechanism when employed for query-aware user modeling, highlighting some shortcomings that can cause the user model to be excessively noisy or skewed towards a single source of user interest. We address those shortcomings by introducing a novel Attention variant called Denoising Attention. By employing a robust normalization scheme and a filtering mechanism, our proposal can finely filter out noisy user-related information and produce a balanced representation of the user interests w.r.t. the current search, outperforming other attention variants when it comes to personalization.

4. In Chapter 5, we introduce the a novel Personalized Query Expansion method designed for contextual word embeddings. Our approach employs an offline

clustering-based procedure to group the user-related terms and identify those that better represent the user interests. This mechanism allows us to avoid adding redundant expansion terms to the query, a very relevant problem when employing contextual word embeddings for Query Expansion. Furthermore, by implementing an approximation mechanism based on the user term clusters, our Personalized Query Expansion method achieve extremely low latency while outperforming previously proposed methods at the state-of-the-art in terms of effectiveness.

5. In Chapter 6, we discuss the available benchmark datasets for Personalized Search evaluation and introduce a novel large-scale benchmark. In particular, the proposed benchmark spans across four domains and accounts for more than 18 million documents and 1.9 million queries. We also provide baselines for future works, opening room for the evaluation of Personalized Search approaches, as well as Domain Adaptation and Transfer Learning methods in the context of Personalization.

6. In Chapter 7, we first introduce a novel large-scale dataset of manually annotated queries for Semantic Query Labeling. Then, we introduce a method for generating semantically labeled synthetic queries directly from a document collection and show that a semantic tagger trained on those queries achieves results comparable to those obtained by the same model trained with real-world data. We also investigate the effect of pre-training the model with the synthetic queries and assess whether such an approach can improve the performance of the model trained only with real-world data. Lastly, we simulate a dynamic environment and evaluate the consistency of the performance improvements brought by pre-training as real-world training data becomes available.

7. In Chapter 8, we present `ranx`, a Python library implementing a user-friendly

interface to compute Information Retrieval evaluation metrics, run statistical tests, and visualize comparison summaries, all in a few lines of code. It also offers a convenient way of managing the evaluation results, allowing to export them in LaTeX format for scientific publications. Finally, it provides several Metasearch algorithms and dedicated functionalities previously unavailable to the research community. By leveraging modern technologies for high-speed vector operations and automatic parallelization, `ranx` also outperforms currently available alternatives in terms of efficiency.

8. In Chapter 9, we summarize the contribution from the previous chapters, highlight some remaining open issues, and discuss several future directions.

1.7 Publications

The list of the papers related to the research activity presented in this dissertation follows below. The contributions of the author of this dissertation, Elias Bassani, to each paper are indicated using the Contributor Roles Taxonomy[3] (CRediT).

- **Elias Bassani** and Gabriella Pasi, "*A Multi-Representation Re-Ranking Model for Personalized Product Search*", in Information Fusion (*journal*), 2022.

  Contributions: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, and Writing - Original Draft.

  Discussed in Chapter 3.

- **Elias Bassani**, Pranav Kasela, and Gabriella Pasi, "*Denoising Attention for Query-aware User Modeling in Personalized Search*", *Submitted to journal*, *TBA*.

  Contributions: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, and Writing - Original Draft.

  Discussed in Chapter 4.

[3] https://www.elsevier.com/authors/policies-and-guidelines/credit-author-statement

- **Elias Bassani**, Nicola Tonellotto, and Gabriella Pasi, "*Personalized Query Expansion with Contextual Word Embeddings*", *Submitted to journal*, *TBA*.

  Contributions: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, and Writing - Original Draft.

  Discussed in Chapter 5.

- **Elias Bassani**, Pranav Kasela, Alessandro Raganato, and Gabriella Pasi, "*A Multi-domain Benchmark for Personalized Search Evaluation*", in Proceedings of the 31st ACM International Conference on Information and Knowledge Management, 2022.

  Contributions: Conceptualization, Software, Validation, Investigation, Resources, Data Curation, and Writing - Original Draft.

  Discussed in Chapter 6.

- **Elias Bassani** and Gabriella Pasi, "*Semantic Query Labeling Through Synthetic Query Generation*", in Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021.

  Contributions: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, and Writing - Original Draft.

  Discussed in Chapter 7.

- **Elias Bassani** and Gabriella Pasi, "*On Building Benchmark Datasets for Understudied Information Retrieval Tasks: the Case of Semantic Query Labeling*", in Proceedings of the 11th Italian Information Retrieval Workshop, 2021.

  Contributions: Conceptualization, Data Curation, and Writing - Original Draft.

  Discussed in Chapter 7.

- **Elias Bassani** and Gabriella Pasi, "*Evaluating the Use of Synthetic Queries for Pre-training a Semantic Query Tagger*", in Proceedings of the 44th European Conference on Information Retrieval, 2022.

  Contributions: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, and Writing - Original Draft.

  Discussed in Chapter 7.

- **Elias Bassani** "*ranx: A Blazing-Fast Python Library for Ranking Evaluation and Comparison*", in Proceedings of the 44th European Conference on Information Retrieval, 2022.

  Contributions: Conceptualization, Software, Resources, and Writing - Original Draft.

  Discussed in Chapter 8.

- **Elias Bassani** "*Towards an Information Retrieval Evaluation Library*", in Proceedings of the 12th Italian Information Retrieval Workshop, 2022.

  Contributions: Conceptualization, and Writing - Original Draft.

  Discussed in Chapter 8.

- **Elias Bassani** and Luca Romelli "*ranx.fuse: A Python Library for Metasearch*", in Proceedings of the 31st ACM International Conference on Information and Knowledge Management, 2022.

  Contributions: Conceptualization, Software, Resources, and Writing - Original Draft.

  Discussed in Chapter 8.

CHAPTER 2

PERSONALIZATION IN INFORMATION RETRIEVAL

Personalization in Information Retrieval aims to tailor search results to specific users to overcome the one-size-fits-all behavior of search engines. In this context, Personalization is mainly conducted by refining the users' search queries and adapting search results to their specific preferences. Personalization pipelines usually comprise three stages [96, 161]: 1) user-related information gathering, 2) representation of the gathered information to define a user model, and 3) exploitation of the user-related information to improve the quality of the search outcome. In the following sections, we introduce and review those stages.

2.1 User-Related Information Gathering

Gathering user-related information means employing different tools and approaches to collect information about the users' interests and preferences that the system can use to create profiles for the users to exploit during personalization. This process can be carried out both explicitly by asking for explicit relevance feedback from the users [189, 61, 112] (*i.e.*, asking for users' preferences, interests, opinions about search results, etc.) and implicitly by collecting implicit relevance feedback from the users' behavior and related contents [252, 245, 250, 62, 3, 260], such as previous queries, clicks, browsing activity, authored contents, email, etc., and/or contextual factors (*i.e.*, user background, search task, etc.). Although users may be a good

source of relevance feedback, multiple works have shown the usefulness of directly asking the users for feedback to be limited by the tradeoff between user effort and performance improvement [27, 229]. Moreover, concerns have been raised regarding the additional time and effort required to supply the information needed to assist the personalization process, which is also sometimes provided inconsistently by the users [49, 56]. Generally, systems that collect implicit information are more likely to be used as they do not require the users to perform any additional actions other than those they normally carry out during a search session. Furthermore, in practice, they perform as well or better than those requiring explicit user feedback to be collected [94]. Nowadays, studies on Personalization and industrial applications widely rely on implicit relevance feedback because of its unobtrusive nature for the user and lower data collection cost, which allows gathering large amounts of data for training Deep Learning-based Personalization models. In this dissertation, we mainly rely on implicit feedback as our main source of user-related information.

2.2 Definition of the User Model

At the core of Personalization is User Modeling, the process of building a representation of the previously-gathered user-related information to assist the personalization process. Over time, researchers have proposed several modeling approaches to represent the information about the users, from methods based on language [258, 248] and topic [116, 53, 284] modeling to Deep Learning models [153, 247, 295, 274]. The outcome of this process is a user profile that stores the user's preferences and search interests.

User modeling techniques are traditionally classified as long-term and short-term, depending on whether they embed the general and persistent user interests or the immediate preferences related to the current user's information need. Generally

speaking, long-term user models are derived from information gathered over long periods, spanning multiple search sessions, and employed for personalization in the long run, while short-term user models are based on the user behavior in the current search session and are used to personalize the search outcome in the same session. Short-term user models play the same role as long-term profiles in providing the system with additional information to personalize search results.

The kinds of user-related information flowing into short-term and long-term user models greatly impact their personalization potential. On the one hand, short-term user models are much more focused on the user preferences regarding her immediate information needs. However, they often struggle to provide meaningful improvements to the retrieval process at the very beginning of a search session as the system collects user-related information from the session itself. On the other hand, long-term user models, by accumulating user-related information over time, allow the system to leverage information regarding a wider variety of user interests during personalization and can do it successfully even for the first search in a session. Unfortunately, the broader scope of user-related information accumulated in the long-term user models could eventually lead to noisy — w.r.t. the current search — user representations if not carefully handled, potentially harming the retrieval improvements that personalization can provide. Not surprisingly, a number of studies [31, 83] have found a conjunct use of both short-term and long-term user models to improve personalization performances over those obtained by the two approaches taken separately. In general, the more information about the user we provide to the system, the better it can understand the user's interests and preferences, leading to better retrieval performances.

In this dissertation, we focus on long-term user modeling and how to manage and use the user-related information collected over time to robustly enhance the retrieval process.

2.3 Exploitation of the User-Related Information

The last stage of the personalization process consists of the actual implementation and execution of personalization. Generally, a system implements personalization by adapting the user's query or the retrieved results. The adaptation process of a query takes place in a pre-processing phase and consists of expanding its original terms with additional ones, aiming to retrieve more relevant results [183]. This process is usually called Query Expansion. In the personalization context, the system generally draws the expansion terms from a user vocabulary inferred from user-related contents (*e.g.*, documents previously accessed or authored by the user, previous queries submitted to the system by the user). The adaptation of a results list is generally carried out by means of result scoring, results re-ranking, or result filtering [96]. Results re-ranking consists in performing an additional ranking round as a post-processing step to re-order the documents retrieved by the system in response to the user query, aiming to display specific results at higher ranks. In the context of personalization, re-raking aims to adapt the result lists to user preferences. Results filtering consists of hiding the documents that the system considers irrelevant w.r.t. the user interests. Results scoring techniques aim to incorporate adaptation features directly in the primary scoring function adopted by the system. In practice, Information Retrieval systems often incorporate both query and result adaptation components.

In the following sections, we introduce the two main personalization techniques we focused on in the works presented in this dissertation: 1) Query Expansion and 2) Results Re-Ranking.

2.3.1 Personalized Query Expansion

In this section, we introduce Personalized Query Expansion, one of the Personalized Information Retrieval tasks we tackled in the works presented in this dissertation.

Nowadays, most search engines provide users with a simple interface to specify their information needs through short keyword-based queries, which are usually two-to-three terms long in the case of Web search [124]. However, as a query only broadly describes a user's information need, search engines may struggle to provide satisfactory results. Multiple factors related to how users choose terms for their queries can affect a system's retrieval effectiveness [18]. For example, the terms composing a query can be related to multiple topics, leading the system to provide results not focused on the user's topic of interest. Moreover, out of habit, users often issue queries too short to clearly express complex information needs, ultimately failing to find documents valuable to fulfill them. Finally, users sometimes have only a broad idea of the information they need, and hence they issue queries that are not appropriate to find documents that can answer their information needs.

A well-known technique proposed to overcome those issues is Query Expansion, whereby the user's original query is augmented with new terms, known as *expansion terms*, to improve the system's effectiveness. The identification of proper expansion terms aims to clarify the user's search intent and bridges the gap between the original query terms and the documents' vocabulary [55], addressing the well-known vocabulary mismatch problem [90].

Query Expansion techniques can leverage user-related information previously gathered to derive the expansion terms, in which case we talk about Personalized Query Expansion. Personalized Query Expansion techniques rely on user-related documents, such as previously accessed Web pages and user-generated content [140], e.g., product reviews or tweets, to extract expansion terms directly from the users' vocabulary or the vocabulary used in documents of their interest.

A Personalized Query Expansion pipeline is depicted in Figure 2.1. As shown in the figure, the user issues a query to the system, which leverage the user profile to select the personalized expansion terms with which to expand the original query.

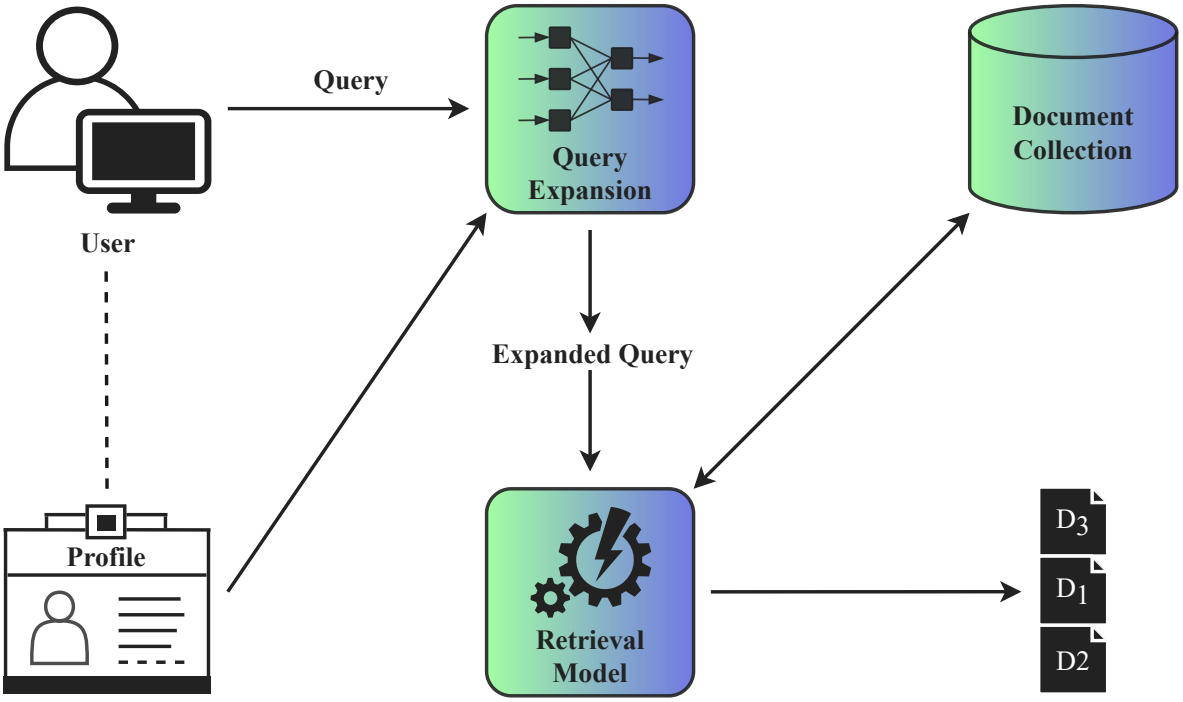


Figure 2.1: Personalized Query Expansion.

Then, the system provides the expanded query to a retrieval component, which computes a set of potentially relevant documents to show to the user. Finally, the ranked list of results is provided to the user.

2.3.2 Personalized Results Re-Ranking

In this section, we introduce Personalized Results Re-ranking, an Information Retrieval task we considered in many of the works presented in this dissertation for evaluating the proposed Personalized Search approaches.

In Results Re-Ranking, a retrieval system's component, commonly called first-stage retriever, retrieves a ranked list of documents in response to a search query. Then, another component, the re-ranker, computes new relevance scores for the initially retrieved documents leveraging additional information. Finally, the new relevance scores computed by the re-ranker, or a combination of those and the scores

computed by the first-stage retriever, are used to re-rank the initially retrieved list of documents. In the latter case, the two relevance scores are often aggregated via convex combination:

$$final\_score = (1 - \lambda) \cdot s_1 + \lambda \cdot s_2 \tag{2.1}$$

where, s_1 and s_2 are the relevance scores computed by the first-stage retriever and the re-ranker, respectively, and λ is a parameter that controls the influence of the two on the final score.

The main difference between the first-stage retriever and the re-ranker is that they usually differ in terms of efficiency and effectiveness. On the one hand, the first-stage retriever aims to efficiently retrieve an initial set of potentially relevant documents focusing on recall rather than precision. On the other hand, the re-ranker is generally a much more effective retrieval model but with little efficiency (*e.g.*, a Transformer-based ranker [158]), which is why it is employed to rank only a limited portion of the document collection, the top-n documents retrieved by the first-stage retriever. By combining those two components, we can efficiently retrieve most of the documents relevant to the query with the first-stage retriever and rank them appropriately by leveraging the effectiveness of the re-ranker, benefitting from the best of both worlds.

This process can also be carried out by leveraging previously-gathered user-related information during the re-ranking step, in which case we talk about Personalized Results Re-ranking. A Personalized Results Re-ranking pipeline is depicted in Figure 2.2. As shown in the figure, the user issues a query to the system, which retrieves an initial set of documents from the document collection by relying on the first-stage retriever. Then, the system provides the initially retrieved set of documents to the Personalized Re-Ranker, which computes their final ordering by leveraging the user profile, usually comparing the user's interests and preferences representation, *i.e.*, the user model, with those of the documents. Finally, the ranked list of results is provided to the user.

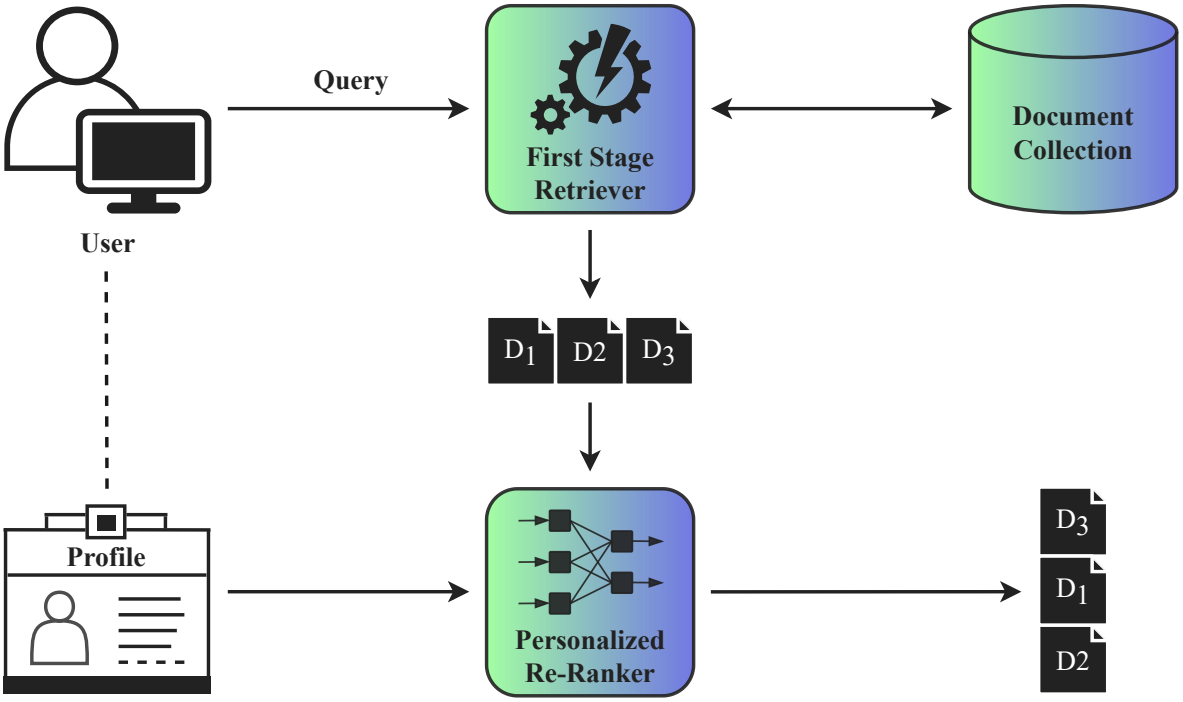


Figure 2.2: Personalized Results Re-Ranking.

In the case of Personalized Results Re-ranking, the λ parameter of Eq. 2.1 is generally used to control the influence of personalization on the ordering of the documents provided to the user. The higher the value of λ, the more personalized the search results shown to the user.

The Employed First-Stage Retriever: BM25

Since all the works on Results Re-Ranking presented in this dissertation rely on the classic retrieval model BM25 as a first-stage retriever, we introduce it in this section.

BM25 is a popular bag-of-words-based probabilistic relevance model [227] proposed by Robertson and Walker [226], which extends the classic TF-IDF relevance model [233]. BM25 assumes the relevance of a document w.r.t. a user query to be binary. It also assumes statistical independence between term occurrences to provide a simple and tractable scoring function, which assesses the relevance of a document

d w.r.t. a given query q as follows:

$$s_{q,d} = \sum_{i=1}^{|q|} IDF(q_i, C) \cdot \frac{tf(d_i, d) \cdot (k_1 + 1)}{tf(d_i, d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avg(|d|)})} \tag{2.2}$$

where $s_{q,d}$ is the relevance score of d w.r.t. q, q_i is the i-th query term, $IDF(q_i, C)$ is the inverse document frequency [129] of q_i in the corpus C, $tf(q_i, d)$ is the term frequency [171] of q_i in d, $|d|$ is the length of d, and $avg(|d|)$ is — with a slight abuse of notation — the average length of the documents in the corpus. The coefficients k_1 and b are the hyper-parameters of the model. The former is the term frequency saturation coefficient, which regulates the contribution of each term so that it cannot exceed a saturation point. The latter controls the normalization effect of document length. k_1 and b need to be tuned according to both the queries and the corpus for maximizing the model effectiveness.

We relied on BM25 as our first-stage retriever for several of the experimental evaluations discussed in this dissertation because, over the years, it has proven multiple times to be a robust retrieval model and a competitive baseline for retrieval evaluation. Moreover, it has been widely adopted in real-world applications thanks to being the default retrieval algorithm of the popular commercial search engine Elasticsearch[1].

[1] https://www.elastic.co

CHAPTER 3

A MULTI-REPRESENTATION RE-RANKING MODEL FOR PERSONALIZED PRODUCT SEARCH

The last 25 years have witnessed the birth of major e-commerce websites as well as a multitude of smaller ones. Online shopping is a popular activity nowadays, and it is expected to become even more popular in the next years, reaching over 2 billion people [253]. In 2020, retail e-commerce sales worldwide amounted to 4.28 trillion US dollars [254] and accounted for 18% of all retail sales [255].

Users often decide which items to buy after they have searched the available products through a search engine. In the context of Product Search, users' needs are highly personal, and a search engine should tailor the result lists on the user preferences, as users' diversity largely affects the notion of relevance of the retrieved products. Therefore, personalization is inherently an integral part of Product Search. Generally, e-commerce websites allow users to express their opinions and considerations on the products they have purchased. This feedback takes the form of ratings and reviews. Customers' reviews provide valuable information for modeling both users and items, as they contain clues about user preferences and product properties, which are often not specified in their descriptions. While user-generated content allows capturing the specificity and diversity of users, the analysis of users' purchasing behavior can provide complementary information to enrich both user and item representations. This information allows capturing the similarities among users as well as items' popularity.

To address personalization in Product Search many Neural Network-based retrieval models have been recently proposed. They make use of auxiliary side information to infer item properties and users' interest towards them. Recent efforts mainly focused on leveraging user reviews [6, 106, 296, 7, 33, 291, 8, 107], brands and categories [8], and product images [106] to personalize the user search experience. Guo et al. [107] also studied the effect of the long and short-term preferences in Product Search, while Zamani and Croft [291] proposed to jointly model personalization in Product Search and Product Recommendation tasks. Bi et al. [33] and Zhang et al. [296] approached the problem of personalization in Product Search in a conversational context. Inspired by Gysel et al. [109], these approaches share the assumption that Product Search is an inherently semantic task, due to the severe vocabulary mismatch [90] between user queries and item descriptions. Because of that, the authors model the internal query matching procedure on the semantic similarity between queries and item information, by mapping them in the same latent space.

Differently from recent works where personalization is directly injected into the retrieval model, we tackle personalization in Product Search as a results re-ranking task, where the list of items retrieved by a search engine is re-ranked based on the computation of a new relevance score obtained by fusing the relevance score assessed by the search engine with several user-item compatibility scores. More specifically, we propose a simple yet effective Personalized Results Re-Ranking model based on the fusion of the relevance score computed by the well-known ranking model BM25 [226] (introduced in Section 2.3.2) with a popularity-based value of the items (an important relevance signal that appears to have been overlooked in previous works) and three compatibility scores computed between latent representations of users and items built upon both content-based and collaborative information (*i.e.*, reviews, categorical information, purchasing behaviors). Despite the preeminent adoption of semantic matching-based models by recent works in the literature, we opt

for a classic lexical matching retrieval model. This choice is driven by the assumption that in Product Search, user queries usually contain "a producer's name, a brand or a set of terms that describe the category of the product" [230] and this kind of information is usually present as-is in product-related information. Finally, our approach is fast and scalable, it can be added on the top of any search engine, and it is easily extendable to accommodate additional relevance/compatibility scores.

To verify the effectiveness of the proposed approach we have performed several experiments. In particular, we have comparatively evaluated its effectiveness with respect to recently proposed Neural Network-based approaches specifically designed for Product Search [6, 8, 109], on a variety of datasets from Amazon[1], which have been previously employed in the literature. Our model consistently increases the retrieval effectiveness of the underlying retrieval model, BM25, and, in the great majority of cases, considerably outperforms modern Neural Network-based baselines.

The main contributions of this chapter are threefold:

- we propose a score fusion-based approach for personalized re-ranking in Product Search that leverages multiple user/item representations;
- the proposed model makes use of both content-based information (*i.e.*, reviews, categorical information) and collaborative information (*i.e.*, representations extracted from the user-item interaction graph);
- the proposed approach is fast and scalable, it can be added on top of any search engine and it is easily extendable to include additional user/item representations.

This chapter is structured as follows: after reviewing the related works related to Product Search in Section 3.1, we present the proposed Personalized Re-Ranking model in Section 3.2. In Section 3.3, we introduce the experimental setup of the

[1] https://www.amazon.com

performed evaluation, and in Section 3.4, we present and discuss the evaluation results, conducting an in-depth performance analysis.

3.1 Related Works

With the widespread of online shopping in the past few years, the task of Product Search has received increasing attention from the research community. Early works in this area focused on modelling the interaction between users and products-related information stored in relational databases through faceted search [28, 157, 270, 271]. Later, to reduce the gap between Web search, the kind of search to which users are most familiar, and search on products' structured data, which usually requires the formulation of structured queries, some works investigated the application of language modeling [218] approaches to Product Search [81, 80, 79]. In the meantime, other studies tackled the problem of results diversity [210, 290], paving the way for the more recent efforts on personalization in this area. More recently, learning-to-rank strategies were also studied in Product Search [15, 236, 122].

The great majority of the works recently published in the context of Product Search rely on the application of Neural Network-based models to tackle both problems of vocabulary mismatch between queries and item descriptions, and personalization. Gysel et al. [109] introduced a latent vector space model for Product Search to address the problem of vocabulary mismatch. The proposed model maps queries and items in the same latent space where their semantic similarity can be directly computed. Ai et al. [6] enhanced the approach proposed in [109] by adding personalization. This effect was obtained by mapping users in the same latent space of queries and items. Ai et al. [8] refined their first model by adding side information, such as item brands and item categories, to user and item representations. Guo et al. [107] modelled user preferences from both long-term and short-term perspectives. Guo

et al. [106] studied the effect of the visual modality on user preferences by leveraging item images in the personalization process. Their approach resulted particularly effective on fashion-related domains.

Other directions in the context of Product Search have also been explored. For example a recent study [7] addressed the problem of when and how to rely on personalization to enhance product retrieval. Zamani and Croft [291] investigated the possibility of jointly modeling and optimizing product retrieval and recommendation tasks, due to their complementary nature. They proposed a general framework to simultaneously learn a retrieval model and a recommendation model by optimizing a joint loss function. Bi et al. [33] and Zhang et al. [296] tackled Product Search from a conversational perspective. Lin et al. [159] proposed an unsupervised method relying on implicit user's feedback from clicks to collect a large amount of query classification data and highlighted some shortcomings of neural approaches in learning useful representations for queries, due to the fact that queries are composed of a few words. Sondhi et al. [246] recently focused on understanding user search behavior in E-Commerce search applications and how it relates to user query generation, by proposing a query taxonomy for Product Search. Other studies focused on query intent for query refinement [182] and term weighting [181], and perceived satisfaction [256] in Product Search.

In this chapter, we focus on the task of personalization in Product Search. In particular, instead of injecting personalization in the retrieval process, we propose a novel model for re-ranking the results produced by an underlying search engine; our model aims at fusing a number of different personalization scores, which are obtained by considering both content-based and collaborative features, related to both users and items. As the underlying search engine, we employ a classic probabilistic retrieval model, *i.e.* BM25, and we perform comparative evaluations finalized at the assessment of the effectiveness of our approach with respect to state-of-the-art

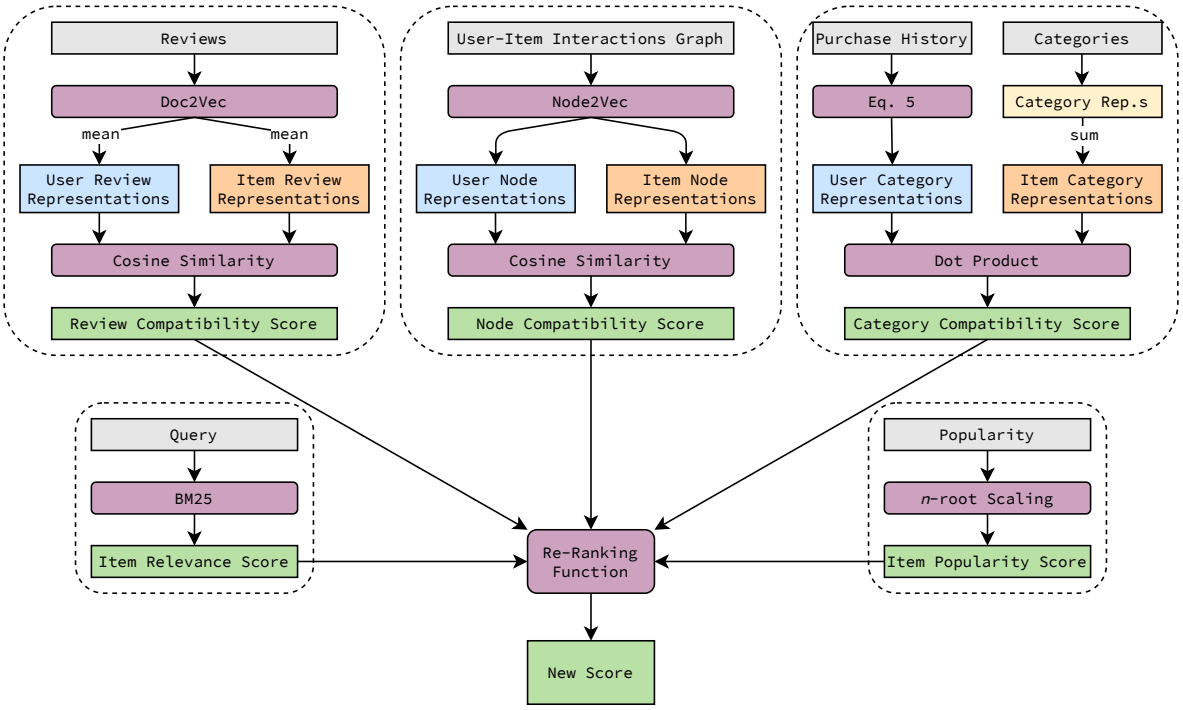


Figure 3.1: Overview of the re-ranking model.

approaches in personalized Product Search. We rely on the re-ranking process to both personalize the ranking of the initially retrieved relevant items and smooth the lexical matching score of BM25 with scores computed from latent representations of users and items. Finally, our approach is much less resource-intensive both in training and online with respect to the recent Neural Network-based approaches.

3.2 The Proposed Re-Ranking Approach

In this section, we introduce an extendable approach for Personalized Results Re-Ranking in Product Search. As previously outlined, our approach relies on a novel multi-faceted personalized re-ranking model applied to the ranked list of items (products) computed by a traditional search engine in response to a user query. The proposed re-ranking algorithm makes use of various information (relevance signals) related to users/items, which may indicate the possible relevance of the products w.r.t. the user's preferences. The considered relevance signals are carried by various

kinds of information shared by users and items (*e.g.*, reviews and user-system interactions) and by specific properties of the items, such as items' popularity. Each of the above kinds of information is formally represented, and a score associated with a given relevance aspect is computed either as a compatibility score between the related representation of users and items or as a score assessing the items' properties. The various relevance signals concur to an informed definition of the re-ranking process to make it effective, as they provide complementary information regarding the compatibility between users and items, which we express through the computation of distinct relevance scores, as previously commented. The re-ranking function employed by the proposed model is based on the fusion of these multiple scores, which provide evidence of the possible relevance of an item to a user from different perspectives. The approach we propose can be easily extended to accommodate additional representations to enrich the model.

In the system implemented and evaluated in this dissertation (Figure 3.1), we consider three relevance signals that are supported by three distinct formal representations of users and items: 1) a *review-based representation* built by employing the neural text embedding model PV-DBOW [147] on the product reviews written by the users, 2) an *interaction-based representation* built by leveraging the node embedding model Node2Vec [103] on the user-item interactions graph, which is built upon user-item purchasing relations, and 3) a *category-based representation* that captures the user's categorical interests towards the items' categories. For the items only, we also use a *popularity-based score*, which provides a valuable relevance indicator.

In the following, we first introduce the proposed user and item representations and the related compatibility scores. Finally, we outline the re-ranking function used to compute the personalized ranking scores for the items retrieved by BM25 in response to a user query.

3.2.1 Review-based Representations

A first information that may carry a relevance signal is constituted by the textual reviews that the users write to express their opinions towards the products they purchased. In fact, reviews are short textual descriptions that contain information regarding user preferences and product characteristics. We first collect the reviews written by each user and associated with each item. Then, we define a vector representation for each of the reviews by employing the PV-DBOW model [147], which we will describe later in this section, and compute the *review-based representations* for both users and items as the arithmetic mean of the vector representations of the related reviews, *i.e.*, the reviews written by the users for representing the users and the reviews associated with the items for representing the items. Finally, when a user queries the system, we compute a compatibility score for every item in the top-k results retrieved by BM25 as the *cosine similarity* between the user's and the items' *review-based representations*. The re-ranking function uses those scores to compute the personalized relevance scores used for re-ranking, as described later.

We now describe the neural text embedding model employed to create the review representations. PV-DBOW [147] is a text embedding model that maps documents — in our case reviews — into a low-dimensional latent vector space where semantically similar documents are close to each other. Following the *distributional hypothesis* [115, 87, 232] and similarly to the Word2Vec Skip-Gram model [147], where the latent representation of each word is learned by predicting nearby words, the PV-DBOW model is trained to predict for each document the words it contains. PV-DBOW operates under the bag-of-words assumption, *i.e.*, it assumes independence between words, and models the generative probability of a word w in document d (a *review* in our case) through the *softmax* function over the vocabulary as follows:

$$P(w|d) = \frac{exp(\vec{w} \cdot \vec{d})}{\sum_{w' \in V} exp(\vec{w'} \cdot \vec{d})} \tag{3.1}$$

where $\vec{w}$ and $\vec{d}$ are the vector representations of w and d, and V is the vocabulary of the training corpus. The softmax function outputs a probability distribution over the input. As the softmax function becomes prohibitively heavy to compute for large vocabularies, *Negative Sampling* [147] is employed to approximate its computations. *Negative Sampling* is a procedure that samples a certain number of words from the vocabulary, which are then used in the softmax instead of the whole corpus vocabulary, drastically increasing the training speed and avoiding unnecessary numerical computations.

3.2.2 Interaction-based Representations

The analysis of the user behavior can unveil meaningful information regarding the user's interests. In our case, the user behavior coincides with the actions performed by him/her on an e-commerce platform. Typically, those actions correspond to user-item interactions, such as visiting the page of an item (*view*), assigning a numerical evaluation to an item (*rate*), expressing an opinion towards an item through text (*review*), purchasing an item (*buy*), and so on. In our work, to define an interaction-based representation of both users and items, we consider only the interactions corresponding to purchasing actions.

First of all, we build the user-item interaction graph (Figure 3.2a). The nodes of this graph represent users and items while the edges represent the user-item interactions. Specifically, if a user purchased an item an edge is drawn between the node representing the user and the node representing the item. Secondly, we create vector representations of the user and item nodes by employing the Node2Vec model [103], which we will describe later in this section. Finally, similar to the review-based representations, when a user queries the system, we compute a compatibility score for every item in the top-k results retrieved by BM25 as the *cosine similarity* between the user's and the items' *interaction-based representations*. Again, the re-

ranking function employs those scores to compute the personalized relevance scores used for re-ranking.

We now review the node embedding model employed to create the node representations. Node2Vec [103] is a neural model that takes a graph as input and maps each node of the graph into a low-dimensional vector space where the nodes that share similar neighbors are close to each other. Firstly, Node2Vec generates training samples by employing a sampling strategy based on a random walk procedure. While the model explores the graph following the random walk procedure, it stores the sequences of nodes visited during each random walk (Figure 3.2b). These sequences are *ordered lists* of nodes' unique identifiers (*e.g.*, $walk = [n_1, n_5, n_2]$). Secondly, the model converts the node sequences to strings (*e.g.*, $[n_1, n_5, n_2] \rightarrow$ "$n_1\ n_5\ n_2$") so that they can be treated as sequences of words, *i.e.*, *sentences*. Finally, Node2Vec feeds the string version of the node sequences to the Word2Vec Skip-Gram model [192] that, by treating the node sequences as word sentences, learns vector representations of the nodes (Figure 3.2c). As previously described, Word2Vec learns representations of words by predicting nearby words, *i.e.*, the words that co-occur in the same sentences. As Node2Vec feeds Word2Vec with sequences of nodes, the model embeds the neighborhood information carried by the node sequences in the same fashion as learning representations of words by predicting their nearby words. By construction of the user-item interactions graph, the neighborhood of a user node comprises 1) the items he/she purchased, 2) the users who bought the same items purchased by the user, and 3) the other items they bought. Therefore, Node2Vec will map users with similar purchasing behaviors and the items they purchased in the same region of the latent space. Consequently, the similarity between the vector representations of user and item nodes is suitable to evaluate the likelihood of a user purchasing an item. As the vector representations embed behavioral neighborhood information, this approach is foundationally similar to the collaborative filtering approaches used

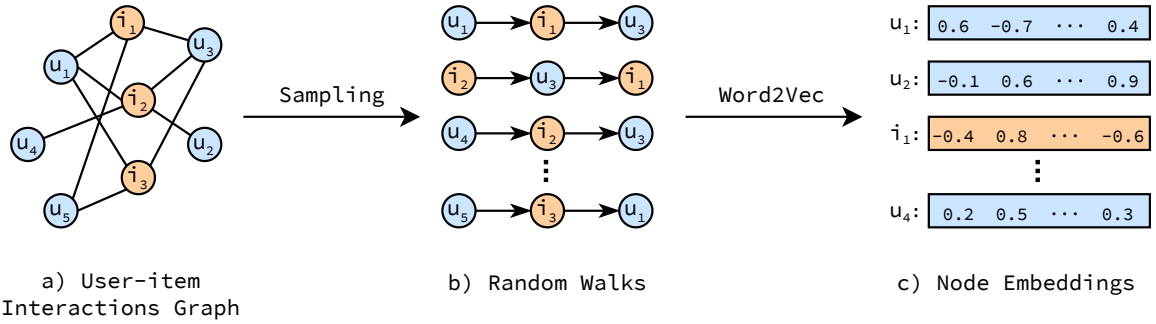


Figure 3.2: Node2Vec

in Recommender Systems.

3.2.3 Category-based Representations

E-commerce websites organize items into product categories (*e.g.*, smartphones, sports clothing, home products, etc.), which are often structured into hierarchies. This information can be leveraged to evaluate the user's interest in different product categories. In particular, by combining the customer's purchase history with the product categories of the items he/she purchased, we can infer his/her categorical interest towards unseen products. To do so, we rely on the item category tree structure found in many e-commerce platforms and the users' purchase history to define a *category-based representation* for both users and items, as described later in this section. Again, when a user queries the system, we compute a category interest-related score of the user towards every item in the top-k results retrieved by BM25 as the *dot product* between their *category-based representations*. The re-ranking function uses those scores to compute the personalized relevance scores used for re-ranking, as described later in this section.

We now describe the method proposed for modelling *the category-based representations* of both users and items. First of all, we leverage the hierarchical structure of the categories to assign to each of them a weight equal to the inverse of its position

in the hierarchy it belongs to, so that *root* categories weights more than *intermediate* and *leaf* categories (Figure 3.3a). The rationale behind this process is that generic categories better capture the long-term user interests (*e.g.*, a user that recently bought a new laptop will probably be more interested in acquiring computer accessories than another laptop). In case a category tree structure is missing, we can assign the same weight to each category. User and item *category-based representations* are then computed as follows. Firstly, categories are represented as *weighted* one-hot vectors, with the non-zero entries set to their associated weights (Figure 3.3a). Then, items are represented as the sum of the representations of the categories they belong to (Figure 3.3b). To map users in the category space, we first initialize their representations to a vector with all components set to 1:

$$user\_init_u = \vec{1} \tag{3.2}$$

In this way, we set a *minimum interest level* for all the categories regardless of the user purchase history. This initialization allows avoiding penalizing item categories for which the user interest is not known when we compute the compatibility scores (*dot product*) between users' and items' *category-based representations*. Then, we compute a vector representation of the user purchase history from a *category-based perspective* through a diminishing return formula based on exponential decay as follows:

$$purchase\_history_u = \vec{1} - exp(-\lambda \vec{p_u}) \tag{3.3}$$

where $\vec{1}$ is a vector with all components set to 1, $exp(\cdot)$ is the element-wise exponential function, λ is the decay constant and $\vec{p_u}$ is a vector representing the actual purchases of the user u. The components of vector $\vec{p_u}$ correspond to the item categories and its entries are equal to the number of items purchased by u in the corresponding categories. Eq. 3.3 has a smoothing effect that balances the representation of the

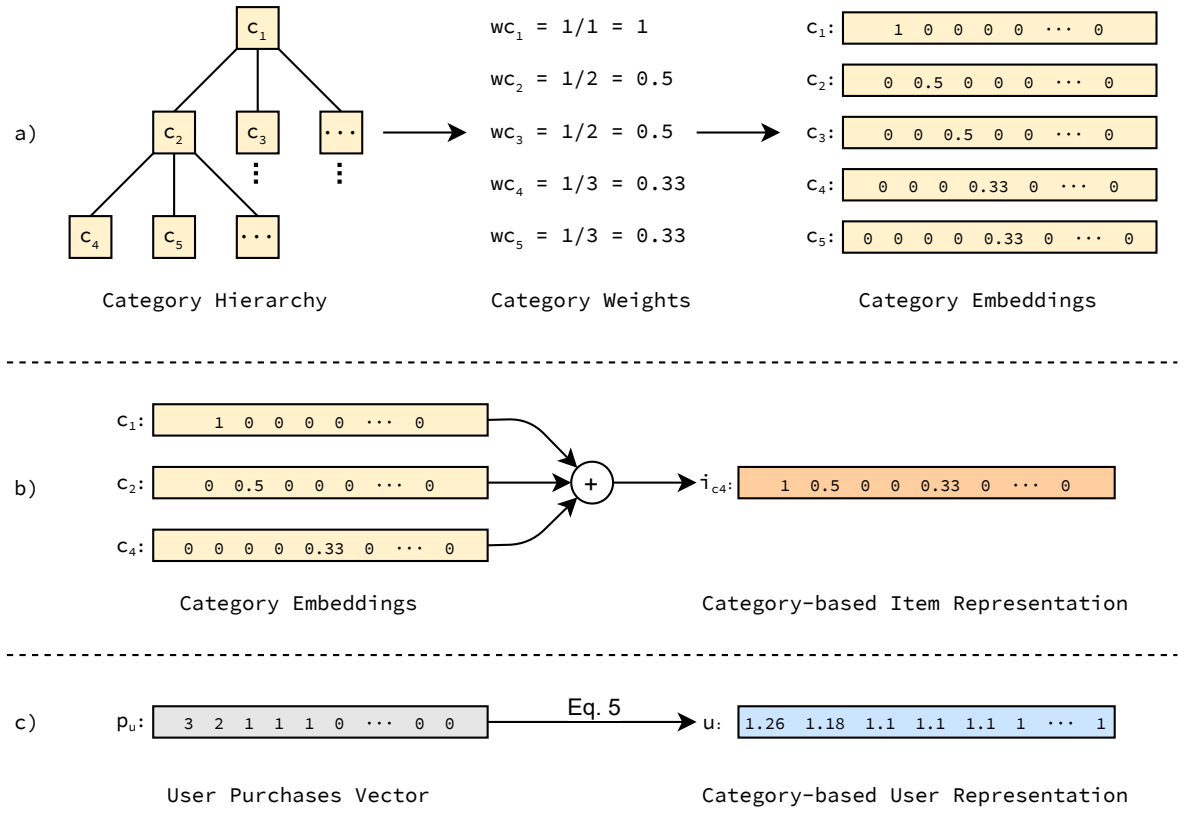


Figure 3.3: Category-based representations.

user's purchase history and avoids excessively skewing it towards already purchased items' categories. Finally, we sum the user's initialization vector with the vector representation of her/his purchase history:

$$user_u = user\_init_u + purchase\_history_u \tag{3.4}$$

If a user did not purchase items from a specific category yet, then the value of the corresponding component of the user representation will be equal to 1 as the purchase history vector's entry for that category will be zero. This mechanism allows not penalizing the products belonging to the categories for which we do not know the user's interest while increasing the importance of the other items according to the user's interest towards the related categories.

3.2.4 Item Popularity

Item popularity is inherently valuable in the context of Product Search as users often buy the most purchased, most rated, or most reviewed items corresponding to their needs. Therefore, we employ an item-popularity-based score during re-ranking to account for this relevance indicator and promote items usually purchased by the users. The popularity score of a given item is computed as the n-root of the total number of times the item has been purchased. The n-root scaling allows us to avoid penalizing low popular items and to smooth the popularity gap between low and high popular items. Although this score could skew the re-ranked results list towards popular items, the employed scaling mechanism, and the other scores — that are all personalized — counterbalance this effect.

3.2.5 Re-Ranking Function

As introduced in Section 2.3.2, Results Re-Ranking is usually performed by taking a convex combination of the scores computed by the first-stage retriever and the re-ranker, following Eq. 2.1. Since our approach accounts for multiple relevance scores besides that computed by the underlying search engine (BM25), we extend Eq. 2.1 to account for those multiple relevance scores and calculate the final score for each retrieved item i taking into account the preferences of the user u as follows:

$$new\_score_{u,q,i} = \left(1 - \sum_{k=1}^{n} \frac{w_k}{n}\right) \cdot r_{q,i} + \sum_{k=1}^{n} \frac{w_k}{n} \cdot s_k \tag{3.5}$$

where n is the number of the considered user/item representations; $r_{q,i}$ is the relevance score computed by the employed search engine — based on BM25 in our case — for the item i and a given query q. s_k and w_k are the values of the score related to the user/item representation k and the weight associated with it, respectively. In our case the scores are the cosine similarity between user and item review-based

representations, the cosine similarity between user and item interaction-based representations, the dot product between user and item category-based representations and the item popularity score. The value $\sum_{k=1}^{n} \frac{w_k}{n}$ in the linear combination depends on the weights associated with the considered representations. Both the representation weights and their sum range in the interval $[0, 1]$. As reported later (see Section 3.3.3), the representation weights choice has been automatically conducted as an hyper-parameter search problem using the Python package Optuna [9]. The search result lists are re-ranked according to the ranks computed by the linear combination in Eq. 3.5.

Depending on the items for sale in e-commerce sites, users may not want to buy the same product twice. This is not the case of grocery products, but if we think to products such as books or CDs a user may not like to see among the top ranked results the items he/she already purchased. To avoid this situation, we suggest to employ a simple mechanism to alter the ranking position of the already purchased items regardless of their relevance score. In the experiments presented in the following, we have pushed to the bottom of the re-ranked list the products already bought by the user as this is consistent with the datasets we have used for the evaluation of the proposed approach (see Section 3.3.1). However, other strategies can be employed for this purpose. For example, we could insert the already purchased items at a fixed position in the results list so that the user would be able to find them easily. We think there is no correct choice in general, as it depends on specific real-world applications, and it is strictly related to the products an e-commerce platform sells and to the behavior of its typical user.

3.3 Experimental Setup

In this section we describe the experimental settings and introduce the baseline methods used in our experiments to perform comparative evaluations. We also present and discuss our model settings and hyper-parameters.

3.3.1 Datasets

Due to the lack of publicly available datasets for Product Search, the Amazon Review 5-Core dataset [186] has been recently adopted as a benchmark dataset for Product Search along with synthetically generated queries [6, 106, 296, 7, 33, 291, 8, 107, 109], as it does not contain query logs. The Amazon Review 5-Core dataset has been widely used for the evaluation of Recommender Systems as it contains millions of users and items as well as user-generated reviews and ratings, item descriptions, and item categories. Also, each user and each item have at least 5 associated reviews, which correspond to actual purchases.

To the aim of generating synthetic queries to be used with the Amazon Review 5-Core dataset as an appropriate benchmark for Product Search evaluation, previous research works [6, 106, 296, 7, 33, 291, 8, 107, 109] have followed the query extraction method proposed by Gysel et al. [109]. This approach is based on the assumption that users search for "a producer's name, a brand or a set of terms which describe the category of the product" [230] and it works as follows: for each category c generate a query q by 1) concatenating the terms in the category hierarchy of c, 2) removing the stop-words, and 3) removing duplicated words in reverse order (*e.g.*, *Camera & Photo* $\rightarrow$ *Digital Camera* would generate "*photo digital camera*"). Each item belonging to the category c is considered as relevant to the query q.

To comparatively evaluate the proposed approach with the considered baselines,

Table 3.1: Statistics of the benchmark datasets.

	Electronics	**Kindle Store**	**CDs & Vinyl**	**CP & A**
# users	192 403	68 233	75 258	27 879
# items	63 001	61 934	64 663	10 429
# reviews	1 689 188	982 618	1 097 591	194 439
# queries	989	4 603	694	165

which are detailed in the next section, we relied on the datasets proposed and shared[2] by Ai et al. [6]. These datasets contain data from four subsets of the Amazon Review 5-Core dataset — *Electronics*, *Kindle Store*, *CDs & Vinyl*, and *Cell Phones & Accessories* — along with queries generated following Gysel et al. [109]. The datasets proposed by Ai et al. [6] are suitable for studying personalization in Product Search as the authors have built user-query pairs by linking user-item pairs with item-related queries. In this setting, only the items purchased by a user and related to a query are considered as relevant for that user-query pair. Each dataset comes already partitioned into training and test sets. Partitioning was done so that every query and user-query-item triplet in the test set is new and unobserved. Reviews related to user-query-item triplets in the test sets were removed from the training sets. Table 3.1 reports some statistics about the datasets; the reader can refer to [6] for additional information. Note that in our work and all previous efforts, the four datasets were considered separately without sharing information, despite coming from the same e-commerce website.

3.3.2 Baselines

We have compared the proposed model with four different retrieval approaches. The first one is a traditional domain-agnostic retrieval model based on bag-of-words representations, BM25 [226]. The other three are recent Neural Network-based re-

[2] https://github.com/QingyaoAi/Amazon-Product-Search-Datasets

trieval models specifically designed for Product Search, namely Latent Semantic Entity [109], the Hierarchical Embedding Model [6] and the Dynamic Relation Embedding Model [8].

BM25 The first baseline is the classic probabilistic retrieval model BM25 [226] introduced in Section 2.3.2, which is also used as first-stage retriever by our proposed re-ranking approach. We consider BM25 as a baseline for the specific purpose of assessing if our re-ranking approach is able to enhance the retrieval effectiveness of the underlying retrieval model. BM25 scores are computed on product titles, descriptions, and reviews as a whole, by first removing stop-words and applying the Krovetz stemmer [139].

Latent Semantic Entity (LSE) LSE [109] is a Product Search model based on a Neural Network trained to project both items and queries in the same latent space where their semantic representations can be directly compared to evaluate the relevance of an item with respect to a given query. LSE relies on a tunable parameter to control the embedding dimension. We considered LSE as a baseline mainly to compare the effectiveness of lexical matching approaches (BM25) with respect to semantic matching ones in Product Search.

Hierarchical Embedding Model (HEM) HEM [6] is a personalized Product Search model based on Neural Networks. Through HEM, the users, items, and queries are projected in the same latent space where they can be directly compared. The distributed representations of users, items, and queries are learned in a generative fashion, by maximizing the likelihood of the observed user-query-item triplets. Similarly to LSE, HEM is based on a purely semantic retrieval model. HEM makes use of tunable hyper-parameters to control the personalization impact and the embeddings dimension.

Dynamic Relation Embedding Model (DREM) DREM [8] is a state-of-the-art Neural Network-based model that, similarly to HEM, maps users, items, and queries in the same latent space. Unlike HEM, which makes use only of reviews, DREM takes advantage of many additional side information, such as item brands and item categories. DREM shares the same query representation model of HEM, and therefore it can be considered a purely semantic retrieval model. DREM delivers the best retrieval performance among the considered baselines at the state-of-the-art in Product Search. DREM relies on two tunable hyper-parameters to control the personalization impact and the embeddings dimension.

3.3.3 Model Training and Hyper-Parameters Tuning

To the aim of defining the user and item representations described in Section 3, both Node2Vec and PV-DBOW were trained using the hyper-parameters configurations proposed by their respective authors. Lemmatization and stop-words removal were performed on reviews before feeding them to PV-DBOW. The weights w_k in Eq. 3.5 and the root smoothing parameter for item popularity were tuned using the hyper-parameter optimization Python package Optuna [9]. The search space for each parameter ranges from 0.01 to 1.0 — note that the n-root operation used for item popularity smoothing has been implemented as power-of-n. These values were sampled from a discrete uniform distribution, with a discretization step equal to 0.01. This means that our search space was composed of 10^{10} possible combinations. The number of trials was limited to 10. The decay constant λ in Eq. 3.3 was set to 0.1 for all datasets.

3.3.4 Evaluation Metrics

To evaluate the proposed approach and compare it with the baselines, we employed three different evaluation metrics: 1) mean average precision (MAP), 2) mean reciprocal rank (MRR) and normalized discounted cumulative gain (NDCG). MAP is the arithmetic mean of the average precision values — the mean of the precision scores after each relevant item is retrieved. MRR of a query results list is computed as the inverse of the rank of the first relevant item. MRR gives information about the expected number of results a user needs to view before finding one she/he is interested in. NDCG gives insights into how good a ranked list is compared to the optimal one. MAP and MRR were computed on the top 100 items retrieved by each model, whereas NDCG was computed on the top 10: these cutoffs have been chosen accordingly to previous works [6, 8] for evaluation consistency. Statistical significance testing was conducted with the Fisher's randomization test [244] with $p \leq 0.01$.

3.4 Results and Discussion

In this section, we present the results obtained on different Product Search benchmark datasets by our model as well as by the considered baselines. First, we discuss the retrieval performance of all the considered models and analyze the results in detail. Then, we conduct an ablation study of the side information employed by our re-ranking model and discuss the effect of personalization over the underlying retrieval function, BM25.

3.4.1 Retrieval Performance

Table 3.2 shows the effectiveness of our model and those of the baselines on the datasets *Electronics*, *Kindle Store*, *CDs & Vinyl* and *Cell Phones & Accessories*. Our model's results refer to the application of the proposed re-ranking approach to the

Table 3.2: Effectiveness of our model and those of the baselines on four benchmark datasets. Best values are highlighted in boldface. $*$, $\star$, †, ‡ denote significant differences w.r.t. BM25, LSE, HEM and DREM respectively, in Fisher's randomization test with $p \leq 0.01$.

Model	Electronics			Kindle Store		
	MAP@100	**MRR@100**	**NDCG@10**	**MAP@100**	**MRR@100**	**NDCG@10**
BM25	0.320*	0.321*	0.368*†	0.015*	0.017*	0.017*
LSE	0.233	0.234	0.239	0.006	0.007	0.007
HEM	0.308*	0.309*	0.329*	0.029**	0.035**	0.033**
DREM	0.366**†	0.367**†	0.408**†	**0.057****†	**0.067****†	**0.067****†
Our	**0.405****†‡	**0.406****†‡	**0.451****†‡	0.046**†	0.054**†	0.055**†
Model	**CDs & Vinyl**			**Cell Phones & Accessories**		
	MAP@100	**MRR@100**	**NDCG@10**	**MAP@100**	**MRR@100**	**NDCG@10**
BM25	0.027*	0.031*	0.032*	0.205*†	0.205*†	0.212*†
LSE	0.018	0.022	0.020	0.098	0.098	0.084
HEM	0.034**	0.040**	0.040**	0.124	0.124	0.153*
DREM	0.074**†	0.084**†	0.086**†	0.249**†	0.249**†	0.282**†
Our	**0.077****†‡	**0.088****†‡	**0.092****†‡	**0.294****†	**0.294****†	**0.306****†

top-1000 results produced by BM25 in response to the user queries. HEM's and DREM's results refer to the best hyper-parameter configuration found by their respective authors on the test sets of the same benchmark datasets, while LSE's results refer to the best hyper-parameter configuration found by Ai at al. [6] for these same datasets. Note that the evaluation datasets used here as well as their training/test splits are the same of the previous works and so are the performance scores of previous models. BM25's hyper-parameters were tuned using Optuna [9], with a number of trials limited to 10.

BM25 performed consistently better than LSE, achieving a NDCG increase ranging from 54% on *Electronics* to 143% on *Kindle Store*. The employed lexical retrieval model outperforms LSE, although the latter was specifically designed for Product Search. We claim that the main reason for this is related to the fact that vocabulary mismatch — the main issue that drove the design of LSE — is not so severe in Product Search. Moreover, we think lexical matching is fundamental when searching for

products as real-world objects usually have a well-defined related lexicon. Users and sellers usually know this lexicon, and they use it to describe their needs and the characteristics of the products they sell, respectively. Finally, as already pointed out by Guo et al. [104] in an ad-hoc retrieval scenario, "relevance matching requires proper handling of the exact matching signal" and, therefore, supporting vocabulary mismatch through semantic search while decreasing lexicon-based matching capabilities is not ideal.

Surprisingly, BM25 also outperformed HEM on two datasets out of four, achieving a NDCG increase of 12% on *Electronics* and 39% on *Cell Phones & Accessories*, respectively. We suspect HEM failed to deliver better performance than BM25 on two of the considered datasets, despite its personalization mechanism, due to its semantic-based retrieval function, similarly to LSE.

Our proposed model consistently improved BM25 performances by a considerable margin across all the benchmark datasets, demonstrating the effectiveness of the proposed re-ranking approach as well as of the employed user and item representations. Our approach also outperformed all the considered baselines across all datasets with the sole exception of DREM on the *Kindle Store* dataset. Our approach achieved a NDCG increase over LSE ranging from 89% on *Electronics* to 686% on *Kindle Store* and a NDCG increase over HEM ranging from 37% on *Electronics* to 130% on *CDs & Vinyl*. This again demonstrates the superiority of BM25 as a ranking function over the ranking function learned by LSE and HEM as well as the better performance of the proposed personalization approach over that of HEM.

Regarding the comparison with the state-of-the-art model DREM, our approach achieved strong improvements on the *Electronics* and the *CDs & Vinyl* datasets, $+11\%$ and $+7\%$ on the NDCG score, respectively, while performing similarly on the *Cell Phones & Accessories*, as no statistically significant difference was detected by the Fisher's randomization test [244]. DREM achieves better results then our model on

the *Kindle Store* dataset. By further analyzing the results, it is easy to spot that DREM outperforms our model on the dataset where BM25 achieved the worst performance. As a consequence of being a re-ranking approach, our model struggles when the recall of the underlying retrieval function is poor, despite being able to consistently improve the ranking of the initially retrieved items. This is because, by design, re-ranking approaches do not have full access to the product catalog but only to the initial result list. Therefore, if the underlying retrieval function fails to retrieve the relevant items, a re-ranking model cannot improve the result list. However, we suppose the poor performances of BM25 on *Kindle Store* and *CDs & Vinyl* are due to the synthetic queries contained in the benchmark datasets [6] used for conducting the comparative evaluation and the available information about the products in those datasets and not to the model itself. Following Gysel et al. [109], queries were generated from categorical information related to the items. There is no query related to authors or book titles for *Kindle Store* nor artist names or album titles for *CDs & Vinyl*. This seems to be largely unlikely as music can be listened to before buying on many web platforms and more. We guess that users often issue queries containing the title — or part of it — of a book or its author's name because they already know what they are looking for. Often people buy books because of word-of-mouth or because they watched their authors interviewed in television programs. A similar discussion can be done for *Electronics* and *Cell Phones & Accessories* where, for example, brands have a high impact on sales. However, the categorical information used to generate the queries for *Electronics* and *Cell Phones & Accessories* always contains item types, such as "screen protector" or "bluetooth speaker", that are highly discriminative and resemble real user queries, despite the lack of more specific information, such as the storage capacity for an external Hard Drive. On the other hand, *Kindle Store* and *CDs & Vinyl* contains very few item types and the categorical information is mostly based on literature and music genres, respectively. In addition, *Kindle Store* and *CDs*

& Vinyl have a very high number of missing item titles (100% and 23%, respectively) and descriptions (75% and 27%, respectively), probably due to how the item-related data were obtained by McAuley et al. [186]. Finally, it is worth mentioning that DREM makes use of item categories as a source of item-related information when computing their latent representations. Despite the author of DREM *anonymized* item categories when computing item representations, their model can still learn to draw strong relationships between the information about item categories and the terms of the queries, that following [109] were constructed from the item categories, as extensively described in Section 3.3.1. BM25 would suffer from a very similar problem if we index the item categories as text along with the other item-related information (titles, descriptions, and reviews). Therefore, to not trivialize the retrieval task, we did not index the item categories. However, it would probably be a good practice in a real-world scenario as users often search for a product using "terms that describe the category of the product" [230]. As our re-ranking approach does not have direct access to the query terms and a fixed formula is used to compute user and item category-based representations, as discussed in Section 3.2, it is not affected by the aforementioned issue.

3.4.2 Ablation Study

In this section we analyze the contribution of each of the employed compatibility scores and item popularity on our proposed personalized re-ranking process, by leveraging them in isolation. In Table 3.3 the results of the ablation study are reported. *Review*, *Interaction*, *Category* and *Popularity* refer to our personalized re-ranking approach with the sole use of *Review-based representations*, *Interaction-based representations*, *Category-based representations* and *Item popularity*, respectively.

As shown in the table, the chosen side information used for re-ranking purpose always increases BM25's effectiveness also when used in isolation. There is no

Table 3.3: Effectiveness of BM25 and our approach with the use of each kind of side information in isolation.

Model	Electronics			Kindle Store		
	MAP@100	**MRR@100**	**NDCG@10**	**MAP@100**	**MRR@100**	**NDCG@10**
BM25	0.320	0.321	0.368	0.015	0.017	0.017
Review	0.320	0.321	0.372	0.023	0.028	0.027
Interaction	0.340	0.340	0.387	**0.057**	**0.068**	**0.068**
Category	0.371	0.371	0.422	0.016	0.018	0.018
Popularity	0.346	0.347	0.396	0.016	0.018	0.018
Model	**CDs & Vinyl**			**Cell Phones & Accessories**		
	MAP@100	**MRR@100**	**NDCG@10**	**MAP@100**	**MRR@100**	**NDCG@10**
BM25	0.027	0.031	0.032	0.205	0.205	0.212
Review	0.047	0.054	0.056	0.230	0.230	0.231
Interaction	0.066	0.077	0.079	0.200	0.200	0.216
Category	0.034	0.039	0.040	0.240	0.240	0.264
Popularity	0.029	0.034	0.035	0.214	0.214	0.218

clear insight on which of those information is the strongest/weakest for re-ranking products, as their effect vary on a dataset bases. Interestingly, the sole use of the *Interaction-based representations* in the *Kindle Store* dataset allows our approach to achieve state-of-the-art performance even on the toughest of the benchmark dataset. This mainly indicates the necessity of a smarter score fusion approach, able to dynamically select and weight the contribution of the side information when computing the re-ranking scores. We leave this for future studies.

We also performed an ablation study of the procedure we employed to build the category-based representations described in Section 3.2.3. In particular, we changed the category weighing scheme, trying two different alternatives. Firstly, we reversed the category importance, giving more weight to more specific categories instead of the broader ones. Secondly, we applied a flat weighing scheme so that each category has the same importance. Table 3.4 shows the results of this analysis. *Reversed* refers to using the compatibility scores obtained by reversing the weighing scheme, while *Flat* refers to using a flat weighing scheme instead. The results of this analysis confirm that the formulation we proposed is more effective.

Table 3.4: Effectiveness of our model and those of its variants using the alternative categories weighing schemes. Best values are highlighted in boldface.

Model	Electronics			Kindle Store		
	MAP@100	MRR@100	NDCG@10	MAP@100	MRR@100	NDCG@10
Our	**0.405**	**0.406**	0.451	**0.046**	**0.054**	**0.055**
Reversed	0.401	0.402	**0.455**	0.042	0.050	0.051
Flat	0.402	0.403	0.446	0.045	0.053	0.054
Model	**CDs & Vinyl**			**Cell Phones & Accessories**		
	MAP@100	MRR@100	NDCG@10	MAP@100	MRR@100	NDCG@10
Our	**0.077**	**0.088**	**0.092**	**0.294**	**0.294**	**0.306**
Reversed	0.071	0.082	0.085	0.286	0.286	0.295
Flat	0.073	0.084	0.087	0.282	0.282	0.291

3.4.3 Analysis of the Efficiency of the Proposed Approach

In this section, we analyze the overhead deriving from the employed re-ranking model on top of BM25 and the time needed to train the review-based and the interaction-based representation models. To conduct this evaluation, we selected 1000 user-query pairs from the *Electronics* dataset. We assumed the various user and item representations to be previously computed and stored in the system's RAM. Note that the size of the representations of all the *Electronics*' users (192k) and items (63k) is less than 40MB in total. The average time required by BM25 to retrieve a set of documents from the *Electronics* dataset in response to a user query amount to 152ms in our test system. Preparing all the compatibility scores required by our re-ranking function, computing the new scores, and re-order the retrieved list of documents only takes 17ms. Therefore, the overhead deriving from our personalization approach at run-time is negligible and should not negatively impact the user experience while increasing the overall retrieval performances. Regarding the training time of the neural models employed by the proposed approach (PV-DBOW and Node2Vec), they are very computationally efficient, and only one hour is required to train each model on a single CPU core (Intel® Core i7-4790k) on the largest of the evaluation datasets (*Electronics*). For comparison, training HEM [6], the smaller model among the per-

sonalized baselines, on an NVidia® Titan X GPU usually requires 7-8 hours on the same dataset.

3.5 Summary

In this chapter, we addressed the problem of Personalized Results Re-Ranking in the context of Product Search. In particular, we investigated the use of four different user/item representations to enhance BM25 performances on the top 1000 results. We employed representations derived from user-generated content, user purchasing behavior, categorical information, and item popularity. Our empirical evaluations show that the proposed approach consistently enhances BM25 and outperforms recently proposed Neural Network-based models specifically designed for Product Search on multiple benchmark datasets. Finally, our proposed Personalized Results Re-Ranking approach is fast and scalable and it is easily extendable to accommodate additional relevance/compatibility scores.

CHAPTER 4

DENOISING ATTENTION FOR QUERY-AWARE USER MODELING IN PERSONALIZED SEARCH

The past few years have witnessed an increasing interest in the application of Deep Learning techniques for tackling various tasks of Information Retrieval [105], such as Personalized Search.

Two of the main challenges of Personalized Search are *how* and *when* personalization should take place. First, not all the data gathered to represent specific users' preferences in their user models are equally related to each of their searches, as users usually have multiple and diverse interests. Second, personalization is not always beneficial to the retrieval process [261] as it could cause the information need expressed by the user to be misinterpreted by the system. For example, this might happen when the user's interests in a specific domain are *unknown*, but the system still applies personalization, which, in this case, leverages user-related information *potentially unrelated* to the user's information need; this could ultimately decrease the system's effectiveness.

A recent trend in Personalized Search [95, 166, 302, 7, 298, 288, 126, 301, 34, 36, 165] is *query-aware user modeling*, which consists in building a representation of the user preferences, *i.e.*, the user model, at query time, based on various sources of user interest and by giving more importance to those related to the current search performed by the user. Since a user is typically interested in different and even *unrelated* topics, a desirable property for defining reliable personalization models

is the ability to discern between beneficial and noisy user-related information on a query basis. Previous works in this context that make use of neural models rely on the *Attention* mechanism [19], to differently weigh the contribution of distinct sources of user-related information in building the user representation at query time. Despite the increasing use of the *Attention* mechanism in user modeling, there is still a lack of an in-depth analysis of its behavior and effects on personalization, as well as a systematic comparison with simpler operators in this context.

In this chapter, we first describe and analyze the *Attention* mechanism when used for query-aware user modeling, highlighting some shortcomings of the standard *Attention* formulation related to its use of the *Softmax* function (Section 4.2). Specifically, the *exponential* function employed by the *Softmax* can cause the user model to be excessively noisy or skewed towards a single piece of user information. Moreover, as it will be extensively discussed in Section 4.2, due to the fact that the standard *Attention* mechanism uses the *Softmax*'s outputs to weigh the contribution of the sources of user information to build the user model at query time, personalization is performed even when those sources are not related to the current search conducted by the user. To overcome these weaknesses, in Section 4.3, we propose the *Denoising Attention* mechanism, an *Attention* variant specifically designed to finely filter out noisy user-related information and produce a balanced representation of the user interests w.r.t. the current search. Firstly, we introduce a novel filtering mechanism based on the *Rectifier Linear Unit* [199] and a threshold value. Secondly, we depart from the *Softmax* function and opt for a more straightforward and robust weighting scheme. To evaluate our proposal, we tackle the task of *Personalized Results Re-Ranking*; to make a comparative evaluation of the proposed user model with alternative user models at the state-of-the-art, we rely on a framework that allows us to switch the user representation model with ease. We introduce the considered task and the related framework in Section 4.4. Then, we present the research questions we

addressed and describe the experimental setup of our comparative evaluation (Section 4.5). Finally, in Section 4.6, we present the comparison between the *Denoising Attention*, the standard *Attention*, the *Zero Attention strategy* [7], an *Attention* variant previously proposed for user modeling, and the *Multi-Head Attention* [272], and we also ablate our proposed *Attention* variant. The results of our evaluation clearly show the advantages of *Denoising Attention* and the importance of the filtering mechanism it implements. We will share all the code to reproduce the experimental evaluation, and make available the implementation of *Denoising Attention* for future works once the related article will be published.

4.1 Related Work

Personalization of search results has received considerable attention from both academia [274, 95, 166, 288, 301, 34, 36, 165, 53, 242, 258, 116, 219, 302, 134] and industry [101, 102, 247, 295, 153, 31, 78, 248, 185, 284, 54, 261, 7, 298, 126, 220, 260]. Over time, many different approaches have been proposed to tackle this task. Early personalization models relied on click-based features [31, 78, 261, 263], language models [258, 248], topic modeling [116, 53, 284], ontologies [242, 219], content-based features [185, 261], social network analysis [54], and other sources of user-related information as well as other formal means to build user representations. Recently, researchers have focused on the application of Deep Learning and Word Embeddings for the personalization of search results [153, 247, 295, 274]. These works take advantage of the opportunity given by Representation Learning [30] to build latent semantic vector representations of queries, documents, and the gathered user-related information. Among those works, a new trend in modeling user interests has recently emerged. In particular, several works [95, 166, 302, 7, 298, 288, 301, 126, 34, 36] rely on the *Attention* mechanism to weigh and aggregate the available user-related information on a query

basis, thus building a representation of the user preferences, *i.e.*, the user model, at query time. With the aim of applying query-aware personalization, these models try to take advantage of the diverse interests that a user may have. Many previous works [95, 166, 288] rely on the *Attention* mechanism to weigh, w.r.t. the current query, the contribution of a user's prior search sessions (represented as a combination of the representations of the previous queries and those of the documents accessed by the user after issuing them) for composing the user model employed for conducting session-based personalization. Zhou et al. [302] tackle the problem of user re-finding behavior and rely on *Attention* to weigh, w.r.t. a user's current search, the previous queries she issued, and the documents she accessed in the past, which are then used as sources of the user interests for personalizing the current search results. By leveraging the *Attention* mechanism, Zhong et al. [298] weigh user-related terms w.r.t. the query the user is typing, and use a weighted combination of those terms and the original query terms to generate personalized query suggestions. Jiang et al. [126] propose an *attentive* Personalized Item Retrieval model leveraging the *Attention* mechanism to estimate the importance of each item in the user history while conducting personalization. Despite the increasing application of the *Attention* mechanism for user modeling, the vast majority of the previous works did not conduct an in-depth analysis of its behavior and effects on personalization. The sole exception is represented by the *Zero Attention Model* proposed by Ai et al. [7]. The authors propose an *Attention* variant defined to allow the retrieval model to avoid personalization when no source of user information is related to her current search, which is not possible using the standard *Attention* formulation, as we will discuss in Section 4.2.2. Although the promising results obtained by the authors, successive works [34, 35, 126] have shown that the *Zero Attention Model* may perform inconsistently, and often it exposes equal or lower effectiveness than the standard *Attention* formulation.

In this chapter, we first discuss some shortcomings of the standard *Attention*

formulation that prevent it from being an optimal solution when it comes to personalization. Then, we propose a novel *Attention* variant, called *Denoising Attention*, designed to solve these issues and show the benefits of our approach over other user modeling mechanisms.

4.2 Preliminaries on Query-aware User Modeling

Users usually have diverse interests in multiple domains. Although the representation of multifaceted user preferences is a powerful resource for personalization, not all those preferences are equally relevant to a specific user's information need. For example, if a user is looking for a new book to read, her apparel preferences probably do not matter for personalizing the results of her current query. *Query-aware User Modeling* consists in building a user model at query time, based on previously gathered sources of user interest, by giving more importance to those related to the current search performed by the user. In the literature, the definition of a user model with the previous characteristics has been provided by relying on the *Attention* mechanism [19], which allows weighing the contribution of the user-related data w.r.t. the current search query.

In the following sections, we first describe the *Attention* mechanism as it is usually employed in the context of Personalized Search. Then, we discuss some shortcomings of its standard formulation when used for personalization.

4.2.1 Attention Mechanism

The *Attention* mechanism, introduced by Bahdanau et al. [19] in Neural Machine Translation, aims at computing a context vector by weighing the available contextual information w.r.t. a given input. A context vector can enrich the information carried by the input, helping, *for example*, to disambiguate its meaning.

In Personalized Search, the context vector is interpreted as the *user's context vector*, *i.e.*, the *user model*; the contextual information is intended as the *user's contextual information*, *i.e.* the available user-related data, and the input is the *search query*. In the following, we assume that the user-related information sources and data are documents in the form of textual documents written by the users, previously accessed documents (*e.g.*, web pages), user-generated content [140] (*e.g.*, product reviews or tweets), previously issued queries, or other content related to the users, their preferences, and behavior. At query time, the *Attention* mechanism weighs the vector representations of these documents w.r.t. the query vector and aggregates them to produce the user model employed in the personalization process. The *Attention* mechanism comprises three steps aiming to build the context vector: *scoring*, *normalization*, and *aggregation*. The three steps are presented here below.

Scoring First of all, an *alignment model* [19] (or *scoring function*) a is used to score how well the representations of the user-related documents match with the input query:

$$e_{\boldsymbol{q},\boldsymbol{d}} = a(\boldsymbol{q}, \boldsymbol{d}) \tag{4.1}$$

where, $\boldsymbol{d} \in \mathbb{R}^m$ and $\boldsymbol{q} \in \mathbb{R}^n$ are the vector representations of a user document and a given query, respectively, and $e_{\boldsymbol{q},\boldsymbol{d}} \in \mathbb{R}$ is the matching score computed by the *alignment model* $a : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$ for the vectors $\boldsymbol{d}$ and $\boldsymbol{q}$. It is important to outline that, usually, the dimension of the representations of the user document $\boldsymbol{d}$ and the query $\boldsymbol{q}$ are the same as they are projected in the same latent space.

The *alignment model* can be as simple as the dot-product, in which case we talk about *Dot-Product Attention* [172] and *Scaled Dot-Product Attention* [272], or the cosine similarity, usually called *Content-based Attention* [100]. Alternatively, the *alignment model* can be a parameterized function, such as a Neural Network [19].

Normalization The second step of the *Attention* mechanism consists in the *normalization* of the matching scores computed by the *alignment model* to generate a probability distribution of the contextual information. The normalized matching scores are commonly called *attention weights*. This step is usually accomplished through the use of the *Softmax* function [19, 172, 272, 100]:

$$\alpha(\boldsymbol{q}, \boldsymbol{d}) = \text{Softmax}(e_{\boldsymbol{q},\boldsymbol{d}}) = \frac{\exp(e_{\boldsymbol{q},\boldsymbol{d}})}{\sum_{\boldsymbol{d}' \in \boldsymbol{D_u}} \exp(e_{\boldsymbol{q},\boldsymbol{d}'})} \tag{4.2}$$

where, exp is the *exponential* function, $\boldsymbol{D_u}$ is the set of all the documents related to the user u, and $\alpha(\boldsymbol{q}, \boldsymbol{d}) \in \mathbb{R}$ is the *attention weight* of $\boldsymbol{d}$ w.r.t. $\boldsymbol{q}$.

Aggregation Finally, the third step consists in the *aggregation* of the contextual information to produce the context vector $\boldsymbol{u}$, which, in our case, represents the *user model*. This process is carried out by summing the user document vector representations weighed by their corresponding *attention weights*:

$$\boldsymbol{u} = \sum_{\boldsymbol{d} \in \boldsymbol{D_u}} \alpha(\boldsymbol{q}, \boldsymbol{d}) \cdot \boldsymbol{d} \tag{4.3}$$

4.2.2 Attention-based User Modeling Shortcomings

Although the *Attention* mechanism allows building user models at query time, some shortcomings prevent it from being an optimal solution for personalization. These issues are related to the *Normalization* step and specifically to the use of the *Softmax* and the *exponential* function. The *Softmax* function, which is based on the *Luce's choice axiom* [170], was proposed by Bridle [46] as a *softened* (*continuous* and *differentiable*) generalization of the *Arg max* function. *Arg max* is an operation that aims to find the point of the domain of a function in which it assumes maximum value. *Arg max* is neither continuous nor differentiable and, therefore, it does not allow for gradient-

based optimizations [97] (*i.e.*, it cannot be used in Neural Networks, as it would break back-propagation). *Softmax* is primarily used by classifiers for computing a probability distribution over n output classes. During training, the model *should* learn to maximize the output of the correct class.

Because of the nature of the *Softmax* function, whose goal is to select one among n options, and its use of the *exponential*, a *Softmax*-based user modeling approach naturally tends to skew the user representation towards a single user document, the one that best *aligns* with the query. Such drawback is usually not ideal for personalization as the other user documents could concur to a more informed and balanced representation of the user interests and preferences. *For example*, given the following vector of *alignment scores* $[7.0, 3.0, 1.0, -2.0]$, by applying Eq. 4.2 for normalization, we obtain the following *attention weights* $[0.9796, 0.0179, 0.0024, 0.0001]$. These weights cause the user model to be strongly biased towards the contextual information contained in a single user document.

A possible solution could be to constrain the *alignment function*'s output so that the *Normalization* step cannot produce an *overly narrow* probability distribution of the contextual information. However, if, *for example*, we constrain the *alignment scores* near zero by using the cosine similarity as the *alignment model*, the *Softmax* normalization will *overly smooth* the scores, thus causing that noisy information will be injected into the user model and have a strong influence on the current search. *For example*, given the following vector of *alignment scores* $[0.7, 0.3, 0.1, -0.2]$, by applying Eq. 4.2 for normalization we obtain the following *attention weights* $[0.3809, 0.2553, 0.2090, 0.1548]$. These weights highly reduce the initial *alignment scores*' diversity, giving considerable importance to the less relevant user-related information and penalizing the most relevant one. Moreover, the user information source whose *alignment score* with the query is negative, indicating very low relatedness, gets a positive *attention weight*. In this case, we can say that *Softmax* promotes the presence of *potentially* noisy in-

formation in the user model instead of penalizing or *filtering* it out. Note that, to properly work with Softmax, which produces higher values for the higher alignment scores among its input, an alignment model must assign high alignment scores to the user documents appropriate for personalization and low alignment scores to those potentially harmful.

Lastly, as the *Softmax* normalizes its input into a probability distribution, it follows that the *attention weights* from Eq. 4.2 are all positives and sum to 1 [63]. Even if all the alignment scores were zero or negative, the *attention weights* would *all* be positive and sum to 1. *For example*, given the following vector of *alignment scores* $[0.0, 0.0, 0.0, 0.0]$, by applying Eq. 4.2 for normalization we obtain the following *attention weights* $[0.25, 0.25, 0.25, 0.25]$. The same happens when all the *alignment scores* are negative: $[-7.0, -3.0, -1.0, -2.0] \rightarrow [0.0016, 0.0899, 0.6641, 0.2443]$. As there will always be at least one positive *attention weight* and the sum of the *attention weights* will always be 1, the context vector cannot be filtered out. In the context of personalization, this means that the user's context vector will never be zero, causing the personalization of search results to be performed even when no source of user-related information is in line with her current search. In such cases, personalization could hurt the effectiveness of the search engine instead of improving it.

4.3 Denoising Attention Mechanism

In this section, we present our proposal to address the shortcomings of the standard *Attention* when used for personalizing search results. As extensively discussed in Section 4.2.2, the principal issues of the *Attention* are related to its *normalization* step, described in Section 4.2.1, and specifically to the use of the *Softmax* function to produce the *attention weights*. To counteract these issues, we need a mechanism able to avoid *overly narrowing* or *overly smoothing* the *attention weights*, which can cause the

model either to focus only on a single source of user-related information or to reduce the diversity of their estimated importance. Moreover, this mechanism should finely filter out noisy contextual information, thus preventing it from flowing into the user model. Finally, it should zero out the user's context vector when personalization could harm the retrieval process, *i.e.*, when *all* the user-related information is noisy or irrelevant with respect to the current search. In this regard, we propose the *Denoising Attention* mechanism. The *Denoising Attention* mechanism departs from the *Softmax* function by adopting a more straightforward and robust normalization scheme, and it introduces a filtering mechanism based on the *Rectifier Linear Unit* [199] and a threshold value. To complement those changes, we rely on a cosine similarity-based alignment model to evaluate the relatedness of the sources of user-related information w.r.t. the current search.

Scoring For an *alignment model* to act in a complementary way with the changes introduced in the next paragraph, we need a function $a(\boldsymbol{q}, \boldsymbol{d})$ bounded between 0 and 1, as an unbounded function would make it difficult to control which information flows into the user model. To compute the *alignment scores* $e_{\boldsymbol{q},\boldsymbol{d}}$, we then rely on the following cosine similarity-based function:

$$a(\boldsymbol{q}, \boldsymbol{d}) = \frac{\cos(\boldsymbol{q}, \boldsymbol{d}) + 1}{2} \tag{4.4}$$

This function shifts the cosine similarity codomain to $[0, 1]$.

Filtering The first change we propose to the standard *Attention* mechanism is the explicit addition of a *filtering* step. First, we introduce a *threshold* parameter t that we use to *negativize* the alignment scores of the user data loosely related to the input

query. We call this operation *alignment scores shifting* and we define it as follows:

$$shifted\_e_{\boldsymbol{q},\boldsymbol{d}} = e_{\boldsymbol{q},\boldsymbol{d}} - \sigma(t) \tag{4.5}$$

where σ is a *squashing function* [120], which allows us to constrain t in $[0, 1]$ during training. We relied on the *Sigmoid* function in our final implementation. Secondly, we apply the *Rectifier Linear Unit* (*ReLU*) [199] to the *shifted alignment scores*. *ReLU* is a very popular activation function used in Neural Networks, and it is formulated as follows:

$$\mathrm{ReLU}(x) = \max\{0, x\} \tag{4.6}$$

What makes *ReLU* convenient in the personalization context is its ability to zero out negative values, in our case, the *shifted alignment scores* of noisy user-related information, while leaving unaltered the others:

$$filtered\_e_{\boldsymbol{q},\boldsymbol{d}} = \mathrm{ReLU}(shifted\_e_{\boldsymbol{q},\boldsymbol{d}}) \tag{4.7}$$

By combining these two operations, we can both control the information flow from the user data to the user model in diverse search scenarios and filter out the noisy user-related information that could harm the retrieval process. To avoid the well-known dying ReLU problem [164, 2] and let the model learn to zero out the user model correctly, we sum the user model to the query representation during training.

Normalization The second major change we propose to the standard *Attention* mechanism is the use of the *plain* normalization operation in place of the *Softmax* for the computation of the *attention weights*, which is defined as follows:

$$\alpha(\boldsymbol{q}, \boldsymbol{d}) = \frac{filtered\_e_{\boldsymbol{q},\boldsymbol{d}}}{\sum_{\boldsymbol{d'} \in \boldsymbol{D_u}} filtered\_e_{\boldsymbol{q},\boldsymbol{d'}}} \tag{4.8}$$

As the *plain* normalization can cause numerical instability when all the $\mathit{filtered\_e}_{\boldsymbol{q},\boldsymbol{d}}$ are zero, we slightly change Eq. 4.8 into:

$$\alpha(\boldsymbol{q},\boldsymbol{d}) = \frac{\mathit{filtered\_e}_{\boldsymbol{q},\boldsymbol{d}}}{\max\left\{\sum_{\boldsymbol{d'}\in\boldsymbol{D_u}} \mathit{filtered\_e}_{\boldsymbol{q},\boldsymbol{d'}}, \varepsilon\right\}} \tag{4.9}$$

where ε is a very low positive value.

Note that the proposed filtering mechanism cannot work correctly with *Softmax*. First of all, the *Softmax* function is *translationally invariant*, which means that adding or removing the same value to all the components of its input does not change its output. For example, $\mathrm{Softmax}([0.7, 0.3, 0.1, -0.2]) = \mathrm{Softmax}([0.7, 0.3, 0.1, -0.2] - 0.3) = [0.3809, 0.2553, 0.2090, 0.1548]$. Secondly, zeroing out negative values through *ReLU* does not prevent the *Softmax* from producing positive attention weights for those values, as already shown in Section 4.2.2. On the contrary, Eq. 4.9 does not suffer from those issues and can produce zero *attention weights*. By avoiding the use of the *Softmax* and its *exponential* function, the *Denoising Attention normalization* step does not suffer from many of the *Attention* shortcomings discussed in Section 4.2.2.

Aggregation The aggregation of the user-related information is performed as a linear combination, following the standard *Attention* formulation (Section 4.2.1). However, as normalizing with Eq. 4.9 allows to produce zero *attention weights*, the *aggregation* step can yield a *zero context vector*, which ultimately allows avoiding personalization when no source of user information is related to her current search.

Denoising Attention Weights To sum up, we propose to compute the weights for the user-related information as follows:

$$\alpha(\boldsymbol{q},\boldsymbol{d}) = \frac{\mathrm{ReLU}\left(e_{\boldsymbol{q},\boldsymbol{d}} - \sigma(t)\right)}{\sum_{\boldsymbol{d'}\in\boldsymbol{D_u}} \mathrm{ReLU}\left(e_{\boldsymbol{q},\boldsymbol{d'}} - \sigma(t)\right)} \tag{4.10}$$

In contrast with the standard *Attention* formulation, *Denoising Attention* is able to 1) selectively filter out the noisy contextual information from the user-related data before aggregating them in the context vector, and 2) zero out the context vector when all the sources of user-related information are unrelated to her current search. Moreover, the combined use of our *filtering mechanism* and *normalization* function makes our *Attention* variant prone to avoid *overly narrow* or *overly smooth* attention weights. This way, the model preserves the estimated importance of the user-related information sources and does not focus only on one of them, thus composing a balanced representation of the user preferences related to the current query while filtering those unrelated. For a sake of comparison, the *alignment scores* $[0.7, 0.3, 0.1, -0.2]$ produce the *attention weights* $[0.3809, 0.2553, 0.2090, 0.1548]$ when fed to Eq. 4.2, whereas they produce the *attention weights* $[0.75, 0.25, 0.0, 0.0]$ when fed to Eq. 4.10 with $\sigma(t) = 0.1$.

4.4 Evaluation Task and Framework

To evaluate the proposed user modeling approach we address the task of *Personalized Results Re-Ranking*, as described in Section 2.3.2. To conduct the comparative evaluation reported in the following sections, we employed a *Personalized Results Re-Ranking Framework* that allowed us to test different user modeling techniques with ease.

Fig. 4.1 depicts the *Personalized Results Re-Ranking Framework* we relied on for comparing various user modeling techniques for Personalized Search (Section 4.5 and 4.6). The framework comprises two modules that generate the vector representations of the top-k results retrieved by the first stage retriever and those of the user-related documents. Once computed the user-related document representations, the *user representation module* aggregates them into the user model. In the case of query-aware user modeling, as the query is involved in weighing the contribution of each user-related document, an additional module is employed to produce the query rep-

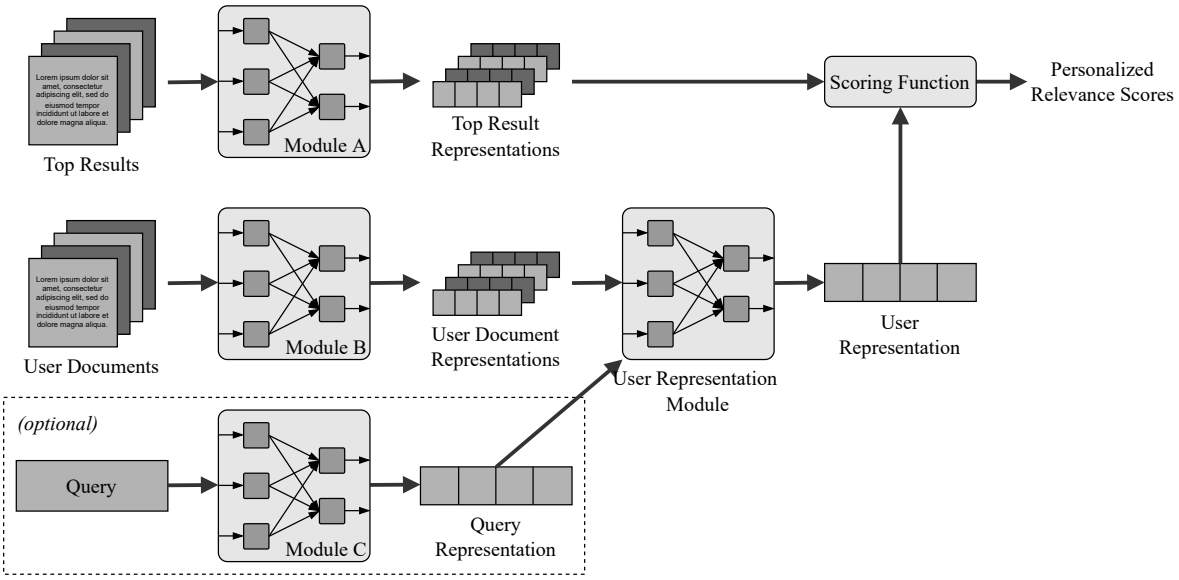


Figure 4.1: Personalized Results Re-Ranking Framework.

resentation used in that process. Finally, a *scoring function* computes a personalized relevance score for each initially retrieved result by comparing its representation with the user representation. Those scores are then combined with the scores computed by our first stage retriever BM25 (see Section 2.3.2) as in Eq. 2.1.

In the experiments presented in Section 4.6, we relied on TinyBERT [128] followed by a *mean pooling* operation to embed both the top retrieved documents, the user information, and the query (*if needed*), as they all are in text format in the dataset we employed for the evaluation. TinyBERT is a distilled [118, 99] version of the well-known Transformer-based [272] Neural Language Model BERT [76]. We chose TinyBERT due to its training and inference speed, lightweight GPU memory consumption, and, above all, due to our hardware limitations (for experimentation, we used an NVidia® RTX 2080 Ti GPU with 11 GBs of VRAM, which is not enough to fine-tune BERT in our specific setting). As with many recent ranking models [224, 93, 133, 149, 175] that rely on learned dense representations of documents [158], we employed the cosine similarity as our *scoring function*.

As the main purpose of the contribution we present in this chapter is to propose a

novel Attention variant for defining user models at query time, the simple technique we implement for re-ranking aims at making it possible to comparatively evaluate the effectiveness of the proposed model with alternatives at the state-of-the-art with ease, allowing us to switch the compared user models seamlessly. We also point out that the authors of previous contributions did not share their code or did not provide adequate instructions to reproduce the results they obtained or train/run their proposed models, posing several reproducibility issues.

4.5 Experimental Setup

The experiments reported in this section aim to answer the following research questions:

RQ1 Do the query-aware *Attention*-based user models increase the effectiveness of a personalized retrieval model w.r.t. simpler operations for aggregating user data?

RQ2 Does the *Denoising Attention*-based user model increase the effectiveness of a personalized retrieval model w.r.t. other *Attention*-based user models?

RQ3 Is the query-aware user representation produced by the *Denoising Attention* better-balanced w.r.t. the query-related user preferences than those of other *Attention* variants?

RQ4 Is the *Denoising Attention*-based personalization more *robust*, *i.e.*, less likely to decrease the system's effectiveness due to noisy user-related data, than the other considered approaches?

To answer the research questions **RQ1** and **RQ2**, we conducted a comparative evaluation of the retrieval effectiveness of the personalized re-ranking pipeline described

Table 4.1: Statistics of the employed datasets.

Web Search Dataset			
# documents	1 291 695	# users	30 166
# train queries	212 386	avg. query length	3.57 ± 1.51
# val queries	31 064	avg. relevants	1.15 ± 0.46
# test queries	36 052	avg. user docs	136.62 ± 134.17
Academic Search Dataset			
# documents	4 201 265	# users	63 738
# train queries	419 004	avg. query length	7.53 ± 2.64
# validation queries	4 241	avg. relevants	5.33 ± 5.11
# test queries	24 056	avg. user docs	53.59 ± 50.94

in Section 4.4 using several different user models. Then, we compared the retrieval effectiveness of the user models for the queries personalized by the *Denoising Attention*-based model to answer question **RQ3**. Finally, to answer the research question **RQ4**, we compared the number of times the considered user models decreased the retrieval effectiveness of our first-stage retriever, BM25.

In the following, we present the datasets we employed for conducting our evaluations (Section 4.5.1), we introduce the baselines we have selected (Section 4.5.2), and we outline the training setup and evaluation procedure (Section 4.5.3). We make all our code available for future works and reproducibility purposes[1].

4.5.1 Datasets

To conduct our experimental evaluations, we relied on two datasets that account for different search scenarios. First, we considered a Web Search dataset based on the AOL query log [214]. Then, we relied on a synthetic dataset we built following the procedure described by Tabrizi et al. [257] to simulate a domain-specific search scenario, in our case Academic Search. We describe both datasets in detail in the following sections.

[1]We will add a link to the repository upon acceptance

Web Search Dataset

Thanks to its potential, personalization in Web Search has been a hot topic for many years and has attracted the attention of several researchers both from academia ([111, 242, 203]) and private companies ([260, 261, 262]). Users search for a myriad of information on the Web, and discerning among the diverse - and often *unrelated* - interests a user can have could highly influence the personalization effectiveness in this scenario. This characteristic makes Web Search a perfect fit to evaluate the *Attention* variant we propose in this work.

The AOL query log [214] is one of the most known large-scale set of data for the evaluation of *session-based* personalization models ([4, 5, 287, 301, 166, 303, 288, 289, 75]). Although we are not focusing on *session-based* personalization, we can rely on this same query log to evaluate the effectiveness of our proposal. Unfortunately, the authors of the previous works that relied on the AOL query log did not release the instructions to re-build the datasets they employed in their evaluations. Therefore, we derive a novel Web Search dataset suited for our evaluation from it. We acknowledge the availability of other large-scale Web Search datasets, such as the Yandex[2] dataset, but, unfortunately, those datasets provide only anonymized texts for queries and documents, they are not publicly available, or they lack user unique identifiers.

We now review the procedure we followed to build our Web Search dataset from the AOL query log and make all the scripts to re-build such a dataset available for future research[3].

Retrieving documents A noticeable limitation of the AOL query log is that it does not provide the document contents but only the URL of clicked documents (*if any*). Because the logs date back to 2006, many of the clicked URLs are not available today, or the content of the documents they point to has changed since users accessed them.

[2] https://www.kaggle.com/competitions/yandex-personalized-web-search-challenge
[3] We will add a link to the repository upon acceptance

To retrieve document contents similar to those seen by the users when the logs were collected, we relied on the recently proposed *aolia-tools* [177], which leverage the Internet Archive's Wayback Machine service. We refer the reader to MacAvaney et al. [177] for an in-depth discussion on this topic. Once retrieved the document contents, we identified and removed non-English documents by analyzing them using Google's *Compact Language Detector* v3[4].

Query logs cleaning The AOL query log comprises queries issued by real users between March 1, 2006, and May 31, 2006. To derive a dataset suited for our evaluation, we operated a cleaning process aiming at obtaining an high quality query set while reducing noise that could interfere in our evaluation. First, we discarded all the queries with no related clicks from the query log and those pointing to non-English documents. Then, following MacAvaney et al. [177], which reported that several queries from the AOL query log have a navigational nature, we discarded those containing Internet domain references (*e.g.*, *.com*, *.org*, etc.) or website names as they do not need personalization and can be easily identified during pre-processing. For ethical reasons, we also discarded all the queries containing or pointing to adult or illegal contents. Following Sordoni et al. [249], we removed non-alphanumeric characters from the queries, applied a spelling corrector (*SymSpell*[5]) and lower-cased the queries. Then, we discarded all the queries shorter than three characters. To avoid introducing in the test set ⟨*query, user, document*⟩ triplets also present in the train set, we kept only the first appearance of such triplets by comparing their associated timestamps. Note that we are not interested in re-finding behavior in our work [267].

[4] https://github.com/google/cld3
[5] https://github.com/wolfgarbe/SymSpell

Training / Validation / Test Splits Following previous works [249, 5], we considered the queries formulated in the first five weeks as a background set. We discarded all the queries from users with less than 20 associated queries in this set to ensure having enough user-related data to conduct personalization, which, in this case, is based on previously accessed web pages. We then temporally split the remaining weeks' worth of queries. We used six weeks for *training* queries, one week for *validation* queries, and one week for *test* queries. Then, we fine-tuned the hyper-parameters of BM25 [226], which we use as our first stage retriever in the experimental evaluation, on thousands of non-test queries. Finally, we discarded the queries for which BM25 does not retrieve any relevant document in the top 1000 results. Likewise, for the remaining ones, we retain only the relevant documents present in the top 1000 results retrieved by BM25, as we are interested in results re-ranking.

Table 4.1 reports the statistics of the final dataset.

Academic Search Dataset

Alongside Web Search, Domain-specific Search is a popular research topic nowadays. Unlike Web Search, in domain-specific search scenarios, the user interests are more focused on particular topics, which could make finely discerning among the user-related data to pick those most promising for personalizing the current search conducted by the user more challenging.

Due to the lack of a publicly available Domain-specific Search dataset for studying personalization, researchers have recently tackled personalization in Product Search scenarios relying on synthetic datasets built upon product reviews from a popular e-commerce platform [6, 35, 34, 36, 25]. However, due to the number of different queries present in these datasets, a few hundred in most cases, and their low quality [25], we did not employ them in our comparative evaluation. Instead, we followed the procedure described by Tabrizi et al. [257] to build an Academic Search dataset

that allow us to test our *Attention* variant in a domain-specific search scenario. In particular, we relied on the *ArnetMiner*'s *Citation Network Dataset V12* [259], which makes available the metadata, such as *titles*, *abstracts*, list of *authors*, and list of *citations*, of $4\,894\,081$ academic papers (papers' full texts are not available). We now describe the process we followed to build our dataset.

Query Generation Firstly, we removed all non-English papers by leveraging Google's *Compact Language Detector* v3[6]. Then, following the approach described by Tabrizi et al. [257], we generated user-query-document triplets as follows: for each academic paper, we considered its title as a query, the list of its citations as the documents relevant to that query, and we assumed that the first author is the user submitting the query. Tabrizi et al. [257] proposed other methods to generate synthetic queries from research papers, but they only reported the evaluation of the one we employ here. As the titles of academic papers are written in well-formed natural language, we applied stop-word removal using the *NLTK*'s [38] stop-words list and a non-destructive stemmer, *i.e.*, the *Krovetz stemmer* [139], to obtain queries that resemble real-world ones. Finally, we discarded all the generated queries whose related users have less than 20 associated documents, *i.e.*, published papers.

Training / Validation / Test Splits We split the obtained dataset into *training* and *test* sets chronologically, *i.e.* by using the queries generated from papers published *after* 2018 as the *test set*. We then randomly split the *training set* to obtain a *training set* and a *validation set*, using a splitting ratio of $99 : 1$. We opted for a chronological *training* / *test* split instead of a random partitioning so that the dataset is closer to a real scenario, where all the searches in the *test set* happen after the searches in the *training set*.

[6] https://github.com/google/cld3

Dataset Refining As discussed by Tabrizi et al. [257], not all the references of a paper are necessarily relevant (from an Information Retrieval perspective) to the topic expressed by its title, which we use as a query. The authors, however, claim that since mistakenly considering some irrelevant documents as relevant will be the same for all the compared models, their presence does not violate the fairness of the comparisons if evaluation measures are averaged over many queries (*thousands*, in our case). To reduce the presence of spurious relevant documents and malformed queries, we applied simple heuristics, similarly to Tabrizi et al. [257]. As we are comparing different user modeling techniques in the context of Personalized Results Re-Ranking, we consider well-formed only the queries for which BM25 [226], which we use as a first-stage retriever, retrieves relevant documents in the top results. Although this is a strong assumption, the reader should consider two key factors. First, every re-ranking approach is inherently bounded by the recall of the first stage retriever. Therefore, as all the compared models re-rank the BM25's results, they all share the same limitations. Secondly, only an exiguous subset of a paper's references can be considered relevant w.r.t. its main topic. For example, if considering our paper, we would consider only the works concerning query-aware personalization, presented in Section 4.1, as truly relevant documents for a query built using our title. Although filtering the references by their positioning in the paper (the section in which they appear) *could be* a better solution in this case, we lack such information. From the original query set, we removed all the queries for which BM25 does not retrieve any relevant document in the top-1000 results. Likewise, for each of the remaining queries, we retain only the relevant documents present in the top 1000 results retrieved by BM25. Before computing the BM25 results, to maintain most queries and relevant documents as possible, we fine-tuned its parameters on the validation queries. Table 4.1 reports the statistics of the final dataset. We make the

dataset and all the scripts to re-build it available for future research[7].

4.5.2 Baselines

In this section, we introduce the baselines employed in our comparative evaluation. We compared the *Denoising Attention*-based user model with user models based on the standard *Attention* formulation, the *Zero Attention strategy* proposed by Ai et al. [7], and the *Multi-Head Attention* [272]. We also considered a user model built by simply averaging the user-related documents' representations to assess whether *Attention*-based user models can improve over simpler operations for aggregating user data.

For reference, we also performed comparison with the classic probabilistic retrieval model BM25 [226], which we used as first stage retriever.

- **Attention:** The first baseline is a query-aware user model based on the standard *Attention* formulation.

- **Zero Attention:** The second baseline is a query-aware user model based on the *Zero Attention Strategy* proposed by Ai et al. [7]. The *Zero Attention Strategy*, introduced in Section 4.1, was proposed to automatically determines *when* and *how* to conduct personalization.

- **Multi-Head Attention:** The third baseline is a query-aware user model based on the *Multi-Head Attention* [272], a scaled-up variant of the *Attention* mechanism. *Multi-Head Attention allows the model to jointly attend to information from different representation subspaces* [272]. We use 4 *Attention* heads in our experiments.

- **Mean:** The fourth baseline is *static* user model that computes user representations as the arithmetic mean of the user-related documents' representations. As

[7]We will add a link to the repository upon acceptance

averaging is a simple form of vector aggregation, its addition to the evaluation allows us to assess whether query-aware user modeling techniques are *truly* beneficial.

We trained three variants for both the *Attention*-based and the *Zero Attention*-based user models by employing different *alignment functions*. The first variant employs the *scaled-dot product*, popularized by the Transformer architecture [272]. The second one uses the cosine similarity, similarly to our *Denoising Attention*. The latter relies on the *alignment model* proposed by Bahdanau et al. [19] in the paper where the *Attention* mechanism was proposed, which is commonly called *Additive Attention* [272].

We leave experimentation and comparison with Transformer models [272] for future work. We note that our proposed *Attention* variant could also be used in place of the standard formulation in these complex architectures.

4.5.3 Setup & Evaluation Metrics

We relied on ElasticSearch [98] for BM25, HuggingFace's Transformers [280] for TinyBERT[8], and PyTorch [215] for the implementation of all the neural models. We fine-tuned BM25's $k1$ and b parameters on thousands of non-test data before computing the result lists for all the queries. BM25 scores were computed on the concatenation of documents' title and body (*the papers' abstracts*) by first removing stop-words and applying the Krovetz stemmer [139]. We trained each variation of the *Personalized Results Re-Ranking Framework* introduced in Section 4.4 on an NVidia® RTX 2080 Ti GPU for 20 epochs using a hinge loss [279] defined over a triplet, similarly to [93], with a margin set to 0.1 (*we found larger margins to decrease performances in all cases*), and AdamW optimizer [162, 163] with learning rate set to 5×10^{-5}, and batch size set to 32. We train the model with hard negatives sampled from the top results retrieved by BM25 and in-batch random negatives samples.

[8] https://huggingface.co/huawei-noah/TinyBERT_General_4L_312D

During training, due to our hardware limitations, we randomly sampled the titles of 20 user documents to use for personalization, while during the evaluation, we used those from all the available user documents. After training, we fine-tuned the λ parameter of Eq. 2.1 and the *Denoising Attention*'s threshold with grid-search on the *validation set*. In the case of the Academic Search Dataset, during the experiments, for each query, we skipped all the documents published after the release of the manuscript used for generating the query as none of them has relevant judgments for it because of dataset construction and, therefore, they would only add noise to the evaluation as suggested by Tabrizi et al. [257]. For the final evaluation, we re-ranked the top 1000 results retrieved by BM25 with each of the considered user models.

To evaluate the effectiveness of the compared models, we employed 1) *Mean Average Precision* (MAP), 2) *Mean Reciprocal Rank* (MRR), and 3) *Normalized Discounted Cumulative Gain* (NDCG). MRR and NDCG were computed on the top 10 documents retrieved by each model, whereas MAP was computed on the top 100. Statistical significance testing was conducted using a Bonferroni corrected Fisher's randomization test [244] with $p < 0.001$. Metrics computation and comparison were conducted using the Python evaluation library `ranx` [23].

4.6 Results and Discussion

In this section, we present the results of our comparative evaluations. First, we discuss the retrieval effectiveness of the personalized re-ranking pipeline described in Section 4.4 when considering different user modeling techniques. Second, we evaluate how balanced the user preferences expressed by the considered user models are w.r.t. each query. Third, we analyze the robustness of the compared user models, evaluating the probability they decrease the system's effectiveness in the presence of noisy user-related data. Finally, we analyze the performance of our proposed

Attention variant from multiple perspectives and ablate our proposal. We remind the reader that the only difference between the compared personalization models is the technique used for defining the user model, while the other components are the same for all the considered models.

4.6.1 Overall Retrieval Effectiveness

In this section, we review the results obtained when combining a document's relevance score produced by different user models with the document's relevance score produced by BM25, as described in Section 4.4, aiming to answer questions **RQ1** and **RQ2**.

As reported in Table 4.2, combining personalized relevance scores with those coming from our first stage retriever, BM25, improved the retrieval effectiveness of the latter regardless of the user modeling mechanism employed, thus confirming the utility of personalization in both the considered search scenarios and datasets.

The *Attention* and *Zero Attention*-based user models generally improved over the *Mean* user model. However, we notice this is not always the case. Of all the different variants, only those using the *scaled-dot* as an *alignment model* significantly improved over *Mean* on both the considered datasets. On the other hand, those relying on the *additive* and the *cosine alignment models* achieved mixed results, sometimes even decreasing w.r.t. *Mean*. Moreover, we highlight that in the case of the *Web Search Dataset*, the best-performing *Attention* baselines' improvements are not that pronounced (only 3% in MAP, MRR, and NDCG, respectively). The *Zero Attention*-based user models generally achieved slightly worse results than their *Attention*-based counterparts, which raises questions regarding the efficacy of its employed mechanism for conducting differentiated personalization. Our findings on the *Zero Attention*-based user model are consistent with results from previous works [34, 35, 126]. The results obtained by both the standard *Attention*-based user model and the *Zero Attention*-based

user model with the cosine similarity as *alignment model* confirm that constraining the *alignment scores* causes noisy information to leak in the user model, as discussed in Section 4.2.2. Finally, the *Multi-Head Attention*-based user model's results are among the lowest for both datasets. The additional complexity introduced by this approach did not deliver improvements over the other *Attention*-based models while introducing additional overhead. We suspect this is because we are employing high-quality text representations obtained using TinyBERT [128], a distilled version of the well-known Transformer [272] model BERT [76], which alleviates the need for *attending information from different representation sub-spaces* [272] during personalization. We leave further investigation for future work. If we consider only the *Attention*-based user model with the *scaled-dot alignment model*, the obtained results positively answer our first research question, **RQ1**. However, this is not the case for all the other *Attention* baselines, which confirms the need for the in-depth investigation we are conducting on the use of the *Attention* mechanism for query-aware personalization.

When employing the *Denoising Attention*-based user model, the results re-ranking pipeline achieved substantial improvements over both that using the *Mean*-based user model and those relying on *Attention*-based user models, corroborating our intuitions about the shortcomings of the standard *Attention* formulation when it comes to personalization (Section 4.2.2) and the advantages brought by our proposal (Section 4.3). In particular, it improved over *Mean* by 20%, 22%, and 19% in MAP, MRR, and NDCG, respectively, on the *Web Search Dataset* and by 23%, 15%, and 21% in MAP, MRR, and NDCG, respectively, on the *Academic Search Dataset*. Moreover, it increased over the best-performing *Attention*-based baseline by 17%, 18%, and 16% in MAP, MRR, and NDCG, respectively, on the *Web Search Dataset* and by 14%, 10%, 13% in MAP, MRR, and NDCG, respectively, on the *Academic Search Dataset*. Finally, it enhanced the BM25 effectiveness by 38%, 41%, and 40% in MAP, MRR, and NDCG, respectively, on the *Web Search Dataset* and by 50%, 29%, and 41% in MAP,

MRR, and NDCG, respectively, on the *Academic Search Dataset*. The obtained results clearly show the robustness of our proposed *Attention* variant to search scenarios with noticeable structural differences. On one side, for *Web Search*, it is fundamental to finely select among the user-related data those most promising for conducting personalization to improving over standard operations for building user models, *i.e.*, averaging over the representations of the user-related data. On the other side, in the case of *Academic Search*, user-related information is very focused and, therefore, it is easier to improve a user model that averages the representations of the user-related data by simply weighing the contribution of those data w.r.t. the current query. Nonetheless, *Denoising Attention* still exhibits significant advantages over the other *Attention* variants. These results, which positively answer our second research question, **RQ2**, highlight the importance of correctly managing the user-related information in personalization and the potential of deepening this research area.

4.6.2 Weighting Schemes Comparison

In this section, aiming to answer our third research question (**RQ3**), we compare the mechanism employed by our proposed Attention variant to weigh the contribution of the user-related data in composing the user model at query time with the other considered Attention baselines. We also consider the Mean-based user model, which uses an even weighting scheme, as a reference. To conduct this evaluation, we consider only the queries for which Denoising Attention outputs a non-zero user model and employ only the scores deriving from the comparisons between the user models and the documents to re-rank the initially retrieved BM25 result lists. We assume that if a user model achieves significantly better retrieval effectiveness than the others, then its weighting scheme is better, as we employ the same user-related information to build all the user models. Since the goal of re-ranking is to improve

Table 4.2: Effectiveness of BM25 and those of the Personalized Results Re-Ranking Framework with different user models. $*$ and † denote significant improvements in a Bonferroni corrected Fisher's randomization test with $p < 0.001$ over *Mean* and over all the baselines, respectively. Best results are highlighted in boldface.

Web Search Dataset						
Model	**Alignment**	**MAP@100**	**MRR@10**	**NDCG@10**	λ	$\sigma(t)$
BM25	—	0.245	0.238	0.280	—	—
Mean	—	0.282	0.276	0.329	0.2	—
Attention	Additive	0.281	0.276	0.328	0.2	—
	Cosine	0.287*	0.281*	0.335*	0.2	—
	Scaled-Dot	0.290*	0.285*	0.339*	0.2	—
Zero Attention	Additive	0.277	0.272	0.325	0.2	—
	Cosine	0.286*	0.281*	0.334*	0.2	—
	Scaled-Dot	0.290*	0.285*	0.338*	0.2	—
Multi-Head	Scaled-Dot	0.275	0.269	0.324	0.2	—
Denoising	Cosine-based	**0.338**†	**0.336**†	**0.393**†	0.4	0.7
Academic Search Dataset						
Model	**Alignment**	**MAP@100**	**MRR@10**	**NDCG@10**	λ	$\sigma(t)$
BM25	—	0.119	0.294	0.171	—	—
Mean	—	0.146	0.328	0.200	0.6	—
Attention	Additive	0.156*	0.340*	0.213*	0.6	—
	Cosine	0.151	0.332	0.206	0.6	—
	Scaled-Dot	0.157*	0.343*	0.214*	0.6	—
Zero Attention	Additive	0.155*	0.338	0.211*	0.6	—
	Cosine	0.150	0.330	0.204	0.6	—
	Scaled-Dot	0.156*	0.341*	0.212*	0.6	—
Multi-Head	Scaled-Dot	0.152	0.336	0.207	0.6	—
Denoising	Cosine-based	**0.179**†	**0.378**†	**0.241**†	0.6	0.6

over a first-stage retriever, we also consider BM25 results for reference.

As reported in Table 4.3, there is generally a clear difference in the retrieval effectiveness of the *Attention*-based user models and that of the *Mean*-based user model, highlighting the potential of weighing the contribution of multiple sources of user-related information during personalization. In the best case scenario (*Scaled-*

dot), the *Attention*-based baselines increased over *Mean* by 11%, 13%, and 11% in MAP, MRR, and NDCG, respectively, on the *Web Search Dataset*, and by 34%, 30%, and 34% in MAP, MRR, and NDCG, respectively, on the *Academic Search Dataset*.

Despite the already good improvements brought by the Attention as a mechanism for weighting user-related data in query-aware personalization, the Denoising Attention reached a significantly higher level of effectiveness. With respect to the best performing Attention baselines, the Denoising Attention improved by 73%, 86%, and 76% in MAP, MRR, and NDCG, respectively, on the *Web Search Dataset*, and by 57%, 53%, and 55% in MAP, MRR, and NDCG, respectively, on the *Academic Search Dataset*. This great difference is due to both the shortcoming of the Attention mechanism described in Section 4.2.2, which makes it under-perform in many situations, and the solution we have proposed to solve them, which allows filtering noisy information and does not reduce the diversity of the estimated importance of the user-related documents. We also highlight that the Denoising Attention-based user model is the only user model that improved over our first stage retriever, BM25, while of the other considered user models largely decreased its effectiveness. Specifically, the *Denoising Attention*-based user model improved over BM25 by 10%, 10%, and 14% in MAP, MRR, and NDCG, respectively, on the *Web Search Dataset*, and by 19%, 08%, and 13% in MAP, MRR, and NDCG, respectively, on the *Academic Search Dataset*. These results positively answer our third research question, **RQ3**

4.6.3 Robustness

In this section, we evaluate and discuss the *robustness* of the considered user models when used in combination with BM25, aiming to answer our fourth research question, **RQ4**. Specifically, we consider the number of times personalization decreased BM25 effectiveness in terms of MAP@100. We remind the reader that the Web Search Dataset based on the AOL query log we employed has 36 052 test queries, while the

Table 4.3: Effectiveness of BM25 and those of the user models when used in isolation. $*$ and † denote significant improvements in a Bonferroni corrected Fisher's randomization test with $p < 0.001$ over *Mean* and over all the baselines, respectively. Best results are highlighted in boldface.

Web Search Dataset						
Model	**Alignment**	**MAP@100**	**MRR@10**	**NDCG@10**	λ	$\sigma(t)$
BM25	—	0.240*	0.233*	0.274*	—	—
Mean	—	0.136	0.120	0.157	1.0	—
Attention	Additive	0.136	0.120	0.155	1.0	—
	Cosine	0.141*	0.125*	0.166*	1.0	—
	Scaled-Dot	0.152*	0.137*	0.177*	1.0	—
Zero Attention	Additive	0.125	0.108	0.144	1.0	—
	Cosine	0.148*	0.132*	0.169*	1.0	—
	Scaled-Dot	0.153*	0.138*	0.177*	1.0	—
Multi-Head	Scaled-Dot	0.128	0.111	0.148	1.0	—
Denoising	Cosine-based	**0.264**†	**0.256**†	**0.312**†	1.0	0.7
Academic Search Dataset						
Model	**Alignment**	**MAP@100**	**MRR@10**	**NDCG@10**	λ	$\sigma(t)$
BM25	—	0.120*	0.295*	0.172*	—	—
Mean	—	0.068	0.160	0.094	1.0	—
Attention	Additive	0.090*	0.205*	0.123*	1.0	—
	Cosine	0.076*	0.172*	0.103*	1.0	—
	Scaled-Dot	0.091*	0.208*	0.125*	1.0	—
Zero Attention	Additive	0.086*	0.195*	0.117*	1.0	—
	Cosine	0.075*	0.171*	0.103*	1.0	—
	Scaled-Dot	0.088*	0.201*	0.120*	1.0	—
Multi-Head	Scaled-Dot	0.074*	0.172*	0.101*	1.0	—
Denoising	Cosine-based	**0.143**†	**0.319**†	**0.194**†	1.0	0.6

Academic Search Dataset has 24 056. Table 4.4 shows the number of times personalization was actually harmful to the retrieval effectiveness in terms of MAP@100. Quite surprisingly, the *Attention*-based user models are often more harmful than the user model based on the average of the user document representations, Mean, although more effective in general, as previously reported. Conversely, the *Denoising*

Table 4.4: Number of times (and ratios) personalization decreased BM25 effectiveness in terms of MAP@100 (lower is better). Best results are highlighted in boldface. Best baselines are highlighted in italic.

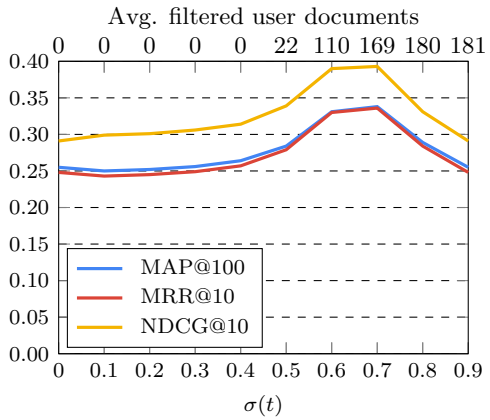
Model	**Alignment**	**Web Search Dataset**	**Academic Search Dataset**
Mean	—	10 798 (30%)	6 165 (26%)
Attention	Additive	11 157 (31%)	6 076 (25%)
	Cosine	9 877 (27%)	6 580 (27%)
	Scaled-Dot	9 426 (26%)	*5 954 (25%)*
Zero Attention	Additive	11 508 (32%)	6 201 (26%)
	Cosine	10 234 (28%)	6 708 (28%)
	Scaled-Dot	*9 356 (26%)*	6 131 (25%)
Multi-Head	Scaled-Dot	12 049 (33%)	6 366 (26%)
Denoising	Cosine-based	**6 780 (19%)**	**5 509 (23%)**

Attention-based user model considerably decreased for both datasets the number of times personalization harmed the retrieval process w.r.t. the other considered user models. Compared to the *Denoising Attention*-based user model, the best baselines on the Web Search Dataset and the Academic Search Dataset decreased the retrieval effectiveness of BM25 for 38% and 8% more queries, respectively. The much more significant difference between the *Denoising Attention*-based user model and the other considered user models on the Web Search Dataset than on the Academic Search Dataset is due to the different nature of the employed datasets. In the Web Search Dataset, the user-related data accounts for many different user interests, while on the Academic Search Dataset, they are much more focused on particular topics. In the first case, personalization is much more likely to harm the retrieval process if a filtering mechanism for the user information is not employed, as in the case of all the considered user models but the one based on our *Attention* variant. We conclude the *Denoising Attention*-based user model is much more robust than the other considered user models regardless of the search scenario, which positively answer our fourth research question, **RQ4**.

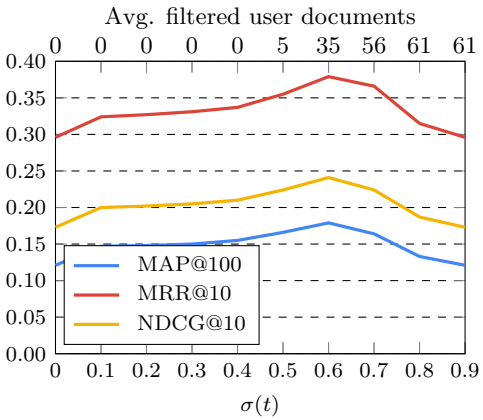
4.6.4 Model Analysis

In this section, we first evaluate the *Denoising Attention*-based user model performances for various threshold values. Then, we compare the performances of the considered user models for queries with various amounts of associated user-related documents. Finally, we ablate the design choices underlying our proposed *Attention* variant.

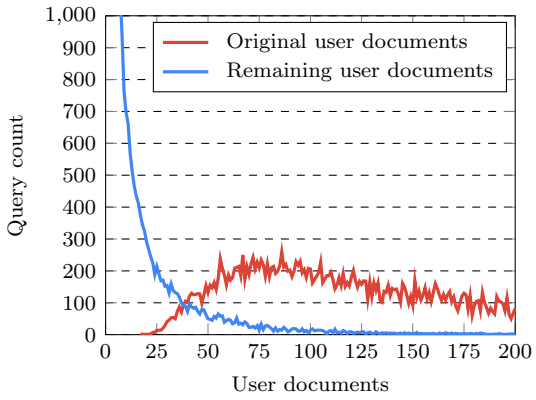
Threshold Figures 4.2a and 4.2b show the performances of the results re-ranking pipeline with the *Denoising Attention*-based user model for different threshold values on the considered datasets. The figures also report the average number of filtered user documents for each considered threshold value. On average, the test queries have 181 and 61 associate user-related documents in the *Web Search Dataset* and the *Academic Search Dataset*, respectively, while the average number of filtered ones for the best threshold values are 169 and 35, respectively. The different ratios of average filtered user-related documents are again due to the distinct nature of the two search scenarios and datasets. Our proposed approach is able to adapt to different search contexts thanks to the threshold parameter and our filtering mechanism. When the threshold is zero, which corresponds to not filtering any user-related document in our case, the model effectiveness is very low for both datasets. When the threshold is equal to 0.5, which corresponds to using the cosine similarity with no modification as our *alignment model*, the model still does not reach its full potential. These results highlight again the need for a filtering mechanism that can be tuned and modulated. In Figures 4.2c and 4.2d, we can observe how the distribution of the documents used for personalization changes using the *Denoising Attention*-based user model. As expected, the number of documents used for personalization decreases drastically in both the considered search scenarios. In particular, for the *Web Search* scenario, which comprises, for each user, a very diverse set of user-related documents, we registered



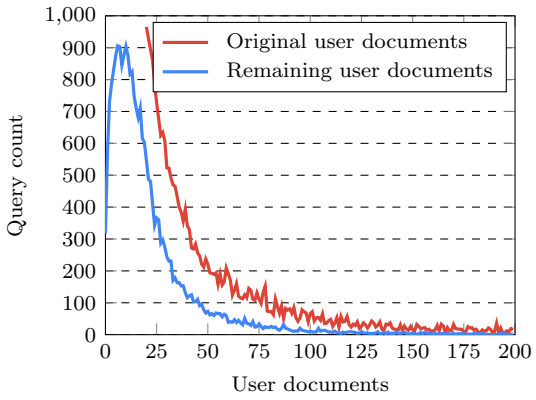
(a) Retrieval effectiveness of the personalized re-ranking pipeline with *Denoising Attention*-based user model on the *Web Search Dataset*



(b) Retrieval effectiveness of the personalized re-ranking pipeline with *Denoising Attention*-based user model on the *Academic Search Dataset*



(c) Remaining user documents for $\boldsymbol{\sigma(t) = 0.7}$ on the *Web Search Dataset*



(d) Remaining user documents for $\boldsymbol{\sigma(t) = 0.6}$ on the *Academic Search Dataset*

Figure 4.2: Threshold analysis.

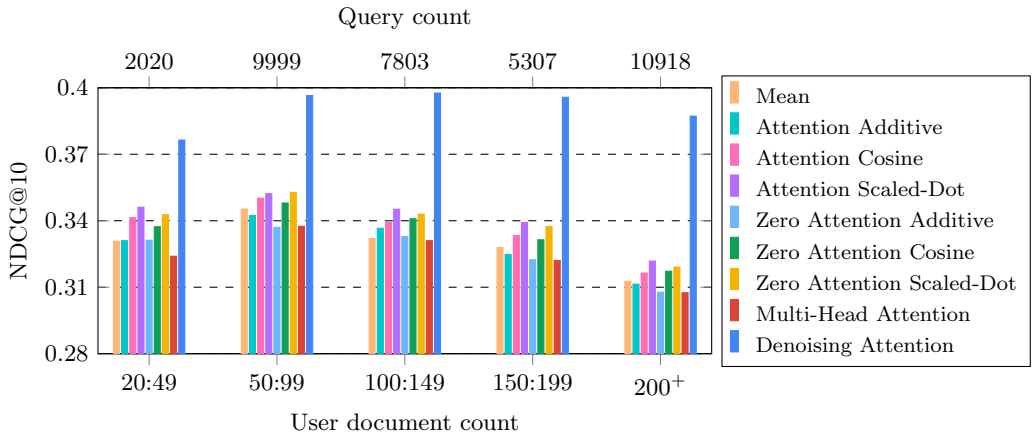
a very pronounced selection of the user data used for personalization.

As the filtering mechanism employed by the *Denoising Attention* is applied to each user document independently, the model can filter numerous documents that are not strictly related to the query.

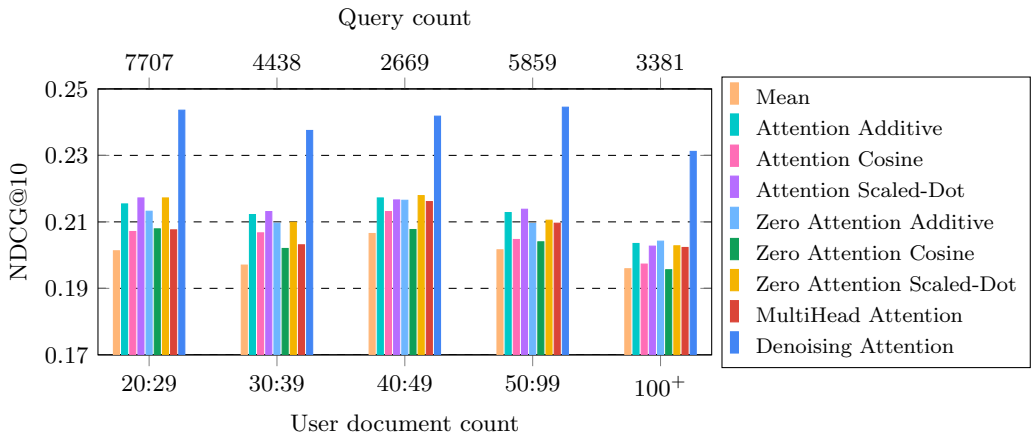
User Document Count Figures 4.3a and 4.3b report the performances of the results re-ranking pipeline with each of the considered user models on the test queries grouped by their amounts of associated user-related documents on the *Web Search*

Dataset and the *Academic Search Dataset*, respectively. We grouped queries by the range in which their associated amount of user documents lays. In particular, we considered the queries in the following ranges: 20:49, 50:99, 100:149, 150:199, and 200^+ for the *Web Search Dataset*, and 20:29, 30:39, 40:49, 50:59, and 100^+ for the *Academic Search Dataset*. We chose different ranges for the two datasets as their user document distribution is very different, as shown in Figures 4.2a and 4.2b. Moreover, by datasets construction (Section 4.5.1), all queries have at least 20 associated user documents. As shown in the figures, the baselines achieved mixed results. It is not clear at a glance which one performed the best overall. Looking closely, we can see that the *Attention*-based user model with the *scaled-dot alignment model* (*purple bar*) generally performed better than or equal to the other baselines in the considered ranges. Conversely, the *Denoising Attention*-based user model consistently outperformed all the considered baselines for each group of queries in each dataset, showing its benefits generalize regardless of the number of available user documents and search scenario.

Ablation Study Table 4.5 shows the performances of the results re-ranking pipeline with the *Denoising Attention*-based user model and with some variations derived by ablating our proposal. For comparison purposes, we added the results of the best performing baseline from previous experiments, the *Attention*-based model with the *scaled-dot alignment model*. The first variation of our proposed *Attention* variant, called *Filter Attention*, employs the *ReLU*-based filtering mechanism we proposed and Eq. 4.9 for the *normalization* step to the *Attention* with the *scaled-dot alignment model*. As discussed before, using Eq. 4.9 for the *normalization* step is mandatory for the filtering mechanism to have effect as *Softmax* normalization is *translationally invariant*. We did not use the threshold parameter in this variation of the *Denoising Attention* as the employed *alignment models*' output is unbounded, making it difficult to calibrate such a parameter. For the other variation we considered the *Denoising*



(a) *Web Search Dataset*



(b) *Academic Search Dataset*

Figure 4.3: Effectiveness of the user models when combined with BM25 for queries with different amounts of associated user-related documents.

Attention with *Softmax* normalization to show the need of using Eq. 4.9 and departing from *Softmax* to make our proposed mechanism to work properly. We called this variation *Denoising Softmax*.

As shown in the table, *Denoising Softmax* performs poorly and decreases the retrieval effectiveness of the system w.r.t. our best performing baseline on both datasets, confirming that using our filtering mechanism with *Softmax* normalization does not perform properly. On the other hand, *Filter Attention* significantly improved over the *Attention*-based model with the *scaled-dot alignment model*, corroborating our intuition that the *Softmax* normalization is not optimal in the context of personalization and

Table 4.5: Effectiveness of the Personalized Results Re-Ranking Framework with different *Denoising Attention* variations. † denotes significant improvements in a Bonferroni corrected Fisher's randomization test with $p < 0.001$ over over all the baselines. Best results are highlighted in boldface.

Web Search Dataset						
Model	**Alignment**	**MAP@100**	**MRR@10**	**NDCG@10**	λ	$\sigma(t)$
Attention	Scaled-Dot	0.290	0.285	0.339	0.2	—
Filter Attention	Scaled-Dot	0.299	0.294	0.351	0.3	—
Denoising Softmax	Cosine-based	0.285	0.280	0.334	0.2	0.1
Denoising	Cosine-based	**0.338**†	**0.336**†	**0.393**†	0.4	0.7
Academic Search Dataset						
Model	**Alignment**	**MAP@100**	**MRR@10**	**NDCG@10**	λ	$\sigma(t)$
Attention	Scaled-Dot	0.157	0.343	0.214	0.6	—
Filter Attention	Scaled-Dot	0.165	0.354	0.223	0.6	—
Denoising Softmax	Cosine-based	0.151	0.332	0.206	0.5	0.1
Denoising	Cosine-based	**0.180**†	**0.382**†	**0.243**†	0.6	0.6

suggesting the proposed alternative is effective regardless of the employed *alignment model*. The *Denoising Attention* significantly outperformed all of its considered variations, verifying the utility of our design choices and their complementarity.

4.7 Summary

In this chapter, we have addressed some issues related to the use of the *Attention* mechanism for query-aware user modeling and proposed a novel user-data aggregation model called *Denoising Attention*, designed to solve the shortcomings of the standard *Attention* formulation and, in particular, filter out noisy user-related information. Experimental evaluation in two different search scenarios, namely *Web Search* and *Academic Search*, shows the benefits of our proposed approach over other *Attention* variants and highlights the potential of correctly managing the user-related information. Finally, the ablation study we conducted clearly illustrates the benefits of our design choices and their synergy.

CHAPTER 5

PERSONALIZED QUERY EXPANSION WITH CONTEXTUAL WORD EMBEDDINGS

Nowadays, most search engines provide users with a simple interface to specify their information needs through short keyword-based queries, which are usually two-to-three terms long in the case of Web search [124]. However, as a query only broadly describes a user's information need, search engines may struggle to provide satisfactory results. Multiple factors related to how users choose terms for their queries can affect a system's retrieval effectiveness [18]. For example, the terms composing a query can be related to multiple topics, leading the system to provide results not focused on the user's topic of interest. Moreover, out of habit, users often issue queries too short to clearly express complex information needs, ultimately failing to find documents valuable to fulfill them. Finally, users sometimes have only a broad idea of the information they need, and hence they issue queries that are not appropriate to find documents that can answer their information needs. A well-known technique proposed to overcome those issues is Query Expansion, whereby the user's original query is augmented with new terms, known as *expansion terms*, to improve the system's effectiveness. The identification of proper expansion terms aims to clarify the user's search intent and bridges the gap between the original query terms and the documents' vocabulary [55], addressing the well-known vocabulary mismatch problem [90]. Query Expansion techniques can leverage user-related information previously gathered to derive the expansion terms, in which case we talk about Personalized Query Expan-

sion [62, 207, 32, 135, 51, 110, 37, 304, 39, 239, 198, 127, 143, 282, 300, 13]. Personalized Query Expansion techniques rely on user-related documents, such as previously accessed Web pages and user-generated content [140], e.g., product reviews or tweets, to extract expansion terms directly from the users' vocabulary or the vocabulary used in documents of their interest.

Historically, most approaches to Personalized Query Expansion [29, 32, 37, 39, 299, 198, 282] focused on leveraging social information derived from folksonomy platforms[1] to extract expansion terms. On those platforms, like the former social bookmarking website `del.icio.us`[2], users apply public tags to online items, such as Web pages. Works in this area addressed the selection of personalized expansion terms by relying on term co-occurrence-based approaches and social relations analyses.

More recently, to overcome the limitations of lexical matching-driven term co-occurrence analysis, which suffers from the vocabulary mismatch problem, researchers [13, 300, 143] started experimenting with word embedding models, which project text into dense low dimensional vector spaces where the semantic similarity among terms can be computed as the cosine similarity of their vector representations. Existing approaches [13, 300, 143] rely on the well-known Word2Vec model [191, 192] to generate word embeddings for both queries and user-related texts and on cosine similarity to evaluate their semantic relatedness, which they use to select the personalized expansion terms. A limitation of the word embeddings produced by Word2Vec and similar models is that terms are always mapped to the same vectors regardless of their context, which usually varies for each occurrence of a term.

Recently, to overcome the limitations of traditional word embeddings, new techniques [217, 76, 47, 221] have been introduced. These new methods map each word occurrence to a unique representation based on its surrounding terms, thereby cap-

[1] https://en.wikipedia.org/wiki/Folksonomy
[2] https://en.wikipedia.org/wiki/Delicious_(website)

turing the different meanings it can assume across varied contexts. The new representations are commonly called *contextual* word embeddings and have allowed reaching a new state-of-the-art in many different Natural Language Processing tasks. In the past few years, contextual word embeddings have also been successfully applied to Information Retrieval [158], advancing the state-of-the-art in multiple tasks and opening new opportunities and challenges for retrieval-enhancing tasks, such as Personalized Query Expansion. In the following, we use the locution "word embedding" to refer to the "word embedding techniques" and "term embedding" to refer to the actual vector representation computed by one of those techniques for a given term.

In this chapter, we address Personalized Query Expansion using contextual word embeddings, which, as mentioned above, open new opportunities for this task while also introducing new challenges. We argue that two of the main challenges in employing contextual word embeddings to select expansion terms are (i) reducing redundancy among expansion terms and (ii) addressing scalability issues. Previous Personalized Query Expansion methods based on word embeddings [13, 300, 143] rely on ranking functions based on cosine similarity to rank all the user-related terms before selecting those for query expansion. However, when working with contextual word embedding models, which produce a unique embedding for each term occurrence, we could end up with very similar embeddings for multiple occurrences of the same term appearing in similar contexts. We argue that, if not carefully handled, this aspect of contextual word embeddings could cause the selection of multiple expansion term embeddings with very close, if not identical, semantic meanings, thus reducing the potential utility of Query Expansion. Because of the lack of a mechanism accounting for the presence of multiple, very similar embeddings, previous methods have a high probability of selecting expansion terms that are redundant with each other, as we will show. Moreover, as previous approaches [13, 300, 143] rely on com-

puting a similarity score between the query and each user-related term embedding to select those most appropriate for expanding the query, they could introduce an overhead proportional to the number of candidate expansion terms. While with traditional word embeddings, such as Word2Vec, each unique term is represented once, with contextual word embeddings, we have a different representation for each term occurrence, potentially making the problem much more severe. This issue could cause query expansion methods based on contextual word embeddings to suffer a low-scalability problem, making their application in data-rich real-world scenarios debatable, such as in Web Search, where we could leverage very long user browsing histories to conduct personalization.

In this chapter, we present PQEWC (pronounced *"quick"*), a **P**ersonalized **Q**uery **E**xpansion method designed to work **W**ith **C**ontextual word embeddings. To address the scalability issues arising from the adoption of contextual word embeddings in Personalized Query Expansion and to reduce the impact of potentially redundant expansion terms, we employ an offline clustering-based procedure aiming at grouping the user-related terms and identifying those that better represent the user interests. By selecting only the expansion term that better represents the user interests w.r.t. the current query from each cluster, we avoid adding to the query multiple expansion terms with similar semantic meanings, thus reducing the chance of expanding it with redundant expansion terms. Finally, we implement an approximation mechanism for selecting the expansion terms, which allows our proposed approach to achieve a sub-millisecond expansion time even in very data-rich scenarios, making it suitable for many real-world applications.

The rest of the chapter is organized as follows. Section 5.1 discusses the related works and positions our work w.r.t. them. In Section 5.2, we present our novel Personalized Query Expansion approach and discuss our design choices. Section 5.3 introduces the retrieval task we tackled to evaluate our proposal. Section 5.4 presents

our research questions and describes the experimental setup of our comparative evaluation. Finally, in Section 5.5, we compare our proposed approach and other query expansion methods at the state-of-the-art [278, 143, 300] both in terms of effectiveness and efficiency, and ablate our design choices. The results of our evaluation clearly show the advantages of PQEWC w.r.t. the considered query expansion baselines, which are outperformed both in retrieval effectiveness and efficiency. Across all the considered datasets, the proposed approach improves by up to 4% in terms of MAP@100 over our base retrieval system, based on BM25 [226] and ColBERT [136], and by up to 3% w.r.t. the best performing baseline [143]. We will share all the code to reproduce the experimental evaluation we conducted once the related article will be published.

5.1 Related Work

Query Expansion is a well-established technique in Information Retrieval. It has received significant attention from the research community in the past few decades and continues to attract many researchers. In this section, we first cover the state-of-the-art of Query Expansion, and then we focus on its Personalized counterpart. In both cases, we pay particular attention to the methods based on word embeddings.

5.1.1 Query Expansion

Among the several approaches proposed for Query Expansion [55, 18], a line of research that still attracts the research community's interest is represented by the methods founded on the pseudo-relevance feedback technique [228]. These methods [123, 60, 285, 52, 173, 67, 150] rely on a first retrieval stage to collect the so-called feedback documents, i.e., documents appearing in the top positions of the ranked list of documents, which are assumed to be relevant w.r.t. the query and from which

terms to expand the initial query are extracted . Query Expansion methods based on pseudo-relevance feedback have proved their effectiveness over the years and are still relevant today. The most important of these models is RM3 [123], which leverages statistical information on the occurrences of terms in feedback documents and in the corpus to select the expansion terms. Intuitively, RM3 expands the initial query with terms that are frequent in the feedback documents and infrequent in the corpus.

With the advent of word embedding techniques, new Query Expansion methods leveraging semantic term representations have been proposed [142, 231, 77]. Instead of exploiting the pseudo-relevance feedback documents using statistical methods to select the expansion terms, those methods choose them by evaluating the semantic similarity between the query terms and the corpus vocabulary. Generally, they expand a query with the closest terms in the word embedding space, i.e., the most semantically similar terms. For example, Kuzi et al. [142] propose to use the Word2Vec model [191, 192] to compute latent representations of all terms appearing in the corpus on which the search is conducted, and to apply cosine similarity to select expansion terms that are semantically related to the query. Similarly, Roy et al. [231] rely on Word2Vec to obtain word embeddings for their corpus vocabulary. To select the expansion terms for a given query, the authors employ a k-nearest neighbor method based on cosine similarity. The authors found their approach could improve over their underlying retrieval model without expansion but not over the pseudo-relevance feedback expansion model RM3. Diaz et al. [77] investigate whether training word embedding models such as Word2Vec and GloVe [216] *locally*, i.e., on the available test collection, instead of using *globally* trained models, i.e., models trained on general domain-agnostic texts, can benefit Query Expansion. The authors found locally-trained word embeddings to generally improve the performance of Query Expansion w.r.t. globally-trained word embeddings. The most significant drawback of those methods, which do not deliver significant improve-

ments over the pseudo-relevance feedback approaches, is the lack of a mechanism to identify the most prominent terms from a retrieval perspective, i.e., the terms that allow improving the identification of the relevant documents, as only the terms' semantic relatedness is considered. Moreover, after expansion, the authors rely on traditional retrieval models based on lexical term matching, which notoriously do not account for semantic relatedness.

More recently, the contextual word embedding techniques renovated the research community's interest in Query Expansion, and novel approaches based on this new kind of embedding were proposed [297, 200, 278]. The authors of new approaches, aware of the limitations of previous methods based on word embeddings, combine the new contextual word embedding techniques with the pseudo-relevance feedback approach.

Zheng et al. [297] propose a novel Query Expansion method based on contextual word embeddings that leverage a BERT-based [76] re-ranker [202] in a pseudo-relevance feedback fashion. After a first re-ranking round, the most relevant text chunks are extracted from the top re-ranked documents and used to compute additional relevance scores for the documents. Finally, the newly computed relevance scores are aggregated with the original ones to obtain the scores to compute the final documents ranking. Their experimental evaluation shows that the proposed model delivers promising retrieval effectiveness improvements. Naseri et al. [200] revisit the pseudo-relevance feedback approaches to Query Expansion by employing the similarity between the query's contextual word embeddings and those of the feedback documents in deriving probability values to use in in place of those of the original formulation. Although improving over non-contextual Query Expansion methods based on word embeddings, the model proposed by Naseri et al. [200] only performs on par with the classic expansion method RM3 [123]. Wang et al. [278] have recently introduced ColBERT-PRF, a novel query expansion method based on

the neural retrieval model ColBERT [136] and pseudo-relevance feedback. After a first ranking stage with ColBERT, this method leverages Kmeans clustering [180] to group the term embeddings of a certain number of feedback documents. Then, it selects the tokens corresponding to the cluster centroids with higher Inverse Document Frequency scores [129] for expanding the original query. The authors report encouraging improvements over ColBERT without query expansion as well as many other baselines. Unfortunately, despite some promising improvements in terms of retrieval effectiveness, previous Query Expansion methods based on contextual word embeddings suffer from poor efficiency [278], limiting their applicability in real-world applications.

5.1.2 Personalized Query Expansion

In the early 2000s, the increasing popularity of social tagging systems, where users can associate public tags with online items such as Web pages, attracted some attention from the research community thanks to the large amount of accessible data provided by those platforms. In particular, researchers leveraged those data to derive test beds for Personalized Information Retrieval in social network-like environments [40, 41, 42, 43, 284, 276, 299]. Among the approaches for personalization proposed in this period, several Personalized Query Expansion methods were presented [29, 32, 37, 39, 299, 198, 282, 44]. Most of the works in this area approach the selection of personalized expansion terms by leveraging both term co-occurrences statistics and social relations among the users. For example, Bender et al. [29], Bertier et al. [32], Mulhem et al. [198], and Wu et al. [282] derive terms for Personalized Query Expansion by leveraging the relations and similarities among users, documents, and tags. Biancalana and Micarelli [37] propose a method for selecting expansion terms based on a three-dimensional co-occurrence matrix from which the authors derive relations among the query terms appearing in a document, terms associated with

similarly tagged documents, and those appearing in user-related documents, among which the authors select the expansion terms. Bouadjenek et al. [39, 44] approach Personalized Query Expansion by employing a combination of social proximity and semantic similarity to identify the terms similar to those mostly used by a given user and his social relatives. Unfortunately, we found most of the previous works to lack comparisons with other Personalized and non-Personalized Query Expansion methods, making it difficult to draw general conclusions about their effectiveness.

Other than the works related to social tagging systems, the literature comprises some approaches leveraging other contextual data. For example, Zhu et al. [304] leverage the co-occurrences of the query terms with terms from user-related documents located in their desktop environment to select personalized expansion terms. Some works focus on building ontology-based user profiles from previous queries formulated by the user [51] and previously accessed documents [110]. Palleti et al. [207] build user profiles by leveraging collaborative information approaches and derive personalized expansion terms from those. Chirita et al. [62] exploit local user-related information to derive personalized terms and expand the queries before submitting them to Web search engines. This way, the authors preserve the users' privacy and anonymity while enhancing their Web search experience. Sarwar et al. [239] leverage users' status messages from social networks to identify personalized expansion terms for their queries. The authors first retrieve the most relevant status messages with BM25 and then select from those the expansion terms relying on their Inverse Document Frequency. Again, those works lack comparisons with other Personalized and non-Personalized Query Expansion methods.

More recently, some researchers have addressed Personalized Query Expansion using word embeddings [143, 300, 13]. Similarly to previous works leveraging non-contextual word embeddings for Query Expansion, the authors mostly employ word embeddings computed with Word2Vec and evaluate cosine similarity to assess the

semantic relatedness of the query terms and, in this case, the user vocabulary to select the expansion terms. Amer et al. [13] conduct an exploratory study on the use of Word2Vec's word embeddings for Personalized Query Expansion. Specifically, they compare the performance of a Query Expansion method that selects expansion terms based on their cosine similarity with the query term embeddings when employing locally trained embeddings, i.e., embeddings trained individually for each user only on the specific user-related texts, and globally trained embeddings, i.e., embeddings trained on the whole corpus. Similarly to previous Query Expansion methods based on word embeddings, the authors employ the word embeddings only during the expansion process and rely on a Language Model with Dirichlet smoothing [293] as their retrieval model. The authors report that the expansion methods did not improve the retrieval effectiveness of the original queries, and the globally trained embeddings outperformed the locally trained ones. Kuzi et al. [143] address the Personalized Query Expansion task in the context of email search. Similarly to the work by Amer et al. [13], the authors compared a Query Expansion method based on word embeddings with globally trained word embeddings and locally trained ones with the pseudo-relevance feedback expansion model RM1 [146]. The authors report findings similar to those of Amer et al. [13], but the personalized variant of their Query Expansion method based on word embeddings allows to improve the performance of the original queries. Zhou et al. [300] focus on enriching user profiles with information from external sources and propose two Personalized Query Expansion methods based on word embeddings and topic modeling. The model based on word embeddings ranks the user-related term embeddings by their cosine similarity with the sum of the query term embeddings and selects the top n for expansion. The authors report good results on folksonomy-based datasets for both the proposed models.

As we reported for the other works about Personalized Query Expansion, most

of the authors of methods based on word embeddings did not compare their approaches with other Personalized Query Expansion methods. We argue that the lack of standard Personalized Search test collections, as we will discuss in Chapter 6, and of publicly available implementations for *all* the presented Personalized Query Expansion methods poses severe issues in determining the state-of-the-art in this context. However, we highlight that many proposals, such as those based on social interactions, are of difficult application in domains different from the original ones.

In this work, we focus on the adoption of contextual word embeddings in Personalized Query Expansion. More specifically, to overcome the limitations of both semantic and lexical methods previously reported, we propose an approach that combines the usage of contextual word embeddings with a clustering-based procedure, which allows for identifying the topic of interest for each user and the terms that better represent the user's specific preferences. Moreover, we also pay particular attention to the efficiency issue affecting the Query Expansion methods based on contextual word embeddings reported in the previous non-personalized works [297, 200, 278] and propose an approximation procedure that allows our approach to achieve a sub-millisecond expansion time and to scale even in very data-rich scenarios.

For transparency and to encourage future work in this area, we conduct our experimental evaluation on publicly available datasets (see Section 5.4.1) and share the code of both the implementation of the novel Personalized Query Expansion approach we present in Section 5.2 and those of the baselines (Section 5.4.2), which we implemented from scratch, as well as all the information needed to reproduce our experimental evaluation (Section 5.5).

5.2 The Proposed Approach

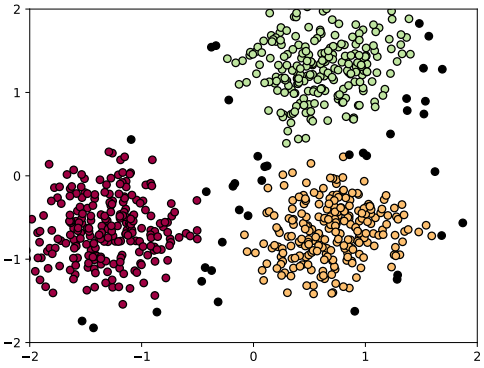
In this section, we present PQEWC[3] the method we propose to tackle the challenges introduced by the adoption of contextual word embeddings in Personalized Query Expansion, as described at the beginning of this chapter and in Section 5.1. First, in Section 5.2.1, we describe the approach we propose to identify the embeddings most representative of the user's interests that are also discriminative from a retrieval perspective (i.e., the embeddings that allow identifying documents relevant w.r.t. the query and the user preferences), and show how to avoid selecting multiple expansion terms with similar semantic meanings. Then, we introduce our expansion terms selection strategy and an effective mechanism to drastically reduce the number of computations required in the expansion terms selection stage (Section 5.2.2). Finally, in Section 5.2.3, we introduce ColBERT [136], a recent state-of-the-art retrieval model, which we enhance with our Personalized Query Expansion approach. We also discuss how we compute the relevance score of a document w.r.t. an expanded query.

In the following, we assume to have gathered a set of related textual content for each user, such as documents authored by the user, previously accessed web pages, user-generated content [140] (e.g., product reviews or tweets), previously issued queries, or other kinds of textual content related to the user.

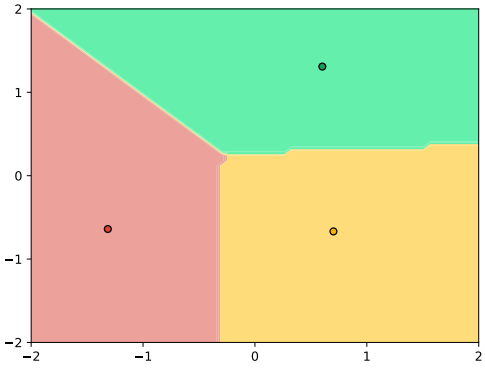
5.2.1 Word Embeddings Representative of the User Interests

In this section, we introduce the first step of our proposed approach, which aims at identifying the word embeddings that better represent the interests of a specific user that are also discriminative from a retrieval perspective (i.e., the embeddings that allow identifying documents relevant w.r.t. the query and the user preferences).

[3] **P**ersonalized **Q**uery **E**xpansion **W**ith **W**ontextual word embeddings, pronounced *"quick"*.



(a) Clusters identified by HDBSCAN.



(b) Space partitioning as a result of nearest centroid classifier.

Figure 5.1: Embedding space partitioning example.

The method we propose aims at pre-computing bags of candidate personalized expansion terms (in the form of word embeddings), among which we select those most related to the current search performed by the user (see Section 5.2.2). To identify the specific user's interests, we first partition the embedding space into regions where embeddings with similar semantic meanings lay. We do this by grouping the embeddings of all terms in the document collection using the hierarchical density-based clustering method HDBSCAN [187]. We rely on density-based clustering instead of the more commonly used centroid-based methods, such as k-means [180], because estimating a reasonably good number of centroids/clusters a priori can be difficult, and finding an optimal configuration can be computationally expensive. Once the clusters of the collection's term embeddings have been found, we use the nearest centroid classifier [265] (a.k.a. Rocchio classifier [183]) defined upon them to classify and group the user-related term embeddings. As exemplified in Figure 5.1, we partition the embedding space following the document collection's topic distribution in that same latent semantic space. Thus, relying on the nearest centroid classifier, we classify and group the user-related term embeddings according to the identified document collection's topics.

To identify the clusters that better capture the specific user interests and contain discriminative embeddings, we propose a function $\phi : \mathbb{R} \rightarrow \mathbb{R}$ inspired by the TF-IDF [233] formulation and defined as follows:

$$\phi(c_{ui}) = \frac{|c_{ui}|}{\sum_{j=1}^{k} |c_{uj}|} \cdot \log \frac{\sum_{j=1}^{k} |c_j|}{|c_i|} \tag{5.1}$$

where k is the number of the latent semantic space regions identified by the application of HDBSCAN and the nearest centroid classifier to the embeddings of the document collection, c_{uj} is the set corresponding to the cluster of the user u's term embeddings laying in the semantic space region j, and c_j is the set corresponding to the cluster of the collection's term embeddings laying in that same region. Similarly, c_{ui} and c_i are the sets corresponding to the cluster of the user u's term embeddings and the cluster of the collection's term embeddings that lay in the semantic space region i, respectively. The first part of the formula, inspired by the Term-Frequency [171], tells us how frequently a term embedding of the user u lays within a specific latent region identified by HDBSCAN and the nearest centroid classifier, which we interpret as the user interest in the topic of the document collection represented by that region in the latent space (in other words, it is the percentage of user's term embeddings that are in region c_i). The second part of the formula expresses the specificity of a topic (represented here by a term embedding cluster and its corresponding latent region), quantified as the inverse function of the number of term embeddings of the collection that lay in the related region c_i of the latent space. We use this quantity to weigh the user interest in a specific topic w.r.t. its discriminative power, similarly to the Inverse-Document-Frequency [129] in the TF-IDF formulation.

We use ϕ to rank and identify the top n clusters of user's term embeddings from which we select the expansion terms at query time, as it will be discussed in the next section. This process is conducted for each user separately. The whole procedure is shown in Figure 5.2.

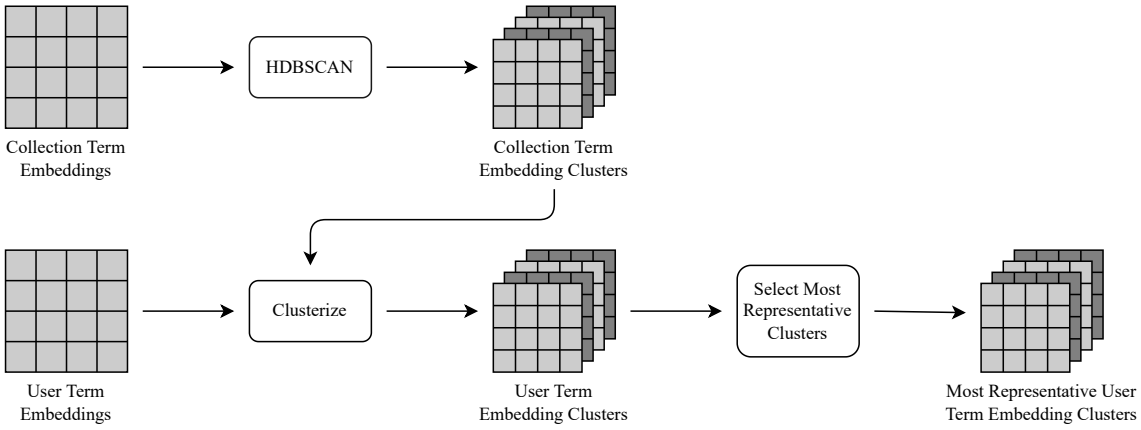


Figure 5.2: Offline step: user term embeddings clustering

We use a collection-level clustering approach instead of separately generating the term clusters from each user's vocabulary so that the resulting user-term clusters reflect the distribution of the topics of the collection in the embedding space. Moreover, as the clustering procedure is independent of the number of term embeddings related to each user, we do not incur to generate low-quality clusters when a user has few associated term embeddings. As we will show in Section 5.5.4, this choice allows us to achieve better results than partitioning the embedding space differently for each user by applying the clustering method directly on each user's term embeddings.

Using a clustering-based approach to pre-compute bags of candidate expansion terms for each user has two beneficial effects. Firstly, it lowers the chances of expanding the queries with terms that are redundant with each other, as we assign semantically close terms to the same clusters and select only one per cluster, as later discussed in Section 5.2.2. Secondly, by only considering the top n most representative clusters for each user, we reduce the computations required to choose the expansion terms at query time as the number of the candidate expansion terms is drastically lower than the total number of user-related terms, thus allowing to achieve far better efficiency and greater scalability than other recent Query Expansion methods (see Section 5.5.2).

5.2.2 Selection of Expansion Terms

In this section, we introduce the procedure we propose to select the expansion term embeddings from the user-related clusters, and the approximation we employ to drastically decrease its computation time.

Once we have built and identified the most representative clusters for a specific user following the method presented in the previous section, we select from each user-related cluster the term embedding with the highest *maximum* cosine similarity with the query term embeddings, and employ those selected for expansion. In other words, given a user-related cluster, for each term embedding in the cluster we compute its cosine similarity with each query term embedding, and take the *maximum* value. Then, we rank the term embeddings in the cluster according to their maximum similarity score and pick the one with the highest value. We repeat this operation for each of the most representative user-related cluster identified following the method presented in the previous section.

During the selection of expansion term embeddings, we are only interested in the maximum similarity value between each user term embedding and query term embeddings. Therefore, all the comparisons that do not produce a maximum similarity score are potentially unnecessary. As we cannot predict which comparison will result in a maximum similarity value without actually performing all of them, we propose to approximate our selection procedure to maximize efficiency. To do so, we first assign to each cluster the query term embedding most similar to the cluster centroid, computed as the average of the user term embeddings belonging to it. In doing so, we implicitly suppose the assigned query term embedding would produce the maximum similarity values for all the term embeddings belonging to that cluster. Consequently, for each user-related cluster, we *only* evaluate the similarity between its term embeddings and the embedding of the query term assigned to that cluster. Finally, for each cluster, we select the embedding that obtained the

highest similarity score to expand the query. This way, we considerably reduce the number of comparisons needed to select the Personalized Query Expansion terms, while leaving the effectiveness practically unaltered, as later shown in Section 5.5.4. For example, for a query representation composed of 32 embeddings, such as those computed by ColBERT [136] (introduced in the next section), and a user with 16 associated term clusters of 100 terms each, we reduce the number of comparisons from $32 \times 100 \times 16 = 51\,200$ to only $32 \times 16 + 16 \times 100 = 2\,112$, thus drastically decreasing the computation time.

5.2.3 Query Expansion with ColBERT

ColBERT is a neural retrieval model recently introduced by Khattab and Zaharia [136] that achieves state-of-the-art performances. Unlike other recent retrieval models based on Neural Networks and contextual word embeddings [158], ColBERT directly leverages query and document term embeddings to estimate the relevance scores of the documents in response to a query instead of, for example, comparing query and document embeddings obtained by a pooling operation over their term embeddings [224, 133, 283, 93, 167], such as taking their average. This characteristic makes ColBERT a good candidate model to study Query Expansion with contextual word embeddings, as we can add the additional expansion term embeddings to the query representation before computing the documents' relevance scores seamlessly. More formally, given a text t consisting of a sequence of tokens $[t_1, \ldots, t_n]$, ColBERT computes a matrix of size $n \times D$, where n is the number of tokens in the text and D is the dimension of each token representation. Under the hood, Colbert relies on BERT [76] to generate contextual vector representations of queries' and documents' terms. On top of BERT, a linear layer with no activation function controls the embeddings' dimension D, compressing the BERT representations to reduce memory consumption. In addition, Colbert leverages BERT's capabilities to augment queries

shorter than a predefined length, generating additional vectors that contribute to the estimation of the documents' relevance scores. The final query representations have a fixed size of 32 embeddings. ColBERT computes the relevance score of a document d in response to a query q as the sum of the maximum cosine similarities among the document's and the query's term embeddings:

$$s_{\boldsymbol{q},\boldsymbol{d}} = \sum_{\boldsymbol{q_i} \in \boldsymbol{q}} \max_{\boldsymbol{d_j} \in \boldsymbol{d}} \cos(\boldsymbol{q_i}, \boldsymbol{d_j}) \tag{5.2}$$

where $\boldsymbol{q}$ and $\boldsymbol{d}$ are the sets of the query term embeddings and the document term embeddings, respectively, and $\boldsymbol{q_i}$ and $\boldsymbol{d_j}$ are the embeddings of specific query and document terms. In the actual implementation, Colbert normalizes the term representations to a unit L2 norm and evaluates the similarity between queries' and documents' term embeddings using the dot product, which is equivalent to the cosine similarity in this particular case.

Although the query augmentation mechanism leveraged by ColBERT is effective in enhancing its retrieval effectiveness, Wang et al. [278] have shown that an additional query expansion stage can improve it even further, paving the way for future studies on query expansion with contextual word embeddings.

In this work, we enhance ColBERT through Personalized Query Expansion with contextual word embeddings and show that our proposed approach significantly improves its retrieval effectiveness with minimal overhead. Our proposed method outperforms Wang et al. [278] approach and recent Personalized Query Expansion methods based on word embeddings [143, 300] both in retrieval effectiveness (Section 5.5.1) and efficiency (Section 5.5.2).

For Query Expansion purposes, we extend Equation (5.2) to account for the expansion term embeddings by taking a convex combination of the scores of the original query term embeddings and those produced by the expansion term embeddings as

follows:

$$s_{\boldsymbol{q},\boldsymbol{e},\boldsymbol{d}} = (1-\gamma) \cdot \sum_{\boldsymbol{q_i} \in \boldsymbol{q}} \max_{\boldsymbol{d_j} \in \boldsymbol{d}} \cos(\boldsymbol{q_i}, \boldsymbol{d_j}) + \gamma \cdot \sum_{\boldsymbol{e_k} \in \boldsymbol{e}} \max_{\boldsymbol{d_j} \in \boldsymbol{d}} \cos(\boldsymbol{e_k}, \boldsymbol{d_j}) \tag{5.3}$$

where $\boldsymbol{q}$, $\boldsymbol{e}$ and $\boldsymbol{d}$ are the sets of the query term embeddings, the personalized expansion term embeddings, and the document term embeddings, respectively, $\boldsymbol{q_i}$, $\boldsymbol{e_k}$ and $\boldsymbol{d_j}$ are the embeddings of specific query, expansion, and document terms, and γ is a parameter that controls the influence of the original and the expansion term embeddings on the final score.

5.3 Personalized Query Expansion Framework

In this section, we describe the Personalized Query Expansion framework we employed for the comparative evaluation reported in the following sections. This framework allowed us to test different Personalized Query Expansion approaches isolating their contribution from the rest on the retrieval pipeline.

Figure 5.3 depicts the *Personalized Query Expansion framework* we relied on for comparing the Personalized Query Expansion methods presented in Section 5.4.2 and our newly proposed approach introduced in Section 5.2. The framework comprises one module that generates the vector representations of the terms of each document of the collection, those of the user-related documents' terms, and those of the query terms. Once computed the user-related term embeddings and the query term embeddings, the *Expansion Module* selects the term embeddings for expansion among those of the user and adds them to the query. Finally, a *scoring function* computes a personalized relevance score for each document of the document collection by comparing the representations of its terms with those of the expanded query terms. In our experiments, we rely on ColBERT [136] to generate the term representations, one of the Personalized Query Expansion baselines described in Section 5.4.2, or our novel

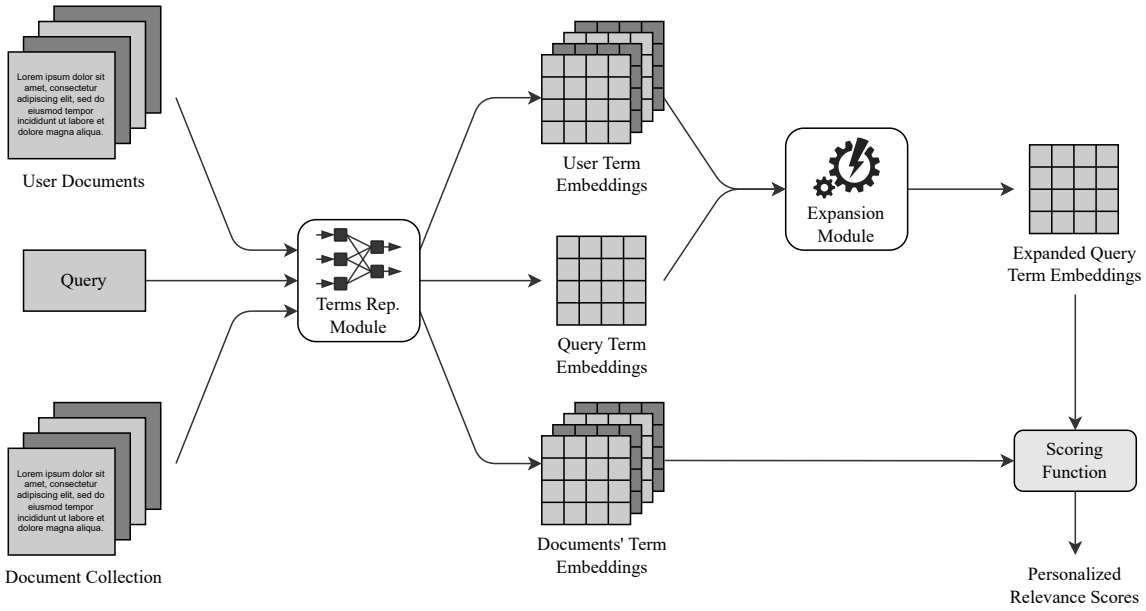


Figure 5.3: Personalized Query Expansion Framework.

approach introduced in Section 5.2 as the *Expansion Module*, and we employed Equation (5.3) to compute the personalized relevance scores for the documents. As the main contribution we present in this work is the novel Personalized Query Expansion method introduced in Section 5.2, the framework we implement for the evaluation is functional to comparatively evaluate the effectiveness of the proposed approach with previous methods at the state-of-the-art with ease, allowing us to switch between the Query Expansion models seamlessly.

For evaluation purposes, we apply our framework to re-rank the results retrieved for the initial queries by BM25 [226]. This choice was conditioned by the employed benchmark, which, by construction, is meant to be used for re-ranking BM25 results, as described in Chapter 6. The lack of publicly available large-scale datasets of high-quality is a known issue in Personalized Search Evaluation, as reported by Tabrizi et al. [257] and as we will discuss in greater detail in Chapter 6. Moreover, the approach used to derive evaluation datasets for Personalized Query Expansion from the data of social tagging platforms used in the past (see Section 5.1.2) has been criticized for the low-quality of the obtained benchmarks [257] and none of

these datasets is currently available. We refer the reader to Chapter 6 for a detailed description of the current state of the datasets for Personalized Search Evaluation. We also acknowledge, the re-ranking setting is often considered for evaluation purposes of novel retrieval models [202, 136, 158] based on contextual word embeddings and Transformer architectures [272], such as ColBERT, to leverage the efficiency of fast first-stage retriever while retaining much of the effectiveness on these new models.

In Section 5.5, we report both retrieval effectiveness statistics when only the re-ranker scores are employed and those obtained when combining them with the BM25 scores. In the latter case, we aggregated the two relevance scores via the weighted sum fusion algorithm provided by `ranx`, a Python library presented in Chapter 8. In this context, weighted-sum fusion works as a convex combination of the BM25 and re-ranker scores:

$$final\_score = (1 - \lambda) \cdot a + \lambda \cdot b \tag{5.4}$$

where a and b are the relevance scores computed by BM25 and the re-ranker, respectively, and λ is a parameter that controls the influence of the two on the final score.

5.4 Experimental Setup

The experiments reported in this section aim to answer the following four research questions:

RQ1 Can a Personalized Query Expansion approach based on contextual word embeddings enhance ColBERT's retrieval effectiveness?

RQ2 Is our approach more effective than previously proposed expansion methods?

RQ3 Is our approach more robust than previously proposed expansion methods?

RQ4 Is our approach more efficient than previously proposed expansion methods?

RQ5 Does our approach improve the expansion terms diversity compared to previous Personalized Query Expansion methods?

RQ6 Does the collection-level term clustering strategy proposed in Section 5.2.1 allow us to achieve better retrieval effectiveness than a user-level one?

RQ7 Is Equation (5.1) effective in identifying the user term clusters that better represent the user's interests, thus enhancing the retrieval effectiveness of our proposed approach?

RQ8 Does our approximated expansion terms selection perform on par of the original procedure proposed in Section 5.2.2 in terms of retrieval effectiveness?

RQ9 Does the approximation proposed to select the personalized expansion terms increase the efficiency w.r.t. the original procedure proposed in Section 5.2.2?

To answer the research questions **RQ1**, **RQ2**, **RQ3**, **RQ4**, and **RQ5** we conduct a comparative evaluation of the retrieval effectiveness, robustness, and efficiency of different personalized and non-personalized Query Expansion methods and analyze the similarity among the terms they select for expansion. Similarly, to answer the research questions **RQ6**, **RQ7**, **RQ8**, and **RQ9**, we compare our proposed Personalized Query Expansion approach described in Section 5.2 with several variants.

In the following sections, we present the dataset we employ for conducting our evaluations (Section 5.4.1), introduce the baselines we have selected (Section 5.4.2), outline the training setup (Section 5.4.3) and the hyper-parameters optimization procedure (Section 5.4.4), and introduce the evaluation metrics (Section 5.4.5) used to assess the models' effectiveness. We make all our code available for future works and reproducibility purposes[4].

[4]We will add a link to the repository upon acceptance

5.4.1 Datasets

In this section, we introduce the datasets employed to conduct our experimental evaluation. Due to the lack of standardized test collections for Personalized Search, we rely on the Personalized Results Re-Ranking benchmark we later propose in Chapter 6. This benchmark accounts for 18 million documents and 1.9 million queries divided into four datasets in the following domains: Computer Science, Physics, Psychology, and Political Science. As later described, we built the datasets by applying and refining the PERSON methodology [257], which consists in leveraging academic papers to derive user-query-document triplets. Specifically, the authors of PERSON proposed to consider, for each paper, the title as a query, the documents listed in its references section as relevant documents, and one of its authors as the user submitting the query. Since titles of academic papers are well-formed natural language, we applied stop-word removal and Krovetz stemming to obtain queries closer to real-world ones.

To compose the benchmark datasets, we started by collecting paper titles, abstracts, references, and other metadata for several millions of papers across multiple disciplines from the Microsoft Academic Knowledge Graph [86, 243]. Once composed and cleaned the document collections, we generated candidate queries following the approach we previously discussed. Then, to ensure the personalization potential for those queries, we discarded the queries whose users published less than 20 papers before the one used as the query. As discussed by Tabrizi et al. [257], the authors of PERSON, not all the documents listed in the references section of a paper are necessarily relevant — from an Information Retrieval perspective — to the topic expressed by a query constructed from the paper's title. Therefore, to reduce the presence of spurious relevant documents and malformed queries, we considered well-formed queries only those for which BM25 [226] places relevant documents in the top ranking positions. Likewise, for each of the remaining queries, we retained only the

Table 5.1: Statistics of the employed benchmark datasets.

	Computer Science	Physics	Political Science	Psychology
# documents	4 809 684	4 926 753	4 814 084	4 215 384
# train queries	552 798	728 171	162 597	544 882
# validation queries	5 583	7 355	1 642	5 503
# test queries	6 497	6 366	5 715	12 625
# relevants (avg $\pm$ sd)	3.25 ± 3.27	4.17 ± 4.15	3.88 ± 5.17	4.73 ± 4.4

relevant documents present in the top results retrieved by BM25. Finally, to closely resemble real-world scenarios — where all searches in the test set happen after those in the training set — the datasets were split chronologically into training and test sets. Training sets were then randomly split into training sets and validation sets, using a splitting ratio of $99 : 1$. Table 5.1 reports some statistics about the datasets.

5.4.2 Baselines

In this section, we introduce the baselines employed in our comparative evaluation. First, we compare our proposed Personalized Query Expansion-enhanced ColBERT to its original implementation, to assess whether our proposed approach is able to improve its retrieval effectiveness. Then we considered other query expansion approaches based on word embeddings, three of which take into account the user preferences, to verify if our proposed approach is improving over the state-of-the-art. In all our experiment, we consider BM25 [226], our first-stage retriever, for reference.

- **ColBERT:** ColBERT [136] is the recent BERT-based retrieval model introduced in Section 5.2.3. We consider ColBERT as a baseline to assess whether the compared query expansion methods are able to enhance its retrieval capabilities.

- **ColBERT-PRF:** ColBERT-PRF [278] is a recently introduced query expansion method based on ColBERT relying on pseudo-relevance feedback [228]. Specifically, given a query, it first ranks the documents using ColBERT, then clus-

ters the term embeddings of a certain number of feedback documents with k-Means [180] and selects the tokens corresponding to the cluster centroids with higher Inverse-Document-Frequency scores for expanding the original query. We consider ColBERT-PRF as a baseline to assess whether personalization is meaningful for query expansion in our context.

- **Baseline 1:** It is a Personalized Query Expansion method introduced by Kuzi et al. [143] that selects expansion terms based on the cosine similarity between their embeddings and the query term embeddings. Specifically, it first computes the cosine similarity between each user-related term embedding and each query term embedding. Then, it softmax-normalizes those similarities to get a probability distribution of the importance of user-related term embeddings w.r.t. each query term embedding. Finally, it sums the log probabilities of each user-related term embedding and selects the top-scored ones for expanding the original query.

- **Baseline 2:** It is a Personalized Query Expansion method introduced by Zhou et al. [300] that selects expansion terms based on the cosine similarity between their embeddings and the sum of the query term embeddings. That is, it simply computes the cosine similarities among the user-related-term embeddings and the sum of the query term embeddings and selects the top-scored ones for expanding the original query.

- **Baseline 3:** It is a variant of Baseline 2 we introduce by using the CLS token embedding in place of the sum of the query term embeddings. The CLS token is a special token appended by BERT [76] at the beginning of each text before computing its contextual word embeddings. It was originally introduced for sentence-level classification tasks but its embedding was also used in Information Retrieval as a single embedding representation of queries and

documents [224, 133, 283, 167]. At a theoretical level, the CLS token embedding is a sort of weighted sum of the other token embeddings and represents the semantic meaning of the input text as a whole.

We apply all the expansion methods before re-ranking the BM25 results with ColBERT. We do not consider the Personalized Query Expansion approach based on word embeddings proposed by Wu et al. [282] as it requires data unavailable in our setting. We do not consider other Personalized Query Expansion approaches based on word embeddings, such as that proposed by Amer et al. [13], as they are almost identical to the considered baselines or their authors did not report encouraging results.

5.4.3 Implementation Details

We relied on PyTorch [215], HuggingFace's Transformers [280], and PyTorch Lightning [84] for implementing and training ColBERT. We employed the cuML's GPU-based implementation of HDBSCAN [222] for clustering purposes. Finally, we implemented and optimized all the considered expansion methods with Numpy [114] to allow for a fair CPU-based efficiency comparison. To further ensure the reproducibility of the experiments, we relied on Hydra [286] to store the experiments' configurations.

We trained ColBERT on an NVidia® RTX A6000 GPU for 20 epochs following the instruction reported in its original paper [136]: learning rate set to 3×10^{-6}, batch size set to 32, number of embeddings per query set to 32, Adam optimizer [137], and pairwise softmax cross-entropy loss over a triplet composed of a query, a relevant document, and a non-relevant document. During training, we sampled hard negatives from the top results retrieved by BM25 and used the other documents in the batch as random negative samples. Since the machine used for experimentation only had 64GB of RAM, we set the maximum number of embeddings per document

to 128 and truncated the longer ones. We also set the embedding dimension to 16 to reduce the memory footprint further. As all the compared models but BM25, which is a classical retrieval model, share the embeddings generated by ColBERT, they are all affected in the same way, preserving the fairness of our comparison. To reduce the time needed to find the term embedding clusters with HDBSCAN, we heavily down-sampled the embeddings of each collection from ~500 to 10 million. For ColBERT and all its Query Expansion-enhanced variants, we aggregated the newly computed document relevance scores with the BM25 scores shared with the employed datasets (see Chapter 6), using the weighted sum fusion algorithm provided by `ranx` (see Chapter 8) after optimization on the validation set. Finally, we removed the word embeddings of stop-words from the possible embeddings to choose for query expansion.

5.4.4 Hyper-parameters Tuning

All the baseline expansion methods considered in our comparison and the proposed one have hyper-parameters controlling their behavior, which we optimize on the validation set. Specifically, they all have a parameter controlling the number of expansion terms to add to the queries, which in the case of PQEWC also corresponds to the number of user-related term embedding clusters to consider as the most representative of the user interests. ColBERT-PRF, Baseline 1, and PQEWC also have a parameter controlling the importance of the expansion terms when computing the document relevance scores, i.e., the expansion terms' weight. Finally, ColBERT-PRF has a parameter controlling the number of feedback documents to consider as pseudo-relevance feedback and a parameter controlling the number of clusters for grouping the feedback documents' term embeddings. We also report here the best values for the λ parameter of Equation (5.4) found on the validation set of each dataset for each model. We consider the following intervals and sets to generate the

hyper-parameters configurations during optimization:

- Number of expansion terms in the interval $[1, 32]$;
- Expansion terms weight in the interval $[0.1, 0.9]$ with a step of 0.1;
- Number of feedback documents in the interval $[1, 10]$;
- Number of clusters in the set $[8, 16, 24, 32, 40, 48, 56, 64]$;
- λ in the interval $[0.1, 0.9]$ with a step of 0.1.

We optimized Baseline 2 and Baseline 3 with a greed search on the validation set as they have only one hyper-parameter, the number of expansion terms. We fine-tuned the hyper-parameters of ColBERT-PRF, Baseline 1, and PQEWC with the Python optimization package Optuna [9], testing 100 hyper-parameters configurations for each of them. After the models' parameters optimization, we proceeded optimizing the λ parameter of Equation (5.4) using the greed search already implemented in `ranx`, the Python library we employed for score fusion. Table 5.2 reports the best hyper-parameters configuration for each method and dataset.

Table 5.2: Best hyper-parameters configurations. CS, PHY, PS, PSY stand for Computer Science, Physics, Political Science, and Psychology, respectively.

Model	# Feedback Docs				# Clusters				# Expansion Terms				Exp. Terms Weight				λ			
	CS	PHY	PS	PSY	CS	PHY	PS	PSY	CS	PHY	PS	PSY	CS	PHY	PS	PSY	CS	PHY	PS	PSY
ColBERT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0.8	0.8	0.8	0.9
ColBERT-PRF	6	1	1	1	16	24	24	16	8	22	3	13	0.1	0.1	0.1	0.1	0.8	0.8	0.8	0.8
Baseline 1	—	—	—	—	—	—	—	—	32	30	19	30	0.2	0.2	0.3	0.2	0.9	0.9	0.8	0.9
Baseline 2	—	—	—	—	—	—	—	—	16	14	32	22	—	—	—	—	0.9	0.9	0.9	0.9
Baseline 3	—	—	—	—	—	—	—	—	8	4	10	3	—	—	—	—	0.8	0.9	0.8	0.9
PQEWC	—	—	—	—	—	—	—	—	32	32	31	32	0.3	0.3	0.4	0.3	0.9	0.9	0.9	0.9

5.4.5 Evaluation Metrics

To evaluate the effectiveness of the considered models, we re-ranked the top 1000 results retrieved by BM25 and we employed 1) *Mean Average Precision* (MAP), 2) *Mean Reciprocal Rank* (MRR), 3) *Normalized Discounted Cumulative Gain* (NDCG), and

4) *Rank-biased Precision* (RBP). MRR and NDCG were computed on the top 10 documents retrieved by each model, whereas MAP was computed on the top 100. RBP' persistence was set to 0.95. Statistical significance testing was conducted using a Bonferroni corrected Two-sided Paired Student's t-Test [244] with $p < 0.005$. To evaluate the robustness of the query expansion methods, we employ the Robustness Index (RI) [66]. RI is defined as $\frac{N^+ - N^-}{|Q|}$, where N^+ and N^- are the amounts of queries whose results lists are improved or worsened by an expansion method in terms of Average Precision (at 100) w.r.t. ColBERT, and $|Q|$ is the total number of queries. The higher the RI, the more robust an expansion method is. Metrics computation and comparison were conducted using the Python evaluation library `ranx` [24].

5.5 Results and Discussion

In this section, we present the results of our comparative evaluation. First, we discuss the retrieval effectiveness, the efficiency, and the diversity of the terms chosen for expansion by the compared models in Section 5.5.1, 5.5.2, 5.5.3, respectively. Then, we ablate the design choices of our proposal in Section 5.5.4. Finally, we summarize our findings in Section 5.5.5.

5.5.1 Effectiveness

In this section, we discuss the performances of each of the compared models as well as the results of their fusion with the first-stage retriever, BM25, aiming to answer our research questions **RQ1**, **RQ2**, and **RQ3**.

First, we compare the results of the re-ranking models without the document scores interpolation of Equation (5.4). As shown in Table 5.3, all the ColBERT-based re-rankers were able to consistently outperform the first-stage retriever, BM25, by a considerable margin. However, there are some clear differences in the benefits

Table 5.3: Effectiveness of the compared models. † and ‡ denote significant improvements in a Bonferroni corrected Two-sided Paired Student's t-Test with $p < 0.005$ over ColBERT and over all the other considered models, respectively. Best results are highlighted in boldface. Best baselines' results are underlined.

Model	Computer Science					Physics				
	MAP@100	MRR@10	NDCG@10	RBP.95	RI	MAP@100	MRR@10	NDCG@10	RBP.95	RI
BM25	12.25	48.92	22.45	13.22	—	12.77	53.68	26.88	16.05	—
ColBERT	18.10	56.56	28.24	17.62	—	17.91	61.86	32.92	20.20	—
ColBERT-PRF	18.56†	56.82	28.68†	17.90†	**20**	18.77†	61.50	33.76†	20.75†	17
Baseline 1	18.46†	56.88	28.64†	17.86†	5	18.35†	62.50	33.40†	20.55†	9
Baseline 2	17.92	56.23	28.11	17.47	0	17.93	61.83	32.97	20.26	4
Baseline 3	18.18	56.86	28.43	17.66	6	18.05†	62.56	33.14†	20.30	10
PQEWC	**19.03**‡	**57.66**†	**29.23**‡	**18.23**‡	15	**19.17**‡	**63.81**‡	**34.46**‡	**21.12**‡	**22**
Model	Political Science					Psychology				
	MAP@100	MRR@10	NDCG@10	RBP.95	RI	MAP@100	MRR@10	NDCG@10	RBP.95	RI
BM25	13.27	50.23	24.07	14.24	—	12.58	51.19	23.93	13.84	—
ColBERT	16.06	53.51	26.40	16.11	—	21.39	62.78	33.39	20.17	—
ColBERT-PRF	16.42†	53.51	26.86†	16.33†	11	21.92†	62.53	33.83†	20.43†	10
Baseline 1	16.39†	53.61	26.66†	16.39†	7	21.60†	62.98	33.54	20.29†	3
Baseline 2	15.92	52.80	26.11	16.04	-1	21.22	63.26	33.40	19.99	-2
Baseline 3	15.49	52.85	25.84	15.74	-5	21.37	62.76	33.37	20.15	4
PQEWC	**17.24**‡	**55.10**‡	**27.71**‡	**16.99**‡	**14**	**22.30**‡	**64.21**‡	**34.47**‡	**20.75**‡	**12**

brought by the Query Expansion methods to ColBERT. ColBERT-PRF, our non-personalized Query Expansion baseline, achieved statistically significant improvements over vanilla ColBERT for all the considered datasets in MAP, NDCG, and RBP, but not MRR. In two cases, Physics and Psychology, ColBERT-PRF even decreased in MRR w.r.t. ColBERT. Baseline 1 achieved the best performances among the Personalized Query Expansion baselines, generally improving vanilla ColBERT's retrieval effectiveness for all the considered datasets and metrics. Compared to ColBERT-PRF, Baseline 1 achieved slightly worse results in all the considered evaluation metrics but MRR. Baseline 2 and Baseline 3 were not able to significantly improve over vanilla ColBERT and sometimes they even harmed its retrieval effectiveness, especially in the Political Science dataset. The main difference between those baselines and the other Query Expansion approaches is that they employ a pooling operation over the query term embeddings before selecting the expansion ones. In other words, they select the expansion term embeddings by their relatedness with the query represented as a whole (single embedding) instead of evaluating it by considering the query term em-

Table 5.4: Effectiveness of the compared models when the document scores they compute are interpolated with those computed by the first-stage retriever BM25 using Equation (5.4). † and ‡ denote significant improvements in a Bonferroni corrected Two-sided Paired Student's t-Test with $p < 0.005$ over BM25 + ColBERT and over all the other considered models, respectively. Best results are highlighted in boldface. Best baselines' results are underlined.

Model	Computer Science					Physics				
	MAP@100	MRR@10	NDCG@10	RBP.95	RI	MAP@100	MRR@10	NDCG@10	RBP.95	RI
BM25	12.25	48.92	22.45	13.22	—	12.77	53.68	26.88	16.05	—
BM25 + ColBERT	20.08	60.72	30.99	18.83	—	19.93	65.26	35.74	21.74	—
BM25 + ColBERT-PRF	20.21	60.56	30.95	18.87†	6	20.12†	64.96	35.77	21.80	1
BM25 + Baseline 1	20.17	60.44	31.01	18.92†	5	20.27†	65.87	36.15†	22.00†	8
BM25 + Baseline 2	19.87	60.55	30.86	18.70	1	20.06	65.56	36.00	21.88†	8
BM25 + Baseline 3	20.21	60.72	31.04	18.88	4	19.98	65.73	35.91	21.80	4
BM25 + PQEWC	**20.73**‡	**61.45**‡	**31.53**‡	**19.26**‡	**11**	**20.92**‡	**66.28**†	**36.85**‡	**22.45**‡	**21**
Model	Political Science					Psychology				
	MAP@100	MRR@10	NDCG@10	RBP.95	RI	MAP@100	MRR@10	NDCG@10	RBP.95	RI
BM25	13.27	50.23	24.07	14.24	—	12.58	51.19	23.93	13.84	—
BM25 + ColBERT	19.03	59.55	30.39	18.15	—	23.06	66.02	35.73	21.22	—
BM25 + ColBERT-PRF	19.13†	59.21	30.43	18.23†	4	23.02	65.55	35.48	21.12	-4
BM25 + Baseline 1	19.27†	59.48	30.58	18.36†	5	23.33†	66.10	35.96†	21.38†	6
BM25 + Baseline 2	18.95	59.32	30.37	18.10	0	23.11	66.53	35.98	21.22	0
BM25 + Baseline 3	18.82	58.97	30.22	18.07	0	23.07	66.02	35.74	21.22	5
BM25 + PQEWC	**19.81**‡	**60.57**‡	**31.24**‡	**18.72**‡	**11**	**23.96**‡	**67.14**‡	**36.71**‡	**21.76**‡	**14**

beddings separately. These results suggest that a single embedding representation for the query is less effective than considering the query term embeddings separately when selecting the expansion terms.

Our proposed Personalized Query Expansion method, PQEWC, achieved the best results on all the considered datasets and significantly improved over ColBERT and all the considered baselines in all the considered search scenarios. On average, it improved over ColBERT by 6%, 2%, 4%, and 4% in MAP, MRR, NDCG, and RBP, respectively, and over the best performing baselines by 3%, 2%, 2%, and 2% in MAP, MRR, NDCG, and RBP, respectively. Moreover, it scored a higher Robustness Index than all the other Personalized Query Expansion methods for all the considered datasets and higher than ColBERT-PRF on three datasets out of four. We also notice that our proposed approach is the only Query Expansion approach to achieve statistically significant increments over ColBERT w.r.t. MRR. These results confirm that a pre-processing phase aimed at identifying the most relevant user interests

and reducing the impact of redundant expansion terms is beneficial for Personalized Query Expansion with contextual word embeddings.

When the document scores produced by ColBERT and its Query Expansion-enhanced variants are aggregated with the scores produced by the first-stage retriever (BM25) following Equation (5.4), we record much less difference between vanilla ColBERT and its variants, with the sole exception of the one employing our proposed Personalized Query Expansion method PQEWC. As shown in Table 5.4, there is generally a little-to-no difference between vanilla ColBERT and the considered baselines regarding retrieval effectiveness. Conversely, PQEWC-enhanced ColBERT achieved the best performances for all the considered metrics and datasets. These results highlight that our proposed method captures personalized relevance signals that are complementary to both those of vanilla ColBERT and BM25. On average, it improved over BM25 + ColBERT by 4%, 2%, 3%, and 3% in MAP, MRR, NDCG, and RBP, respectively.

With and without interpolation, PQEWC outperformed ColBERT and all the other considered baselines and generally reached a higher Robustness Index. These results positively answer both our first, second, and third research questions, **RQ1**, **RQ2** and **RQ3**.

5.5.2 Efficiency

In this section, we compare the efficiency of the considered expansion methods in terms of the average time required to expand a query on the CPU (an AMD Ryzen™ 5950X, in our case), aiming to answer our research question **RQ4**. We suppose to have already loaded all the data needed for query expansion into memory. This way, we can focus on the expansion term selection latency. Note that all the compared models took less than one millisecond to re-rank the top-1000 BM25 results with ColBERT on our GPU. Therefore, we do not report the re-ranking times.

The second-to-fifth rows of Table 5.5 reports the expansion time required by the considered Query Expansion methods on the datasets employied in our evaluation. ColBERT-PRF was the least efficient of them, requiring 32 ms to expand a query on average, which makes its applicability to real-world scenarios questionable. Baseline 1, which performed the best among the personalized baseline methods, required 4 ms to expand a query on average, almost than ten times less than ColBERT-PRF while delivering similar retrieval performances. Baseline 2 and Baseline 3, which delivered very similar results in terms of effectiveness, both took less than one millisecond to expand a query on average. Finally, our proposed Personalized Query Expansion approach, PQEWC, achieved an expansion time inferior to one millisecond while delivering the best retrieval performances across the line.

Although all the considered Personalized Query Expansion methods are very efficient in our context, the number of operations needed by PQEWC is much lower than the other methods. PQEWC only compares the query term embeddings with small subsets of the user term embeddings, as discussed in Section 5.2.1, allowing our proposed method to be much more scalable than the others. On average, it compares the query term embeddings with less than 15% of the term embeddings associated with a user. In contrast, all the other methods consider all of them. Moreover, as introduced in Section 5.2.2, it only compares a single query term embedding with the most representative ones of each user.

To further prove our claims on PQWEC scalability, we conducted an empirical evaluation based on synthetically generated data. Since ColBERT-PRF latency was already high on our test sets, we did not consider it in this additional experiment. For each personalized expansion method, we investigated several different scenarios. Specifically, we considered three different embedding dimensions, denoted d, and four different amounts of average user-associated term embeddings, denoted T. As for embedding dimensions d, we considered 16, 128, and 768, which are the

dimension of the embedding we used in the experiment previously reported, the embedding dimension originally proposed for ColBERT, and the dimension of the uncompressed BERT embeddings, respectively. As for the average number of user term embeddings T, we considered 1 000, 10 000, 100 000, and 1 000 000.

As reported in Table 5.5, Baseline 1 rapidly saturates as more user-related terms becomes available, making it not suitable for real-world scenarios with high availability of user-related texts. On average, Baselines 2 and 3 require the same time to expand a query. Their applicability is mainly affected by the number of available user term embeddings rather than their dimension. As expected, PQEWC is the most scalable of the compared query expansion methods, and it is suited even for data-intensive scenarios. As shown in the table, it took just a fraction of the time required by the other models to expand the queries in each considered scenario. The embedding dimension has a noticeable impact on all the Personalized Query Expansions' execution times. However, the efficiency advantages of PQEWC make it the sole model able to scale to both high user data availability scenarios and highl dimensional vector spaces. To conclude, these results corroborate our claims regarding the scalability of PQEWC and positively answer our fourth research question **RQ4**.

5.5.3 Expansion Terms Diversity

In this section, we compare the diversity of the user term embeddings selected for expansion by the considered Personalized Query Expansion methods aiming at answering our fifth research question, **RQ5**. Moreover, this analysis allows us to verify our intuitions regarding the potential issues of employing previous Personalized Query Expansion methods based on word embeddings with contextual word embeddings, as discussed at the beginning of this chapter. In this regard, we introduce a novel metric to evaluate the percentage of semantically non-overlapping expansion terms per query. For each query, we first count the number of expansion term em-

Table 5.5: Query expansion methods execution time in milliseconds. d is the embedding dimension. T is the average number of user term embeddings.

	ColBERT-PRF	Baseline 1	Baseline 2	Baseline 3	PQEWC
Computer Science	39	5	< 1	< 1	< 1
Physics	34	5	< 1	< 1	< 1
Political Science	28	2	< 1	< 1	< 1
Psychology	27	4	< 1	< 1	< 1
$d = 16, T = 1\,000$	—	< 1	< 1	< 1	< 1
$d = 16, T = 10\,000$	—	5	< 1	< 1	< 1
$d = 16, T = 100\,000$	—	61	7	7	< 1
$d = 16, T = 1\,000\,000$	—	651	88	88	1
$d = 128, T = 1\,000$	—	< 1	< 1	< 1	< 1
$d = 128, T = 10\,000$	—	7	1	1	< 1
$d = 128, T = 100\,000$	—	72	12	12	< 1
$d = 128, T = 1\,000\,000$	—	740	141	140	9
$d = 768, T = 1\,000$	—	1	< 1	< 1	< 1
$d = 768, T = 10\,000$	—	8	2	2	< 1
$d = 768, T = 100\,000$	—	147	23	23	3
$d = 768, T = 1\,000\,000$	—	1494	249	249	26

beddings having a maximum cosine similarity score w.r.t. the other expansion term embeddings below a certain *semantic overlap threshold*. Then, we divide this counter by the number of terms selected for expansion and take the average across all queries. Our Expansion Terms Diversity (ETD) metric is as follows:

$$ETD_\tau = \frac{1}{n}\sum_{q=1}^{n}\frac{1}{k}\sum_{i=1}^{k} 1 \text{ if } \max_{\boldsymbol{e_{j\neq i}}\in\boldsymbol{E_q}} \cos(\boldsymbol{e_i}, \boldsymbol{e_j}) < \tau \text{ and }, 0 \text{ otherwise} \tag{5.5}$$

where n is the number of queries, k is the number of expansion terms for the query q, $\boldsymbol{E_q}$ is the set of expansion term embeddings selected for the query q, $\boldsymbol{e_i}$ is the i-th expansion term embedding, and τ is the semantic overlap threshold. We further propose to evaluate ETD with the following values for the semantic overlap threshold parameter τ: 0.99, 0.95, and 0.90. The rationale behind those values is as follows: low ETD.95/ETD.99 scores mean the query expansion method selects term embeddings with high/extremely-high semantic overlap, i.e., almost-duplicate term embeddings,

while we interpret EDT.90 scores near 0.5 as an indication that the expansion terms are topically focused but not semantically overlapping, thus they are diverse but semantically related. Note that a high EDT.90 score means the expansion terms are loosely correlated. We highlight that the expansion terms diversity score is not directly correlated nor proportional to the effectiveness gain brought by a Query Expansion method. However, it can help us understand why a method performs better or worse than another relative to a specific application domain.

Table 5.6 shows the ETD scores for the compared Personalized Query Expansion methods. The low ETD.99 and ETD.95 scores that Baseline 2 and Baseline 3 achieved in all datasets tell us those methods are prone to select expansion term embeddings that suffer from semantic overlap, a property not desirable in our search evaluation domains given the unsatisfactory results achieved by those methods and discussed in Section 5.5.1. Although Baseline 1 selected much more diverse expansion term embeddings than the other two baselines, it could not reach the expansion terms diversity of PQWEC because of the lack of a mechanism that prevents the selection of multiple semantically overlapping expansion terms. Generally, those results corroborate our intuitions regarding the potential issues of employing previous Personalized Query Expansion methods based on word embeddings with contextual word embeddings discussed at the beginning of this chapter. By employing a clustering-based procedure to group and find the term embeddings that better represent the user interests and preferences (Section 5.2.1) and selecting only one embedding per user-related term embedding cluster for query expansion purposes (Section 5.2.2), PQEWC achieved very high ETD.99 and ETD.95 scores. Therefore, term embeddings selected for query expansion by PQEWC benefit from great diversity. Moreover, the EDT.90 scores tell us PQEWC generally select topically focused but not semantically overlapping expansion term embeddings. These results positively answer our fifth research questions, **RQ5**.

Table 5.6: Expansion terms diversity of the compared Personalized Query Expansion methods. Higher is better for ETD.99 and ETD.95. Values near 0.5 for ETD.90 are better. Reported results are in percentages. Best results are highlighted in boldface.

Model	Computer Science			Physics			Political Science			Psychology		
	ETD.99	ETD.95	ETD.90	ETD.99	ETD.95	ETD.90	ETD.99	ETD.95	ETD.90	ETD.99	ETD.95	ETD.90
Baseline 1	97.10	75.09	**54.28**	96.00	75.91	57.65	96.70	79.65	63.29	93.99	72.64	55.31
Baseline 2	85.05	41.35	20.37	83.43	33.82	11.14	91.40	42.44	13.16	88.86	34.13	11.94
Baseline 3	89.52	40.44	17.99	89.02	27.91	7.71	93.04	39.02	15.12	86.14	33.90	14.38
PQEWC	**99.56**	**88.38**	64.39	**99.41**	**81.24**	**50.06**	**99.68**	**84.72**	**55.35**	**99.38**	**82.41**	**53.93**

5.5.4 Ablation Study

In this section, we conduct the ablation study of our proposal to assess whether our design choices are effective.

Effectiveness To evaluate whether our design choices are functional effectiveness-wise and the approximation method we proposed in Section 5.2.2 does not harm the retrieval effectiveness of our proposal, we compared it with the three following variants:

- **Local:** This variant relies on user-level term clustering instead of assigning user terms to collection-level clusters as proposed in Section 5.2.1. As there is no direct relations between local and global clusters (i.e., Equation (5.1) is not applicable), we consider as the most representative user term embedding clusters for each user those with the highest number of associated embeddings. This variant allows us to assess whether clustering the user terms following the collection topic distribution in the embedding space improves the retrieval performances over partitioning the embedding space user-wise (**RQ6**).

- **Top Clusters**: Instead of relying on the user-term clusters that better represent the user interests, this variant focuses on the user-term clusters that are most related to the query. First, it selects the top n user-term clusters most similar to the query. Then, following the procedure proposed in Section 5.2.2, it chooses from each top user-term cluster a term embedding to use for query expansion.

This variant allows us to assess whether considering the user-term clusters identified with Equation (5.1) as the best clusters to draw the expansion term embeddings from is an effective design choice (**RQ7**).

- **Non-Approximated:** This variant does not employ the approximated expansion terms selection strategy described in Section 5.2.2. It allows us to assess the impact on the retrieval effectiveness of the approximation we proposed to reduce the expansion term selection time (**RQ8**).

As for the main evaluation, all the expansion methods are applied before re-ranking the BM25 results with ColBERT and their hyper-parameters have been optimized on the validation set.

Table 5.7 reports the retrieval effectiveness of PQEWC, those of its considered variants, and — for reference — those of ColBERT. The results show that clustering the embeddings at the collection level and considering as best clusters to draw the expansion terms embeddings from following Equation (5.1) is more effective than the considered alternatives for all the considered datasets and evaluation metrics. Furthermore, PQEWC and Non-Approximated reached comparable effectiveness and robustness on all datasets but Political Science, where PQEWC increased over Non-Approximated. In general, our intuition regarding the computation of many unnecessary comparisons between the query term embeddings and the user-related term embeddings, discussed in Section 5.2.2, proved to be true. These results positively answer our research questions **RQ6**, **RQ7**, and **RQ8**.

Efficiency In this section, we compare the efficiency of PQEWC with that of its Non-Approximated variant. This comparison aims to evaluate in which contexts the approximation mechanism proposed in Section 5.2.2 is required and in which it is not. As shown in Table 5.8, for all the considered datasets PQEWC and its Non-Approximated variant achieved sub-millisecond execution time. This result means

Table 5.7: Overall effectiveness of the our proposal variants and that of ColBERT. $\star$, †, and ‡ denote significant improvements in a Bonferroni corrected Two-sided Paired Student's t-Test with $p < 0.005$ over ColBERT, Local, and Top Clusters, respectively.

Model	Computer Science					Physics				
	MAP@100	MRR@10	NDCG@10	RBP.95	RI	MAP@100	MRR@10	NDCG@10	RBP.95	RI
ColBERT	18.10	56.56	28.24	17.62	—	17.91	61.86	32.92	20.20	—
Local	18.52*	57.27	28.68*	17.88*	7	18.48*	62.83*	33.57*	20.60*	11
Top Clusters	18.80*†	58.47*†	29.33*†	17.98*	11	18.59*	63.36*	33.97*†	20.66*	13
Non-Approximated	19.05*†‡	57.58*	29.25*†	18.24*†‡	15	19.17*†‡	63.88*†	34.44*†‡	21.12*†‡	23
PQEWC	19.03*†‡	57.66*	29.23*†	18.23*†‡	15	19.17*†‡	63.81*†	34.46*†‡	21.12*†‡	22
Model	Political Science					Psychology				
	MAP@100	MRR@10	NDCG@10	RBP.95	RI	MAP@100	MRR@10	NDCG@10	RBP.95	RI
ColBERT	16.06	53.51	26.40	16.11	—	21.39	62.78	33.39	20.17	—
Local	16.14	53.24	26.38	16.28*	1	21.56*	62.94	33.56	20.27*	3
Top Clusters	16.78*†	54.92*†	27.41*†	16.56*†	9	21.89*†	64.02*†	34.17*†	20.43*†	12
Non-Approximated	16.87*†	53.91	27.18*†	16.73*†‡	7	22.33*†‡	64.24*†	34.51*†‡	20.76*†‡	12
PQEWC	17.24*†‡	55.10*†	27.71*†	16.99*†‡	14	22.30*†‡	64.21*†	34.47*†‡	20.75*†‡	12

the prominent factor in achieving top efficiency is limiting the search for expansion term embeddings to only the most representative user-term embedding clusters. However, the reader should consider that the similar expansion times of PQEWC and Non-Approximated are also due to the efficient vector operations offered by the Intel® Math Kernel Library [275], which we use as the back-end for Numpy [114]. In fact, the number of term comparisons performed without approximation is 32 times larger than when employing our approximation mechanism, as ColBERT's query representations are always composed of 32 embeddings. Table 5.8 also reports the average expansion time needed by the two approaches in the same simulated scenarios described in Section 5.5.2. As shown in the table, the Non-Approximated variant suffers in very data-rich scenarios. Nonetheless, it achieves far better efficiency than all the considered Personalized Query Expansion baselines, whose execution times are reported in Table 5.5. These results positively answer our ninth research question, **RQ9**. We conclude that the most important factor efficiency-wise is restricting the expansion term embeddings selection to a small portion of the user-related term embeddings (15% of all of them, in our case). However, as the effectiveness of PQEWC is not inferior to its Non-Approximated variant, the additional efficiency brought by the proposed approximation mechanism comes at no cost and still noticeably improves

Query Expansion latency.

Table 5.8: PQEWC variants execution time in milliseconds. d is the embedding dimension. T is the average number of user term embeddings.

	Non-Approximated	PQEWC
Computer Science	< 1	< 1
Physics	< 1	< 1
Political Science	< 1	< 1
Psychology	< 1	< 1
$d = 16, T = 1\,000$	< 1	< 1
$d = 16, T = 10\,000$	< 1	< 1
$d = 16, T = 100\,000$	2	< 1
$d = 16, T = 1\,000\,000$	13	1
$d = 128, T = 1\,000$	< 1	< 1
$d = 128, T = 10\,000$	< 1	< 1
$d = 128, T = 100\,000$	2	< 1
$d = 128, T = 1\,000\,000$	20	9
$d = 768, T = 1\,000$	< 1	< 1
$d = 768, T = 10\,000$	1	< 1
$d = 768, T = 100\,000$	5	3
$d = 768, T = 1\,000\,000$	49	26

5.5.5 Findings

In this section, we summarize our findings w.r.t. the research questions introduced in Section 5.4, which we report here for simplicity.

- **RQ1** *Can a Personalized Query Expansion approach based on contextual word embeddings enhance ColBERT's retrieval effectiveness?* Baseline 1 performed the best among the considered Personalized Query Expansion baselines, and it often significantly improved over ColBERT. Baseline 2 and Baseline 3 rarely improved over ColBERT, and sometimes they even decreased ColBERT's retrieval effectiveness. Finally, PQWEC consistently and significantly outperformed ColBERT in all the considered datasets and for all the considered evaluation metrics.

- **RQ2** *Is our proposed Personalized Query Expansion approach more effective than previously proposed expansion methods?* Among both the compared Query Expansion approaches, PQWEC performed the best, significantly improving over the other approaches w.r.t. MAP@100, MRR@10, NDCG@10, and RBP.95 in almost all cases.

- **RQ3** *Is our proposed Personalized Query Expansion approach more robust than previously proposed expansion methods?* Robustness-wise, PQEWC achieved much higher Robustness Index scores than all the other considered Personalized Query Expansion methods. It also outperformed ColBERT-PRF in all cases but one, Computer Science without BM25's scores fusion.

- **RQ4** *Is our proposed Personalized Query Expansion approach more efficient than previously proposed expansion methods?* PQEWC is more efficient than every other expansion method in our experimental evaluation, especially w.r.t. the best performing baselines ColBERT-PRF and Baseline 1, and achieved sub-millisecond expansion time even in very data-rich scenarios.

- **RQ5** *Does our approach improve the expansion terms diversity compared to previous Personalized Query Expansion methods?* Our approach selects topically focused but not semantically overlapping expansion term embeddings, which benefit from higher diversity compared to those picked by previous methods. This characteristic allowed our approach to reach significantly higher retrieval effectiveness than all the considered baselines in our comparative evaluation.

- **RQ6** *Does the collection-level term clustering strategy proposed in Section 5.2.1 allow us to achieve better retrieval effectiveness than a user-level one?* Our collection-level term clustering strategy allowed us to reach far better improvements over ColBERT than the user-level one, which often achieved mixed results.

- **RQ7** *Is Equation (5.1) effective in identifying the user term clusters that better represent the user's interests, thus enhancing the retrieval effectiveness of our proposed approach?* Using Equation (5.1) for identifying the user term clusters that better represent the user's interests was generally more effective than selecting user term clusters using other means.

- **RQ8** *Does our approximated expansion terms selection perform on par of the original procedure proposed in Section 5.2.2 in terms of retrieval effectiveness?* For all the considered datasets, the approximation we proposed to improve the efficiency of the original expansion terms selection procedure proposed in Section 5.2.2 did not affect the retrieval effectiveness improvements brought by our Personalized Query Expansion method.

- **RQ9** *Does the approximation proposed to select the personalized expansion terms increase the efficiency w.r.t. the original procedure proposed in Section 5.2.2?* Although the original expansion terms selection procedure proposed in Section 5.2.2 is already very efficient, our approximated variant still significantly improved the time needed to expand a given query.

5.6 Summary

In this chapter, we have addressed some issues arising from employing contextual word embeddings with current Personalized Query Expansion methods and proposed PQEWC, an approach designed to counteract those problems and take full advantage of contextual word embeddings. Specifically, our proposed method employs a clustering-based technique to group and identify the term embeddings most representative of the user interests and preferences and an approximation procedure of the personalized expansion terms selection to increase efficiency. Experimental evaluation shows the benefits of our proposed approach both in terms of efficiency

and effectiveness. Moreover, it highlights how the effectiveness and the efficiency of Personalized Query Expansion based on word embeddings can be greatly improved with effective procedures. Finally, the ablation study we conducted clearly illustrates the benefits of our design choices and the lack of drawbacks deriving from those.

CHAPTER 6

A MULTI-DOMAIN BENCHMARK FOR PERSONALIZED SEARCH EVALUATION

Personalization in Information Retrieval has been a hot topic in both academia and industry for the past two decades. However, there is still a lack of high-quality standard benchmark datasets for conducting offline comparative evaluations in this context. To mitigate this problem, in the past few years, approaches to derive synthetic datasets suited for evaluating Personalized Search models have been proposed. In this dissertation, we put forward a novel evaluation benchmark for Personalized Search with more than 18 million documents and 1.9 million queries across four domains.

In the following sections, we first discuss the current state of Personalized Search evaluation (Section 6.1). Then, in Section 6.2, we introduce a novel evaluation benchmark for Personalized Search and present a detailed description of the construction procedure we undertook, highlighting its characteristics and challenges. Finally, we run a comparative evaluation to establish baseline performances on the novel introduced datasets (Section 6.3 and 6.4), including pre-trained neural models, opening room for the evaluation of personalized approaches, as well as domain adaptation and transfer learning scenarios.

6.1 State of Personalized Search Evaluation

Personalization in Information Retrieval has been long studied by academia [250, 240, 190, 219] and industry [82, 260, 261, 220] from both modeling and evaluation perspectives. It is well known that evaluations based on the Cranfield [64] paradigm are not suited for Personalised Search, which produces search results tailored to specific users with their own perception of relevance. Unfortunately, text collections shared by initiatives on search evaluation do not usually provide the information needed for evaluating personalization, *i.e.*, information about specific users and their preferences.

In the past, evaluation campaigns targeting Personalized Search, such as the Interactive [206], HARD [10, 11, 12], Contextual Suggestion [71, 72, 73, 74, 117], and Session tracks [130, 131, 132, 57, 58], were promoted by the Text Retrieval Conference (TREC). However, all the above are limited in the amount of user-related information considered for the evaluation process; for this reason, the problem of defining a standard approach to the evaluation of Personalized Search is a hot research topic needing effective solutions. A first attempt to go towards the generation of a collection accounting for specific user search behaviors was made at the 2011 FIRE Conference [91]. Later, within CLEF 2017, the PIR-CLEF [213, 212, 211] benchmark was launched with the purpose of combining user-centered methods with the Cranfield evaluation paradigm, with the potential benefit of producing evaluation results that are easily reproducible.

Recently, some efforts have been devoted to the definition of large-scale task-related datasets (on which we focus in this paper). Specifically, three available real-world test collections that have also been employed to evaluate Personalized Search are the following: 1) the AOL Query Log [214] (Web Search), 2) the CIKM Cup

2016 dataset[1] (Product Search), and 3) the Yandex Query Log[2] (Web Search). Unfortunately, all these datasets come with specific issues, from content availability and privacy concern [22, 1], to anonymized texts, making personalized semantic retrieval approaches not usable. Due to the above issues, researchers have proposed several methodologies to define synthetic datasets for evaluating personalized retrieval systems, most notably the folksonomy-based datasets [40, 41, 42, 43, 284, 276, 299], the Amazon Product Search Datasets [6], and PERSON [257].

In folksonomy websites, users assign tags to online items, such as web pages. Datasets based on these data consider the tags as queries issued by the users to retrieve the tagged web pages. The main concern about this methodology is that tags can be very generic and far from expressing a real user's information need. Moreover, tag-based queries are usually composed of one or two terms only.

The Amazon Product Search Datasets [6] are designed to evaluate Personalized Search approaches in e-commerce scenarios. They are based on the Amazon Review 5-Core dataset [186], which contains millions of users, items, and user-generated reviews. Lacking user queries, the authors take advantage of the Amazon Product structural properties, using item categories to generate synthetic ones. Recently, some concerns were raised regarding the quality of the generated queries and the item data, as the first are far from realistic for many search scenarios, and the latter miss many descriptions and titles [25]. Moreover, the number of different queries present in these datasets is very low, a few hundred in most cases. Finally, many subsequent works [36, 34, 85, 209] generated alternative datasets following the same methodology instead of employing those previously shared by Ai et al. [6], making it challenging to conduct comparative evaluations.

Tabrizi et al. [257] have recently proposed PERSON, a new methodology for building synthetic datasets for Personalized Search based on citation networks. The main

[1] https://competitions.codalab.org/competitions/11161
[2] https://www.kaggle.com/c/yandex-personalized-web-search-challenge

idea behind this methodology is to leverage academic papers to derive user-query-document triplets. To this end, the authors consider the paper titles as queries, the papers cited in the references section as relevant documents w.r.t. the queries, and one of the papers' authors as the user issuing the generated query. The authors conduct an extensive evaluation of their approach, showing its benefits over other methodologies, such as those based on folksonomy data. For example, datasets built using PERSON methodology provide thousands of queries, making them suitable for training Deep Learning-based retrieval models. They also inherit social network-like information that can be exploited for personalization purposes and contain temporal information, which allows tracking the evolution of user preferences over time. Unfortunately, the dataset shared by Tabrizi et al. [257] requires users to build custom parsers to extract the data and lacks potentially valuable information for personalization, such as author affiliations and papers' publication dates and venues, despite being present in the original data. Finally, since it relies on ArnetMiner's Citation Network Dataset[3] [259], it covers one discipline only, preventing an evaluation across different domains.

In this work, we revisit and extend the PERSON methodology, overcoming the aforementioned limitations while keeping its benefits, paying particular attention to data processing, query generation, selection, and data splitting. As a result of our pipeline, we build and share a large-scale benchmark across four domains, with more than 18 million documents and 1.9 million queries, designed for evaluating Personalized Search approaches, including transfer learning scenarios. We perform experiments to establish baseline performances, including personalization approaches and recent pre-trained neural models. Data, code, and models for reproducing the experiments are publicly available online.[4]

[3] https://www.aminer.org/citation
[4] https://github.com/AmenRa/a-multi-domain-benchmark-for-personalized-search-evaluation

6.2 Benchmark Datasets

In this section, we detail the procedure we employed to build our proposed benchmark datasets, suited for the task of Personalized Results Re-Ranking (Section 2.3.2) in the Academic Search setting.

Data gathering and License We rely on the Microsoft Academic Knowledge Graph dump[5] [86, 243], a large-scale knowledge graph, to collect the metadata of 240M papers across several disciplines, which enables us to build multiple datasets. The dataset accounts for paper titles, abstracts, references, keywords, related fields of studies, publication dates and venues, and authors and their affiliations, and it comes under the CC BY 4.0 license[6], allowing us to transform and redistribute the original data.

Data processing After gathering the dataset, we proceeded by parsing its content as it comes in RDF format. Then, we cleaned the data to obtain a high-quality document collection. The cleaning process comprises several steps. First, we selected all and only the English papers by relying on the language attribute already present in the original data. Then, we divided the documents by discipline. Since some papers may belong to more than one discipline, we have duplicated them and placed a copy of each in the document collections of the disciplines they belong. Then, we discarded all the papers with no abstract, no date, no authors, and no associated keywords. Finally, we decreased the collections' sizes to reduce the hardware requirements needed to handle them, making them manageable with consumer-grade hardware. To do so by retaining only the most cited papers, we pruned the citation graph, where nodes represent the papers and edges represent the citation among them. In particular, we iteratively removed the nodes/papers with no incoming citations until

[5] https://zenodo.org/record/4617285
[6] https://creativecommons.org/licenses/by/4.0

we reached a size of fewer than five million documents. During each iteration, we removed all the papers with no incoming citations. As already noted by Clough et al. [65], an academic citation network *should be* a Direct Acyclic Graph as the edges (citations) must always point backward in time. However, this is not always the case due to wrong dates or updated versions of papers. Therefore, we discarded the citations not pointing backward to avoid a non-controllable pruning behavior and ease the future application of graph-based personalization approaches.

Query generation Once we had cleaned the document collections, we proceeded by generating the candidate queries for our benchmark datasets, following the approach proposed by Tabrizi et al. [257]. The approach is to generate user-query-document triplets as follows: for each academic paper, the title is considered the query, the papers cited in its references section as the relevant documents, and one of its authors (the first listed in the original data, in our case) is assumed to be the user submitting the query.

Although Tabrizi et al. [257] proposed several methods to generate synthetic queries from research papers, they only reported the evaluation of the title-based ones. We also considered using the keywords associated with each paper as queries, but we found this method to produce lower quality queries w.r.t. the title-based one. Moreover, we noticed that the keywords present in our dataset come from the abstracts rather than from those defined by the authors. However, we do not have additional information on how they were extracted.

Since the titles of academic papers are well-formed natural language, we processed them to obtain queries that resemble real-world ones. Specifically, we applied stop-word removal using the NLTK's [38] stop-word list and stemmed them using a non-destructive stemmer, *i.e.*, the *Krovetz stemmer* [139].

Query selection As we are interested in personalization, we need each query to be associated with at least a minimum amount of user-related data to personalize the search results. We set this minimum to 20 user documents. Consequently, we kept only the queries whose users have published at least 20 papers before the one used as the query. We explicitly considered only the papers published by the user/author before the one used as the query to preserve the temporal aspect and the user interests evolution. Previous works, such as those that relied on the Amazon data to derive Personalized Search datasets [6, 34, 36], ignored these aspects, which we believe are essential to resemble real-world search scenarios and correctly evaluate personalization models.

As discussed by Tabrizi et al. [257], not all the references are necessarily relevant (from an Information Retrieval perspective) to the topic expressed by a paper's title, which we use here as a query. The authors, however, claim that since mistakenly considering some irrelevant documents as relevant will be the same for all the compared models, their presence does not violate the fairness of the comparisons if evaluation measures are averaged over many queries. To reduce the presence of spurious relevant documents and malformed queries, we applied simple heuristics, similarly to Tabrizi et al. [257]. As we are proposing a series of datasets designed for evaluating Personalized Results Re-Ranking approaches, we consider well-formed only the queries for which BM25 [226], which we use as a first-stage retriever, retrieves relevant documents in the top results. Although this is a strong assumption, it should be noted that every re-ranking approach is limited by the first stage retriever's recall. We acknowledge that filtering the references by their positioning in the paper (the section in which they appear) *could be* a better alternative, but unfortunately, we lack such information. Specifically, we kept the queries for which BM25 retrieved at least one relevant document in the top-100 results for the training and validation queries and the queries for which it retrieved at least ten relevant documents in the top-1000

results for the test queries (see the next paragraph for a discussion of the data splits). Likewise, for each of the remaining queries, we retained only the relevant documents present in the top results retrieved by BM25. Before computing the BM25 results, to maintain most queries and relevant documents as possible, we fine-tuned its parameters *b* and *k1* on 5000 randomly selected non-test queries using the hyper-parameter optimization library `Optuna` [9] and evaluating the retrieval effectiveness of each configuration with `ranx` [23].

Data splits We split the obtained datasets into *training* and *test* sets chronologically, *i.e.* by using the queries generated from papers published starting from 2016 for Political Science, 2017 for Computer Science and Physics, and 2019 for Psychology as test queries. Then, we randomly split each *training set* to obtain a *training set* and a *validation set*, using a splitting ratio of $99 : 1$. We opted for a chronological *training*/*test* split instead of a random partitioning, so that the datasets are closer to real scenarios, where all searches in the *test sets* happen after those in the *training sets*. We choose the splitting dates to have thousands of queries in each test set. To prevent data leakage due to wrong data manipulation by future users, we removed from the collections all the documents published after the *train*/*test* splitting dates.

Final datasets In this paragraph, we describe the final datasets we share, their content, and some possible usage of the provided data. We opted for Computer Science, Physics, Political Science, and Psychology papers to compose the datasets we share. In the final datasets, each document comprises a title, an abstract, related field of studies, associated keywords, publication date and timestamp, proceeding/journal-related metadata, and the DOI of the original paper. For each query, we provide text, timestamp, relevant document ids, user id, user-related document ids, and the BM25 results (document ids and scores), which can be used both for re-ranking and as hard negatives to train Machine Learning models. We also make available the

data about the paper fields of study hierarchical structures, the author affiliations, the user-document authorship relations, and the paper references. We highlight that many of the provided data are not available in the dataset shared by Tabrizi et al. [257], which also has the limitation of considering a single domain. Table 6.1 reports some statistics of the final datasets. Note that the number of users refers to the number of authors of the documents in the document collections, while the average number of user documents refers to those associated with the queries.

The shared large-scale datasets are suited for training and evaluating content-based personalization models and collaborative-filtering approaches, as we provide a rich set of metadata for each document and all the data to derive the user-document interactions, which could also be leveraged by citation prediction approaches [14]. Moreover, the relations among the data, such as authorship relations and the paper references, can be represented by graph structures and leveraged by graph-based personalization approaches. Finally, our datasets allow the study and design of novel joint Personalized Search and Recommendation models [292].

Limitations Our proposed datasets share some limitations affecting previous synthetic datasets for personalization. Specifically, these datasets cannot be employed for evaluating session-based personalization approaches due to the lack of search sessions. We leave for future work the design of a procedure for generating search sessions for our datasets. Another limitation of those datasets is that all the synthetic user-query-document triplets are unique as they derive from non-repeatable user actions, such as writing a paper or a review or tagging an online item. Therefore, approaches relying on re-finding behavior, such as P-Click [78], cannot be used.

Table 6.1: Statistics of the proposed benchmark datasets.

Computer Science			
# documents	4 809 684	# users	5 260 279
# train queries	552 798	avg. query length	8.01 ± 2.74
# val queries	5 583	avg. user docs	61.94 ± 60.32
# test queries	6 497	avg. relevants	3.25 ± 3.27
Physics			
# documents	4 926 753	# users	5 835 016
# train queries	728 171	avg. query length	9.21± 3.43
# val queries	7 355	avg. user docs	60.98 ± 56.54
# test queries	6 366	avg. relevants	4.17 ± 4.15
Political Science			
# documents	4 814 084	# users	6 347 092
# train queries	162 597	avg. query length	8.97 ± 3.35
# val queries	1 642	avg. user docs	40.64 ± 29.32
# test queries	5 715	avg. relevants	3.88 ± 5.17
Psychology			
# documents	4 215 384	# users	4 825 578
# train queries	544 882	avg. query length	9.36 ± 3.27
# val queries	5 503	avg. user docs	61.66 ± 62.72
# test queries	12 625	avg. relevants	4.73 ± 4.4

6.3 Experimental Setup

In this section, we present the baselines we evaluated on the novel test collections we have constructed, and we introduce the evaluation metrics we adopted for their evaluation.

Our first baseline is BM25, whose results we re-rank with the other baselines. For re-ranking the BM25 results, we consider a document popularity-based model (Pop), a bi-encoder-based retrieval model (BiEnc) similar to several recent neural retrieval approaches [224, 93, 133, 149], and two embedding-based personalized retrieval models. The first (Mean) defines the user models as the average of their associated document representations. The second (QA) adopts a query-aware user

modeling technique [95, 7, 126] using the Attention mechanism [19] to weigh the contribution of the user-related document representations to generate the user model. Both approaches compare the user models with the documents retrieved by BM25 to compute personalized relevance scores. In each case, we aggregate the original BM25 scores with those computed by the re-rankers using the weighted sum fusion algorithm implemented in `ranx.fuse` [26], which we optimize on the validation set. Each model computes the document representations using MiniLM-L6-v2[7] with a mean pooling head on the document titles. To assess domain adaptation and transfer learning opportunities, we train BiEnc on all four domains (All-BiEnc) and evaluate its effectiveness on each dataset separately. Furthermore, we combine the relevance scores of BM25, BiEnc, and the personalization models to assess whether they can all be combined to improve effectiveness further.

As for the evaluation metrics, we consider Mean Average Precision (MAP) at 100, Mean Reciprocal Rank (MRR) at 10, and Normalized Discounted Cumulative Gain (NDCG) at 10.

6.4 Results and Discussion

Table 6.2 summarizes the results achieved by the various models on the provided test sets. All the considered baselines substantially improve over BM25, indicating that the benchmark is suitable for evaluating (Personalized) Re-Ranking models. By comparing the results of the personalized models (Mean and QA), we registered a clear advantage of the query-aware personalization model (QA). However, we notice that the differences between the two models are largely reduced if we also consider the scores of the neural re-ranker (BiEnc) during re-ranking. In this latter case, BiEnc + QA was able to statistically improve over BiEnc + Mean only on two datasets out

[7] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

Table 6.2: Effectiveness of BM25 and those of the re-ranking models. † denote significant improvements in a Bonferroni corrected Two-sided Paired Student's t-Test with $p < 0.001$ over all the baselines. Best results are highlighted in boldface.

Model	Computer Science			Physics		
	MAP@100	**MRR@10**	**NDCG@10**	**MAP@100**	**MRR@10**	**NDCG@10**
BM25	12.25	48.93	22.45	12.77	53.68	26.88
BM25 + Pop	16.64	58.63	28.97	15.45	59.61	30.88
BM25 + BiEnc	18.21	58.02	28.90	16.98	60.99	32.13
BM25 + All-BiEnc	17.82	57.79	28.59	16.81	60.95	32.02
BM25 + Mean	16.37	54.57	26.69	16.44	59.05	31.18
BM25 + QA	17.88	57.21	28.49	17.47	61.80	32.72
BM25 + BiEnc + Mean	19.92	60.64	30.80	18.91	63.88	34.51
BM25 + BiEnc + QA	**20.11**†	**61.17**	**31.15**†	**18.98**	**64.78**	**34.81**
Model	**Political Science**			**Psychology**		
	MAP@100	**MRR@10**	**NDCG@10**	**MAP@100**	**MRR@10**	**NDCG@10**
BM25	13.27	50.23	24.07	12.58	51.19	23.93
BM25 + Pop	16.04	57.39	28.46	15.13	56.46	27.29
BM25 + BiEnc	18.15	57.64	29.25	20.72	63.41	33.11
BM25 + All-BiEnc	18.51	58.82	29.94	20.23	62.91	32.64
BM25 + Mean	16.61	55.05	27.58	16.73	57.05	28.42
BM25 + QA	17.69	57.58	28.94	18.90	60.92	31.12
BM25 + BiEnc + Mean	19.26	59.76	30.63	21.97	65.21	34.71
BM25 + BiEnc + QA	**19.85**†	**61.20**†	**31.41**†	**21.99**	**65.62**	**34.85**

of four. BiEnc generally delivered slightly better results than the personalization models. However, those approaches are not mutually exclusive, as we can observe from the effectiveness reached by their combinations, which achieved the best performances across the board. Regarding domain adaptability, the All-BiEnc model, *i.e.*, the BiEnc model trained across domains, performed slightly worse than BiEnc on every dataset but Political Science. Those results were not entirely unexpected, given how diverse the domains are. Although training on such diverse data usually allows obtaining more robust re-ranking models, models trained on domain-specific data generally outperform the former models in their specific domain. However, we notice that All-BiEnc can obtain comparable performance to the BiEnc trained on specific domains, which requires four separate training instances. Moreover, All-BiEnc outperformed BiEnc on Political Science, the dataset with the lower number of associated training queries, which benefits the most from the additional training

data from the other domains.

6.5 Summary

In this chapter, we presented and shared a novel benchmark for Personalized Search evaluation, composed of four large-scale datasets in multiple domains and accounting for millions of documents and hundreds of thousands of queries. The size of the proposed benchmark, along with its rich structured information, opens up new research opportunities, from adopting graph-based approaches for personalization to designing and training novel Deep Learning-based personalization approaches. Furthermore, we provided performance baselines including recent pre-trained neural models, showing that there is big room for improvement for personalized approaches. Finally, by providing multiple datasets across different domains, we pave the way for domain adaptation and transfer learning in the personalization context.

CHAPTER 7

SEMANTIC QUERY LABELING WITH SYNTHETICALLY GENERATED DATA

Nowadays, many different kinds of vertical online platforms, such as media streaming services (*e.g.*, Netflix[1], Spotify[2]), e-commerce websites (*e.g.*, Amazon[3]), job-seeking platforms (*e.g.*, LinkedIn[4]), and several others, provide access to domain-specific information through a search engine. This information is usually organized in structured documents. In this context, like in Web Search, users typically convey their information needs by formulating short keyword-based queries. However, in Vertical Search[5], users' queries often reference specific structured information contained in the documents. Nevertheless, Vertical Search is often managed as a traditional retrieval task, treating documents as unstructured texts and taking no advantage of the latent structure carried by the queries; exploiting this latent structure would allow leveraging the document structure during retrieval.

Semantic Query Labeling [184, 21], the task of locating the constituent parts of a query and assigning domain-specific predefined semantic labels to each of them, allows unfolding the relations between the query terms and the documents' structure, thus enabling the search engine to leverage the latter during retrieval while leaving unaltered the keyword-based query formulation. For example, the query *"alien ridley*

[1] http://netflix.com
[2] http://spotify.com
[3] http://amazon.com
[4] http://linkedin.com
[5] Search conducted on domain-specific corpora of structured documents.

scott 1979" comprises the title of a movie, *Alien*, the name of a movie director, *Ridley Scott*, and a date, *1979*. In this case, the query could be segmented into *alien*, *ridley scott*, and *1979* and the query segments could be tagged with the labels *Title*, *Director*, and *Year*, respectively.

Semantic Query Labelling is a challenging task that can add context and structure to keyword-based queries, usually composed of a few terms that may be ambiguous. The main challenges of this task are related to the vocabulary overlap among different semantic classes, which could require the use of contextual information and disambiguation techniques, and vocabulary mismatch [90] between the vocabulary employed by the users to express their information need and the vocabulary used to describe the corresponding answers in the document collection. Unfortunately, the production of an appropriate dataset to evaluate the effectiveness of automatic query tagging approaches is costly, and actually, there is a lack of publicly available datasets for this task.

Despite semantic query labeling could play an important role in Vertical Search, very little work has been done in this regard. The majority of past efforts in this context come from private companies, such as Microsoft ([184, 152, 151, 238, 160]) and Yahoo! ([138]). Due to privacy issues, companies cannot release the datasets used in their studies. As well known, this makes it hard to reproduce their approaches and comparatively evaluate them. Moreover, the lack of public datasets makes it difficult for academic researchers to propose novel Semantic Query Labeling models, and evaluate their effectiveness.

The aim of the work presented in this chapter is three-fold: 1) we propose a method to alleviate the need for manually labeled training data for Semantic Query Labeling by leveraging a rule-based query generator that only requires a structured document collection for producing synthetic queries; 2) we conduct an in-depth analysis of the usage of synthetic queries both as a means for training with no further fine-tuning

and pre-training with subsequent fine-tuning on real-world data a semantic query tagger; 3) we release a large-scale dataset of manually annotated queries in the movie domain.

In particular, the query generation method we define leverages the structured information of a domain-specific corpus and simple query variation techniques to produce semantically annotated training queries. By automatically generating training queries, we alleviate the need for large manually labeled query datasets and their cost. Moreover, we show that synthetic and real-world queries can play a complementary role in effectively training a semantic query tagger. We propose a combination of BERT's contextual word embeddings, gazetteers-based features, and Conditional Random Fields as our semantic query tagging model.

To evaluate the proposed approach, we need to compare the performances of our semantic query tagger when trained with synthetic or real-world data or a combination of those. To this aim, we define a novel dataset of manually annotated queries in the movie domain[6]. Our dataset is composed of 6749 unique keyword-based queries similar to those formulated by users on movie streaming platforms.

Experimental evaluation shows that training a semantic query tagger with synthetic queries allows for results comparable to those obtained with real-world ones. Furthermore, pre-training our semantic query tagging model with synthetic queries and fine-tuning it with real-world ones allows for the best results across the line.

In Section 7.1 we discuss the related works. Then, in Section 7.2 we describe our query generation procedure and the semantic query tagger we propose. In Section 7.3, we introduce a novel benchmark dataset for Semantic Query Labeling and discuss the process we undertook to build it. Then, in Section 7.4 we describe the experimental setup. Finally, we discuss the evaluation results in Section 7.5.

[6] https://github.com/AmenRa/semantic-query-tagging-dataset

7.1 Related Work

Although the semantic tagging of query terms could play a key role in advancing Vertical Search, there have been few published works in this regard, mainly due to the lack of publicly available datasets of annotated queries for this task. Because of that, the majority of past research efforts in this context come from private companies [184, 152, 151, 238, 160, 138]).

Manshadi and Li [184] proposed the use of a probabilistic grammar to produce all the possible interpretations of the queries. The interpretations are then re-ranked using a Support Vector Machine-based module [69] and several contextual features. Li et al. [152] trained a Conditional Random Fields-based [144] query tagger in a semi-supervised fashion, leveraging a database comprising 20M products, 750k (*query, product*) pairs, and more than 20k manually labeled queries. Li [151] focused on *noun phrase queries*, *i.e.*, queries composed of *intent heads* and *intent modifiers*, and proposed to use semi-Markov Conditional Random Fields [237] with semantic and syntactic features. Sarkas et al. [238] proposed an unsupervised method that, given a collection of structured tables, produces all possible annotations for a given query and assess the likelihood of each of them by using a generative model. The authors trained their generative model by leveraging a combination of structured data and query logs. Liu et al. [160] studied the problem of expanding the lexicons collected from structured corpora, which often have limited coverage, are ambiguous, and lack terms importance, by exploiting query logs for extracting query patterns and new lexicon elements. Recently, Kozareva et al. [138] studied semantic tagging of product search queries. The authors made use of the Word Embedding-model Word2Vec [192] trained on 2.5M shopping queries for representing query terms and adopted a Long Short-Term Memory network [119] with a Conditional Random Field on top as their query tagger.

Differently from those works, we propose an approach to train a semantic query

tagger with synthetic queries generated by leveraging the structured information of a domain-specific corpus without the need for manually annotated data or a query log. Moreover, we investigate the potential of pre-training a semantic query tagger with synthetic data and evaluate the model performance over time to assess whether this pre-training is always beneficial. Finally, we share a real-world dataset of manually annotated queries for evaluating semantic query tagging models.

7.2 Proposed approach

In this section, we define a method for generating semantically annotated training queries from a structured corpus and describe the proposed semantic query tagger.

7.2.1 Synthetic Query Generation

Building a query generator that produces queries similar to those issued by real users can be very time-consuming, and it could require a large amount of data [174]. These scenarios are incompatible with the underlying motivation of our experiment — the unavailability or scarcity of real-world training data. Therefore, we propose to rely on the concatenation of values extracted from structured corpora as the base query generation method. This choice is also motivated by the empirical observation that, in Vertical Search, users usually issue queries as *bag-of-features* of what they are searching. For example, e-commerce users usually search for *"a producer's name, a brand or a set of terms which describe the category of the product"* [230]. This information is often present in the queried corpus.

Queries generated by simple concatenation of values from structured corpora have, however, some limitations. First of all, real-world queries often contain terms used for connecting query segments, such as *from* in the query *"horrors from 1990s"*, that cannot be directly generated using values contained in a structured corpus.

Moreover, there could be a vocabulary mismatch between the user queries and the structured documents. For example, a user will likely search for *"italian movies"* and not for *"italy movie"* (the value used in the *Country* field of movie corpora for Italian movies).

To solve these issues, we apply some steps finalized at defining query variations. For example, converting country names to their demonyms (*e.g.*, *Italy* $\rightarrow$ *Italian*) is a straightforward operation that can be automated by using publicly available mappings[7]. Other means to obtain variations of the query terms can be derived, for example, from query logs [160], if available, or by using synonyms dictionaries [193]. Note that we explicitly avoid to use external domain-specific information in our experiments, as it is not always available. Regarding the generation of segment connectors, we employ very simple rules based on the semantic classes of the query segments. Also, to the purpose of generating query variations, the query generation process randomly removes words and alter the word order.

For our experiments in the movie domain, we only used: 1) demonyms conversion and date alterations (*e.g.*, *1979* $\rightarrow$ *1970s*), which can be easily automated, 2) some alternative values for movie genres, such as *scary* for *horror*, 3) some terms for connecting query segments, and 4) some values for indicating ordering preferences.

The generation process has been implemented as follows:

1. Randomly compose a query pattern of 1-to-n predefined semantic components — *e.g.*, <**GENRE O YEAR**>
   (**O** indicates segment connectors);

2. Randomly select a movie m from the corpus — *e.g.*, *Alien*;

3. Select the values corresponding to the query pattern semantic classes from m metadata and predefined values — *e.g.*, **GENRE** $\rightarrow$ *horror*, **O** $\rightarrow$ *from*, **YEAR** $\rightarrow$

[7] https://github.com/mledoze/countries

1979;

4. Randomly apply alterations — *e.g.*, *1979* $\rightarrow$ *1970s*;

5. Compose the annotated query with IOB2 labeling format [223, 235] — *e.g.*, *horror* **B-GENRE** *from* **O** *1970s* **B-YEAR**;

6. Accept the query if new and unobserved, discard otherwise. During the experiments described later in this chapter, we discarded all the queries present in the dev and test sets.

7.2.2 Labeling Model

We propose to represent query terms through contextual word embeddings produced using BERT [76] (HuggingFace's [280] *bert-base-uncased*), as they allowed achieving state-of-the-art results in many sequence labeling tasks, and they have not been employed before for Semantic Query Tagging. We also use some custom features based on corpus-derived gazetteers that allow the labeling model to leverage the corpus information. Specifically, for each query term *t*, for each semantic class *c*, we employed the following gazetteers-based features, basing on the structured corpus lexicons: 1) *t in c*; 2) *t and* t_{-1} *in c*; 3) *t and* t_{+1} *in c*; 4) *t*, t_{-1}, *and* t_{-2} *in c*; 5) *t*, t_{+1}, *and* t_{+2} *in c*; 6) *t*, t_{-1}, *and* t_{+1} *in c*; where t_{-i} is the *i*-th term preceding *t* and t_{+j} is the *j*-th term following *t*. The values of these features are based on discriminative probabilities computed w.r.t. corpus lexicons — *i.e.*, $P(c|t) = \frac{occurrences_{t,c}}{occurrences_t}$, where $occurrences_{t,c}$ is the number of occurrences of *t* in *c* w.r.t. our corpus and $occurrences_t$ is the number of occurrences of *t* in our corpus as a whole. As semantic query tagger we employ a model based on Conditional Random Fields [144], a standard sequence labeling model that can incorporate arbitrary features as input. We do not deepen further in the model description due to space constraints, and we leave the ablation study of its components for future work.

7.3 Benchmark Dataset

This section describes the dataset we have defined and the process we followed for manually annotating each query term. Our dataset comprises thousands of manually-labeled real-world queries in the movie domain for training and evaluating novel methods for Semantic Query Labeling.

The choice of working in the movie domain is motivated by the fact that movie streaming platforms are popular nowadays, but they still provide a sub-optimal search experience to their users. Moreover, structured search is fundamental in this context: as we assessed during our work described here, users tend to compose their queries referring to specific movie-related information, such as the name of an actor or a director, a movie genre, a topic, and others, which are usually available as metadata. By conducting a qualitative evaluation of the top 10 results returned by the search engine of one of the most popular movie streaming services, we assessed that it is not able to correctly retrieve movies even for simple queries. For example, *"horror 2015"* retrieved only one horror movie from 2015, many other results were neither horror movies nor movies from 2015. *"2015 horror"* did not retrieved any result at all. Neither *"leone eastwood"* nor *"sergio leone clint eastwood"* retrieved any result despite the presence on the platform of all the movies directed by *Sergio Leone* and starring *Clint Eastwood* at the time of the experiment.

7.3.1 Query Gathering

The first step in building a dataset suitable for studying Semantic Query Labeling is the query gathering. To collect the queries that are part of our dataset, we relied on a publicly available large-scale query log of the AOL Web search engine[8], which was shared by Pass et al. [214]. This query set comprises queries issued by real users

[8] https://www.aol.com

between March 1, 2006, and May 31, 2006. First of all, we defined a list of seed-terms for identifying movie-related queries: *movie*, *movies*, *film*, and *films*. Leveraging these terms, we extracted $39\,635$ unique queries. Then, we *manually* filtered out all the queries that did not fall into our category of interest: keyword-based queries that resemble those used by users for searching movies on movie streaming platforms. We ended up with 9752 candidate queries, as the large majority of the initially extracted queries were related to theaters' movie listings — note that AOL offers a general-purpose Web search engine. After removing the seed-terms used for gathering the queries, *manually* correcting misspellings, normalizing strings, removing the stop-words, and applying lemmatization, our dataset accounts for $6\,749$ unique queries. Our stop-word list was composed only of *a*, *an* and *the*, as we found all the other query terms useful for semantic labeling purposes. We applied lemmatization only to nouns, by leveraging the SpaCy[9] POS-based lemmatizer.

7.3.2 Semantic Labels Assessment

The second step in the building process of our dataset was to define 1) the semantic label set to use for the creation of the ground truth and 2) the procedure to follow to assign the semantic labels to the query terms, ensuring the quality of the proposed dataset.

Semantic Labels

After an initial analysis of the harvested queries, we defined the following semantic classes to assign to each query term: *Title*, *Country*, *Year*, *Genre*, *Director*, *Actor*, *Production company*, *Tag* (mainly topics and plot features), *Sort* (*e.g.*, *new*, *best*, *popular*, etc.). Following previews works in Natural Language Processing and Sequence Labeling [266], we used the IOB2 labeling format [223, 235] for *manually* assigning

[9] https://spacy.io

both semantic labels and segmentation delimiters. For example, the query *"alien by ridley scott 1979"* is labeled as follows: *"alien* ***B-TITLE*** *by* ***O*** *ridley* ***B-DIRECTOR*** *scott* ***I-DIRECTOR*** *1979* ***B-YEAR****"*, where the prefix **B-** indicates the beginning of a segment, the prefix **I-** indicates that the term is *inside* a segment, and the tag **O** is used to label terms with no semantic values, such as the preposition *by* in our example.

Creation of the Ground Truth

One of the main reasons for choosing to work in the movie domain is the public availability of movie-related information. We relied on this information to ensure the quality of the ground truth labels we *manually* assigned to the query terms. In this regard, we consulted many websites that contain movie-related information while labeling the queries, such as Wikipedia[10], IMDb[11], and many others. Furthermore, particular attention was paid in discerning actors from directors, as sometimes a single person is both an actor and a director, such as *Ron Howard*. In these cases, we followed a simple rule: if the query contains elements pointing towards a specific interpretation of the query, we labeled the query accordingly (*e.g.*, in the query *"1999 ron howard"*, *Ron Howard* has been labeled as a *Director* as in 1999 he directed the movie *EDtv* and did not star in any movie), otherwise we assigned the most likely label based on the number of movies the person has directed or starred. Therefore, we can state that, where meaningful, we applied a *contextual* labeling.

7.4 Experimental Setup

The experiments reported in this section aim to answer the following research questions:

[10] https://www.wikipedia.org
[11] https://www.imdb.com

RQ1 How does a semantic query tagger trained with *synthetic* data perform w.r.t. the same model trained with *real-world* queries?

RQ2 Can we improve the performance of a semantic query tagger by pre-training it with *synthetic* data before fine-tuning it with *real-world* queries?

RQ3 Can pre-training with many synthetic queries improve the model consistency in predicting semantic classes under-represented in a real-world training set?

RQ4 Is the performance boost given by pre-training, if any, consistent over time while new real-world training data become available?

RQ5 When does fine-tuning with real-world data become effective for achieving performance improvements over a model trained only on synthetic queries?

To answer the research questions **RQ1**, **RQ2**, and **RQ3**, we conducted a comparative evaluation of the classification effectiveness of the semantic query tagger described in Section 7.2.2 when trained with synthetic data, real-world data, or a combination of both. Then, we simulated a dynamic environment where new labeled queries are collected over time to evaluate the models at regular intervals, allowing us to answer the research questions **RQ4** and **RQ5**.

In this following, we describe the experimental settings, the model training configuration and the evaluation metrics used to assess the effectiveness of the proposed approach.

7.4.1 Compared Models

All the models we compare in our experimental evaluation are based on the query tagger described in Section 7.2.2. What differs from one another is the data and training procedure employed to train the model. Specifically, we considered the following variants:

- **Raw:** It refers to the model trained with synthetic queries generated without the use of query variation techniques described in Section 7.2.1, *i.e.*, the training queries are generated by sole string concatenation. *Raw* was added to the evaluation to assess the importance of adding variations to the query generation procedure.

- **Synthetic:** It refers to the model trained with the synthetic queries generated following the query generation procedure described in Section 7.2.1.

- **Raw:** It refers to the model trained with queries from the proposed benchmark dataset introduced in Section 7.3.

- **Pre-Trained:** It refers to the model pre-trained on the training data of *Synthetic* and fine-tuned with the real-world training queries of *Real*.

7.4.2 Datasets

To promote a realistic evaluation setting, we split the dataset introduced in Section 7.3 into *train*, *dev*, and *test* sets temporally, using the queries issued by the AOL users in the first two months as *train set*, and those from the two subsequent two-weeks periods as *dev set* and *test set*. As the queries issued by a user in the same search session often share several terms, randomly splitting the queries without accounting for the temporal aspect would cause several query term overlaps between the splits and thus lead to unrealistic results.

To conduct a fine-grained evaluation, we built three different scenarios of increasing difficulty by subsetting our proposed benchmark dataset. The first scenario, *Basic*, comprises only queries containing the following semantic components: *Actor*, *Country*, *Genre*, *Title*, *Year*, and *O*. We then added the semantic components *Director* and *Sort* to create the *Advanced* scenario. Finally, we added *Production Company* and *Tag* to create the *Hard* scenario. The rationale behind these choices is as follows: the *Basic*

Table 7.1: Statistics of the benchmark datasets.

	Basic	**Advanced**	**Hard**
# train queries	3938	4292	5131
# dev queries	601	672	822
# test queries	538	610	796
Total	5077	5574	6749

scenario is composed of semantic components whose vocabularies are disjoint; the *Advanced* scenario introduces vocabulary overlaps (actors/directors), and a semantic class with few manually defined values; the *Hard* scenario introduces a semantic class often subject to omissions, *e.g.*, *Walt Disney Pictures* $\rightarrow$ *disney*, and a class, *Tag*, affected by vocabulary overlaps with the others and vocabulary mismatch between queries and documents. Table 7.1 reports some statistics regarding the proposed scenarios.

For each of these scenarios, we generated 100000 unique training queries for *Raw*, *Synthetic*, and *Pre-Trained*. We discarded generated queries present in the *dev* and *test* sets.

7.4.3 Structured Corpus

As a structured corpus to use for query synthetic generation and for computing the gazetteers-based features, we employ a public dataset hosted on Kaggle[12]. This dataset contains structured information about many movies, such as title, genre and year. This information originally comes from The Movie Database[13], a collaborative online database for movies and TV shows. For consistency with our query set, we filtered out every movie released after 2006.

[12] https://www.kaggle.com/rounakbanik/the-movies-dataset
[13] https://www.themoviedb.org

7.4.4 Model Training Configuration

In each of the conducted experiments, we trained the models for 50 epochs using Stochastic Gradient Descent, batch size of 64, and a starting learning rate of 0.1. We used a starting learning rate of 0.01 only for the *Pre-trained* model on the *Basic* and *Advanced* scenarios, as the pre-trained model already achieved good performances (see Section 7.5). We halved the learning rate when the training loss did not decrease for 5 consecutive epochs. We applied Dropout [251] with a probability of 0.5 on the Conditional Random Field's input to help with regularization. As Semantic Query Labeling is a multi-class classification problem, we optimize our models using Softmax Cross-Entropy Loss. We selected the final models basing on their performances on the *dev set* in the best epoch.

7.4.5 Evaluation Metrics

To comparatively evaluate the proposed models, we employed F1, Micro-F1 and Macro-F1 scores. F1 score is defined as the harmonic mean of Precision and Recall. Micro-F1 is the average of the F1 scores computed independently for each class and weighted by their number of samples. For imbalanced datasets, the Micro-F1 score can be skewed towards the most populated classes. Macro-F1 is the average of the F1 scores computed independently for each class. Each class contributes equally to this score. A noticeable discrepancy between the Micro-F1 score (high) and the Macro-F1 score (low) highlights inconsistency in the model predictions, suggesting that the model is skewed towards specific classes, usually the most popular ones in the training set.

7.5 Results and Discussion

7.5.1 Experiment I

The first experiment we conducted aimed to evaluate the performance of the semantic query tagger described in Section 7.2.2 when trained with synthetic data, real-world data, or a combination of both and to answer the research questions **RQ1**, **RQ2**, and **RQ3**. Specifically, we aimed to assess whether the model trained with synthetic data could reach the performance of the model trained with the real-world ones (**RQ1**). Moreover, we investigated the use of synthetic data as a means for pre-training the model before fine-tuning it with the real-world queries (**RQ2**) and whether the use of cheap and abundant training data could improve the model from a consistency perspective (**RQ3**), *i.e.*, the classification effectiveness for the semantic classes under-represented in the real-world training set. We conducted this evaluation by comparing the obtained Micro-F1 and Macro-F1 scores as described in Section 7.4.5.

Table 7.2 shows the results obtained by training the models described in Section 7.4.1 on each of the proposed scenarios. As expected, the addition of variations to the query generation process is mandatory to obtain good results, as shown by the performance differences of *Raw* and *Synthetic* in each of the considered evaluation scenarios. Conversely, we found the comparison between the *Synthetic* and the *Real* models to be very interesting. Both model performed well on the *Basic* dataset, with *Real* slightly outperforming *Synthetic*. Surprisingly, on the *Advanced* dataset *Synthetic* performed better than *Real*, with a 12% increment in Macro-F1. The discrepancy in the Micro-F1 and Macro-F1 scores of *Real* highlights inconsistency in model predictions, suggesting that the model is skewed towards the most popular classes. It also suggests the need for many samples to successfully train a semantic query tagger w.r.t. to less common semantic classes and reinforces the motivations underlying our

query generation proposal. We find both *Synthetic* and *Real* to achieve unsatisfactory results on the *Hard* dataset, although the performance of *Synthetic* is still worth mentioning, being the model trained only on synthetic data. This last observation suggests Semantic Query Labeling is far from being solved in difficult domains, and highlight the importance of the proposed benchmark dataset for advancing in this regard.

The *Pre-trained* model consistently outperforms the other considered models in all the evaluation scenarios, achieving considerable improvements over both the *Synthetic* and the *Real* models. Interestingly, we registered the most noticeable benefits of pre-training / fine-tuning on *Hard*, the most complex scenario among the three. Furthermore, the discrepancies in Micro-F1 and Macro-F1 scores that affected the *Real* model — highlighting inconsistency in the model predictions and suggesting it is skewed towards the most popular classes in the real-world training set — do not affect the *Pre-trained* model. The latter gets its consistency from the large synthetically generated query sets it was pre-trained on, where each semantic class is represented evenly.

Table 7.3 shows the F1 scores computed for each semantic class. To reduce table cluttering we do not report results for the *Raw* model, which underperformed w.r.t. the other considered models. As shown in the table, the obtained results suggest that the synthetically generated queries can play a complementary role w.r.t. real-world queries in effectively training a semantic query tagger. In fact, by pre-training the semantic query tagger with many synthetic queries, we can expose the model to abundant in-domain and task-related information and achieve the best performances across the line.

Table 7.2: Effectiveness of the models on the three proposed evaluation scenarios. Best values are highlighted in boldface. Second-best results are underlined.

Model	*Basic*		*Advanced*		*Hard*	
	Micro-F1	**Macro-F1**	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**
Raw	0.840	0.718	0.756	0.582	0.617	0.514
Synthetic	0.909	0.884	<u>0.903</u>	<u>0.865</u>	0.765	<u>0.756</u>
Real	<u>0.927</u>	<u>0.903</u>	0.896	0.776	<u>0.816</u>	<u>0.756</u>
Pre-trained	**0.934**	**0.910**	**0.925**	**0.893**	**0.840**	**0.828**

Table 7.3: Effectiveness of the models for each semantic class on the three proposed evaluation scenarios. Best values are highlighted in boldface. Second-best results are underlined.

Scenario	**Model**	*Actor* **F1**	*Country* **F1**	*Genre* **F1**	*Title* **F1**	*Year* **F1**	*Director* **F1**	*Sort* **F1**	*Tag* **F1**	*Company* **F1**
Basic	Synthetic	<u>0.898</u>	0.811	<u>0.867</u>	0.917	0.928	*N/A*	*N/A*	*N/A*	*N/A*
Basic	Real	0.865	**0.857**	**0.897**	**0.949**	<u>0.945</u>	*N/A*	*N/A*	*N/A*	*N/A*
Basic	Pre-trained	**0.905**	**0.857**	0.862	<u>0.945</u>	**0.978**	*N/A*	*N/A*	*N/A*	*N/A*
Advanced	Synthetic	<u>0.885</u>	<u>0.833</u>	**0.923**	0.914	<u>0.983</u>	<u>0.667</u>	0.853	*N/A*	*N/A*
Advanced	Real	0.844	0.765	0.880	<u>0.921</u>	0.975	0.111	**0.937**	*N/A*	*N/A*
Advanced	Pre-trained	**0.890**	**0.849**	<u>0.895</u>	**0.937**	**1.000**	**0.750**	<u>0.929</u>	*N/A*	*N/A*
Hard	Synthetic	<u>0.857</u>	0.773	0.855	0.777	<u>0.971</u>	<u>0.550</u>	0.876	0.522	0.623
Hard	Real	0.831	**0.837**	<u>0.873</u>	<u>0.854</u>	0.956	0.222	<u>0.883</u>	<u>0.576</u>	<u>0.771</u>
Hard	Pre-trained	**0.884**	<u>0.809</u>	**0.897**	**0.857**	**0.985**	**0.667**	**0.931**	**0.600**	**0.817**

7.5.2 Experiment II

The second experiment we conducted aimed at two goals. First, we evaluate the consistency over time of the improvements brought by pre-training with synthetic data to assess whether it is always beneficial, aiming to answer our fourth research question, **RQ4**. Then, to answer our fifth research question (**RQ5**), we assess *when* fine-tuning the pre-trained model with real-world data becomes effective in achieving performance gains.

For this experiment, we simulated a dynamic environment where new labeled queries are collected over time by relying on the time-stamps from the original AOL query logs. During the simulation, we evaluated the models from the previous experiment at regular intervals. At each evaluation step, we used the corresponding

Table 7.4: Average effectiveness of the models over time. Best values are highlighted in boldface. Second-best results are underlined.

Model	*Basic*		*Advanced*		*Hard*	
	Micro-F1	**Macro-F1**	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**
Synthetic	<u>0.916</u>	<u>0.892</u>	0.896	<u>0.851</u>	0.778	0.743
Real	0.911	0.845	<u>0.899</u>	0.777	<u>0.825</u>	<u>0.753</u>
Pre-trained	**0.936**	**0.907**	**0.924**	**0.877**	**0.850**	**0.821**

week-worth of queries as our test set, the queries from the week before as the <u>dev set</u>, and all the queries submitted in the antecedent weeks as the <u>train set</u>.

In Table 7.4 are reported Micro-F1 scores and Macro-F1 scores averaged across all the evaluation steps, for all the evaluation scenarios. As shown in the table, the *Synthetic* model registered only a 6% decrease in Micro-F1 on average w.r.t. the *Real* model in the worst-case scenario, *Hard*, while the *Pre-trained* model consistently achieves better performances than both the *Real* model and the *Synthetic* model, corroborating the results of the first experiment.

Figure 7.1 depicts the graphs of the results of the *over time evaluation* we conducted for the *Hard* scenario. As shown in the figure, the *Pre-trained* model overtakes *Real* as soon as it is fine-tuned, achieving top performances. Furthermore, the results highlight that the improvements brought by pre-training are consistent over time. Because of that, the performance penalty between *Synthetic* and *Real* will never take effect as we can replace *Synthetic* with *Pre-trained* as soon as we gather real-world queries. These results suggest that, while we collect real-world training data for conducting fine-tuning, we can employ *Synthetic* with no actual performance loss w.r.t. *Real*.

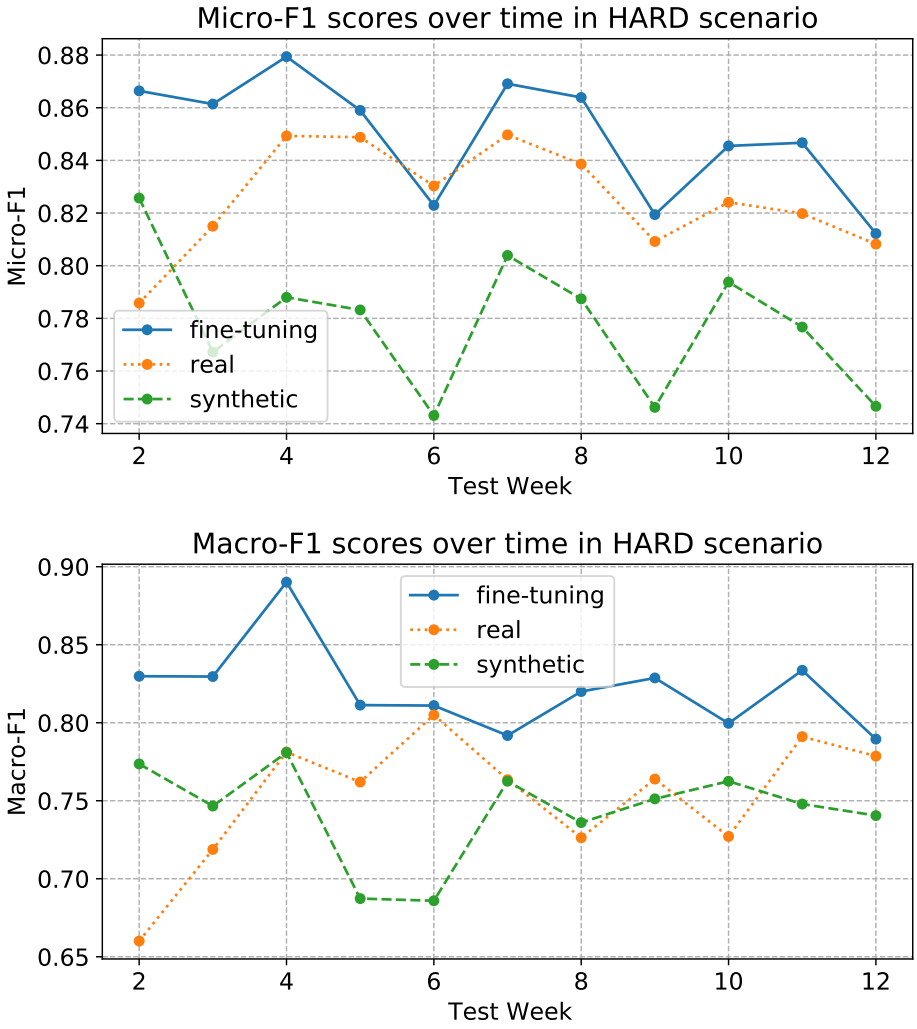


Figure 7.1: Over time effectiveness of the models in the *HARD* scenario.

7.6 Summary

In this chapter, we addressed the lack of a publicly available dataset for studying Semantic Query Labeling by introducing a novel dataset composed of 6749 unique manually annotated queries that we make available for future research. Moreover, we proposed a query generation method that, by leveraging the structure of the documents, automatically produces training data for a semantic query tagger, aiming at reducing the need for manually labeled data. We also studied the effects of using these synthetic training data for pre-training a semantic query tagger before fine-

tuning it with real-world queries. To evaluate our proposal, we compared the same semantic query tagging model when trained with different data.

Our experimental evaluation shows that a semantic query tagger trained with synthetic data performs comparably to the same model trained with real-world queries. Moreover, the performance of a semantic query tagger can be improved by first pre-training it with synthetic data and subsequently fine-tuning it with real-world queries. Pre-training with many synthetic queries also improves the model consistency in predicting semantic classes under-represented in a real-world training set. Furthermore, the performance boost given by pre-training is consistent over time while new real-world training data become available. Finally, fine-tuning with real-world data becomes effective for achieving performance improvements over a model trained only on synthetic queries as soon as minimal amounts of real-world training data become available.

To conclude, leveraging the already available information from a structured corpus is a valuable — *and cheap* — option for achieving performance gains without the need for additional real-world data, which, conversely, is very costly.

CHAPTER 8

RANX: A PYTHON LIBRARY FOR RANKING EVALUATION, COMPARISON, AND FUSION

Nowadays, researchers in Information Retrieval mostly rely on Python [269, 205] as their primary coding language. Because of that, many Python tools providing Information Retrieval experimentation and evaluation functionalities have recently been proposed [178, 179, 176, 108, 208, 45, 169, 168]. Nevertheless, we think there is still the need for user-friendly Python libraries following a truly *Plug & Play* paradigm, especially for young researchers with different backgrounds. For this reason, in this chapter, here we present `ranx`[1], a Python library built following a user-centered design providing several evaluation, comparison, and fusion functionalities. First, we introduce the implemented functionalities to manage the evaluation and comparison of Information Retrieval systems in Section 8.1. Then, we discuss the Metasearch algorithms provided by `ranx` and the advanced functionalities implemented for their optimization in Section 8.2.

8.1 Evaluation and Comparison

Offline evaluation and comparison of different Information Retrieval systems is a fundamental step in developing new solutions [113, 234]. The introduction of

[1] https://github.com/AmenRa/ranx

`trec_eval`[2] by the Text Retrieval Conference (TREC)[273] allowed standardizing evaluation measures in Information Retrieval. This handy tool comes as a standalone C executable that researchers and practitioners must compile and run through a command-line interface. Unfortunately, it does not provide additional functionalities, such as comparing results from different Information Retrieval systems or exporting the evaluation results to specific formats (*e.g.*, LaTeX). Despite some successful attempts to bring Information Retrieval `trec_eval`'s evaluation metrics to Python [108, 208], there are still no libraries offering advanced functionalities for evaluating and comparing multiple retrieval models and offering a user-friendly interface to those.

To these extents, `ranx`, the Python library we propose, lets the user calculate multiple evaluation measures, run statistical tests, and visualize comparison summaries, all in a few lines of code. Furthermore, it offers a convenient way of managing the evaluation results, allowing exporting them in LaTeX format for scientific publications. On top of that, `ranx` achieves top-notch efficiency thanks to `Numba` [145], a *just-in-time* compiler [17] for Python and NumPy [204, 268, 114] code, that we used for implementing all the core functionalities. Specifically, it allows leveraging high-speed vector operations and automatic parallelization with ease. To the best of our knowledge, none of the other available tools support multi-threading, which can vastly improve efficiency and grants `ranx` the ability to scale on large industrial datasets.

In the following, we present the main evaluation and comparison functionalities provided by `ranx`: the `Qrels` and `Run` classes, the `evaluate` method, the `compare` method, and the `Report` class. More details and examples are available in the official repository.

[2] https://github.com/usnistgov/trec_eval

8.1.1 `Qrels` and `Run` Classes

The first step in the offline evaluation of the effectiveness of an Information Retrieval system is the definition of a list of *query relevance judgments* (*qrels*) and the ranked lists of documents retrieved for those queries by the system under evaluation (*run*). To ease the managing of these data, `ranx` implements two dedicated Python classes: 1) `Qrels` for the query relevance judgments and 2) `Run` for the computed ranked lists. As shown in Listing 1, the users can convert the evaluation data from Python dictionaries, commonly employed to store the query relevance judgments and the ranked lists of documents retrieved by a retrieval system when working with Python, into the dedicated `ranx` classes. The users can also import the evaluation data from TREC-style and JSON files and Pandas DataFrames [188]. Moreover, `ranx` integrates seamlessly with *ir-datasets* [176], allowing the users to load query relevance judgments for *several* Information Retrieval datasets, such as those from TREC's challenges[3], BEIR [264], and MS MARCO [201]. `ranx` takes care of sorting and checking the data so that the users do not need to. Finally, `Qrels` and `Run` can be saved as TREC-style or JSON files for sharing. To learn more about `Qrels` and `Run`, we invite the reader to follow our online Jupyter Notebook[4].

8.1.2 Metrics

`ranx` provides the several ranking evaluation metrics such as *Reciprocal Rank*, *Average Precision*, and *Normalized Discounted Cumulative Gain* [125, 50], as show in Table 8.1. We tested the implemented metrics against `trec_eval` for correctness to assess the compliancy of those with the standard implementations used for Information Retrieval evaluation. `ranx` provides access to the supported evaluation metrics through a single interface, the method `evaluate`. As shown in Listing 2, `ranx`

[3] https://trec.nist.gov

[4] https://colab.research.google.com/github/AmenRa/ranx/blob/master/notebooks/2_qrels_and_run.ipynb

```python
from ranx import Qrels

# Default usage, convert Python Dictionary
qrels_dict = { "q_1": { "d_12": 5, "d_25": 3 },
               "q_2": { "d_11": 6, "d_22": 1 } }

qrels = Qrels(qrels_dict)

# Import qrels from JSON file
qrels = Qrels.from_file(path_to_qrels)

# Import qrels from TREC-Style file
qrels = Qrels.from_file(path_to_qrels, kind="trec")

# Import qrels from Pandas DataFrame
qrels = Qrels.from_df(
    qrels_df,
    q_id_col="q_id",
    doc_id_col="doc_id",
    score_col="score",
)

# Import qrels from ir-datasets (FOR QRELS ONLY)
qrels = Qrels.from_ir_datasets("msmarco-document/dev")

# Save as JSON file
qrels.save("qrels.json")

# Save as TREC-Style file
qrels.save("qrels.txt", kind="trec")
```

Listing 1: `Qrels`' methods. *The same methods are also available for* `Run`.

allows evaluating a *run* in a single line of code and allows the user to employ one or multiple metrics at once and define cut-offs using a convenient syntax. Every time the user evaluates a metric over a `Qrels`-`Run` pair, `ranx` stores the metrics' scores for each query and their averages in the Run instance so that they can be accessed later on.

As mentioned above, `ranx` relies on `Numba` to efficiently compute the scores for the evaluation metrics. In this regard, we simulated three scenarios of increasing data intensity to assess both `ranx`'s efficiency in relation to the `pytrec_eval` one and its saturation point. Table 8.2 reports the result of the efficiency comparison between `ranx` and `pytrec_eval` [108], a Python interface to `trec_eval`, we performed. As

the table shows, `ranx` is several times faster than `pytrec_eval`, allowing it to scale seamlessly on hundreds of thousands of queries and, potentially, more. Although we considered very query-rich scenarios, `ranx` was far from reaching a saturation point thanks to its `Numba`-based implementation.

Table 8.1: Provided evaluation metrics. The *cut-off* column indicates whether the metric support cut-offs.

Metric	**Cut-off**
Hits	✓
Hit Rate / Success	✓
Precision	✓
Recall	✓
F1	✓
R-Precision	✗
Bpref [48]	✗
Rank-biased Precision [194]	✗
Mean Reciprocal Rank	✓
Mean Average Precision	✓
NDCG [125]	✓
NDCG Burges [50]	✓

8.1.3 Comparison and Statistical Testing

As comparison is one of the fundamental steps in retrieval systems' evaluation, `ranx` implements a functionality — `compare` — for comparing multiple `Runs`. As shown in Listing 3, it computes the scores for a list of metrics provided by the user and performs statistical testing on those scores through one of the supported statistical tests: Two-sided Paired Student's t-Test, Fisher's Randomization Test [244], and Tukey's HSD Test [59, 89]. As a result, the `compare` method outputs an object of the `Report` class.

```python
from ranx import evaluate

# Compute score for a single metric
evaluate(qrels, run, "ndcg@5")
>>> 0.7861

# Compute scores for multiple metrics at once
evaluate(qrels, run, ["map@5", "mrr"])
>>> {"map@5": 0.6416, "mrr": 0.75}

# Computed metric scores are saved in the Run object
run.mean_scores
>>> {"ndcg@5": 0.7861, "map@5": 0.6416, "mrr": 0.75}

# Access scores for each query
dict(run.scores)
>>> {"ndcg@5": {"q_1": 0.9430, "q_2": 0.6292},
      "map@5": {"q_1": 0.8333, "q_2": 0.4500},
       "mrr": {"q_1": 1.0000, "q_2": 0.5000}}
```

Listing 2: Usage example of the `evaluate` method.

8.1.4 The `Report` Class

The `Report` class store all the data produced by performing a *comparison* as described previously. The user can access this information by simply printing a `Report` in a Python shell, as shown in Listing 3. It also allows exporting a LaTeX table presenting the average scores for each computed metric for each of the compared models, enriched with superscripts denoting the statistical significance of the improvements (if any), as well as a pre-defined caption. Table 8.3 was generated by `report.to_latex()`.

8.2 Metasearch

In this section we introduce and discuss the Metasearch functionalities provided by `ranx`.

Metasearch [16], sometimes called data-fusion [156, 281, 154, 241, 155, 121], is the problem of combining the results returned by multiple search engines in response

Table 8.2: Efficiency comparison between `ranx` (using different number of threads) and `pytrec_eval` (pytrec), a Python interface to `trec_eval`. The comparison was conducted with synthetic data. Queries have 1-to-10 relevant documents. Retrieved lists contain 100 documents. NDCG, MAP, and MRR were computed on the entire lists. Results are reported in milliseconds. Speed-ups were computed w.r.t. `pytrec_eval`.

metric	queries	pytrec	ranx t=1		ranx t=2		ranx t=4		ranx t=8	
NDCG	1 000	28	4	7.0×	3	9.3×	2	14.0×	2	14.0×
	10 000	291	35	8.3×	24	12.1×	18	16.2×	15	19.4×
	100 000	2 991	347	8.6×	230	13.0×	178	16.8×	152	19.7×
MAP	1 000	27	2	13.5×	2	13.5×	1	27.0×	1	27.0×
	10 000	286	21	13.6×	13	22.0×	9	31.8×	7	40.9×
	100 000	2 950	210	14.0×	126	23.4×	84	35.1×	69	42.8×
MRR	1 000	28	1	28.0×	1	28.0×	1	28.0×	1	28.0×
	10 000	283	7	40.4×	6	47.2×	4	70.8×	4	70.8×
	100 000	2 935	74	39.7×	57	51.5×	44	66.7×	38	77.2×

Table 8.3: Overall effectiveness of the models. Best results are highlighted in boldface. Superscripts denote statistically significant differences in Fisher's Randomization Test with $p \leq 0.01$.

#	Model	MAP@100	MRR@100	NDCG@10
a	model_1	0.3202^{b}	0.3207^{b}	0.3684^{bc}
b	model_2	0.2332	0.2339	0.239
c	model_3	0.3082^{b}	0.3089^{b}	0.3295^{b}
d	model_4	0.3664^{abc}	0.3668^{abc}	0.4078^{abc}
e	model_5	$\mathbf{0.4053}^{abcd}$	$\mathbf{0.4061}^{abcd}$	$\mathbf{0.4512}^{abcd}$

to a given query in a way that optimizes the performance of their combination. Previous works [16, 88, 148, 197, 20, 156, 281, 154, 68, 241, 155, 196] have shown this combination to consistently improve the retrieval effectiveness of the combined systems. As discussed by Aslam and Montague [16], Metasearch algorithms can be applied *externally*, when the combined search engines are completely independent from each other, or *internally*, when a single search engine comprises multiple retrieval models. In the latter case, Metasearch techniques allow decomposing a large and monolithic search engine into smaller and more specialized modules, whose results

```python
from ranx import compare

# Compare different runs and perform statistical tests
report = compare(
    qrels=qrels,
    runs=[run_1, run_2, run_3, run_4, run_5],
    metrics=["map@100", "mrr@100", "ndcg@10"],
    max_p=0.01  # P-value threshold
)

print(report)
>>>
#    Model    MAP@100      MRR@100      NDCG@10
---  -------  -----------  -----------  -----------
a    model_1  0.3202^b     0.3207^b     0.3684^bc
b    model_2  0.2332       0.2339       0.2390
c    model_3  0.3082^b     0.3089^b     0.3295^b
d    model_4  0.3664^abc   0.3668^abc   0.4078^abc
e    model_5  0.4053^abcd  0.4061^abcd  0.4512^abcd
```

Listing 3: Usage example of the `compare` method.

are fused after the retrieval phase. Metasearch techniques are usually classified as score-based [88, 148, 281] or rank-based [16, 196, 154, 241, 155, 68, 156, 197, 20] methods depending on whether they need relevance scores or just the ranking positions of the retrieved documents. They can also be divided into supervised and unsupervised methods, whether they require training data or not.

Despite Metasearch has long been studied in Information Retrieval and simple combination schemes, such as the weighted sum of multiple relevance scores, are often used for fusing the rankings produced by modern multi-stage retrieval systems [277, 92, 158], most of the proposed algorithms do not have a publicly available implementation. Moreover, there is a lack of a dedicated library providing several of these algorithms for research and optimization in a single package. As far as we know, `TrecTools`[5] [208] and `Polyfuse`[6] [141] are the only available tools exposing some Metasearch algorithms. Unfortunately, they both provide few fusion methods and lack advanced functionalities for fusion optimization. `TrecTools` is

[5] https://github.com/joaopalotti/trectools
[6] https://github.com/rmit-ir/polyfuse

meant for meta-analysis purposes of TREC-like campaigns' submissions rather than Metasearch research or optimization, which explains the lack of more algorithms and advanced features. `Polyfuse` was a companion tool for the SIGIR 2018's tutorial on fusion [141]. It comes as a simple command-line tool with limited functionalities rather than a full-fledged library for Metasearch, limiting its adoption as a day-to-day tool for Information Retrieval researchers and practitioners. The unavailability of a dedicated Metasearch library providing working implementations of several algorithms and advanced functionalities for fusion optimization motivates the work we have undergone to extend `ranx` to accommodate Metasearch functionalities.

Specifically, `ranx` provides a user-friendly interface to 25 Metasearch algorithms, implemented leveraging `Numba`. Our library offers both score-based and rank-based fusion approaches, all of which can be accessed through a convenient interface, the method `fuse`, as we will discuss in the next section. Additionally, `ranx` allows the user to experiment with six normalization strategies to transform the results of different search engines to make them comparable, which is mandatory for the correct application of many Metasearch algorithms [195, 70]. Finally, since several Metasearch algorithms require a training or optimization phase, `ranx` provides a convenient feature for their optimization that automatically evaluates pre-defined hyper-parameters configurations via grid search.

In the next section, we introduce the implemented functionalities and provide usage examples that show their user-centered design and usability. Then, we compare the Metasearch functionalities offered by `ranx` with those of already available tools for Metasearch, *i.e.*, `TrecTools` and `Polyfuse`, and highlight their differences. Finally, we discuss some use cases for our library to show the multi-faceted utility of a publicly available tool providing Metasearch functionalities. More details and examples are available in the official repository[7], documentation[8], and Jupyter

[7] https://github.com/AmenRa/ranx
[8] https://amenra.github.io/ranx

Notebook[9].

8.2.1 Normalization

The first step when fusing the results of multiple retrieval systems with score-based fusion methods is to normalize them to make them comparable. This operation is required as different retrieval models estimate relevance scores on different scales and ranges and distribute them differently [195]. For example, the classic probabilistic retrieval model BM25 [226] outputs unbounded positive relevance scores, while modern Deep Learning-based retrieval systems, relying on the dot-product or the cosine similarity to compute relevance scores, often output unbounded scores or scores in the interval $[-1, 1]$. To this extent, `ranx` offers six normalization strategies: Min-Max Norm, Max Norm, Sum Norm [195], ZMUV Norm [195], Rank Norm [225], and Borda Norm [225]. Note that Rank Norm and Borda Norm transform the original relevance-based scores assigned to the documents into scores based on their positioning in the list of ranked results. Therefore, when using these normalization strategies, score-based fusion methods act as rank-based methods. The reader can refer to the package documentation for further details on the provided normalization strategies [10].

8.2.2 Fusion

Table 8.4 lists the fusion algorithms provided by `ranx` and reports whether they require a training phase or have hyper-parameters that need to be optimized. We classify those algorithms into four categories: score-based methods, rank-based methods, probabilistic methods, and voting-based methods. Score-based meth-

[9] https://colab.research.google.com/github/AmenRa/ranx/blob/master/notebooks/5_fusion.ipynb

[10] https://amenra.github.io/ranx/normalization

ods [88, 148, 281] combine the relevance scores given to the documents retrieved by multiple search engines to derive the scores used to decide the final arrangement for the retrieved documents. Rank-based methods [68, 197, 20] rely only on the positioning of the documents retrieved by the considered search engines to derive the final ranking. Rank-based methods are particularly useful when the relevance scores given to the documents retrieved by the different search engines are unavailable (this is the case of Web search aggregators such as Kayak[11] and Skyscanner[12]). A special case of the rank-based methods are the probabilistic methods [16, 154, 241, 155, 156], which derive a probability distribution of the relevance over the ranking positions, *i.e.*, for each search engine, they assign to each ranking position the probability of finding a relevant document in that specific position. Probabilistic methods require a training phase to estimate this probability distribution. The voting-based methods [16, 196] adapt voting procedures, such as Borda Count and the Condorcet election method, to Metasearch, combining the preferences of multiple "experts", *i.e.*, the search engines. The voting-based methods can be considered as a special case of rank-based methods as they only rely on the ranking positions of the documents.

To simplify the use of the provided algorithms, `ranx` exposes a single interface to access them: the method `fuse`, shown in Listing 4. This function takes as input the runs to be combined (the ranked lists of documents retrieved by the search systems for multiple queries), the name of the normalization strategy to apply before fusion, and the name of the fusion method. Optionally, as discussed in the next section, the `fuse` function can take in input some parameters required by the chosen fusion method.

[11] https://www.kayak.it
[12] https://www.skyscanner.it

Table 8.4: Provided fusion algorithms. Supervised means the algorithm requires a training phase. Params column indicates whether the algorithm has hyper-parameters that need to be optimized. TT column indicates whether the algorithm is provided by `TrecTools`. PF column indicates whether the algorithm is provided by `Polyfuse`.

Score-based Methods				
Name	**Supervised**	**Params**	**TT**	**PF**
CombANZ [88]	✗	✗	✓	✓
CombMAX [88]	✗	✗	✓	✓
CombMED [88]	✗	✗	✓	✓
CombMIN [88]	✗	✗	✓	✓
CombMNZ [88]	✗	✗	✓	✓
CombSUM [88]	✗	✗	✓	✓
CombGMNZ [148]	✗	✓	✗	✗
Mixed [281]	✗	✓	✗	✗
WMNZ [281]	✗	✓	✗	✗
Weighted Sum	✗	✓	✗	✗
Rank-based Methods				
Name	**Supervised**	**Params**	**TT**	**PF**
ISR [197]	✗	✗	✗	✓
Log_ISR [197]	✗	✗	✗	✓
LogN_ISR [197]	✗	✓	✗	✗
RBC [20]	✗	✓	✓	✓
RRF [68]	✗	✓	✓	✓
Probabilistic Methods				
Name	**Supervised**	**Params**	**TT**	**PF**
BayesFuse [16]	✓	✗	✗	✗
MAPFuse [156]	✓	✗	✗	✗
PosFuse [156]	✓	✗	✗	✗
ProbFuse [154]	✓	✓	✗	✗
SegFuse [241]	✓	✗	✗	✗
SlideFuse [155]	✓	✓	✗	✗
Voting-based Methods				
Name	**Supervised**	**Params**	**TT**	**PF**
BordaFuse [16]	✗	✗	✓	✓
Weighted BordaFuse [16]	✗	✓	✗	✗
Condorcet [196]	✗	✗	✗	✗
Weighted Condorcet [196]	✗	✓	✗	✗

```python
from ranx import fuse

combined_run = fuse(
    # The list of Run instances to fuse
    runs=[run_1, run_2],
    # The normalization strategy to use
    norm="min-max",
    # The fusion algorithm to use
    method="sum",  # Alias for CombSUM
)
```

Listing 4: Usage example of the `fuse` method.

8.2.3 Fusion Optimization

As reported in Table 8.4, many fusion algorithms require a training or optimization step. To this extent, `ranx` implements the functionalities needed to optimize those algorithms. Instead of exposing several training and optimization functions, `ranx` conveniently provides the users with a single easy-to-use interface, the `optimize_fusion` method, to optimize those algorithms with ease. Under the hood, it routes the input parameters to the correct functions and performs other operations if needed. In the case of the algorithms requiring hyper-parameters optimization, `ranx` comes with pre-defined hyper-parameters search spaces, which can be altered by the user. During optimization, `ranx` automatically generates and evaluates several hyper-parameters configurations via grid search, relieving the users of the burden of implementing such procedure for each of the provided algorithms. Once the optimization is complete, `optimize_fusion` outputs the trained/optimized parameters for correctly fusing the ranked lists produced by the search engines the user want to combine.

Listing 5 shows a usage example of the `optimize_fusion` method. In this case, the goal is to find the Weighted Sum algorithm's weights configuration that maximize the Normalized Discounted Cumulative Gain [125] at depth 100 (NDCG@100). To do so, the user provides some training data: the query relevance judgments for some

```python
from ranx import fuse, optimize_fusion

best_params = optimize_fusion(
    qrels=train_qrels,
    runs=[train_run_1, train_run_2],
    norm="min-max",
    method="wsum",  # Alias for Weighted Sum
    # The metric to maximize during optimization
    metric="ndcg@100",
)

combined_test_run = fuse(
    runs=[test_run_1, test_run_2],
    norm="min-max",
    method="wsum",
    params=best_params,
)
```

Listing 5: Usage example of the `optimize_fusion` method.

training queries (qrels) and the results lists (runs) produced for those queries by the retrieval systems she aims to combine. After normalization, `optimize_fusion` will automatically evaluate several weights configuration and output the best one. At this point, the user provides the runs for the test queries and the best weights configuration previously found to the `fuse` method to compute the final combined run.

In addition to finding the best hyper-parameters configuration for a given Metasearch algorithm, `optimize_fusion` can optionally return a report of the evaluated configurations for inspection. This behavior is achieved by setting the `optimize_fusion`'s parameter `return_optimization_report` to `True`. Table 8.5 shows the report for a Weighted Sum optimization over two runs using NDCG@100 as the metric to maximize. The first column of the table shows the configuration, while the second shows the score of NDCG@100 obtained with that configuration.

Advanced usage examples are provided in the Jupyter Notebook previously referenced.

Table 8.5: Optimization report.

Weights	NDCG@100
(0.0, 1.0)	0.502
(0.1, 0.9)	0.517
(0.2, 0.8)	0.531
(0.3, 0.7)	0.553
(0.4, 0.6)	**0.556**
(0.5, 0.5)	0.543
(0.6, 0.4)	0.528
(0.7, 0.3)	0.511
(0.8, 0.2)	0.493
(0.9, 0.1)	0.480
(1.0, 0.0)	0.452

8.2.4 Comparison with Available Tools

In this section, we compare `ranx` with existing tools for Metasearch, *i.e.*, `TrecTools` [208] and `Polyfuse` [141], highlighting their differences and pointing out the innovative aspects of the software library we present in this dissertation.

First, we introduce the tools considered for the comparison. `TrecTools` was proposed by Palotti et al. [208] as an analysis tool to support TREC-like campaigns. It provides several functionalities, such as evaluation metrics for Information Retrieval and fusion algorithms. `Polyfuse` was a companion software shared with the participants of the SIGIR 2018's tutorial on fusion [141]. It comes as a command-line tool providing Metasearch functionalities.

Table 8.4 reports the fusion approaches provided by `ranx` and whether they are available in `TrecTools` (TT column) or `Polyfuse` (PL column). Both `TrecTools` and `Polyfuse` provide a small subset of the fusion methods available in `ranx`. Specifically, `TrecTools` and `Polyfuse` implement nine and eleven Metasearch algorithms, respectively. Conversely, `ranx` provides all the algorithms supported by the other considered tools for Metasearch, and many others, accounting for 25 fusion methods. We also notice neither `TrecTools` nor `Polyfuse` provides methods requiring a supervised training phase, while `ranx` supports six.

Unfortunately, both `TrecTools` and `Polyfuse` lack advanced functionalities for fusion optimization. In contrast, `ranx` implements the fusion optimization functionality described in Section 8.2.3, which we think will be very convenient for both researchers and practitioners willing to maximize the performance of their Information Retrieval systems. While `ranx` and `Polyfuse` implement normalization strategies, we noticed `TrecTools` does not provide any. Note that we found those implemented by `Polyfuse` by digging into its source code, as they are not documented.

From a usability perspective, we argue `ranx` shines among the other tools, providing easy-to-use functionalities developed following a user-centered design and accounting for young researchers with different backgrounds. As pointed out in Section 8.2, `ranx` offers documentation (which we will enrich over time) pointing to the original papers of the implemented Metasearch algorithms and providing their BibTex references. Moreover, it provides a Jupyter Notebook showing its main features through a hands-on approach. `TrecTools` offers standard access to the implemented fusion algorithms through distinct functions, while `Polyfuse` comes as a command-line tool only. Both `TrecTools` and `Polyfuse` provide a simple usage example in their repository's readme files as the only documentation for the fusion algorithms they implement.

8.2.5 Use Cases

In this section, we briefly discuss some use cases for the Metasearch functionalities our proposed library implements. First, `ranx` provides the users with working implementations for several Metasearch algorithms that were previously unavailable, allowing the users to try different approaches to fuse the results of multiple search engines. Second, our library offers new opportunities to combine the rankings produced by modern two-stage retrieval pipelines, usually composed of a term matching first-stage retriever, *e.g.*, BM25, and a neural re-ranker. In those pipelines,

multiple rankings computed in response to the same queries are often combined through a weighted sum of the scores of the documents assigned by the different retrieval modules [277, 92, 158]. Moreover, finding the best way to combine those rankings could also be convenient for mining hard negatives to use for training Deep Learning-based retrieval models [294]. Finally, researchers can leverage the functions our library exposes to test novel normalization strategies with several Metasearch algorithms in a simple and efficient way. To do so, they can set the `norm` parameter to `null` when calling `fuse` or `optimize_fusion`, causing the normalization step to be bypassed. By doing so, they can test novel normalization strategies by providing data normalized with the approach at test.

8.3 Summary

In this chapter, we presented `ranx`, a Python library for the evaluation, comparison, and fusion of retrieval results powered by `Numba`. It provides a user-friendly interface to the most commonly used ranking evaluation metrics and a procedure for comparing the results of multiple models and export them as a LaTeX table. Moreover, it implement 25 Metasearch algorithms, both score-based and rank-based methods, six normalization strategies, and an automatic procedure for optimizing the algorithms requiring a training or optimization phase. We compared `ranx` with available tools for both evaluation and Metasearch, highlighting our library efficiency and pointing out the innovative aspects from both a functional and a usability perspective.

CHAPTER 9

CONCLUSIONS AND FUTURE WORK

In this chapter, we provide a broad summary of our work. We first summarize the issues and challenges addressed in our work and our primary contributions and results. We then highlight open challenges related to Personalized Search and several potential research directions for future work.

9.1 Overview of our Contributions and Results

In this dissertation, we addressed several topics related to Personalized Information Retrieval, Semantic Query Labeling, Evaluation, and Fusion. We now summarize our contributions and results.

Multi-Representation User Modeling In chapter 3, we addressed the problem of personalization representing the users' interests and preferences from multiple perspectives and proposed a novel personalized results re-ranking approach for Product Search. In particular, we investigated the use of four different user/item representations to enhance BM25 performances. We employed representations derived from user-generated content, user purchasing behavior, categorical information, and item popularity. Our empirical evaluations show that the proposed approach consistently enhances BM25 and outperforms recently proposed Neural Network-based models ([109, 6, 8]) specifically designed for Product Search on multiple benchmark datasets.

Finally, our proposed approach is fast, scalable, and easily extendable to accommodate additional representations.

Query-Aware User Modeling In chapter 4, we analyzed the effects of the Attention mechanism when employed for query-aware user modeling, highlighting some shortcomings that can cause the user model to be excessively noisy or skewed towards a single source of user interest. Furthermore, we addressed those shortcomings by proposing a novel user-data aggregation model called Denoising Attention, which can finely filter out noisy user-related information. Experimental evaluation in two different search scenarios, namely Web Search and Academic Search, shows the benefits of our proposed approach over other Attention variants ([19, 7, 272]) and highlights the potential of correctly managing the user-related information.

Personalized Query Expansion with Contextual Word Embeddings In chapter 5, we have addressed some issues arising from employing contextual word embeddings with current Personalized Query Expansion methods and proposed PQEWC, an approach designed to counteract those problems and take full advantage of contextual word embeddings. Specifically, our proposed method employs a clustering-based technique to group and identify the term embeddings most representative of the user interests and preferences and an approximation procedure of the personalized expansion terms selection to increase efficiency. Experimental evaluation shows the benefits of our proposed approach both in terms of efficiency and effectiveness over other recent Query Expansion methods, both personalized ([143, 300]) and non personalized ([278]). Moreover, it highlights how the effectiveness and the efficiency of Personalized Query Expansion based on word embeddings can be greatly improved with effective procedures.

Personalized Search Evaluation In Chapter 6, we discussed the current state of Personalized Search evaluation and the lack of large-scale datasets for training and evaluating Neural Networks-based Personalized Information Retrieval models. Furthermore, we introduced a novel large-scale benchmark spanning four domains and accounting for more than 18 million documents and 1.9 million queries. The size of the proposed benchmark, along with its rich structured information, opens up new research opportunities, from adopting graph-based approaches for Personalization to designing and training novel Neural Networks-based Personalization approaches. Finally, we provided baselines for future works, opening room for the evaluation of Personalized Search approaches, as well as Domain Adaptation and Transfer Learning methods in the context of Personalization.

Semantic Query Labeling In Chapter 7, we addressed the lack of a publicly available dataset for studying Semantic Query Labeling by introducing a novel dataset composed of 6749 unique manually annotated queries, which we released for future research. Moreover, we proposed a query generation method that, by leveraging the structure of the documents, automatically produces training data for a semantic query tagger, aiming at reducing the need for manually labeled data. We also studied the effects of using these synthetic training data for pre-training a semantic query tagger before fine-tuning it with real-world queries. Lastly, we simulated a dynamic environment and evaluated the consistency of the performance improvements brought by pre-training as real-world training data becomes available.

Our experimental evaluation showed that leveraging the already available information from a structured corpus is a valuable — *and cheap* — option for achieving performance gains without additional real-world data, which is very costly. Specifically, synthetic training queries can be employed to achieve tagging performances comparable to those obtained with real-world training queries and as a means to ef-

fectively pre-train a tagger before fine-tuning it with real-world data. We also found that pre-training with many synthetic queries improves the model consistency in predicting semantic classes under-represented in a real-world training set and that the performance boost given by pre-training is consistent over time while new real-world training data become available.

Evaluation, Comparison, and Fusion In Chapter 8, we presented `ranx`, a Python library for the evaluation, comparison, and fusion of retrieval results. It provides a user-friendly interface to the most commonly used ranking evaluation metrics and a procedure for comparing the results of multiple models. It also offers a convenient way of managing the evaluation results, allowing to export them in LaTeX format for scientific publications. Moreover, it implement 25 Metasearch algorithms, both score-based and rank-based methods, six normalization strategies, and an automatic procedure for optimizing the algorithms requiring a training or optimization phase, functionalities previously unavailable to the research community. We compared `ranx` with available tools for both evaluation and Metasearch, pointing out the innovative aspects from both a functional and a usability perspective. Finally, by leveraging modern technologies for high-speed vector operations and automatic parallelization, `ranx` also outperforms the available alternatives in terms of efficiency.

9.2 Future work

Multi-Representation User Modeling The Multi-Representation Personalization model we presented in 3 employs textual, collaborative, and categorical modalities to represent the user interests. However, other information could be available for personalization purposes, such as images, videos, and other relations among users and documents. The proposed model aggregates the contribution of the different user and document representations through a convex combination of user-document

compatibility scores with globally defined coefficients. As demonstrated later in Chapter 4, weighing the contribution of the user-related information w.r.t. the current search performed by the user can greatly benefit retrieval performances. Therefore, it could be interesting to apply this kind of approach in a multi-representation setting, which comes with its own challenges deriving from the intrinsic differences among the representations and the data that originated them.

Query-Aware User Modeling Despite the significant improvements brought by the Denoising Attention mechanism proposed in Chapter 4 when applied for selecting user-related information for query-aware personalization, some related problems are worth further study. The alignment model we employed, the scaled cosine similarity, could be replaced by a parameterized function that could leverage additional information other than the textual-based representations of a user-related document and the query. For example, the dates associated with the user-related documents might play a role in personalization, as documents written or consulted long before the query might be less relevant to personalization than more recent ones, despite being semantically related to the current search. Furthermore, the fixed value threshold parameter we employed could be sub-optimal in many cases. As shown by the difference in the threshold parameter values for the two considered datasets, different queries could benefit from more user-related information or require a finer selection of the user-related data employed in the personalization process. To conclude, the management of the user-related information during personalization is fundamental and far from being a solved issue, leaving room for further improvements.

Personalized Query Expansion with Contextual Word Embeddings Despite the significant improvements brought by our proposed Personalized Query Expansion method proposed in Chapter 5 both in terms of efficiency and effectiveness, we think there still are related topics worth further study. As in previous works, we relied

on cosine similarity to rank and select the expansion term embeddings. However, cosine similarity could be replaced by a more sophisticated parameterized function that, instead of just selecting the expansion terms by their semantic similarity with the original query terms, could evaluate their utility and assign them specific importance weights. Moreover, all the compared methods define the number of terms to add to the query as a fixed parameter, but this number is only generally good and not optimal for all the queries. Different queries could benefit from more expansion terms or work better without expansion. Furthermore, a weighing mechanism that can balance the importance of expansion terms one by one could improve the effectiveness brought by Query Expansion. Therefore, trying to predict the number of and the related weights for the expansion terms is a research direction still with much unexplored potential.

Personalized Search Evaluation Throughout this dissertation, we showed that Personalization in Information Retrieval can be performed with different methodologies and data. The benchmark datasets proposed in Chapter 6 fill a huge gap in Personalized Search evaluation and open many new opportunities and possibilities for future works, such as studying content-based personalization models and collaborative-filtering approaches, as we provide a rich set of metadata for each document and all the data to derive the user-document interactions. Moreover, the relations among the data, such as authorship relations and the paper references, can be represented by graph structures and leveraged by graph-based personalization approaches. Finally, our datasets allow the study and design of novel joint Personalized Search and Recommendation models [292]. Unfortunately, the proposed benchmark is limited in scope, as it is not suited for training and evaluating short-term user modeling techniques and session-based personalization approaches due to the lack of search sessions. However, in our opinion, proposing novel methodologies for evaluating

those approaches to Personalization is a direction worth pursuing in the future.

Semantic Query Labeling As described in Chapter 7, Semantic Query Labeling is a very understudied task. In our work, we focused on obtaining cheap-but-reliable training data for a semantic query tagger from a structured document collection, but we did not design a retrieval model able to leverage the semantic labels produced by this tagger due to time constraints. However, we think unfolding the relations between the query terms and the documents' structure could allow for designing sophisticated retrieval models that can potentially overcome the matching-based nature of current retrieval approaches, both lexical and semantic, introducing reasoning engines able to exploit those relations from a retrieval perspective. In other words, labeled queries, which contain both lexical (the textual terms), semantic (if projected into a latent space with word embedding techniques), and symbolic information (the labels), could enable novel approaches to Information Retrieval that are not possible without this kind of labeling. We acknowledge that part of the symbolic information is intrinsically captured by modern word embedding techniques, although not in a humanly interpretable way, undermining both explainability and the use of human-designed reasoning rules. Finally, we think designing methods to leverage the information already contained in a document collection is a research direction worth pursuing, and that should be more popular among the research community.

`ranx` Currently, we are working on `ranxhub`[1], a public hub for sharing state-of-the-art Information Retrieval Systems' results (*runs*) for multiple benchmarks. We think such a hub could be helpful for both researchers and reviewers, promote transparency, and speed up Information Retrieval research, lightening the burden of implementing and training modern state-of-the-art retrieval models based on Neural Networks. We aim to provide the community with a tool for finding appropriate

[1]https://amenra.github.io/ranxhub

baselines for their models and performing comparisons with their results in just a few minutes and less than ten lines of code, avoiding hardware-demanding and time-consuming computations and thus reducing our environmental footprint. In this regard, we are extending our Python library `ranx` to integrate direct access to this hub so that all the provided evaluation, comparison, and fusion functionalities will be readily available to be applied to the shared results.

BIBLIOGRAPHY

[1] Eytan Adar. User 4xxxxx9: Anonymizing query logs. In *Proc of Query Log Analysis Workshop, International Conference on World Wide Web*, 2007.

[2] Abien Fred Agarap. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375, 2018.

[3] Eugene Agichtein, Eric Brill, and Susan T. Dumais. Improving web search ranking by incorporating user behavior information. In Efthimis N. Efthimiadis, Susan T. Dumais, David Hawking, and Kalervo Järvelin, editors, *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6-11, 2006*, pages 19–26. ACM, 2006. doi:10.1145/1148170.1148177.

[4] Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. Multi-task learning for document ranking and query suggestion. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[5] Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. Context attentive document ranking and query suggestion. In Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer, editors, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 385–394. ACM, 2019. doi:10.1145/3331184.3331246.

[6] Qingyao Ai, Yongfeng Zhang, Keping Bi, Xu Chen, and W. Bruce Croft. Learning a hierarchical embedding model for personalized product search. In Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White, editors, *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 645–654. ACM, 2017. doi:10.1145/3077136.3080813.

[7] Qingyao Ai, Daniel N. Hill, S. V. N. Vishwanathan, and W. Bruce Croft. A zero attention model for personalized product search. In Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu, editors, *Proceedings of the 28th ACM International Conference*

on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019, pages 379–388. ACM, 2019. doi:10.1145/3357384.3357980.

[8] Qingyao Ai, Yongfeng Zhang, Keping Bi, and W. Bruce Croft. Explainable product search with a dynamic relation embedding model. *ACM Trans. Inf. Syst.*, 38(1):4:1–4:29, 2020. doi:10.1145/3361738.

[9] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis, editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 2623–2631. ACM, 2019. doi:10.1145/3292500.3330701.

[10] James Allan. HARD track overview in TREC 2003: High accuracy retrieval from documents. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of The Twelfth Text REtrieval Conference, TREC 2003, Gaithersburg, Maryland, USA, November 18-21, 2003*, volume 500-255 of *NIST Special Publication*, pages 24–37. National Institute of Standards and Technology (NIST), 2003.

[11] James Allan. HARD track overview in TREC 2004 - high accuracy retrieval from documents. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of the Thirteenth Text REtrieval Conference, TREC 2004, Gaithersburg, Maryland, USA, November 16-19, 2004*, volume 500-261 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2004.

[12] James Allan. HARD track overview in TREC 2005 high accuracy retrieval from documents. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of the Fourteenth Text REtrieval Conference, TREC 2005, Gaithersburg, Maryland, USA, November 15-18, 2005*, volume 500-266 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2005.

[13] Nawal Ould Amer, Philippe Mulhem, and Mathias Géry. Toward word embedding for personalized information retrieval. *CoRR*, abs/1606.06991, 2016.

[14] Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, Rodney Kinney, Sebastian Kohlmeier, Kyle Lo, Tyler Murray, Hsu-Han Ooi, Matthew E. Peters, Joanna Power, Sam Skjonsberg, Lucy Lu Wang, Chris Wilhelm, Zheng Yuan, Madeleine van Zuylen, and Oren Etzioni. Construction of the literature graph in semantic scholar. In Srinivas Bangalore, Jennifer Chu-Carroll, and Yunyao Li, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 3 (Industry Papers)*, pages 84–91. Association for Computational Linguistics, 2018. doi:10.18653/v1/n18-3011.

[15] Kamelia Aryafar, Devin Guillory, and Liangjie Hong. An ensemble-based approach to click-through rate prediction for promoted listings at etsy. In *Proceedings of the ADKDD'17, Halifax, NS, Canada, August 13 - 17, 2017*, pages 10:1–10:6. ACM, 2017. doi:10.1145/3124749.3124758.

[16] Javed A. Aslam and Mark H. Montague. Models for metasearch. In W. Bruce Croft, David J. Harper, Donald H. Kraft, and Justin Zobel, editors, *SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, September 9-13, 2001, New Orleans, Louisiana, USA*, pages 275–284. ACM, 2001. doi:10.1145/383952.384007.

[17] John Aycock. A brief history of just-in-time. *ACM Comput. Surv.*, 35(2):97–113, 2003.

[18] Hiteshwar Kumar Azad and Akshay Deepak. Query expansion techniques for information retrieval: A survey. *Inf. Process. Manag.*, 56(5):1698–1735, 2019. doi:10.1016/j.ipm.2019.05.009.

[19] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[20] Peter Bailey, Alistair Moffat, Falk Scholer, and Paul Thomas. Retrieval consistency in the presence of query variations. In Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White, editors, *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 395–404. ACM, 2017. doi:10.1145/3077136.3080839.

[21] Krisztian Balog. *Entity-Oriented Search*, volume 39 of *The Information Retrieval Series*. Springer, 2018.

[22] Michael Barbaro, Tom Zeller, and Saul Hansell. A face is exposed for aol searcher no. 4417749. *New York Times*, 9(2008):8, 2006.

[23] Elias Bassani. ranx: A blazing-fast python library for ranking evaluation and comparison. In Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty, editors, *Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part II*, volume 13186 of *Lecture Notes in Computer Science*, pages 259–264. Springer, 2022. doi:10.1007/978-3-030-99739-7_30.

[24] Elias Bassani. ranx: A blazing-fast python library for ranking evaluation and comparison. In Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty, editors, *Advances in*

Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part II, volume 13186 of *Lecture Notes in Computer Science*, pages 259–264. Springer, 2022. doi:10.1007/978-3-030-99739-7_30.

[25] Elias Bassani and Gabriella Pasi. A multi-representation re-ranking model for personalized product search. *Inf. Fusion*, 81:240–249, 2022. doi:10.1016/j.inffus.2021.11.010.

[26] Elias Bassani and Luca Romelli. ranx.fuse: A python library for metasearch. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management, CIKM 2022, Atlanta, Georgia, USA, October 17-21, 2022*. ACM, 2022.

[27] Nicholas J. Belkin, Michael J. Cole, Jacek Gwizdka, Yuelin Li, Jingjing Liu, Gheorghe Muresan, Catherine Smith, Arthur R. Taylor, Xiaojun Yuan, and Dmitri Roussinov. Rutgers information interaction lab at TREC 2005: Trying HARD. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of the Fourteenth Text REtrieval Conference, TREC 2005, Gaithersburg, Maryland, USA, November 15-18, 2005*, volume 500-266 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2005.

[28] Ori Ben-Yitzhak, Nadav Golbandi, Nadav Har'El, Ronny Lempel, Andreas Neumann, Shila Ofek-Koifman, Dafna Sheinwald, Eugene J. Shekita, Benjamin Sznajder, and Sivan Yogev. Beyond basic faceted search. In Marc Najork, Andrei Z. Broder, and Soumen Chakrabarti, editors, *Proceedings of the International Conference on Web Search and Web Data Mining, WSDM 2008, Palo Alto, California, USA, February 11-12, 2008*, pages 33–44. ACM, 2008. doi:10.1145/1341531.1341539.

[29] Matthias Bender, Tom Crecelius, Mouna Kacimi, Sebastian Michel, Thomas Neumann, Josiane Xavier Parreira, Ralf Schenkel, and Gerhard Weikum. Exploiting social relations for query expansion and result ranking. In *Proceedings of the 24th International Conference on Data Engineering Workshops, ICDE 2008, April 7-12, 2008, Cancún, Mexico*, pages 501–506. IEEE Computer Society, 2008. doi:10.1109/ICDEW.2008.4498369.

[30] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35 (8):1798–1828, 2013. doi:10.1109/TPAMI.2013.50.

[31] Paul N. Bennett, Ryen W. White, Wei Chu, Susan T. Dumais, Peter Bailey, Fedor Borisyuk, and Xiaoyuan Cui. Modeling the impact of short- and long-term behavior on search personalization. In William R. Hersh, Jamie Callan, Yoelle Maarek, and Mark Sanderson, editors, *The 35th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '12, Portland, OR, USA, August 12-16, 2012*, pages 185–194. ACM, 2012. doi:10.1145/2348283.2348312.

[32] Marin Bertier, Rachid Guerraoui, Vincent Leroy, and Anne-Marie Kermarrec. Toward personalized query expansion. In Tao Stein and Meeyoung Cha, editors, *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems, SNS 2009, Nuremberg, Germany, March 31, 2009*, pages 7–12. ACM, 2009. doi:10.1145/1578002.1578004.

[33] Keping Bi, Qingyao Ai, Yongfeng Zhang, and W. Bruce Croft. Conversational product search based on negative feedback. In Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu, editors, *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 359–368. ACM, 2019. doi:10.1145/3357384.3357939.

[34] Keping Bi, Qingyao Ai, and W. Bruce Croft. A transformer-based embedding model for personalized product search. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1521–1524. ACM, 2020. doi:10.1145/3397271.3401192.

[35] Keping Bi, Qingyao Ai, and W. Bruce Croft. A review-based transformer model for personalized product search. *CoRR*, abs/2004.09424, 2020.

[36] Keping Bi, Qingyao Ai, and W. Bruce Croft. Learning a fine-grained review-based transformer model for personalized product search. In Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai, editors, *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 123–132. ACM, 2021. doi:10.1145/3404835.3462911.

[37] Claudio Biancalana and Alessandro Micarelli. Social tagging in query expansion: A new way for personalized web search. In *Proceedings of the 12th IEEE International Conference on Computational Science and Engineering, CSE 2009, Vancouver, BC, Canada, August 29-31, 2009*, pages 1060–1065. IEEE Computer Society, 2009. doi:10.1109/CSE.2009.492.

[38] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly, 2009. ISBN 978-0-596-51649-9.

[39] Mohamed Reda Bouadjenek, Hakim Hacid, Mokrane Bouzeghoub, and Johann Daigremont. Personalized social query expansion using social bookmarking systems. In Wei-Ying Ma, Jian-Yun Nie, Ricardo Baeza-Yates, Tat-Seng Chua, and W. Bruce Croft, editors, *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*, pages 1113–1114. ACM, 2011. doi:10.1145/2009916.2010075.

[40] Mohamed Reda Bouadjenek, Amyn Bennamane, Hakim Hacid, and Mokrane Bouzeghoub. Evaluation of personalized social ranking functions of information retrieval. In Florian Daniel, Peter Dolog, and Qing Li, editors, *Web Engineering - 13th International Conference, ICWE 2013, Aalborg, Denmark, July 8-12, 2013. Proceedings*, volume 7977 of *Lecture Notes in Computer Science*, pages 283–290. Springer, 2013. doi:10.1007/978-3-642-39200-9_24.

[41] Mohamed Reda Bouadjenek, Hakim Hacid, and Mokrane Bouzeghoub. Sopra: a new social personalized ranking function for improving web search. In Gareth J. F. Jones, Paraic Sheridan, Diane Kelly, Maarten de Rijke, and Tetsuya Sakai, editors, *The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13, Dublin, Ireland - July 28 - August 01, 2013*, pages 861–864. ACM, 2013. doi:10.1145/2484028.2484131.

[42] Mohamed Reda Bouadjenek, Hakim Hacid, Mokrane Bouzeghoub, and Athena Vakali. Using social annotations to enhance document representation for personalized search. In Gareth J. F. Jones, Paraic Sheridan, Diane Kelly, Maarten de Rijke, and Tetsuya Sakai, editors, *The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13, Dublin, Ireland - July 28 - August 01, 2013*, pages 1049–1052. ACM, 2013. doi:10.1145/2484028.2484130.

[43] Mohamed Reda Bouadjenek, Hakim Hacid, Mokrane Bouzeghoub, and Athena Vakali. Persador: Personalized social document representation for improving web search. *Inf. Sci.*, 369:614–633, 2016. doi:10.1016/j.ins.2016.07.046.

[44] Mohamed Reda Bouadjenek, Hakim Hacid, and Mokrane Bouzeghoub. Personalized social query expansion using social annotations. *Trans. Large Scale Data Knowl. Centered Syst.*, 40:1–25, 2019. doi:10.1007/978-3-662-58664-8_1.

[45] Timo Breuer, Nicola Ferro, Maria Maistro, and Philipp Schaer. repro_eval: A python interface to reproducibility measures of system-oriented IR experiments. In *ECIR (2)*, volume 12657 of *Lecture Notes in Computer Science*, pages 481–486. Springer, 2021.

[46] John S. Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In David S. Touretzky, editor, *Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27-30, 1989]*, pages 211–217. Morgan Kaufmann, 1989.

[47] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish,

Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[48] Chris Buckley and Ellen M. Voorhees. Retrieval evaluation with incomplete information. In Mark Sanderson, Kalervo Järvelin, James Allan, and Peter Bruza, editors, *SIGIR 2004: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK, July 25-29, 2004*, pages 25–32. ACM, 2004. doi:10.1145/1008992.1009000.

[49] Jay Budzik and Kristian J. Hammond. User interactions with everyday applications as context for just-in-time information access. In Doug Riecken, David Benyon, and Henry Lieberman, editors, *Proceedings of the 5th International Conference on Intelligent User Interfaces, IUI 2000, New Orleans, LA, USA, January 9-12, 2000*, pages 44–51. ACM, 2000. doi:10.1145/325737.325776.

[50] Christopher J. C. Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Gregory N. Hullender. Learning to rank using gradient descent. In *ICML*, volume 119 of *ACM International Conference Proceeding Series*, pages 89–96. ACM, 2005.

[51] Silvia Calegari and Gabriella Pasi. Personalized ontology-based query expansion. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology - Workshops, 9-12 December 2008, Sydney, NSW, Australia*, pages 256–259. IEEE Computer Society, 2008. doi:10.1109/WIIAT.2008.242.

[52] Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, pages 243–250. ACM, 2008. doi:10.1145/1390334.1390377.

[53] Mark James Carman, Fabio Crestani, Morgan Harvey, and Mark Baillie. Towards query log based personalization using topic models. In Jimmy Huang, Nick Koudas, Gareth J. F. Jones, Xindong Wu, Kevyn Collins-Thompson, and Aijun An, editors, *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 1849–1852. ACM, 2010. doi:10.1145/1871437.1871745.

[54] David Carmel, Naama Zwerdling, Ido Guy, Shila Ofek-Koifman, Nadav Har'El, Inbal Ronen, Erel Uziel, Sivan Yogev, and Sergey Chernov. Personalized social search based on the user's social network. In David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy Lin, editors, *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, pages 1227–1236. ACM, 2009. doi:10.1145/1645953.1646109.

[55] Claudio Carpineto and Giovanni Romano. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.*, 44(1):1:1–1:50, 2012. doi:10.1145/2071389.2071390.

[56] John M Carroll and Mary Beth Rosson. Paradox of the active user. In *Interfacing thought: Cognitive aspects of human-computer interaction*, pages 80–111. 1987.

[57] Ben Carterette, Ashraf Bah, Evangelos Kanoulas, Mark M. Hall, and Paul D. Clough. Overview of the TREC 2013 session track. In Ellen M. Voorhees, editor, *Proceedings of The Twenty-Second Text REtrieval Conference, TREC 2013, Gaithersburg, Maryland, USA, November 19-22, 2013*, volume 500-302 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2013.

[58] Ben Carterette, Evangelos Kanoulas, Mark M. Hall, and Paul D. Clough. Overview of the TREC 2014 session track. In Ellen M. Voorhees and Angela Ellis, editors, *Proceedings of The Twenty-Third Text REtrieval Conference, TREC 2014, Gaithersburg, Maryland, USA, November 19-21, 2014*, volume 500-308 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2014.

[59] Benjamin A. Carterette. Multiple testing in statistical analysis of systems-based information retrieval experiments. *ACM Trans. Inf. Syst.*, 30(1):4:1–4:34, 2012. doi:10.1145/2094072.2094076.

[60] Marc-Allen Cartright, James Allan, Victor Lavrenko, and Andrew McGregor. Fast query expansion using approximations of relevance models. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 1573–1576. ACM, 2010. doi:10.1145/1871437.1871675.

[61] Liren Chen and Katia P. Sycara. Webmate: A personal agent for browsing and searching. In Katia P. Sycara and Michael J. Wooldridge, editors, *Proceedings of the Second International Conference on Autonomous Agents, AGENTS 1998, St. Paul, Minnepolis, USA, May 9-13, 1998*, pages 132–139. ACM, 1998. doi:10.1145/280765.280789.

[62] Paul-Alexandru Chirita, Claudiu S. Firan, and Wolfgang Nejdl. Personalized query expansion for the web. In Wessel Kraaij, Arjen P. de Vries, Charles L. A. Clarke, Norbert Fuhr, and Noriko Kando, editors, *SIGIR 2007: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, July 23-27, 2007*, pages 7–14. ACM, 2007. doi:10.1145/1277741.1277746.

[63] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 577–585, 2015.

[64] Cyril Cleverdon. The cranfield tests on index language devices. In *Aslib proceedings*. MCB UP Ltd, 1967.

[65] James R. Clough, Jamie Gollings, Tamar V. Loach, and Tim S. Evans. Transitive reduction of citation networks. *Journal of Complex Networks*, 3(2):189–203, 09 2014. ISSN 2051-1310. doi:10.1093/comnet/cnu039.

[66] Kevyn Collins-Thompson. Reducing the risk of query expansion via robust constrained optimization. In David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy Lin, editors, *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, pages 837–846. ACM, 2009. doi:10.1145/1645953.1646059.

[67] Kevyn Collins-Thompson and Jamie Callan. Estimation and use of uncertainty in pseudo-relevance feedback. In *SIGIR 2007: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands, July 23-27, 2007*, pages 303–310. ACM, 2007. doi:10.1145/1277741.1277795.

[68] Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *SIGIR*, pages 758–759. ACM, 2009.

[69] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, 1995. doi:10.1007/BF00994018.

[70] W Bruce Croft. Combining approaches to information retrieval. In *Advances in information retrieval*, pages 1–36. Springer, 2002.

[71] Adriel Dean-Hall, Charles L. A. Clarke, Jaap Kamps, Paul Thomas, and Ellen M. Voorhees. Overview of the TREC 2012 contextual suggestion track. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of The Twenty-First Text REtrieval Conference, TREC 2012, Gaithersburg, Maryland, USA, November 6-9, 2012*, volume 500-298 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2012.

[72] Adriel Dean-Hall, Charles L. A. Clarke, Nicole Simone, Jaap Kamps, Paul Thomas, and Ellen M. Voorhees. Overview of the TREC 2013 contextual suggestion track. In Ellen M. Voorhees, editor, *Proceedings of The Twenty-Second Text REtrieval Conference, TREC 2013, Gaithersburg, Maryland, USA, November 19-22, 2013*, volume 500-302 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2013.

[73] Adriel Dean-Hall, Charles L. A. Clarke, Jaap Kamps, Paul Thomas, and Ellen M. Voorhees. Overview of the TREC 2014 contextual suggestion track. In Ellen M. Voorhees and Angela Ellis, editors, *Proceedings of The Twenty-Third Text REtrieval Conference, TREC 2014, Gaithersburg, Maryland, USA, November 19-21, 2014*,

volume 500-308 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2014.

[74] Adriel Dean-Hall, Charles L. A. Clarke, Jaap Kamps, Julia Kiseleva, and Ellen M. Voorhees. Overview of the TREC 2015 contextual suggestion track. In Ellen M. Voorhees and Angela Ellis, editors, *Proceedings of The Twenty-Fourth Text REtrieval Conference, TREC 2015, Gaithersburg, Maryland, USA, November 17-20, 2015*, volume 500-319 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2015.

[75] Chenlong Deng, Yujia Zhou, and Zhicheng Dou. Improving personalized search with dual-feedback network. In K. Selcuk Candan, Huan Liu, Leman Akoglu, Xin Luna Dong, and Jiliang Tang, editors, *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, pages 210–218. ACM, 2022. doi:10.1145/3488560.3498447.

[76] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi:10.18653/v1/n19-1423.

[77] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. Query expansion with locally-trained word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016. doi:10.18653/v1/p16-1035.

[78] Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. A large-scale evaluation and analysis of personalized search strategies. In Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy, editors, *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 581–590. ACM, 2007. doi:10.1145/1242572.1242651.

[79] Huizhong Duan and ChengXiang Zhai. Mining coordinated intent representation for entity search and recommendation. In James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis, and Jeffrey Xu Yu, editors, *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 333–342. ACM, 2015. doi:10.1145/2806416.2806557.

[80] Huizhong Duan, ChengXiang Zhai, Jinxing Cheng, and Abhishek Gattani. A probabilistic mixture model for mining and analyzing product search log. In

Qi He, Arun Iyengar, Wolfgang Nejdl, Jian Pei, and Rajeev Rastogi, editors, *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 2179–2188. ACM, 2013. doi:10.1145/2505515.2505578.

[81] Huizhong Duan, ChengXiang Zhai, Jinxing Cheng, and Abhishek Gattani. Supporting keyword search in product database: A probabilistic approach. *Proc. VLDB Endow.*, 6(14):1786–1797, 2013. doi:10.14778/2556549.2556562.

[82] Susan T. Dumais, Edward Cutrell, Jonathan J. Cadiz, Gavin Jancke, Raman Sarin, and Daniel C. Robbins. Stuff i've seen: a system for personal information retrieval and re-use. In Charles L. A. Clarke, Gordon V. Cormack, Jamie Callan, David Hawking, and Alan F. Smeaton, editors, *SIGIR 2003: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 28 - August 1, 2003, Toronto, Canada*, pages 72–79. ACM, 2003. doi:10.1145/860435.860451.

[83] Carsten Eickhoff, Kevyn Collins-Thompson, Paul N. Bennett, and Susan T. Dumais. Personalizing atypical web search sessions. In Stefano Leonardi, Alessandro Panconesi, Paolo Ferragina, and Aristides Gionis, editors, *Sixth ACM International Conference on Web Search and Data Mining, WSDM 2013, Rome, Italy, February 4-8, 2013*, pages 285–294. ACM, 2013. doi:10.1145/2433396.2433434.

[84] William Falcon et al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3:6, 2019.

[85] Lu Fan, Qimai Li, Bo Liu, Xiao-Ming Wu, Xiaotong Zhang, Fuyu Lv, Guli Lin, Sen Li, Taiwei Jin, and Keping Yang. Modeling user behavior with graph convolution for personalized product search. In Frédérique Laforest, Raphaël Troncy, Elena Simperl, Deepak Agarwal, Aristides Gionis, Ivan Herman, and Lionel Médini, editors, *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 203–212. ACM, 2022. doi:10.1145/3485447.3511949.

[86] Michael Färber. The microsoft academic knowledge graph: A linked data source with 8 billion triples of scholarly data. In Chiara Ghidini, Olaf Hartig, Maria Maleshkova, Vojtech Svátek, Isabel F. Cruz, Aidan Hogan, Jie Song, Maxime Lefrançois, and Fabien Gandon, editors, *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II*, volume 11779 of *Lecture Notes in Computer Science*, pages 113–129. Springer, 2019. doi:10.1007/978-3-030-30796-7_8.

[87] John R Firth. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*, 1957.

[88] Edward A. Fox and Joseph A. Shaw. Combination of multiple searches. In *TREC*, volume 500-215 of *NIST Special Publication*, pages 243–252. National Institute of Standards and Technology (NIST), 1993.

[89] Norbert Fuhr. Some common mistakes in IR evaluation, and how they can be avoided. *SIGIR Forum*, 51(3):32–41, 2017. doi:10.1145/3190580.3190586.

[90] George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. The vocabulary problem in human-system communication. *Commun. ACM*, 30(11):964–971, 1987. doi:10.1145/32206.32212.

[91] Debasis Ganguly, Johannes Leveling, and Gareth J. F. Jones. Overview of the personalized and collaborative information retrieval (PIR) track at FIRE-2011. In Prasenjit Majumder, Mandar Mitra, Pushpak Bhattacharyya, L. Venkata Subramaniam, Danish Contractor, and Paolo Rosso, editors, *Multilingual Information Access in South Asian Languages - Second International Workshop, FIRE 2010, Gandhinagar, India, February 19-21, 2010 and Third International Workshop, FIRE 2011, Bombay, India, December 2-4, 2011, Revised Selected Papers*, volume 7536 of *Lecture Notes in Computer Science*, pages 227–240. Springer, 2011. doi:10.1007/978-3-642-40087-2_22.

[92] Luyu Gao, Zhuyun Dai, Zhen Fan, and Jamie Callan. Complementing lexical retrieval with semantic residual embedding. *CoRR*, abs/2004.13969, 2020.

[93] Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. Complement lexical retrieval model with semantic residual embeddings. In Djoerd Hiemstra, Marie-Francine Moens, Josiane Mothe, Raffaele Perego, Martin Potthast, and Fabrizio Sebastiani, editors, *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part I*, volume 12656 of *Lecture Notes in Computer Science*, pages 146–160. Springer, 2021. doi:10.1007/978-3-030-72113-8_10.

[94] Susan Gauch, Mirco Speretta, Aravind Chandramouli, and Alessandro Micarelli. User profiles for personalized information access. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*, pages 54–89. Springer, 2007. doi:10.1007/978-3-540-72079-9_2.

[95] Songwei Ge, Zhicheng Dou, Zhengbao Jiang, Jian-Yun Nie, and Ji-Rong Wen. Personalizing search results using hierarchical RNN with query-aware attention. In Alfredo Cuzzocrea, James Allan, Norman W. Paton, Divesh Srivastava, Rakesh Agrawal, Andrei Z. Broder, Mohammed J. Zaki, K. Selçuk Candan, Alexandros Labrinidis, Assaf Schuster, and Haixun Wang, editors, *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 347–356. ACM, 2018. doi:10.1145/3269206.3271728.

[96] M. Rami Ghorab, Dong Zhou, Alexander O'Connor, and Vincent Wade. Personalised information retrieval: survey and classification. *User Model. User Adapt. Interact.*, 23(4):381–443, 2013. doi:10.1007/s11257-012-9124-1.

[97] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[98] Clinton Gormley and Zachary Tong. *Elasticsearch: the definitive guide: a distributed real-time search and analytics engine*. " O'Reilly Media, Inc.", 2015.

[99] Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge distillation: A survey. *Int. J. Comput. Vis.*, 129(6):1789–1819, 2021. doi:10.1007/s11263-021-01453-z.

[100] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *CoRR*, abs/1410.5401, 2014.

[101] Mihajlo Grbovic. Search ranking and personalization at airbnb. In Paolo Cremonesi, Francesco Ricci, Shlomo Berkovsky, and Alexander Tuzhilin, editors, *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*, pages 339–340. ACM, 2017. doi:10.1145/3109859.3109920.

[102] Mihajlo Grbovic and Haibin Cheng. Real-time personalization using embeddings for search ranking at airbnb. In Yike Guo and Faisal Farooq, editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 311–320. ACM, 2018. doi:10.1145/3219819.3219885.

[103] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 855–864. ACM, 2016. doi:10.1145/2939672.2939754.

[104] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. A deep relevance matching model for ad-hoc retrieval. In Snehasis Mukhopadhyay, ChengXiang Zhai, Elisa Bertino, Fabio Crestani, Javed Mostafa, Jie Tang, Luo Si, Xiaofang Zhou, Yi Chang, Yunyao Li, and Parikshit Sondhi, editors, *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pages 55–64. ACM, 2016. doi:10.1145/2983323.2983769.

[105] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W. Bruce Croft, and Xueqi Cheng. A deep look into neural ranking models for information retrieval. *Inf. Process. Manag.*, 57(6):102067, 2020. doi:10.1016/j.ipm.2019.102067.

[106] Yangyang Guo, Zhiyong Cheng, Liqiang Nie, Xin-Shun Xu, and Mohan S. Kankanhalli. Multi-modal preference modeling for product search. In Susanne Boll, Kyoung Mu Lee, Jiebo Luo, Wenwu Zhu, Hyeran Byun, Chang Wen Chen,

Rainer Lienhart, and Tao Mei, editors, *2018 ACM Multimedia Conference on Multimedia Conference, MM 2018, Seoul, Republic of Korea, October 22-26, 2018*, pages 1865–1873. ACM, 2018. doi:10.1145/3240508.3240541.

[107] Yangyang Guo, Zhiyong Cheng, Liqiang Nie, Yinglong Wang, Jun Ma, and Mohan S. Kankanhalli. Attentive long short-term preference modeling for personalized product search. *ACM Trans. Inf. Syst.*, 37(2):19:1–19:27, 2019. doi:10.1145/3295822.

[108] Christophe Van Gysel and Maarten de Rijke. Pytrec_eval: An extremely fast python interface to trec_eval. In *SIGIR*, pages 873–876. ACM, 2018.

[109] Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. Learning latent vector spaces for product search. In Snehasis Mukhopadhyay, ChengXiang Zhai, Elisa Bertino, Fabio Crestani, Javed Mostafa, Jie Tang, Luo Si, Xiaofang Zhou, Yi Chang, Yunyao Li, and Parikshit Sondhi, editors, *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pages 165–174. ACM, 2016. doi:10.1145/2983323.2983702.

[110] Gyeong June Hahm, Mun Yong Yi, Jae-Hyun Lee, and Hyo-Won Suh. A personalized query expansion approach for engineering document retrieval. *Adv. Eng. Informatics*, 28(4):344–359, 2014. doi:10.1016/j.aei.2014.04.002.

[111] Aniko Hannak, Piotr Sapiezynski, Arash Molavi Kakhki, Balachander Krishnamurthy, David Lazer, Alan Mislove, and Christo Wilson. Measuring personalization of web search. In Daniel Schwabe, Virgílio A. F. Almeida, Hartmut Glaser, Ricardo Baeza-Yates, and Sue B. Moon, editors, *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 527–538. International World Wide Web Conferences Steering Committee / ACM, 2013. doi:10.1145/2488388.2488435.

[112] Donna Harman. Relevance feedback revisited. In Nicholas J. Belkin, Peter Ingwersen, and Annelise Mark Pejtersen, editors, *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Copenhagen, Denmark, June 21-24, 1992*, pages 1–10. ACM, 1992. doi:10.1145/133160.133167.

[113] Donna Harman. *Information Retrieval Evaluation*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2011.

[114] Charles R. Harris, K. Jarrod Millman, Stéfan van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy,

Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with numpy. *Nat.*, 585:357–362, 2020.

[115] Zellig S Harris. Distributional structure. *Word*, 1954.

[116] Morgan Harvey, Fabio Crestani, and Mark James Carman. Building user profiles from topic models for personalised search. In Qi He, Arun Iyengar, Wolfgang Nejdl, Jian Pei, and Rajeev Rastogi, editors, *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 2309–2314. ACM, 2013. doi:10.1145/2505515.2505642.

[117] Seyyed Hadi Hashemi, Jaap Kamps, Julia Kiseleva, Charles L. A. Clarke, and Ellen M. Voorhees. Overview of the TREC 2016 contextual suggestion track. In Ellen M. Voorhees and Angela Ellis, editors, *Proceedings of The Twenty-Fifth Text REtrieval Conference, TREC 2016, Gaithersburg, Maryland, USA, November 15-18, 2016*, volume 500-321 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2016.

[118] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

[119] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997. doi:10.1162/neco.1997.9.8.1735.

[120] Kurt Hornik, Maxwell B. Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. doi:10.1016/0893-6080(89)90020-8.

[121] D. Frank Hsu and Isak Taksa. Comparing rank and score combination methods for data fusion in information retrieval. *Inf. Retr.*, 8(3):449–480, 2005. doi:10.1007/s10791-005-6994-4.

[122] Yujing Hu, Qing Da, Anxiang Zeng, Yang Yu, and Yinghui Xu. Reinforcement learning to rank in e-commerce search engine: Formalization, analysis, and application. In Yike Guo and Faisal Farooq, editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 368–377. ACM, 2018. doi:10.1145/3219819.3219846.

[123] Nasreen Abdul Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah S. Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. Umass at TREC 2004: Novelty and HARD. In *Proceedings of the Thirteenth Text REtrieval Conference, TREC 2004, Gaithersburg, Maryland, USA, November 16-19, 2004*, volume 500-261 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2004.

[124] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. Real life, real users, and real needs: a study and analysis of user queries on the web. *Inf. Process. Manag.*, 36(2):207–227, 2000. doi:10.1016/S0306-4573(99)00056-4.

[125] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[126] Jyun-Yu Jiang, Tao Wu, Georgios Roumpos, Heng-Tze Cheng, Xinyang Yi, Ed Chi, Harish Ganapathy, Nitin Jindal, Pei Cao, and Wei Wang. End-to-end deep attentive personalized item retrieval for online content-sharing platforms. In Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen, editors, *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2870–2877. ACM / IW3C2, 2020. doi:10.1145/3366423.3380051.

[127] Xu Jianmin and Liu Chang. Personalized query expansion based on user interest and domain knowledge. In *2012 Third Global Congress on Intelligent Systems*, pages 394–399. IEEE, 2012.

[128] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling BERT for natural language understanding. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4163–4174. Association for Computational Linguistics, 2020. doi:10.18653/v1/2020.findings-emnlp.372.

[129] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972.

[130] Evangelos Kanoulas, Paul D. Clough, Ben Carterette, and Mark Sanderson. Overview of the TREC 2010 session track. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of The Nineteenth Text REtrieval Conference, TREC 2010, Gaithersburg, Maryland, USA, November 16-19, 2010*, volume 500-294 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2010.

[131] Evangelos Kanoulas, Mark M. Hall, Paul D. Clough, Ben Carterette, and Mark Sanderson. Overview of the TREC 2011 session track. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of The Twentieth Text REtrieval Conference, TREC 2011, Gaithersburg, Maryland, USA, November 15-18, 2011*, volume 500-296 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2011.

[132] Evangelos Kanoulas, Ben Carterette, Mark M. Hall, Paul D. Clough, and Mark Sanderson. Overview of the TREC 2012 session track. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of The Twenty-First Text REtrieval Conference, TREC 2012, Gaithersburg, Maryland, USA, November 6-9, 2012*, volume 500-298 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2012.

[133] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics, 2020. doi:10.18653/v1/2020.emnlp-main.550.

[134] Diane Kelly and Jaime Teevan. Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum*, 37(2):18–28, 2003. doi:10.1145/959258.959260.

[135] Hamid Khalifi, Walid Cherif, Abderrahim El Qadi, and Youssef Ghanou. Query expansion based on clustering and personalized information retrieval. *Prog. Artif. Intell.*, 8(2):241–251, 2019. doi:10.1007/s13748-019-00178-y.

[136] Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 39–48. ACM, 2020. doi:10.1145/3397271.3401075.

[137] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[138] Zornitsa Kozareva, Qi Li, Ke Zhai, and Weiwei Guo. Recognizing salient entities in shopping queries. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics, 2016. doi:10.18653/v1/p16-2018.

[139] Robert Krovetz. Viewing morphology as an inference process. In Robert Korfhage, Edie M. Rasmussen, and Peter Willett, editors, *Proceedings of the 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Pittsburgh, PA, USA, June 27 - July 1, 1993*, pages 191–202. ACM, 1993. doi:10.1145/160688.160718.

[140] John Krumm, Nigel Davies, and Chandra Narayanaswami. User-generated content. *IEEE Pervasive Comput.*, 7(4):10–11, 2008. doi:10.1109/MPRV.2008.85.

[141] Oren Kurland and J. Shane Culpepper. Fusion in information retrieval: SIGIR 2018 half-day tutorial. In Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz, editors, *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1383–1386. ACM, 2018. doi:10.1145/3209978.3210186.

[142] Saar Kuzi, Anna Shtok, and Oren Kurland. Query expansion using word embeddings. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pages 1929–1932. ACM, 2016. doi:10.1145/2983323.2983876.

[143] Saar Kuzi, David Carmel, Alex Libov, and Ariel Raviv. Query expansion for email search. In Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White, editors, *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 849–852. ACM, 2017. doi:10.1145/3077136.3080660.

[144] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Carla E. Brodley and Andrea Pohoreckyj Danyluk, editors, *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann, 2001.

[145] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: a llvm-based python JIT compiler. In *LLVM@SC*, pages 7:1–7:6. ACM, 2015.

[146] Victor Lavrenko and W. Bruce Croft. Relevance-based language models. In *SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, September 9-13, 2001, New Orleans, Louisiana, USA*, pages 120–127. ACM, 2001. doi:10.1145/383952.383972.

[147] Quoc V. Le and Tomás Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1188–1196. JMLR.org, 2014.

[148] Joon Ho Lee. Analyses of multiple evidence combination. In *SIGIR*, pages 267–276. ACM, 1997.

[149] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 6086–6096. Association for Computational Linguistics, 2019. doi:10.18653/v1/p19-1612.

[150] Kyung-Soon Lee, W. Bruce Croft, and James Allan. A cluster-based resampling method for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, pages 235–242. ACM, 2008. doi:10.1145/1390334.1390376.

[151] Xiao Li. Understanding the semantic structure of noun phrase queries. In Jan Hajic, Sandra Carberry, and Stephen Clark, editors, *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 1337–1345. The Association for Computer Linguistics, 2010.

[152] Xiao Li, Ye-Yi Wang, and Alex Acero. Extracting structured information from user queries with semi-supervised conditional random fields. In James Allan, Javed A. Aslam, Mark Sanderson, ChengXiang Zhai, and Justin Zobel, editors, *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, pages 572–579. ACM, 2009. doi:10.1145/1571941.1572039.

[153] Xiujun Li, Chenlei Guo, Wei Chu, Ye-Yi Wang, and Jude Shavlik. Deep learning powered in-session contextual ranking using clickthrough data. In *In Proc. of NIPS*, 2014.

[154] David Lillis, Fergus Toolan, Rem W. Collier, and John Dunnion. Probfuse: a probabilistic approach to data fusion. In Efthimis N. Efthimiadis, Susan T. Dumais, David Hawking, and Kalervo Järvelin, editors, *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6-11, 2006*, pages 139–146. ACM, 2006. doi:10.1145/1148170.1148197.

[155] David Lillis, Fergus Toolan, Rem W. Collier, and John Dunnion. Extending probabilistic data fusion using sliding windows. In Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven, and Ryen W. White, editors, *Advances in Information Retrieval , 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings*, volume 4956 of *Lecture Notes in Computer Science*, pages 358–369. Springer, 2008. doi:10.1007/978-3-540-78646-7_33.

[156] David Lillis, Lusheng Zhang, Fergus Toolan, Rem W. Collier, David Leonard, and John Dunnion. Estimating probabilities for effective data fusion. In Fabio Crestani, Stéphane Marchand-Maillet, Hsin-Hsi Chen, Efthimis N. Efthimiadis, and Jacques Savoy, editors, *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, Geneva, Switzerland, July 19-23, 2010*, pages 347–354. ACM, 2010. doi:10.1145/1835449.1835508.

[157] Soon Chong Johnson Lim, Ying Liu, and Wing Bun Lee. Multi-facet product information search and retrieval using semantically annotated product family ontology. *Inf. Process. Manag.*, 46(4):479–493, 2010. doi:10.1016/j.ipm.2009.09.001.

[158] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. *Pretrained Transformers for Text Ranking: BERT and Beyond*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2021. doi:10.2200/S01123ED1V01Y202108HLT053.

[159] Yiu-Chang Lin, Ankur Datta, and Giuseppe Di Fabbrizio. E-commerce product query classification using implicit user's feedback from clicks. In Naoki Abe, Huan Liu, Calton Pu, Xiaohua Hu, Nesreen K. Ahmed, Mu Qiao, Yang Song, Donald Kossmann, Bing Liu, Kisung Lee, Jiliang Tang, Jingrui He, and Jeffrey S. Saltz, editors, *IEEE International Conference on Big Data, Big Data 2018, Seattle, WA, USA, December 10-13, 2018*, pages 1955–1959. IEEE, 2018. doi:10.1109/BigData.2018.8622008.

[160] Jingjing Liu, Xiao Li, Alex Acero, and Ye-Yi Wang. Lexicon modeling for query understanding. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011, May 22-27, 2011, Prague Congress Center, Prague, Czech Republic*, pages 5604–5607. IEEE, 2011. doi:10.1109/ICASSP.2011.5947630.

[161] Jingjing Liu, Chang Liu, and Nicholas J. Belkin. Personalization in text information retrieval: A survey. *J. Assoc. Inf. Sci. Technol.*, 71(3):349–369, 2020. doi:10.1002/asi.24234.

[162] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017.

[163] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[164] Lu Lu, Yeonjong Shin, Yanhui Su, and George E. Karniadakis. Dying relu and initialization: Theory and numerical examples. *CoRR*, abs/1903.06733, 2019.

[165] Shuqi Lu, Zhicheng Dou, Xu Jun, Jian-Yun Nie, and Ji-Rong Wen. PSGAN: A minimax game for personalized search with limited and noisy click data. In Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer, editors, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 555–564. ACM, 2019. doi:10.1145/3331184.3331218.

[166] Shuqi Lu, Zhicheng Dou, Chenyan Xiong, Xiaojie Wang, and Ji-Rong Wen. Knowledge enhanced personalized search. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 709–718. ACM, 2020. doi:10.1145/3397271.3401089.

[167] Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. Sparse, dense, and attentional representations for text retrieval. *Trans. Assoc. Comput. Linguistics*, 9:329–345, 2021.

[168] Claudio Lucchese, Cristina Ioana Muntean, Franco Maria Nardini, Raffaele Perego, and Salvatore Trani. Rankeval: An evaluation and analysis framework for learning-to-rank solutions. In *SIGIR*, pages 1281–1284. ACM, 2017.

[169] Claudio Lucchese, Cristina Ioana Muntean, Franco Maria Nardini, Raffaele Perego, and Salvatore Trani. Rankeval: Evaluation and investigation of ranking models. *SoftwareX*, 12:100614, 2020.

[170] Robert Duncan Luce. *Individual Choice Behavior: A Theoretical Analysis*. Wiley, 1959.

[171] Hans Peter Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317, 1957.

[172] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. The Association for Computational Linguistics, 2015. doi:10.18653/v1/d15-1166.

[173] Yuanhua Lv and ChengXiang Zhai. Positional relevance model for pseudo-relevance feedback. In *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, Geneva, Switzerland, July 19-23, 2010*, pages 579–586. ACM, 2010. doi:10.1145/1835449.1835546.

[174] Ji Ma, Ivan Korotkov, Yinfei Yang, Keith Hall, and Ryan McDonald. Zero-shot neural retrieval via domain-targeted synthetic query generation. *arXiv preprint arXiv:2004.14503*, 2020.

[175] Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. Expansion via prediction of importance with contextualization. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1573–1576. ACM, 2020. doi:10.1145/3397271.3401262.

[176] Sean MacAvaney, Andrew Yates, Sergey Feldman, Doug Downey, Arman Cohan, and Nazli Goharian. Simplified data wrangling with ir_datasets. In *SIGIR*, pages 2429–2436. ACM, 2021.

[177] Sean MacAvaney, Craig Macdonald, and Iadh Ounis. Reproducing personalised session search over the AOL query log. In Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty, editors, *Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part*

I, volume 13185 of *Lecture Notes in Computer Science*, pages 627–640. Springer, 2022. doi:10.1007/978-3-030-99736-6_42.

[178] Craig Macdonald and Nicola Tonellotto. Declarative experimentation in information retrieval using pyterrier. In *ICTIR*, pages 161–168. ACM, 2020.

[179] Craig Macdonald, Nicola Tonellotto, Sean MacAvaney, and Iadh Ounis. Pyterrier: Declarative experimentation in python from BM25 to dense retrieval. In *CIKM*, pages 4526–4533. ACM, 2021.

[180] J. MacQueen. Some methods for classification and analysis of multivariate observations. 1967.

[181] Saurav Manchanda, Mohit Sharma, and George Karypis. Intent term weighting in e-commerce queries. In Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu, editors, *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 2345–2348. ACM, 2019. doi:10.1145/3357384.3358151.

[182] Saurav Manchanda, Mohit Sharma, and George Karypis. Intent term selection and refinement in e-commerce queries. *CoRR*, abs/1908.08564, 2019.

[183] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008. ISBN 978-0-521-86571-5. doi:10.1017/CBO9780511809071.

[184] Mehdi Manshadi and Xiao Li. Semantic tagging of web search queries. In Keh-Yih Su, Jian Su, and Janyce Wiebe, editors, *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, pages 861–869. The Association for Computer Linguistics, 2009.

[185] Nicolaas Matthijs and Filip Radlinski. Personalizing web search using long term browsing history. In Irwin King, Wolfgang Nejdl, and Hang Li, editors, *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*, pages 25–34. ACM, 2011. doi:10.1145/1935826.1935840.

[186] Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In Ricardo Baeza-Yates, Mounia Lalmas, Alistair Moffat, and Berthier A. Ribeiro-Neto, editors, *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*, pages 43–52. ACM, 2015. doi:10.1145/2766462.2767755.

[187] Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.*, 2(11):205, 2017. doi:10.21105/joss.00205.

[188] Wes McKinney et al. pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, 14(9):1–9, 2011.

[189] Alessandro Micarelli and Filippo Sciarrone. Anatomy and empirical evaluation of an adaptive web-based information filtering system. *User Model. User Adapt. Interact.*, 14(2-3):159–200, 2004. doi:10.1023/B:USER.0000028981.43614.94.

[190] Alessandro Micarelli, Fabio Gasparetti, Filippo Sciarrone, and Susan Gauch. Personalized search on the world wide web. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*, pages 195–230. Springer, 2007. doi:10.1007/978-3-540-72079-9_6.

[191] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[192] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013.

[193] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38 (11):39–41, 1995. doi:10.1145/219717.219748.

[194] Alistair Moffat and Justin Zobel. Rank-biased precision for measurement of retrieval effectiveness. *ACM Trans. Inf. Syst.*, 27(1):2:1–2:27, 2008. doi:10.1145/1416950.1416952.

[195] Mark H. Montague and Javed A. Aslam. Relevance score normalization for metasearch. In *Proceedings of the 2001 ACM CIKM International Conference on Information and Knowledge Management, Atlanta, Georgia, USA, November 5-10, 2001*, pages 427–433. ACM, 2001. doi:10.1145/502585.502657.

[196] Mark H. Montague and Javed A. Aslam. Condorcet fusion for improved retrieval. In *Proceedings of the 2002 ACM CIKM International Conference on Information and Knowledge Management, McLean, VA, USA, November 4-9, 2002*, pages 538–548. ACM, 2002. doi:10.1145/584792.584881.

[197] André Mourão, Flávio Martins, and João Magalhães. Multimodal medical information retrieval with unsupervised rank fusion. *Comput. Medical Imaging Graph.*, 39:35–45, 2015. doi:10.1016/j.compmedimag.2014.05.006.

[198] Philippe Mulhem, Nawal Ould Amer, and Mathias Géry. Axiomatic term-based personalized query expansion using bookmarking system. In Sven Hartmann and Hui Ma, editors, *Database and Expert Systems Applications - 27th International Conference, DEXA 2016, Porto, Portugal, September 5-8, 2016, Proceedings, Part II*, volume 9828 of *Lecture Notes in Computer Science*, pages 235–243. Springer, 2016. doi:10.1007/978-3-319-44406-2_17.

[199] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814. Omnipress, 2010.

[200] Shahrzad Naseri, Jeff Dalton, Andrew Yates, and James Allan. CEQE: contextualized embeddings for query expansion. In Djoerd Hiemstra, Marie-Francine Moens, Josiane Mothe, Raffaele Perego, Martin Potthast, and Fabrizio Sebastiani, editors, *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part I*, volume 12656 of *Lecture Notes in Computer Science*, pages 467–482. Springer, 2021. doi:10.1007/978-3-030-72113-8_31.

[201] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. In Tarek Richard Besold, Antoine Bordes, Artur S. d'Avila Garcez, and Greg Wayne, editors, *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016.

[202] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *CoRR*, abs/1901.04085, 2019.

[203] Michael G. Noll and Christoph Meinel. Web search personalization via social bookmarking and tagging. In Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 367–380. Springer, 2007. doi:10.1007/978-3-540-76298-0_27.

[204] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[205] Travis E. Oliphant. Python for scientific computing. *Comput. Sci. Eng.*, 9(3): 10–20, 2007.

[206] Paul Over. The TREC interactive track: an annotated bibliography. *Inf. Process. Manag.*, 37(3):369–381, 2001. doi:10.1016/S0306-4573(00)00053-4.

[207] Pallavi Palleti, Harish Karnick, and Pabitra Mitra. Personalized web search using probabilistic query expansion. In *Proceedings of the 2007 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology - Workshops, 2-5 November 2007, Silicon Valley, CA, USA*, pages 83–86. IEEE Computer Society, 2007. doi:10.1109/WIIATW.2007.4427545.

[208] João R. M. Palotti, Harrisen Scells, and Guido Zuccon. Trectools: an open-source python library for information retrieval practitioners involved in trec-like campaigns. In *SIGIR*, pages 1325–1328. ACM, 2019.

[209] Yaoxin Pan, Shangsong Liang, Jiaxin Ren, Zaiqiao Meng, and Qiang Zhang. Personalized, sequential, attentive, metric-aware product search. *ACM Trans. Inf. Syst.*, 40(2):36:1–36:29, 2022. doi:10.1145/3473337.

[210] Nish Parikh and Neel Sundaresan. Beyond relevance in marketplace search. In Craig Macdonald, Iadh Ounis, and Ian Ruthven, editors, *Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, Glasgow, United Kingdom, October 24-28, 2011*, pages 2109–2112. ACM, 2011. doi:10.1145/2063576.2063902.

[211] Gabriella Pasi, Gareth J. F. Jones, Stefania Marrara, Camilla Sanvitto, Debasis Ganguly, and Procheta Sen. Overview of the CLEF 2017 personalised information retrieval pilot lab (PIR-CLEF 2017). In Gareth J. F. Jones, Séamus Lawless, Julio Gonzalo, Liadh Kelly, Lorraine Goeuriot, Thomas Mandl, Linda Cappellato, and Nicola Ferro, editors, *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 8th International Conference of the CLEF Association, CLEF 2017, Dublin, Ireland, September 11-14, 2017, Proceedings*, volume 10456 of *Lecture Notes in Computer Science*, pages 338–345. Springer, 2017. doi:10.1007/978-3-319-65813-1_29.

[212] Gabriella Pasi, Gareth J. F. Jones, Keith Curtis, Stefania Marrara, Camilla Sanvitto, Debasis Ganguly, and Procheta Sen. Overview of the CLEF 2018 personalised information retrieval lab (PIR-CLEF 2018). In Linda Cappellato, Nicola Ferro, Jian-Yun Nie, and Laure Soulier, editors, *Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum, Avignon, France, September 10-14, 2018*, volume 2125 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018.

[213] Gabriella Pasi, Gareth J. F. Jones, Lorraine Goeuriot, Liadh Kelly, Stefania Marrara, and Camilla Sanvitto. Overview of the CLEF 2019 personalised information retrieval lab (PIR-CLEF 2019). In Fabio Crestani, Martin Braschler, Jacques Savoy, Andreas Rauber, Henning Müller, David E. Losada, Gundula Heinatz Bürki, Linda Cappellato, and Nicola Ferro, editors, *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 10th International Conference of the CLEF Association, CLEF 2019, Lugano, Switzerland, September 9-12, 2019, Proceedings*, volume 11696 of *Lecture Notes in Computer Science*, pages 417–424. Springer, 2019. doi:10.1007/978-3-030-28577-7_31.

[214] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. A picture of search. In *Proceedings of the 1st International Conference on Scalable Information Systems, Infoscale 2006, Hong Kong, May 30-June 1, 2006*, volume 152 of *ACM International Conference Proceeding Series*, page 1. ACM, 2006.

[215] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019.

[216] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014. doi:10.3115/v1/d14-1162.

[217] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018. doi:10.18653/v1/n18-1202.

[218] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. *SIGIR Forum*, 51(2):202–208, 2017. doi:10.1145/3130348.3130368.

[219] Alexander Pretschner and Susan Gauch. Ontology based personalized search. In *11th IEEE International Conference on Tools with Artificial Intelligence, ICTAI '99, Chicago, Illinois, USA, November 8-10, 1999*, pages 391–398. IEEE Computer Society, 1999. doi:10.1109/TAI.1999.809829.

[220] Filip Radlinski and Susan T. Dumais. Improving personalized web search using result diversification. In Efthimis N. Efthimiadis, Susan T. Dumais, David Hawking, and Kalervo Järvelin, editors, *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6-11, 2006*, pages 691–692. ACM, 2006. doi:10.1145/1148170.1148320.

[221] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of

transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21: 140:1–140:67, 2020.

[222] Sebastian Raschka, Joshua Patterson, and Corey Nolet. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *arXiv preprint arXiv:2002.04803*, 2020.

[223] Adwait Ratnaparkh. Maximum entropy models for natural language ambiguity resolution. In *Ph.D. Dissertation in Computer and Information Science*. University of Pennsylvania, 1998.

[224] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics, 2019. doi:10.18653/v1/D19-1410.

[225] M. Elena Renda and Umberto Straccia. Web metasearch: Rank vs. score based rank aggregation methods. In Gary B. Lamont, Hisham Haddad, George A. Papadopoulos, and Brajendra Panda, editors, *Proceedings of the 2003 ACM Symposium on Applied Computing (SAC), March 9-12, 2003, Melbourne, FL, USA*, pages 841–846. ACM, 2003. doi:10.1145/952532.952698.

[226] Stephen E. Robertson and Steve Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In W. Bruce Croft and C. J. van Rijsbergen, editors, *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Dublin, Ireland, 3-6 July 1994 (Special Issue of the SIGIR Forum)*, pages 232–241. ACM/Springer, 1994. doi:10.1007/978-1-4471-2099-5_24.

[227] Stephen E. Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, 2009. doi:10.1561/1500000019.

[228] Joseph Rocchio. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*, pages 313–323, 1971.

[229] Henning Rode, Djoerd Hiemstra, Georgina Ramírez, Thijs Westerveld, and Arjen P. de Vries. The lowlands' TREC experiments 2005. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of the Fourteenth Text REtrieval Conference, TREC 2005, Gaithersburg, Maryland, USA, November 15-18, 2005*, volume 500-266 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 2005.

[230] Jennifer Rowley. Product search in e-shopping: A review and research propositions. *Journal of Consumer Marketing*, 17, 2000.

[231] Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. Using word embeddings for automatic query expansion. *CoRR*, abs/1606.07608, 2016. URL http://arxiv.org/abs/1606.07608.

[232] Magnus Sahlgren. The distributional hypothesis. *The Italian Journal of Linguistics*, 20:33–54, 2008.

[233] Gerard Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manag.*, 24(5):513–523, 1988. doi:10.1016/0306-4573(88)90021-0.

[234] Mark Sanderson. Test collection based evaluation of information retrieval systems. *Found. Trends Inf. Retr.*, 4(4):247–375, 2010.

[235] Erik F. Tjong Kim Sang and Jorn Veenstra. Representing text chunks. In *EACL 1999, 9th Conference of the European Chapter of the Association for Computational Linguistics, June 8-12, 1999, University of Bergen, Bergen, Norway*, pages 173–179. The Association for Computer Linguistics, 1999.

[236] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. On application of learning to rank for e-commerce search. In Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White, editors, *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 475–484. ACM, 2017. doi:10.1145/3077136.3080838.

[237] Sunita Sarawagi and William W. Cohen. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pages 1185–1192, 2004.

[238] Nikos Sarkas, Stelios Paparizos, and Panayiotis Tsaparas. Structured annotations of web queries. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 771–782, 2010.

[239] Sheikh Muhammad Sarwar, Md. Anowarul Abedin, A. H. M. Sofi Ullah, and Abdullah Al-Mamun. Personalized query expansion for web search using social keywords. In Edgar R. Weippl, Maria Indrawan-Santiago, Matthias Steinbauer, Gabriele Kotsis, and Ismail Khalil, editors, *The 15th International Conference on Information Integration and Web-based Applications & Services, IIWAS '13, Vienna, Austria, December 2-4, 2013*, page 610. ACM, 2013. doi:10.1145/2539150.2539266.

[240] Xuehua Shen, Bin Tan, and ChengXiang Zhai. Implicit user modeling for personalized search. In Otthein Herzog, Hans-Jörg Schek, Norbert Fuhr, Abdur Chowdhury, and Wilfried Teiken, editors, *Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management, Bremen, Germany, October 31 - November 5, 2005*, pages 824–831. ACM, 2005. doi:10.1145/1099554.1099747.

[241] Milad Shokouhi. Segmentation of search engine results for effective data-fusion. In Giambattista Amati, Claudio Carpineto, and Giovanni Romano, editors, *Advances in Information Retrieval, 29th European Conference on IR Research, ECIR 2007, Rome, Italy, April 2-5, 2007, Proceedings*, volume 4425 of *Lecture Notes in Computer Science*, pages 185–197. Springer, 2007. doi:10.1007/978-3-540-71496-5_19.

[242] Ahu Sieg, Bamshad Mobasher, and Robin D. Burke. Web search personalization with ontological user profiles. In Mário J. Silva, Alberto H. F. Laender, Ricardo A. Baeza-Yates, Deborah L. McGuinness, Bjørn Olstad, Øystein Haug Olsen, and André O. Falcão, editors, *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal, November 6-10, 2007*, pages 525–534. ACM, 2007. doi:10.1145/1321440.1321515.

[243] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan Wang. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, page 243–246, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450334730. doi:10.1145/2740908.2742839.

[244] Mark D. Smucker, James Allan, and Ben Carterette. A comparison of statistical significance tests for information retrieval evaluation. In Mário J. Silva, Alberto H. F. Laender, Ricardo A. Baeza-Yates, Deborah L. McGuinness, Bjørn Olstad, Øystein Haug Olsen, and André O. Falcão, editors, *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal, November 6-10, 2007*, pages 623–632. ACM, 2007. doi:10.1145/1321440.1321528.

[245] Barry Smyth and Evelyn Balfe. Anonymous personalization in collaborative web search. *Inf. Retr.*, 9(2):165–190, 2006. doi:10.1007/s10791-006-7148-z.

[246] Parikshit Sondhi, Mohit Sharma, Pranam Kolari, and ChengXiang Zhai. A taxonomy of queries for e-commerce search. In Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz, editors, *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1245–1248. ACM, 2018. doi:10.1145/3209978.3210152.

[247] Yang Song, Hongning Wang, and Xiaodong He. Adapting deep ranknet for personalized search. In Ben Carterette, Fernando Diaz, Carlos Castillo, and Donald Metzler, editors, *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*, pages 83–92. ACM, 2014. doi:10.1145/2556195.2556234.

[248] David A. Sontag, Kevyn Collins-Thompson, Paul N. Bennett, Ryen W. White, Susan T. Dumais, and Bodo Billerbeck. Probabilistic models for personalizing web search. In Eytan Adar, Jaime Teevan, Eugene Agichtein, and Yoelle Maarek,

editors, *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*, pages 433–442. ACM, 2012. doi:10.1145/2124295.2124348.

[249] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In James Bailey, Alistair Moffat, Charu C. Aggarwal, Maarten de Rijke, Ravi Kumar, Vanessa Murdock, Timos K. Sellis, and Jeffrey Xu Yu, editors, *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 553–562. ACM, 2015. doi:10.1145/2806416.2806493.

[250] Mirco Speretta and Susan Gauch. Personalized search based on user search histories. In Andrzej Skowron, Rakesh Agrawal, Michael Luck, Takahira Yamaguchi, Pierre Morizet-Mahoudeaux, Jiming Liu, and Ning Zhong, editors, *2005 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2005), 19-22 September 2005, Compiegne, France*, pages 622–628. IEEE Computer Society, 2005. doi:10.1109/WI.2005.114.

[251] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

[252] Sofia Stamou and Alexandros Ntoulas. Search personalization through query and page topical analysis. *User Model. User Adapt. Interact.*, 19(1-2):5–33, 2009. doi:10.1007/s11257-008-9056-y.

[253] Statista. Number of digital buyers worldwide from 2014 to 2021, 2020.

[254] Statista. Retail e-commerce sales worldwide from 2014 to 2024, 2021.

[255] Statista. E-commerce share of total global retail sales from 2015 to 2024, 2021.

[256] Ning Su, Jiyin He, Yiqun Liu, Min Zhang, and Shaoping Ma. User intent, behaviour, and perceived satisfaction in product search. In Yi Chang, Chengxiang Zhai, Yan Liu, and Yoelle Maarek, editors, *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, pages 547–555. ACM, 2018. doi:10.1145/3159652.3159714.

[257] Shayan A. Tabrizi, Azadeh Shakery, Hamed Zamani, and Mohammad Ali Tavallaei. PERSON: personalized information retrieval evaluation based on citation networks. *Inf. Process. Manag.*, 54(4):630–656, 2018. doi:10.1016/j.ipm.2018.04.004.

[258] Bin Tan, Xuehua Shen, and ChengXiang Zhai. Mining long-term search history to improve search accuracy. In Tina Eliassi-Rad, Lyle H. Ungar,

Mark Craven, and Dimitrios Gunopulos, editors, *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 718–723. ACM, 2006. doi:10.1145/1150402.1150493.

[259] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In Ying Li, Bing Liu, and Sunita Sarawagi, editors, *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*, pages 990–998. ACM, 2008. doi:10.1145/1401890.1402008.

[260] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Personalizing search via automated analysis of interests and activities. In Ricardo A. Baeza-Yates, Nivio Ziviani, Gary Marchionini, Alistair Moffat, and John Tait, editors, *SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, August 15-19, 2005*, pages 449–456. ACM, 2005. doi:10.1145/1076034.1076111.

[261] Jaime Teevan, Susan T. Dumais, and Daniel J. Liebling. To personalize or not to personalize: modeling queries with variation in user intent. In Sung-Hyon Myaeng, Douglas W. Oard, Fabrizio Sebastiani, Tat-Seng Chua, and Mun-Kew Leong, editors, *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, pages 163–170. ACM, 2008. doi:10.1145/1390334.1390364.

[262] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Potential for personalization. *ACM Trans. Comput. Hum. Interact.*, 17(1):4:1–4:31, 2010. doi:10.1145/1721831.1721835.

[263] Jaime Teevan, Daniel J. Liebling, and Gayathri Ravichandran Geetha. Understanding and predicting personal navigation. In Irwin King, Wolfgang Nejdl, and Hang Li, editors, *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*, pages 85–94. ACM, 2011. doi:10.1145/1935826.1935848.

[264] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021.

[265] Robert Tibshirani, Trevor Hastie, Balasubramanian Narasimhan, and Gilbert Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences*, 99(10):6567–6572, 2002.

[266] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of*

the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, pages 142–147, 2003.

[267] Sarah K. Tyler, Jian Wang, and Yi Zhang. Utilizing re-finding for personalized information retrieval. In Jimmy Huang, Nick Koudas, Gareth J. F. Jones, Xindong Wu, Kevyn Collins-Thompson, and Aijun An, editors, *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 1469–1472. ACM, 2010. doi:10.1145/1871437.1871649.

[268] Stéfan van der Walt, S. Chris Colbert, and Gaël Varoquaux. The numpy array: A structure for efficient numerical computation. *Comput. Sci. Eng.*, 13(2):22–30, 2011.

[269] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[270] Damir Vandic, Jan-Willem van Dam, and Flavius Frasincar. Faceted product search powered by the semantic web. *Decis. Support Syst.*, 53(3):425–437, 2012. doi:10.1016/j.dss.2012.02.010.

[271] Damir Vandic, Flavius Frasincar, and Uzay Kaymak. Facet selection algorithms for web product search. In Qi He, Arun Iyengar, Wolfgang Nejdl, Jian Pei, and Rajeev Rastogi, editors, *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 2327–2332. ACM, 2013. doi:10.1145/2505515.2505664.

[272] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[273] E Voorhees and D Harman. Experiment and evaluation in information retrieval, 2005.

[274] Thanh Vu, Dat Quoc Nguyen, Mark Johnson, Dawei Song, and Alistair Willis. Search personalization with embeddings. In Joemon M. Jose, Claudia Hauff, Ismail Sengör Altingövde, Dawei Song, Dyaa Albakour, Stuart N. K. Watt, and John Tait, editors, *Advances in Information Retrieval - 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8-13, 2017, Proceedings*, volume 10193 of *Lecture Notes in Computer Science*, pages 598–604, 2017. doi:10.1007/978-3-319-56608-5_54.

[275] Endong Wang, Qing Zhang, Bo Shen, Guangyong Zhang, Xiaowei Lu, Qing Wu, and Yajuan Wang. Intel math kernel library. In *High-Performance Computing on the Intel® Xeon Phi™*, pages 167–188. Springer, 2014.

[276] Qihua Wang and Hongxia Jin. Exploring online social activities for adaptive search personalization. In Jimmy Huang, Nick Koudas, Gareth J. F. Jones, Xindong Wu, Kevyn Collins-Thompson, and Aijun An, editors, *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 999–1008. ACM, 2010. doi:10.1145/1871437.1871564.

[277] Shuai Wang, Shengyao Zhuang, and Guido Zuccon. Bert-based dense retrievers require interpolation with BM25 for effective passage retrieval. In Faegheh Hasibi, Yi Fang, and Akiko Aizawa, editors, *ICTIR '21: The 2021 ACM SIGIR International Conference on the Theory of Information Retrieval, Virtual Event, Canada, July 11, 2021*, pages 317–324. ACM, 2021. doi:10.1145/3471158.3472233.

[278] Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. Pseudo-relevance feedback for multiple representation dense retrieval. In Faegheh Hasibi, Yi Fang, and Akiko Aizawa, editors, *ICTIR '21: The 2021 ACM SIGIR International Conference on the Theory of Information Retrieval, Virtual Event, Canada, July 11, 2021*, pages 297–306. ACM, 2021. doi:10.1145/3471158.3472250.

[279] Jason Weston and Chris Watkins. Support vector machines for multi-class pattern recognition. In *ESANN 1999, 7th European Symposium on Artificial Neural Networks, Bruges, Belgium, April 21-23, 1999, Proceedings*, pages 219–224, 1999.

[280] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019.

[281] Shengli Wu and Fabio Crestani. Data fusion with estimated weights. In *Proceedings of the 2002 ACM CIKM International Conference on Information and Knowledge Management, McLean, VA, USA, November 4-9, 2002*, pages 648–651. ACM, 2002. doi:10.1145/584792.584908.

[282] Xuan Wu, Dong Zhou, Yu Xu, and Séamus Lawless. Personalized query expansion utilizing multi-relational social data. In Mária Bieliková and Marián Simko, editors, *12th International Workshop on Semantic and Social Media Adaptation and Personalization, SMAP 2017, Bratislava, Slovakia, July 9-10, 2017*, pages 65–70. IEEE, 2017. doi:10.1109/SMAP.2017.8022669.

[283] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[284] Shengliang Xu, Shenghua Bao, Ben Fei, Zhong Su, and Yong Yu. Exploring folksonomy for personalized search. In Sung-Hyon Myaeng, Douglas W. Oard,

Fabrizio Sebastiani, Tat-Seng Chua, and Mun-Kew Leong, editors, *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, pages 155–162. ACM, 2008. doi:10.1145/1390334.1390363.

[285] Yang Xu, Gareth J. F. Jones, and Bin Wang. Query dependent pseudo-relevance feedback based on wikipedia. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, pages 59–66. ACM, 2009. doi:10.1145/1571941.1571954.

[286] Omry Yadan. Hydra - a framework for elegantly configuring complex applications. Github, 2019.

[287] Jing Yao, Zhicheng Dou, and Ji-Rong Wen. Employing personal word embeddings for personalized search. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1359–1368. ACM, 2020. doi:10.1145/3397271.3401153.

[288] Jing Yao, Zhicheng Dou, Jun Xu, and Ji-Rong Wen. Rlper: A reinforcement learning model for personalized search. In Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen, editors, *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2298–2308. ACM / IW3C2, 2020. doi:10.1145/3366423.3380294.

[289] Jing Yao, Zhicheng Dou, and Ji-Rong Wen. Clarifying ambiguous keywords with personal word embeddings for personalized search. *ACM Trans. Inf. Syst.*, 40(3):43:1–43:29, 2022. doi:10.1145/3470564.

[290] Jun Yu, Sunil Mohan, Duangmanee Putthividhya, and Weng-Keen Wong. Latent dirichlet allocation based diversified retrieval for e-commerce search. In Ben Carterette, Fernando Diaz, Carlos Castillo, and Donald Metzler, editors, *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*, pages 463–472. ACM, 2014. doi:10.1145/2556195.2556215.

[291] Hamed Zamani and W. Bruce Croft. Joint modeling and optimization of search and recommendation. In Omar Alonso and Gianmaria Silvello, editors, *Proceedings of the First Biennial Conference on Design of Experimental Search & Information Retrieval Systems, Bertinoro, Italy, August 28-31, 2018*, volume 2167 of *CEUR Workshop Proceedings*, pages 36–41. CEUR-WS.org, 2018.

[292] Hamed Zamani and W. Bruce Croft. Learning a joint search and recommendation model from user-item interactions. In James Caverlee, Xia (Ben) Hu,

Mounia Lalmas, and Wei Wang, editors, *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 717–725. ACM, 2020. doi:10.1145/3336191.3371818.

[293] ChengXiang Zhai and John D. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, September 9-13, 2001, New Orleans, Louisiana, USA*, pages 334–342. ACM, 2001. doi:10.1145/383952.384019.

[294] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. Optimizing dense retrieval model training with hard negatives. In Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai, editors, *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1503–1512. ACM, 2021. doi:10.1145/3404835.3462880.

[295] Han Zhang, Songlin Wang, Kang Zhang, Zhiling Tang, Yunjiang Jiang, Yun Xiao, Weipeng Yan, and Wenyun Yang. Towards personalized and semantic retrieval: An end-to-end solution for e-commerce search via embedding learning. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 2407–2416. ACM, 2020. doi:10.1145/3397271.3401446.

[296] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. Towards conversational search and recommendation: System ask, user respond. In Alfredo Cuzzocrea, James Allan, Norman W. Paton, Divesh Srivastava, Rakesh Agrawal, Andrei Z. Broder, Mohammed J. Zaki, K. Selçuk Candan, Alexandros Labrinidis, Assaf Schuster, and Haixun Wang, editors, *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 177–186. ACM, 2018. doi:10.1145/3269206.3271776.

[297] Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. BERT-QE: contextualized query expansion for document re-ranking. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 4718–4728. Association for Computational Linguistics, 2020. doi:10.18653/v1/2020.findings-emnlp.424.

[298] Jianling Zhong, Weiwei Guo, Huiji Gao, and Bo Long. Personalized query suggestions. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval,*

SIGIR 2020, Virtual Event, China, July 25-30, 2020, pages 1645–1648. ACM, 2020. doi:10.1145/3397271.3401331.

[299] Dong Zhou, Séamus Lawless, and Vincent Wade. Improving search via personalized query expansion using social media. *Inf. Retr.*, 15(3-4):218–242, 2012. doi:10.1007/s10791-012-9191-2.

[300] Dong Zhou, Xuan Wu, Wenyu Zhao, Séamus Lawless, and Jianxun Liu. Query expansion with enriched user profiles for personalized search utilizing folksonomy data. *IEEE Trans. Knowl. Data Eng.*, 29(7):1536–1548, 2017. doi:10.1109/TKDE.2017.2668419.

[301] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. Encoding history with context-aware representation learning for personalized search. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1111–1120. ACM, 2020. doi:10.1145/3397271.3401175.

[302] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. Enhancing re-finding behavior with external memories for personalized search. In James Caverlee, Xia (Ben) Hu, Mounia Lalmas, and Wei Wang, editors, *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 789–797. ACM, 2020. doi:10.1145/3336191.3371794.

[303] Yujia Zhou, Zhicheng Dou, Yutao Zhu, and Ji-Rong Wen. PSSL: self-supervised learning for personalized search with contrastive sampling. In Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong, editors, *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 2749–2758. ACM, 2021. doi:10.1145/3459637.3482379.

[304] Zhengyu Zhu, Jingqiu Xu, Xiang Ren, Yunyan Tian, and Lipei Li. Query expansion based on a personalized web search model. In *Third International Conference on Semantics, Knowledge and Grid, Xian, Shan Xi, China, October 29-31, 2007*, pages 128–133. IEEE Computer Society, 2007. doi:10.1109/SKG.2007.83.