Reproducible Information Retrieval Research: From Principled System-Oriented Evaluations Towards User-Oriented Experimentation

Von der Fakultät für Ingenieurwissenschaften
Abteilung Informatik und Angewandte Kognitionswissenschaft
der Universität Duisburg-Essen

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

Timo Breuer
aus
Stade

1. Gutachter: Prof. Dr. Norbert Fuhr
2. Gutachter: Prof. Dr. Philipp Schaer
3. Gutachter: Prof. Dr. Matthias Hagen

Tag der mündlichen Prüfung: 31.03.2023

Abstract

The reproducibility of earlier findings is fundamental to the empirical sciences. Even though this circumstance is widely acknowledged, several systematic large-scale reproducibility reviews showed that many earlier findings, e.g., in computer science, psychology, or the biomedical sciences, are not reproducible. Information Retrieval (IR) is rooted in experimentation, and empirical findings mainly drive the research progress. Therefore, the IR community established several initiatives to understand the reproducibility of earlier works better and provided solutions for better research practices to enforce reproducible research. For instance, dedicated reproducibility tracks at IR conferences report on the reproducibility of previous work, while other works introduce solutions to prepare an experimental setup for reuse.

This thesis contributes perspectives on how reproducibility can be evaluated at different levels of validity. The first part of the thesis deals with internal validity covering the scope of system-oriented experimentation. We note that there is no standard approach in IR when evaluating the quality of reimplementations as part of a reproducibility attempt. To this end, this work proposes a more principled approach to reproducibility analysis for system-oriented IR experiments. Building upon an extended version of the PRIMAD taxonomy, we outline how a derived metadata schema can be combined with reproducibility measures to determine the degree and quality of reproduction in a principled way.

The second part of the thesis focuses on external validity by considering user variability in an IR experiment. The user's influence in an IR experiment is a key component that allows us to conclude how well the system-oriented findings can be reproduced in a different experimental context. As an alternative to experiments with real users, simulations provide a more cost-efficient, reproducible, and controllable solution to account for the variation of user behavior. Our simulation experiments specifically focus on the variation of the query formulation and the click behavior. In this regard, we analyze reproducibility by considering different query variants as alternative system inputs and clicks as other forms of relevance signals to the system outputs. Both user interactions are usually not part of system-oriented IR experiments and simulations allow better conclusions about the external validity.

Finally, we provide an outlook of how the validity can be analyzed in real-world user experiments run on a living lab platform. The underlying infrastructure embeds the concept of containerization and allows the integration of technically reproducible IR systems. The corresponding evaluations of online experiments show how the infrastructure adds up to earlier online platforms and exemplify how system-oriented experiments could be accompanied and validated by living lab experiments with real users in the future.

Zusammenfassung

Die Reproduzierbarkeit ist für die empirische Wissenschaft von grundlegender Bedeutung und obwohl dies weithin anerkannt ist, haben mehrere Reproduzierbarkeitsstudien gezeigt, dass viele Ergebnisse, z. B. in der Informatik, der Psychologie oder den biomedizinischen Wissenschaften, nicht reproduzierbar sind. Da der Fortschritt im Information Retrieval (IR) hauptsächlich durch empirische Erkenntnisse vorangetrieben wird, wurden mehrere Initiativen ins Leben gerufen, um die Reproduzierbarkeit früherer Arbeiten besser zu verstehen und Lösungen für bessere Forschungspraktiken zur Durchsetzung reproduzierbarer Forschung zu finden. So bieten beispielsweise IR-Konferenzen die Möglichkeit, über die Reproduzierbarkeit früherer Arbeiten zu berichten, während in anderen Arbeiten Lösungen zur Vorbereitung eines Versuchsaufbaus für die Wiederverwendung vorgestellt werden.

Diese Arbeit leistet einen Beitrag zur Auswertung von Reproduzierbarkeitsstudien auf verschiedenen Ebenen der Validität. Der erste Teil befasst sich mit der internen Validität, die systemorientierte IR-Experimente abdeckt. Häufig wird kein Standardansatz verfolgt, wenn Ergebnisse einer Reproduzierbarkeitsstudie evaluiert werden. Zu diesem Zweck wird in dieser Arbeit ein systematischer Ansatz zur Reproduzierbarkeitsanalyse für systemorientierte IR-Experimente vorgestellt. Aufbauend auf einer Erweiterung der PRIMAD-Taxonomie wird skizziert, wie diese in Form eines Metadatenschemas mit Reproduzierbarkeitsmaßen zur Bestimmung der Reproduktionsqualität verwendet werden kann.

Der zweite Teil der Arbeit befasst sich mit der externen Validität, indem er die Nutzervariabilität in einem Experiment betrachtet. Der Benutzereinfluss in einem IR-Experiment ist eine Schlüsselkomponente, die uns Rückschlüsse darauf erlaubt, wie gut die Ergebnisse in einem geänderten experimentellen Kontext reproduziert werden können. Als Alternative zu Experimenten mit realen Nutzern bieten Simulationen eine kostengünstigere, reproduzierbare und kontrollierbare Lösung, um die Variation des Nutzerverhaltens zu berücksichtigen. Unsere Simulationen konzentrieren sich insbesondere auf die Variation der Anfrageformulierung und des Klickverhaltens. In diesem Zusammenhang analysieren wir die Reproduzierbarkeit, indem wir verschiedene Anfragevarianten als alternative Systemeingaben und Klicks als andere Formen von Relevanzsignalen für die Systemausgaben betrachten.

Zuletzt geben wir einen Ausblick darauf, wie die Validität in realen Benutzerexperimenten analysiert werden kann. Die zugrundeliegende Living-Lab-Infrastruktur beruht auf dem Konzept der Containerisierung und erlaubt die Integration technisch reproduzierbarer IR-Systeme. Die dazugehörigen Auswertungen von Online-Experimenten veranschaulichen, wie die Infrastruktur eine Möglichkeit bietet, systemorientierte Experimente in Zukunft durch Living-Lab-Experimente mit realen Nutzern validieren zu können.

Acknowledgements

Looking back, I want to take the opportunity and thank all the people involved in this dissertation project at some point. I am very thankful to all my co-authors and colleagues. The collaborations were among my best experiences in the last four years. Many of the results were only possible with the outstanding support of the following people, and I consider this dissertation project also a collaborative effort.

First and foremost, I would like to thank my two supervisors, Norbert Fuhr and Philipp Schaer. Thank you for your guidance and advice in every regard. Thank you, Norbert, for the constructive criticism and expertise that helped shape the direction of my research. Thank you, Philipp, for making it possible to work with you on such exciting topics, encouraging me, and for your support at any time. In this regard, I would also like to thank Matthias Hagen. Thank you, Matthias, for the valuable comments on the text and the thoughtful feedback and suggestions.

Many thanks to Maria Maistro and Nicola Ferro for such an interesting collaboration. I learned a lot from the meetings; they inspired me a lot. Our joint work also led to collaboration with Tetsuya Sakai and Ian Soboroff, whom I also want to thank. Furthermore, I would like to thank Dirk Tunger for our joint work and our interesting discussions. Working with all of you also shaped my understanding of approaching a research problem.

Next, I would like to thank the STELLA team for making the collaboration across three institutes possible. Many thanks to Narges Tavakolpoursaleh, Benjamin Wolff, Leyla Jael Castro, Johann "Wanja" Schaible, Daniel Hienert, and Zeljko Carevic.

Furthermore, I want to thank the students I had the opportunity to work with. Many thanks to Jüri Keller for steady support and Melanie Pest and Anh Huy Tran for the excellent work you made as part of your thesis.

In the office, I was surrounded by some very supportive colleagues whom I would like to express my thankfulness. Many thanks to Fabian Haak, Björn Engelmann, Christin Katharina Kreutz, Sven Wöhrle, and Narjes Nikzad Khasmakhi. I would also like to thank my former colleagues, Mandy Neumann and Malte Bonart, for the onboarding and introduction to the research group.

Finally, I would like to express my deepest gratitude to my family for supporting me more than I could ever thank them. Many thanks to Mi, Oli, Sandra, and Thomas. Thank you for your unconditional love and support and the patience and trust you had since I started my studies.

Contents

Chapter 1

Introduction

Reproducibility is a key component of good scientific practice and fundamental to the overall progress of a research field. However, several recent large-scale reproducibility studies in different domains acknowledge that the reproducibility of empirical science cannot be taken for granted, and a large proportion of previously published research is not reproducible. This is critical as it not only puts current research practices into question but also harms the public credibility and trust in science.

The progress in IR research is mainly driven by empirical evidence, which is prone to reproducibility issues. During the last decade, the IR community has dealt with some of these issues and implemented several countermeasures to make IR studies more reproducible. However, evaluating the reproducibility of earlier IR experiments is an ongoing challenge, as can be seen from the dedicated reproducibility tracks inaugurated at major IR conferences like ECIR and SIGIR.

This dissertation project is about reproducible IR research and makes contributions of how it can be evaluated at different levels of validity. Related to the question if an experiment is reproducible within the constraints of the original experimental context, there is the question of to which extent the original findings are valid in a different experimental context. As part of this thesis, we consider the scope of the internal validity to be evaluated by system-oriented reproducibility evaluations. With a particular focus on the user, we consider the external validity to be evaluated under the variation of the user's influence.

In the first part of the thesis, we review the state of the art in reproducible IR research and make contributions that allow a more principled reproducibility evaluation of system-oriented IR experiments. Based on an extension of the PRIMAD taxonomy, we provide a metadata schema that, combined with reproducibility measures, can quantify the reproducibility of reimplementations in a principled way.

In the second part of the thesis, we lower the level of abstraction regarding the user as part of the evaluation of IR experiments. As an alternative to experiments with real users, the simulation of user interaction is a viable solution that allows us to define user behavior in a controlled and reproducible manner. Finally, we present how experiments with real users can be evaluated in a living lab environment that embeds the concept of technical reproducibility.

In the following, we describe the motivation for this dissertation project and our contributions. Afterward, we outline the structure of the thesis.

1.1 Motivation

In the following, we motivate the dissertation's contributions by giving a brief review of reproducibility in the empirical sciences and by outlining how reproducibility relates to in- and external validity.

1.1.1 Reproducibility in the Empirical Sciences

In the empirical sciences, the evidence is often based on experimental results. Recently, concerns have been raised that claims and conclusions drawn from empirical studies in medicine [122, 333], psychology [25, 313], economics [75], or computer science [98, 99] do not hold as the underlying experiment is not reproducible. Irreproducible studies not only slow scientific progress [13] but also negatively affect the public trustworthiness of science in general and, last but not least, unnecessarily increase the use of resources and the environmental impact of science [360].

In 2005, Ioannidis' thought-provoking simulations [198] showed that it is likely that most research claims are false due to influential factors like study power, effect sizes, biases, and other works on the same research question. Recently, a survey with over 1,500 scientists from different scientific disciplines revealed that the scientific community acknowledges these reproducibility issues [26] and some go as far as to say that there is a *reproducibility crisis* [194]. The survey's results showed that more than 70% of the interviewees failed to reproduce another scientist's experiment, and more than 50% even failed to reproduce their own work later in time.

All of these concerns are confirmed as part of large-scale reproducibility meta-evaluations across various scientific fields. For instance, a large-scale reproducibility analysis in psychology revalidated 100 studies published in high-impact journals. Ninety-seven studies originally reported positive findings, but for only 36 studies, significant effects could be reconfirmed [25,313]. By following a similar study design, Camerer et al. [75] conducted a reproducibility analysis in economics and could only confirm comparable effects for 11 out of 18 studies. Similarly, Camerer et al. [76] revalidated 21 studies from the social sciences and found significant effects for only 13 studies. Other examples of meta-evaluations with rather disillusioning conclusions include drug development [333] or cancer research [122].

With a particular focus on computer science, Collberg et al. [98, 99] conducted a systematic reproducibility analysis of over 600 ACM publications but could only successfully reproduce one-third of the analyzed publications when rerunning the original code. Other meta-evaluations from the fields of Natural Language Processing (NLP) [38] or Recommender Systems (RecSys) [108] research further confirm the reproducibility issues in the computational sciences. Earlier work is often not reproducible due to various reasons ranging from mundane aspects like fraud or missing experimental artifacts to more complex circumstances like low statistical power and conclusions that cannot be confirmed in a slightly modified context.

Even though reproducibility in IR research has always been implicitly considered from the early beginnings of Cleverdon's experiments [94, 95], which established the Cranfield paradigm, the IR research community acknowledges these increasing reproducibility concerns, as can be seen by the conference proceedings of ECIR [171] and SIGIR [9] that inaugurated dedicated reproducibility tracks. Since the middle of the previous decade, the IR community developed countermeasures and policies to

enforce reproducible research. To this end, this dissertation project reviews existing solutions but also addresses open points of how reproducibility evaluations can be improved by considering different levels of validity, as outlined in the following.

1.1.2 Internal and External Validity

Related to the reproducibility of an experiment, there is the question regarding the degree of validity. *Internal validity* describes the extent to which the claims about an experiment are supported by the data, whereas *external validity* describes the extent to which the claims can be generalized, for instance, with another population of users or different data in general [293]. In psychology, Brunswick [71] introduced the concept of *ecological validity* as a sub-type of the external validity. It describes to which extent findings from the laboratory hold in the real world. More recently, Kieffer [225] has introduced a framework for human-computer interaction studies, which outlines how ecological validity can be assessed by considering user experience.

The contributions of this dissertation project can be categorized into different levels of validity. Throughout the progress of the chapters, we lower the abstraction level of the user in an IR experiment, i.e., we shift the context of the experimental setup towards ecological validity with regard to the user's influence. Starting with reproducibility evaluations of the internal validity in system-oriented IR experiments, which imply a strong abstraction of the real-world user behavior, we can shift the scope of the evaluations towards external validity in a controlled way by simulating variations of the user behavior. As the conclusions drawn from simulations strongly depend on the fidelity of the user model, real-world online experiments finally allow us to evaluate the ecological validity of an IR experiment. The following section outlines how these concepts are integrated into our contributions.

1.2 Contributions

As pointed out in the previous section, we address the topic of reproducibility and the corresponding evaluations at different levels of validity and similarly align the contributions to these levels. Beforehand, we review the state of the art about reproducible IR research as part of:

C1 **Literature review** about the state of the art regarding reproducible research in computer science and IR (cf. Chapter 2)

Specifically, we answer what kinds of general reproducibility problems exist in computer science with a particular focus on IR research. In addition, we give an overview of how these reproducibility problems have been addressed and how the countermeasures are implemented. Finally, we highlight open points to motivate our following contributions.

Besides the answers to these questions, two major outcomes of the literature review are as follows. First (cf. **Outcome 1** in Figure 1.1), we notice that the PRIMAD taxonomy has not been put into practice yet, and it is described at a very abstract level. In addition, it is outlined separately for system- and user-oriented experiments, but we criticize that the users are not represented well enough as they are only considered as part of the data component, and a more holistic view of the IR experiment is required.

Second (cf. **Outcome 2** in Figure 1.1), we highlight that there is no consistency among authors when evaluating the reproducibility of an IR experiment. Even though specific experiments may require dedicated evaluation approaches, there is no general idea about how to evaluate the reproducibility of an IR experiment in a principled way. Likewise, many software tools exist, helping researchers prepare a computational experiment for reproducibility in a proactive way. However, on the other side, few software tools help researchers evaluate their reimplementations as part of reactive reproducibility attempts. As an answer to the underspecification and sometimes imprecise use of the reproducibility terminology, the first two contributions at the level of internal validity are:

C2 Extension of the PRIMAD taxonomy by an additional user component and additional specifications of the original taxonomy (cf. Chapter 3)

C3 Metadata annotation schema for run files of system-oriented IR experiments (cf. Chapter 3 and also [63])

Building on these contributions, we outline the framework of a reactive reproducibility attempt, including measures for determining the degree of reproducibility and the corresponding software toolkit (cf. **C4**). Combined with our reimplementations (cf. **C5**), all of these contributions serve as the basis to demonstrate how principled reproducibility evaluations (cf. **C6**) can be implemented. Regarding the internal validity of an IR experiment, our additional contributions can be summarized as follows:

C4 Reproducibility framework for reactive reproducibility experiments and a corresponding software toolkit (cf. Chapter 4 and also [60, 61])

C5 Reimplemenations of Cross-Collection Relevance Feedback (CCRF) (cf. Chapter 5 and also [64, 65, 66])

C6 Principled reproducibility analysis of different CCRF reimplementations (cf. Chapter 5 and also [63])

However, all of these contributions describe and evaluate the reproducibility with a strong focus on system-oriented aspects at the level of internal validity. In order to widen the reproducibility scope towards external validity, we consider user variability as one of the most influential components that should be considered. As an alternative to evaluating the IR systems directly in online experiments, we prefer the user simulation as a more cost-efficient and controllable way to include the user variability in the experimental evaluations. As part of this dissertation, we focus on two user-related aspects that might influence the reproducibility of an IR experiment: query formulation and click-based relevance feedback.

In system-oriented experiments, the query formulation is often limited to the evaluation of a single query variant per topic (or information need). This approach does not account for the variability that would result from users who formulate different queries for the same underlying information need. We address this by analyzing different query simulators based on TREC test collections. Besides introducing a new query simulation method and comparing it with other conventional methods, we also introduce a validation framework. The results show the range of variability

of the retrieval effectiveness that can result from different user models of the query formulation and how a general searcher without prior knowledge about the topic compares to a more proficient searcher who searches for a known-item. In addition, we analyze how the simulated queries, specifically those of the introduced simulation method, compare to real user queries, leading to the following contribution:

C7 **Method for query simulations** based on IR test collections and a corresponding **evaluation framework** (cf. Chapter 6 and also [62])

Analogous to queries, which serve as an input to the IR system, there is also variability in the relevance feedback, i.e., how the user perceives the relevance of the returned system results. System-oriented evaluations are based on editorial relevance judgments, which have high organizational costs and are usually only made possible as part of large-scale community efforts. As an alternative, click signals from web search experiments can serve as proxies or alternative relevance indicators. Different types of click models can be parameterized from click logs. We outline how these parameterized click models can be used to estimate the relevance of rankings and analyze to which extent they can be used to evaluate the correct relative effectiveness of IR systems. Click models embed different rules for the user behavior, and thus, their use allows us to simulate different types of users, leading to the following contribution:

C8 **Click model-based evaluations** of IR experiments (cf. Chapter 7)

Our click model-based evaluations address how click models, embedding satisfaction and continuation probabilities, compare to the simpler model based on the Click-Through Rate (CTR). By evaluating simulated interleaving experiments, we bridge the gap to the living lab experiments by addressing how well click model-based evaluations can reproduce the relative system ordering in living labs.

This dissertation project was funded by the DFG project "STELLA - Infrastructures for Living Labs" (project no. 407518790), which had the aim to develop an open infrastructure (cf. **C9**) that can be used to evaluate IR and RecSys experiments with user feedback data. The overall design of the infrastructure was tailored for interleaving experiments with two competing systems from which the results were shown to users and transferred the simulations of Chapter 7 into the real world. The corresponding evaluations are based on a shared task (cf. **C10**).

By considering user variability as one of the key components towards evaluating the external validity of IR experiments, the earlier contributions analyzed simulated user behavior for IR evaluations, whereas the contributions **C9** and **C10** outline how experiments in the real word can determine the ecological validity with the help of living lab experiments. Regarding the external and ecological validity of an IR experiment, our contributions can be summarized as follows:

C9 **Living lab infrastructure** for reproducible experimentation (cf. Chapter 8 and also [67])

C10 **Evaluations of a shared task** that served as a testbed for the infrastructure (cf. Chapter 8 and also [362])

The following Section 1.3 provides more specific details about the structure of this dissertation and the contributions of each particular chapter.

1.3 Outline

As illustrated in Figure 1.1, the contributions of this work can be aligned to reproducibility evaluations at different levels of validity. While the first chapters focus on the evaluation of system-oriented IR experiments and cover the scope of internal validity, the later chapters evaluate the reproducibility of their external validity under the consideration of (simulated) user variability. Different query variants or click models simulate user behavior. Finally, we provide an outlook on how the ecological validity of IR experiments can be validated by living lab experiments with real users.

Chapter 1 describes the motivations of this dissertation project. The initial proposal of this dissertation was contributed to the doctoral consortium at ECIR'20 [59].

Chapter 2 gives answers to the first two research questions, and the literature review shows that the PRIMAD taxonomy has not been put into practice yet and is described at a very abstract level. Furthermore, there are few solutions to evaluate reimplementations as part of reactive reproducibility attempts. Both of these shortcomings will be picked up in the following two chapters.

Chapter 3 provides a more detailed taxonomy of PRIMAD by extending each component with several sub-components. Furthermore, we favor a holistic view on the IR experiment by adding a user component, which makes the user's contributions and influences in an experiment more explicit. The corresponding metadata schema was contributed to SIGIR'22 [63].

Chapter 4 addresses the lack of reproducibility measures and tools for a reactive reproduction analysis. We put the reproducibility measures, introduced at SIGIR'20 [60], into context and outline how different levels of rigor can be used to evaluate the reproducibility. In addition, we provide the reproducibility measures in a Python software library (cf. ECIR'21 [61]).

Chapter 5 combines the outcomes of the previous two chapters and outlines how principled reproducibility experiments based on metadata annotations of the run files can be conducted. For these experiments, we conducted several reproducibility studies that have been published in different proceedings (cf. CENTRE'19 [66], OSIRRC'19 [65], CLEF'21 [64]).

Chapter 6 analyzes different query simulators based on TREC test collections. Besides introducing a new query simulation method and comparing it with other conventional methods, we also introduce a validation framework. The results show the range of variability of the retrieval effectiveness resulting from different query formulations. These experiments were contributed to ECIR'22 [62].

Chapter 7 outlines how click models can be used to estimate the relevance of rankings and analyze to which extent they can be used to evaluate the correct relative effectiveness of IR systems. The analysis of simulated interleaving experiments bridges the gap to the living lab experiments by addressing how well click model-based evaluations can reproduce the relative system ordering in living labs.

Chapter 8 paves the way towards ecological validity of IR experiments by evaluating retrieval systems *in the wild* with real users. In the corresponding chapter, we outline the infrastructure's design (cf. ISI'21 [67]) that was evaluated in a shared task (cf. CLEF'21 [361]).

Chapter 9 concludes and puts the results into context once again.

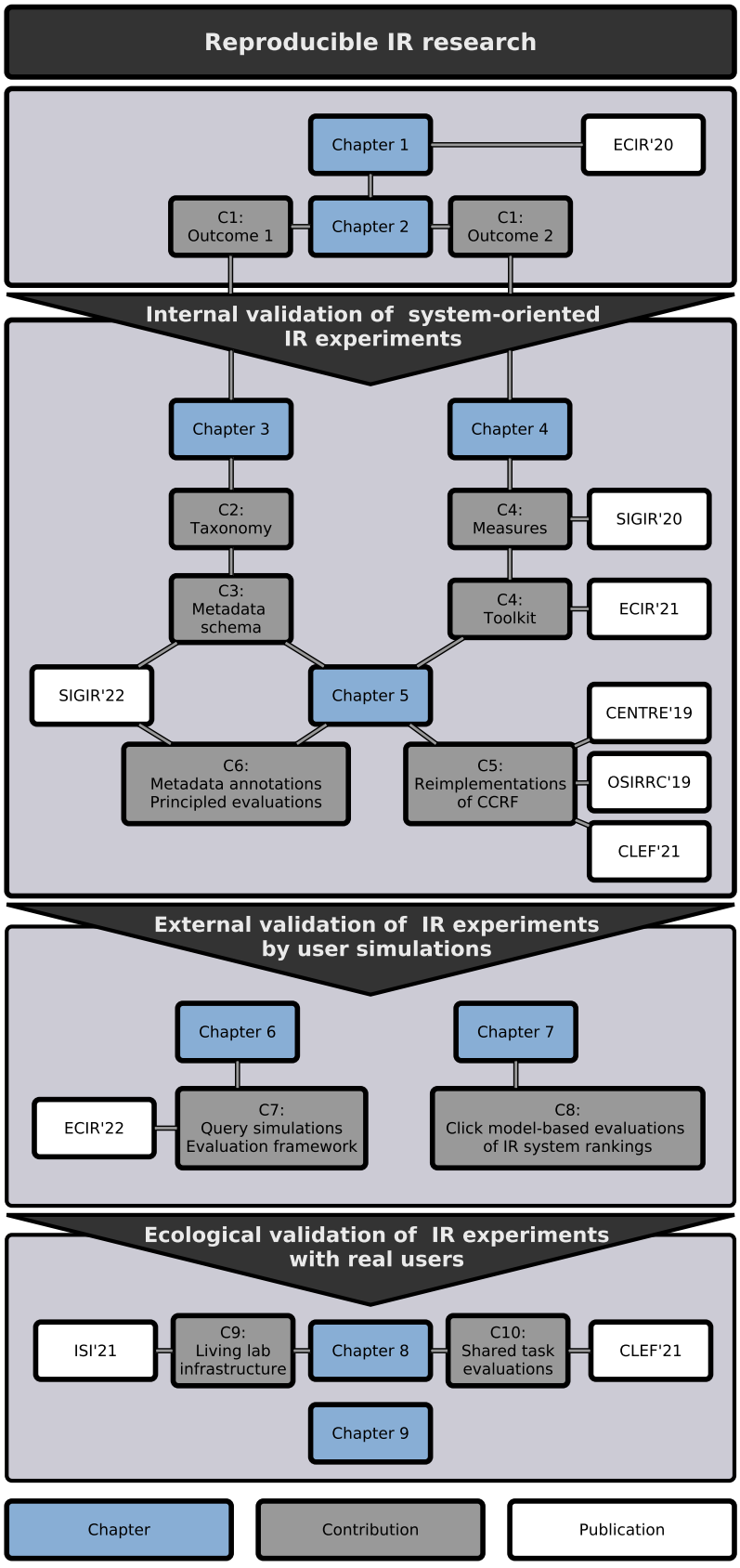

Figure 1.1: Overview of this dissertation project.

Chapter 2

Related Work

This chapter reviews the related work and state of the art about reproducible computational science in general and reproducible IR research in particular, i.e., it contributes the **literature review (C1)**. More specifically, the literature review focuses on the following aspects. First, we provide an overview of possible factors and issues that may cause irreproducible science and how these issues impact IR research. Second, building upon these intermediate results, we identify how these problems have been addressed so far and what kinds of open questions about reproducible IR research exist. More specifically, we address the following two research questions in this chapter:

RQ1 *What kinds of general reproducibility problems are there in computer science and particularly in IR research?*

RQ2 *To what extent have reproducibility problems been addressed in IR research, and how are the countermeasures implemented? What kinds of open points are there?*

In order to provide answers to these questions, we follow a systematic approach to the literature research. We mainly focus on peer-reviewed publications of high-impact conferences and journals, including SIGIR, ECIR, ICTIR, JCDL, CHIIR, CIKM, WWW, KDD, WSDM, TOIS, IRJ, IPM, and JASIST, and we search *dblp.org* with the ACM terminology [448], i.e., stemmed terms of repeatability, reproducibility, and replicability, resulting in structured queries like `sigir repeat | repro | replica` (when browsing the proceedings of the SIGIR conference). In addition, we include all publications of the ECIR and SIGIR reproducibility track from 2015 until 2022 and relevant references of core publications, for which we additionally search the IR Anthology [330] and Google Scholar.

As a starting point for identifying irreproducible factors, we categorize the answers given to Baker's survey [26] in 2016 into five more abstract groups, to which we align the related work in the computational sciences. Furthermore, we build upon the reproducibility taxonomy by Potthast et al. [329] that groups attempts towards reproducibility into supportive, pro- and reactive actions. This dissertation and, likewise, this literature review are inspired mainly by the PRIMAD taxonomy [132]. It considers six components of computational experiments, possibly affecting reproducibility. We review IR-related reproducibility problems and solutions by aligning them to the PRIMAD taxonomy. For each of the six components, we discuss the

factors of irreproducibility as derived by Baker's survey. Finally, we include existing solutions by assigning them to the corresponding actions in the taxonomy by Potthast et al. Our review results in the following main findings.

First, PRIMAD is comprehensive enough to describe system-oriented IR experiments in a reproducible way. However, its components currently lack more detailed specifications, preventing PRIMAD from being put into practice. Furthermore, reproducibility attempts of user-oriented experiments are underrepresented in the literature and deserve more attention. PRIMAD also requires a more integrated user-oriented perspective since user- and system-oriented experiments are originally discussed separately. More detailed specifications as part of an integrated taxonomy that likewise accounts for system- and user-oriented and practical applications of PRIMAD are provided in Chapter 3.

Second, referring to the distinction between supportive, pro-, and reactive actions, we show that current solutions towards reproducible IR research are mainly supportive and proactive. Even though most of the reproducibility efforts as part of conference tracks at ECIR and SIGIR follow a reactive approach, no solutions exist to measure the success of reproducibility for these experiments. This open point is addressed as part of Chapter 4, in which we introduce a reproducibility framework of the general reactive reproducibility study and the corresponding measures and a software toolkit.

The remainder of this chapter is structured as follows. First, we review how reproducibility is implicitly considered in system-oriented experiments according to the Cranfield paradigm. Afterward, we address the terminology that is used throughout this work. Next, we give a general introduction to the taxonomies by Potthast et al. [329] and Ferro et al. [132] in order to align IR-related studies to them. Afterward, we define groups of irreproducible factors in order to categorize the literature from the computational sciences. The existing work and literature are discussed separately for each of the components. Finally, we summarize the related work by addressing the research questions outlined above.

2.1 The Cranfield Paradigm and Reproducibility

Even though this chapter and literature review has a strong focus on what kinds of solutions towards reproducible research were proposed since the mid-2010s, we note that IR research has been deeply rooted in experimentation since the early beginning. IR research has always been based on the implicit assumption of building upon earlier work that is reproducible due to the experimental design that is known as the Cranfield paradigm [95]. Cleverdon was one the first authors to propose systematic evaluations of IR systems on the basis of a document collection and pre-defined search terms or queries for which the search process would be considered to be successful if a relevant document would be returned by the system (cf. Cranfield 1). By building upon this design, the Cranfield 2 approach added graded relevance labels for particular documents. Notably, there is an implicit question about reproducibility as this experimental design evaluates if the system performs similarly — in a reproducible way — if another query is used.

The Cranfield paradigm established the three constituting parts of an IR test collection, which are (1) a collection of (text) documents, (2) a set of topics/queries (resembling an information need) and (3) the corresponding relevance judgments for

particular documents [355]. As the process of curating such a test collection is labor-intensive and costly (especially the relevance annotations), the research community was engaged in developing test collections as part of shared tasks starting with the TREC conference in 1992 [172, 413]. Throughout the following years, several "offshoot" conferences were inaugurated, such as CLEF for European languages [138, 322], NTCIR for Asian languages such as Japanese, Chinese, and Korean [215], and FIRE for Indian languages [272].

For these shared task efforts, the *pooling* process is fundamental [355]. Multiple different retrieval systems contribute rankings to a unified set of documents, the so-called *document pool*, out of which the documents are given to annotation experts in order to make relevance judgments. Overlapping documents in the rankings do not have to be judged twice (reducing annotation costs), and based on the same source of relevance judgments, multiple systems can be systematically evaluated in a fair way. This procedure prevents the system developers from *overfitting* their system to the gold labels, and thus, implicitly forces the experimenter to follow good scientific practice by reasoning about effective retrieval approaches and to formulate a research question or hypothesis.

Furthermore, a diverse set of different retrieval systems contributing to the document pool allows us to reuse the test collection for the evaluation of new retrieval approaches that did not participate in the original shared task, i.e., the test collection is a *reusable tool*, which is essential for reproducibility. Most of the conferences host experimental artifacts such as the submitted rankings (runs) and the resulting test collections (including topics and relevance judgments) in archives. Nowadays, there is a large variety of collections, which makes it possible to systematically evaluate the reproducibility of a retrieval system with different document types in different domains (e.g., newswire or medical), with different languages, and even for different tasks. All of these resources provide an excellent basis for reproducible research.

However, as revealed by several systematic reviews for computational research [99] and cross-domain surveys [26], there are increasing concerns about the reproducibility of modern research, and some go as far as to say that there is a *reproducibility crisis*. The IR community acknowledged these reproducibility concerns, which are not least attributable to the increasing complexity and computational requirements of modern retrieval approaches, by inaugurating a dedicated reproducibility track at ECIR. By explicitly addressing reproducibility, some pitfalls, and methodological flaws, which are an obstacle to reproducibility, were revealed in the last years and will be reviewed in the following sections.

Beyond the Cranfield paradigm, it is of interest to analyze how a retrieval method performs in a real-world context, i.e., to evaluate the ecological validity of the conclusions drawn from a Cranfield experiment. While using the same queries for the topics allows the systematic evaluation of different retrieval systems, it is assumed that the same query formulation always expresses the topic's underlying information need. This does not hold in a real-world setting, where users formulate different queries for the same information needs, e.g., as exemplified by user query variants. Vice versa, the same query might originate from different information needs. Likewise, the somewhat objectified notion of relevance does not consider the pertinence of individual users. In this regard, Chapters 6, 7, and 8 propose solutions by user simulations and evaluations in online experiments.

Table 2.1: Terminology according to the ACM policy and Claerbout.

	ACM policy [448]	Claerbout [92, 113, 347]
Repeatability	Same team, same experimental setup	-
Reproducibility	Different team, same experimental setup	Providing the environment of the experiment in order to recreate the stated results
Replicability	Different team, different experimental setup	Reaching the same results with an independently created experimental environment

2.2 Reproducibility Terminology

In this work, we follow the terminology introduced by the ACM Policy on Artifact and Review Badging [448] when writing about reproducibility in more general terms. The policy specifies the terms *repeatability*, *reproducibility*, and *replicability* by the definitions given in Table 2.1. The three terms are based on the *International Vocabulary of Metrology* [28] and should be understood in succeeding order along an increasing level of generalizability. Repeatability describes experimental outcomes reconfirmed by the *same* team of researchers using the *same* (their own) original experimental setup. Reproducibility describes experimental outcomes reconfirmed by a *different* team of researchers using the *same* original experimental setup. Finally, replicability describes experimental outcomes reconfirmed by a *different* team of researchers using a *different* experimental setup. Similarly, Ivie and Thain [199] speak of *verification* to "see if it [the experiment] produces the claimed output" and of *validation*, when it "is the task of evaluating a result to see if the author's conclusions are warranted". In a wider sense, these definitions relate to the ACM definitions of reproducibility and replicability, respectively.

However, it has to be noted that there is a discourse about the correct way of using the terminology for reproducibility. There is no common sense about how the terminology should be used in general, e.g., Feitelson [127] proposed a reproducibility terminology for the SIGOPS community, which diverges from the previously outlined ACM definitions. For a more in-depth discussion about this topic, we refer the reader to Plesser [327], who reviewed the confusion about the two terms replicability and reproducibility. In conclusion, Plesser favored the current ACM terminology (v1.1) since it is in line with what Plesser referred to as the *Claerbout* terminology.[1] The corresponding background of this terminology was started by Claerbout and Karrenbach [92], who provided one of the earlier works discussing reproducibility in the context of repeatable experiments with digital documents. This chain of thoughts was adopted and extended by Donoho et al. [113] and Peng [347]; it is summarized and aligned with the ACM terminology in Table 2.1. Heroux et al. [178] also emphasized the inconsistent use of the two terms in the computational and computing sciences, and they also concluded that the Claerbout terminology (and

[1] In this work, Plesser compares the ACM terminology to the Clearbout definitions and discusses them as alternatives, since an earlier version of the ACM terminology basically swapped the definitions of reproducibility and replicability, see also [447].

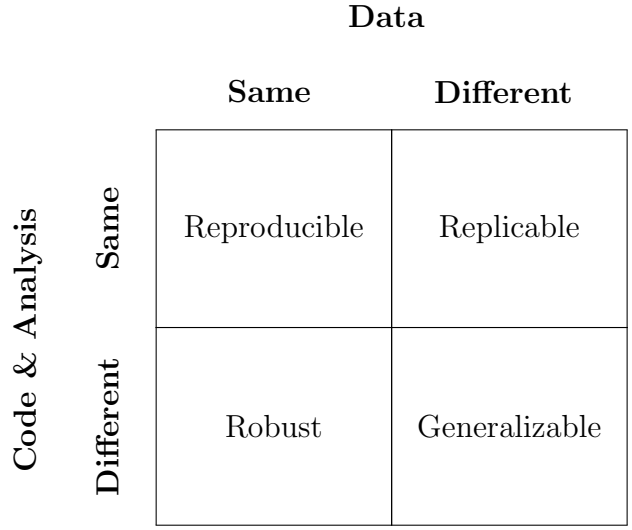

Figure 2.1: Reproducibility terminology according the to the NeurIPS definitions (reproduced from [324, 390]).

thus the ACM terminology v1.1) should be preferred since it received a broader scientific adoption and has been used since the early 1990s. This opinion is also supported by the *National Academies of Sciences, Engineering, and Medicine* [307].

Besides, the NeurIPS conference [324] adopted another terminology from *The Turing Way Community* [390] for which a confusion matrix is illustrated in Figure 2.1. This terminology is based on the four adjectives *"reproducible"*, *"replicable"*, *"robust"*, and *"generalizable"* depending on whether the data and the experimental setup are the same as or different from those in the original experiment. Not least, all of this confusion can be attributed to a plethora of terms. For instance, De Roure [349] listed 21 "r-words" related to reproducible science. Simply applying the ACM terminology makes it unclear what exactly has been changed in reference to the original experiment. PRIMAD, which is discussed in the following section, can be seen as an answer to this underspecification.

2.3 PRIMAD

As mentioned in the previous section, authors use the terminology of reproducibility and related terms in different and sometimes inconsistent ways when reproducing IR experiments and disseminating the results. Despite the ACM Policy on Artifact Review and Badging, there is still enough freedom of interpretation of how these definitions can be applied to the IR experiment in different ways. As an example, we refer the reader to the ECIR reproducibility track [171], where the authors use terms like reproducibility, replicability, robustness, and generalizability as they see fit. For instance, the authors sometimes used the terminology of reproducibility in inconsistent ways: Müllner et al. [303] or Fröbe et al. [147] validated the *reproducibility* by evaluating the reimplemented experiments with new datasets, while Ferro and Silvello [142] referred to a *reproducibility* analysis when reusing the dataset of the original experiment and they referred to a *generalization* when revalidating the reim-

Table 2.2: PRIMAD according to its original definitions (given in [132]).

Component	**General**	**System-oriented**	**User-oriented**
Resarch Goal	Purpose of a study	High-quality ranking	Research question accompanied by a hypothesis
Method	Approach of the study	Mapping of query to document ordering	Experimental setting including study type and other aspects
Implementation	Implementation of a method	Retrieval system	Environment, user group, conditions
Platform	Underlying hard- and software	Retrieval system	Environment, user group, conditions
Data	Input data and parameters	Test collection	Testbed (including document collection) and collected user data (*user = data generator*)
Actor	Experimenter	Agent undertaking the experiment	Agent undertaking the experiment (might affect users)

plemented experiment with new datasets. In contrast, Yang et al. [431] referred to a *generalization* when applying the reimplementations to another classification task. Judging from the terminology alone, it is not clear what exactly has been changed and under which circumstances the former experiments could be validated.

PRIMAD [132, 146] can be seen as an answer to this underspecification. The acronym stems from the components of a typical experiment in the computational sciences, including the **P**latform, **R**esearch Goal, **I**mplementation, **M**ethod, **A**ctor, and the **D**ata. By defining which PRIMAD components were modified ("primed"), it can be specified how the reproduced experiment "adds up" to the former experiment it is compared to. As stated by Rauber et al. [146, p. 129] "reproducibility is never a goal in itself", but rather a means to an end. Successfully reproducing an IR experiment verifies its *internal validity*, but does not provide new insights. By modifying some components of the original experimental setup, we aim to assess the *external validity* [149,293]. For instance, evaluating a retrieval method with another test collection provides insights about the performance in a different context.

Originally, PRIMAD has been outlined in two different ways, covering system- and user-oriented experiments separately. Table 2.3 provides an overview of the definition by Ferro et al. [132]. In the following, we briefly summarize these definitions. The platform comprises the hard- and software underlying the actual implementation in system-oriented experiments; but may also include the experimental environment, the user groups, and the conditions in user-oriented experiments. The research goal describes the purpose of the study. If the experiment is aligned with the Cranfield paradigm, as often in system-oriented IR experiments, the research goal is a high-quality ranking. Nevertheless, the study can focus on other aspects based on research questions and the corresponding hypothesis, as is often the case

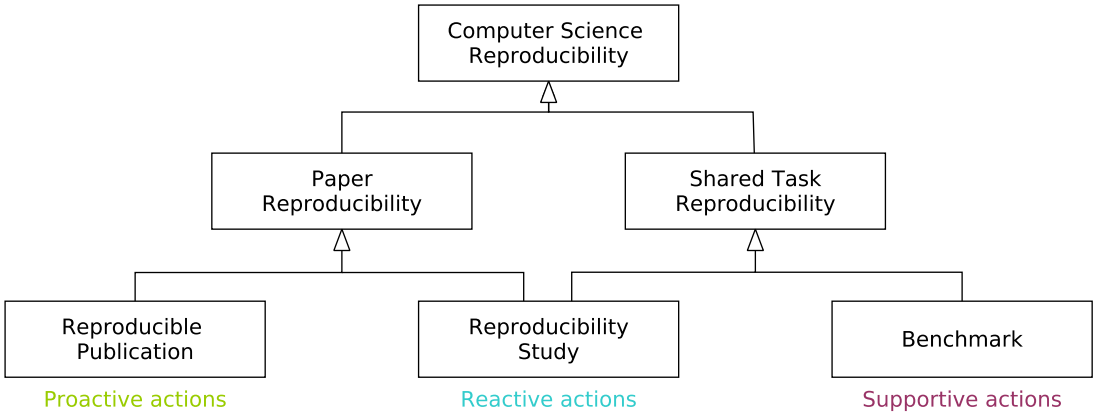

Figure 2.2: Reproducibility taxonomy by Potthast et al. (reproduced from [329]).

for user-oriented experiments. The implementation is closely related to the method. What is more formally described by the method is translated by the implementation into operations that can be conducted *in silico* in the case of a system-oriented experiment. In user-oriented experiments, it is not limited to the retrieval system but also includes the previously mentioned experimental environment. In system-oriented IR research, the study focuses on the actual retrieval approach covered by the method in the PRIMAD model. From a technical point of view, it describes the mapping of query-document pairs to a ranking score; but it also includes experimental setting, i.e., the type of user study. The actor component represents the experimenter who conducts the experiments. It is the one who operates the computer, implements the experiments, types commands, et cetera in a system-oriented IR experiment. In user-oriented experiments, special attention must be paid to how the actor might influence the experiment by affecting user behavior. By its original definition, the data component comprises the input data and the parameters required to run the experiments; in user-oriented experiments, users are considered as *data generators*.

While PRIMAD considers most of the relevant components for reproducible experiments, it is a rather abstract taxonomy leaving certain components underspecified. Furthermore, the separate definitions do not allow a holistic approach to describing an experiment that includes both system- and user-oriented aspects. Both shortcomings are addressed in Chapter 3 by extending the PRIMAD model with an additional user component U and by outlining each subcomponent in detail. As part of the related work, we structure the literature review by the PRIMAD model in combination with the taxonomy by Potthast et al. [329], which is described in the following section.

2.4 Reproducibility Taxonomy by Potthast et al.

The previous section introduced PRIMAD, which proposes a taxonomy based on the experimental components that can affect reproducibility. A different taxonomy is introduced by Potthast et al. [329], whose concept is based on actions towards reproducibility. Figure 2.2 shows the corresponding hierarchy of concepts that is based on *proactive*, *reactive*, and *supportive* actions.

Unlike other scientific domains, the computational sciences benefit from the fact that the experimental setup can be made available for future reuse "with no extra costs" according to Potthast et al. [329]. *Proactive* actions include sharing the (meta) data, the code, and the workflow, as well as the services for hosting these. The original experimenters must mainly undertake these actions and prepare their experimental setup for others, keeping reproducibility in mind.

On the other hand, *reactive* actions include artifact evaluation, reimplementation studies, systematic reviews, and meta-analysis as part of shared tasks. Potthast et al. see these actions realized by the scientific peer-review framework, as part of which the reviewers could also be considered as other experimenters or reproducers.

Finally, *supportive* actions are mostly part of benchmarks and are realized as part of constructing corpora, the measurement theory, the (software) library development, and shared task events. Potthast et al. see these actions implemented by experiment frameworks and Evaluation-as-a-Service (EaaS) platforms [190].

In the following sections, we use these three types of reproducibility actions to categorize countermeasures, which can be made in answer to the problems of irreproducibility related to the particular PRIMAD components.

2.5 Factors of Irreproducibility

As part of the large-scale survey [26], Baker's questionnaire [474] listed *causes* for irreproducible outcomes, from which we derive five more abstract (interrelated) factors of irreproducibility, including **unethical actions** (*fraud, selective reporting*), **issues of scholarly communication** (*pressure to publish, insufficient peer review*), **statistical and experimental flaws** (*no robust results, low statistical power/poor analysis, variability of standard reagents, insufficient oversight, poor experimental design*), **unavailability of the experimental setup** (*unavailability of raw data, methods and code*), and **missing expertise** (*mistakes or inadequate expertise in reproduction efforts, particular technical expertise that is difficult for others to reproduce, bad luck*). We align selected references to these categories in the following, highlighting related reproducibility issues.

2.5.1 Unethical Actions

Baker considered selective reporting and fraud as factors causing irreproducibility [26, 105]. Selecting positive research findings and intentionally withholding negative outcomes can also be seen as a concrete action of fraud [105]. Another action that falls into this category is the falsification of data [125]. These actions are considered unethical and are often done with intention. However, solutions to these problems are not within the scope of this work. Every individual should be obliged to follow ethical guidelines, especially when conducting scientific experiments. But also funding agencies can motivate researchers to follow guidelines of *Good Research Practice*, for instance, those proposed by the German Research Foundation [459]. In general, unethical actions should also be addressed on an organizational level, e.g., by journal editors or the scientific community as a whole [381].

2.5.2 Issues of Scholarly Communication

Research findings are usually communicated to a broader audience in journals or conference proceedings. Before publication, the journal or conference submissions will undergo a peer review process. In her survey, Baker [26] included insufficient peer reviewing as a factor for irreproducibility. Lee et al. [240] as well as García et al. [153] highlighted that the peer review process could be biased. Furthermore, the extent to which technical details in scientific writing can be reported is limited [156]. More concrete details about how the experiments were implemented can only be found in logs or the source code of computational experiments. Similarly, Ivie and Thain [199] considered the required compromises between concrete instructions for a computer and the more abstract means of human communication as a major threat to reproducibility. Finally, the success of an academic career and the scientific impact is often measured in terms of the published output, leading to the *pressure to publish* that may lead to insufficient oversight and a flawed research design [182,403]. Especially in performance-driven domains, Sculley et al. [368] pointed out that the scientific rigor may suffer from the *leaderboard chasing*.

2.5.3 Statistical and Experimental Flaws

Ioannidis' simulations [198] showed that most findings are likely to be false due to low statistical power or poor analysis. Baker [26] included the variability of standard reagents in her survey as an irreproducible factor. Analogously, this complies with uncertainties about why particular combinations of retrieval methods and test collections perform well while others do not. According to Jones et al. [210] and Fuhr [149], it is challenging and only sometimes possible to assess what characterizes a test collection.

In general, a poor research design [26] can be caused by the following reasons, including no pre-registration of hypotheses [255], leading to cherry-picking a hypothesis with adequate p-values (a.k.a. "p-hacking" or "data dredging") [198], multiple comparisons problem [78], or simple holdout without cross-validations [339]. Especially when benchmarking different approaches, it is critical to use strong baseline methods for comparison [13, 247, 430].

2.5.4 Unavailability of the Experimental Setup

For the sake of transparency and reusability, the experimental setup should be preserved for future studies. According to Potthast et al., the computational sciences have the privilege to preserve the underlying setup of the experiments with little or no costs [329] in contrast to other scientific fields like chemistry, geography, or the material sciences. However, recent work has shown that, especially for computational studies, the reported experiments often need more documentation about the methods or the source code is unavailable [26, 99]. Likewise, unreported details about hyperparameters [101], as well as closed/paywalled data, can be an obstacle to making the experimental setup fully reproducible.

2.5.5 Missing Expertise

Finally, reproducibility cannot be guaranteed due to different levels of expertise between the original experimenters and the reproducers [26]. While in some cases, there is the need for particular technical expertise, or likewise, inadequate expertise may lead to mistakes, in other cases, there is also the chance of bad luck. Especially as part of user-oriented or online studies [186], it is critical to be aware of confounding variables and (cognitive) biases that may influence the experimental results and the conclusions drawn from them.

2.6 Reproducible Information Retrieval

This section reviews the reproducibility issues and countermeasures in IR research. In order to provide the reader with a general overview, we start this section with a timeline that covers substantial community-wide achievements. Afterward, we review issues, proactive solutions, and reactive reproducibility studies for each PRIMAD component. Many of the conclusions and outcomes in this section are based on the literature of the ECIR reproducibility track, which is also discussed in Subsection 2.6.1 and reviewed as part of a structured Table A.1 in the Appendix A.

The meta-evaluations by Armstrong et al. [13] questioned the reproducibility of improvements over baselines due to inconsistent evaluation protocols that did not consider state-of-the-art baselines. However, the IR community began only a few years later to enforce countermeasures for reproducibility starting in the middle of the last decade. Table 2.3 provides a timeline of reproducibility attempts and achievements from 2015 until 2022.

Even though earlier work also highlighted the importance of reproducible experimentation [301, 394, 395, 433], the IR community started to enforce the reproducibility efforts from 2015 with the inauguration of the ECIR reproducibility track that invites researchers to report their experiences with reproducibility studies also including negative results [171]. The corresponding body of literature mainly covers reactive reproductions, and a more detailed analysis is provided in the following sections. In the same year, the TREC conference promoted the idea of Open Runs, according to which the submitted run files should be backed by an open-source code repository that is, for instance, hosted on GitHub [414]. Approximately 25% of the participants in 2015 made their run submissions *open*. However, the TREC organizers concluded with moderate success by considering their initial attempts too simplistic. In this regard, they highlighted technical underspecifications, data dependencies, and underestimating the additional overhead required to prepare an experiment for reproducibility. In the following years, these attempts were unfortunately not actively promoted. Finally, SIGIR hosted the RIGOR workshop that offered a venue for reports about repeatability, reproducibility, generalizability, and inexplicability [11]. In addition, workshop participants were invited to contribute open-source systems to a reproducibility challenge that became known as the Open-Source Reproducibility Challenge (OSIRRC) in the later years.

In 2016, the results of OSIRRC were reported as part of the ECIR proceedings by Lin et al. [249]. Motivated by the goal of building robust and reproducible open-source baselines, efforts were made to standardize the evaluation environment and protocol for different implementations of the same retrieval methods. They

Table 2.3: Reproducibility attempts in IR research from 2015 to 2022.

2015	The TREC conference introduces the idea of Open Runs [414]; RIGOR workshop at SIGIR [11]; ECIR inaugurates reproducibility track with three studies [171].
2016	PRIMAD is introduced as a result of the Dagstuhl seminar 16041 [132, 146]; Report of OSIRRC 2015 artifacts resulting from RIGOR [249]; Lucene4IR workshop [20]; ECIR proceedings include four reproducibility studies [131].
2017	Anserini toolkit is introduced at SIGIR [427].
2018	Ferro and Kelly survey the community about the ACM Artifact Review and Badging [134]; First iteration of the CENTRE workshop [136]; ECIR proceedings include four reproducibility studies [320].
2019	Meta-evaluation by Yang et al. [430] reconfirms the problem of weak baselines as already pointed out by Armstrong et al. [13]; OSIRRC workshop at SIGIR [93]; Second iteration of the CENTRE workshop [133]; ECIR proceedings include nine reproducibility studies [21].
2020	ACM Artifact Review and Badging Version 1.1; Pyterrier introduced at ICTIR [267]; Lin and Zhang revalidate the OSIRRC 2015 artifacts [254]; ECIR proceedings include eight reproducibility studies [212]; ACM RecSys inaugurates reproducibility track with 2 reproducibility studies [358].
2021	SIGIR implements the ACM Artifact Review and Badging; Pyserini toolkit is introduced at SIGIR [250]; Data catalog `ir_datasets` is introduced at SIGIR [265]; ECIR proceedings include eleven reproducibility studies [183].
2022	SIGIR inaugurates reproducibility track with seven studies [9]; Reproducibility tutorial at SIGIR [260]; ECIR proceedings include eleven reproducibility studies [170].

showed a large variability of retrieval performance between different systems even when implementing the same method. In addition, they highlighted the challenges like platform dependencies, unavailable scripts, or deviating parameterizations. As part of the Dagstuhl seminar 16041 [132, 146], PRIMAD was introduced as a collaborative result. Even though it discussed relevant components of reproducible IR experiments, it was not put into practice but rather outlined by anecdotal examples for each PRIMAD component.

In 2017, Yang et al. [427] released the Anserini toolkit that has since served as the de facto framework for reproducible baselines. It can be seen as a follow-up of the previous open-source attempts [11, 20, 394, 433]. It provides a more research-friendly interface to the Lucene library and also provides regression tests for many standard test collections. It was used as part of many reactive reproducibility studies [157, 332, 417, 429, 437].

In 2018, Ferro and Kelly [134] surveyed the community about the ACM Artifact and Review Badging that was already successfully implemented as part of ACM SIGMOD. Overall there was a positive attitude towards assigning badges to reproducible papers, as seen from the survey's results. The ACM SIGIR Artifact Badging [450] was inaugurated in 2021 and offered an additional review of accepted submissions to TOIS, SIGIR, CHIIR, and ICTIR. Depending on the degree of reproducibility, the publications are given a badge in the ACM Digital Library. The review process focuses on transparency by using the OpenReview platform [476].

In the same year, the cross-venue workshop CENTRE was introduced at CLEF [136], NTCIR [353], and TREC [375]. CENTRE invited the workshop's participants to reproduce previous submissions to the respective conferences. As part of these conferences, several measures for reproducibility were introduced that will be later on discussed in Chapter 4. Even though the previous years showed that there was an increasing interest in the topic of reproducibility, the number of participants could have been higher, with only one group participating at CLEF, two groups at NTCIR, and one group at TREC. In the following year, the CLEF workshop had one participating group, and TREC discontinued CENTRE altogether. NTCIR continued with CENTRE but moderate participation. While it is out of the scope of this work to reach any definitive conclusions, we assume that these efforts may have competed with the ECIR reproducibility track. Regarding academic approval, there is a higher reward when submitting the experimental results to a peer-reviewed track, considering the laborious work required for a good reproducibility analysis.

In 2019, Yang et al. [430] reconfirmed the problem of weak baselines that were already pointed out by Armstrong et al. [13] by conducting another longitudinal analysis including a more recent time frame and then state-of-the-art Deep Learning (DL) ranking methods. As a follow-up of OSIRRC, SIGIR hosted the workshop once again. This time, participants prepared reproducible systems and experiments with the help of the containerization technology based on Docker [93]. The workshop resulted in a rich library of Docker images that can be used in combination with a dedicated software toolkit in order to rerun the experiments on purpose.

In 2020, the ACM updated the Artifact Review and Badging to Version 1.1 by aligning the terms of reproducibility and replicability to the conventions in other research domains. As part of another reactive reproducibility study, Lin and Zhang [254] revalidated the OSIRRC artifacts from 2015. They showed that the results could be reproduced for one of seven retrieval systems. While the four years

between the original evaluations and the revalidations seem to be a long time in terms of software release cycles, this highlights the importance that reproducibility should also be guaranteed in the long term.

In 2020 and 2021, several helpful software toolkits were introduced to prepare an experiment proactively for reproducibility. Pyterrier [267] is a Python interface for the established retrieval platform Terrier [266]. In addition to the ease of use due to Python, it also introduced a declarative programming style allowing better readability of the implemented retrieval method and reducing the gap between the source code and the descriptions in the publication. Likewise, Pyserini [250] offers an easy-to-use Python interface to the Anserini toolkit. Like Anserini, Pyserini also found application in several reproducibility experiments [243, 263, 332]. Finally, SIGIR inaugurated a reproducibility track in 2022 with seven studies [9].

2.6.1 ECIR Reproducibility Track

In the following, we provide an overview of the ECIR reproducibility track based on the structured Table A.1. The selected publications cover all reproducibility studies of ECIR from 2015 until 2022. We consider these 50 papers to represent how reactive reproducibility studies in IR research are usually conducted. More specifically, we review what topics have been addressed by reproducibility studies so far and by which methods and based on which criteria the authors considered the reproduction successful or failed.

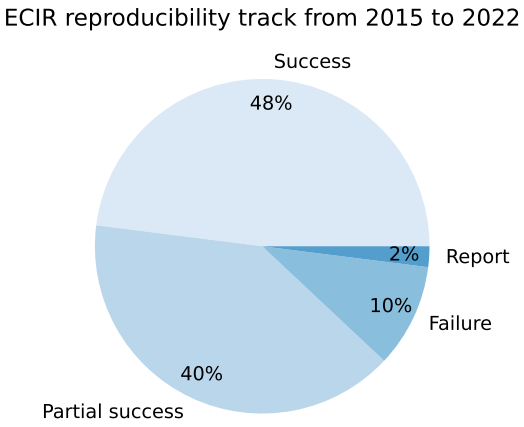

Figure 2.3: Our analysis (cf. Table A.1) regarding the success of studies published in the ECIR reproducibility track.

In 2017, there were no reproducibility studies submitted. However, there is an increasing trend in the number of accepted papers, which underlines that reproducibility has become an integral part of the ECIR community over the past years. Figure 2.3 shows the success rate of the 50 papers. While in 24 papers (48%), the authors considered their reproductions to be a success, in 20 papers (40%), they concluded with partial success. In five papers (10%), they considered their reproductions to be a failure.[2] These statistics show that reproducibility issues are also present in IR and that reproducibility cannot always be taken for granted.

In most cases, the authors reimplemented the original experiments, given the descriptions in the corresponding publications of the original experiment. However, in seven out of 50 papers (14%), the authors were able to reuse existing implementations and used them to evaluate the validity with different datasets or another kind of experimental setup [54, 173, 237, 254, 302, 303, 417]. Some reproducibility studies validated the state of the art of a particular research problem and do not

[2]Note that one paper provides anecdotes about how research outcomes find their way into the industry and the paper does not contain empirical evaluations [157].

focus on a single method but comprised benchmark experiments including multiple methods. These studies compare the selected methods in a systematic and fair way by providing a consistent experimental setup. Normally, the dataset and experimental conditions were kept fixed for all methods. The authors aimed to answer if there is overall progress in the effectiveness and if previous conclusions can be reproduced in the common ground. Examples include sentiment classification in tweets [169], ad-hoc retrieval methods [249, 254], author identification [328], statistical stemmers [370], text summarization of legal texts [49], recommender system bias [53, 54], index compression techniques [276], entity alignment [47], web page segmentation [226], sentiment analysis [302], loss functions for image retrieval [51], or methods for systematic literature reviews [237].

Typically, authors validated a successful reproduction by the Average Retrieval Performance (ARP) (or more specifically, the system effectiveness) in 40 out of 50 papers (80%). For 24 out of these 40 papers (60%), the validation was limited to only comparing averaged performance scores. In comparison, 16 out of these 40 papers (40%) provided additional statistical analysis based on significance testing, often with the help of paired t-tests. In these cases, authors claimed their reproduction to be successful if their reimplemented approach significantly outperformed the retrieval effectiveness of the original experiments. While this methodology complies with the typical evaluation design in IR experiments, when a new system is compared to a baseline, it lowers the rigor of determining a "successful" reproduction. This evaluation approach may confirm that the reproduced experiment results in comparable or even better retrieval effectiveness, but it neglects the exact similarity between the original and reproduced results. In theory, computational experiments would allow a bitwise similarity of the experimental artifacts [199]. However, comparing the ARP does not account for differences between topic score distributions or rankings for particular topics that could result in similar average scores.

We note the overall lack of user-oriented reproducibility studies as part of ECIR and emphasize that future research should focus on user-oriented aspects in reactive reproducibility attempts. Some studies included additional user judgments or simulated click behavior in the evaluations, but the user behavior was not the main focus of these reproducibility studies. For instance, Mackie et al. [271] evaluated their reproduced text summarization methods with additional crowdsourced user judgments. Likewise, Bhattacharya et al. [49] evaluated summarization methods for legal texts with a further qualitative analysis based on judgments by domain experts. Oosterhuis and de Rijke [312] simulated user click behavior, which was also adopted by Wang et al. [417], to validate their reproductions.

To our knowledge, the only reproducibility study involving users was recently presented by Roy et al. [351]. They analyzed four different Search Engine Result Page (SERP) layouts in user studies and validated if the conclusions from earlier studies still hold almost a decade after the original studies were conducted. They concluded by their results that both the SERP layout and the task complexity impact the user interactions, and the earlier observations mainly hold over time.

The following subsections align the reactive reproducibility studies of ECIR to the PRIMAD taxonomy. We review what kind of reproducibility issues can occur and what can be learned from these studies in Subsection 2.6.2, and how countermeasures in the form of proactive solutions are implemented in Subsection 2.6.3.

2.6.2 Reactive Studies and Reproducibility Issues

In the following, we review common issues related to irreproducibility and align them to the six PRIMAD components.

Platform

If the experimental outcomes depend on the computational environment in which they have been conducted, it can impact the reproducibility. For instance, when rerunning the experiment on **an updated platform**, some of the underlying software dependencies might have changed, leading to inaccuracies or even complete execution failures, which do not support the original conclusions. Lin and Zhang [254] revalidated the artifacts of the OSIRRC workshop [249] four years after they were made with relatively moderate success. They could reproduce the same results for only one out of seven retrieval systems by rerunning the old experiments with newer hardware and an updated operating system. The remaining attempts resulted in exceptions, segmentation faults, and compilation errors or depended on external resources that were not available anymore. Four years is a long time under consideration of the pace of software release cycles. This reproducibility attempt shows that providing long-term reproducibility of experimental results requires ongoing software maintenance and revalidations in the form of regression tests.

With regard to DL approaches, it has been acknowledged in several studies that the **non-deterministic behavior** of some GPU operations can cause differences in the effectiveness of the outcomes [137, 306]. For the implementations of DL approaches, GPUs are used to parallelize the training process and the later inference. With a particular focus on reinforcement learning, Nagarajan et. al [306] emphasized that even if the algorithm considers countermeasures to non-deterministic behavior, the non-deterministic influences of the hardware operations can impact the reproducibility. As part of their OSIRRC contribution, Ferro et al. [137] reimplemented the Neural Vector Space Model and showed how these non-deterministic effects influence ranking results. While there were only minor differences in the average system effectiveness, the reproductions based on the GPU implementations led to entirely different document rankings. Likewise, the GPU company NVIDIA acknowledges that bitwise reproducibility is not guaranteed even if the same GPU architecture is used due to truly random floating point rounding errors [475].

Furthermore, Hasibi et al. [173] showed that it could also be critical if the experiment relies on resources retrieved from an **external platform**. They reimplemented experiments with the help of an entity-linking system. They could only partially reproduce the original outcomes due to an updated Application Programming Interface (API) of a web service running on an external platform that was not explicitly documented.

Research Goal

The research goal describes the purpose of the study [132, 146]. If the experiment is aligned with the Cranfield paradigm, as often in IR experiments, the research goal is a high-quality ranking. In this regard, it is an essential question of how it is designed and evaluated. Fuhr [149] emphasizes that it is crucial to formulate the hypotheses or research questions before conducting the experiments. Especially, the low entry

barrier of modern Machine Learning (ML) frameworks allows practitioners to start **experimenting without clearly formulated research questions or any hypothesis** about the outcomes [255], but with the intent to improve the effectiveness leading to the earlier mentioned problem of ***leaderboard chasing*** [368].

Consequently, the experimenter might be inclined to "search for" experiments that satisfy a **post hoc formulated research question** leading to issues like **p-hacking** [78]. In this regard, it is crucial to apply correction methods to avoid the **multiple comparisons problem** [150]. Likewise, **underpowered statistics and effect sizes** can lead to overclaiming of findings and **conclusions that are not supported by empirical evidence** [111,114,149,229,401]. The overall evaluation criteria should give answers to the research questions [231], and more recently, Ferrante et al. [129] stressed that evaluation measures should be interval-scaled when conducting significance tests.

Armstrong et al. [13] pointed out the lacking upwards trend of the overall retrieval effectiveness from 1998 to 2008 and highlighted the importance of including strong baselines in the experimental evaluations. **Improvements over weak baselines** can be illusionary when the method is compared to strong and adequate state-of-the-art baselines. Similar observations were made by Yang et al. [430] in a more recent study by evaluating DL-based approaches for ad-hoc retrieval.

Rendle et al. [343] went as far as to state that findings should be questioned unless they are obtained through extensively tuned baselines by the research community. Many studies do not meet this requirement, as confirmed by several systematic reproducibility benchmarks that follow a principled approach. Usually, the reviewing authors try to include all state-of-the-art approaches that target a narrow research problem (task) and that were accepted at major conferences and journals. Afterward, they try to obtain and execute the corresponding source code or reimplement the methods based on the descriptions in the publication. In this manner, it is possible to compare different approaches for the same research goal in a standardized computational environment, i.e., all of the experiments have the same evaluation setup, including the platform, the same dataset, and evaluation measures.

There exist several of these systematic reproducibility benchmarks in neighboring research disciplines. Dacrema et al. [108,109] showed that reproducibility issues affect recommender systems. Only a minority of previous research was reproducible (12 out of 26 papers), and most of the methods were still outperformed by less complex baseline methods (11 out of 12 papers). Similar observations were made by Ludewig and Jannach [262] for session-based recommender systems. Another systematic evaluation of time series forecasting was made by Makridakis et al. [273], who showed that ML approaches could be outperformed by simpler statistical methods. Comparable observations were made in systematic reproducibility studies of computational linguistics [421] and NLP [38,39].

As part of the ECIR reproducibility track, several systematic reproducibility benchmarks were conducted [47,53,54,226,233,237,276,283,300,308]. Moreo and Sebastiani [300] conducted a systematic analysis of methods for *learning to quantify* and showed that it is critical to **account for properly tuned hyperparameters of the baseline methods** and also carefully consider the evaluation protocol when benchmarking them against novel methods. Otherwise, the reported improvements over the baseline can be illusionary. Similarly, Maurera et al. [283] analyzed generative adversarial networks for collaborative filtering and showed that the imple-

mentations of the original paper could be successfully reused, but the improvements of the analyzed framework were not replicable when compared to a broader range of baselines, i.e., it was not competitive against conventional baselines.

Berrendorf et al. [47] conducted a systematic revalidation of entity alignment methods and provided a more in-depth analysis regarding the influence of hyperparameters and **training/test data splits**. They highlighted the limitations of the state-of-the-art approaches. In this regard, Lipton and Steinhardt [255] pointed out the dangers of **misinterpreting the sources of empirical gains**. Kusa et al. [237] conducted a systematic evaluation of two DL-based methods for systematic literature reviews across 23 datasets and showed that only one method was reproducible. In addition, the authors introduced a simpler yet more effective method.

Boratto et al. [54] conducted a systematic analysis of recommender systems that mitigate consumer unfairness, for which 8 out of 15 systems were usable for the experiments. Likewise, they showed in another study [53] that selected recommender systems could be reproduced when evaluated in the context of *massive online open courses* but also highlighted that undesired effects, i.e., different forms of bias related to the popularity of the recommended items, were reproduced as well. Similarly, Kowald et al. [233] revalidated that the popularity bias in recommender systems is not only present in the movie domain but also in the music domain, and Neophytou et al. [308] reconfirmed that popularity and demographic biases influence the effectiveness of recommender systems.

Other ECIR studies validated if the research goal still holds in a different context. For instance, Bhattacharya et al. [49] analyzed the generalizability of domain-independent summarization algorithms when applied to legal texts and also analyzed domain-specific summarization algorithms for legal texts in different languages. Althammer et al. [5] revalidated a BERT model based on paragraph-level interactions in the legal and patent domain and could show that BM25 is still a strong baseline for the document retrieval task. In contrast, the Transformer-based methods achieved reasonable results in both domains for the paragraph retrieval task. Yang et al. [431] reproduced retrieval methods based on linear transformations of word embeddings and showed that it generalized well on datasets with other languages than in the original experiment. Mukherjee et al. [302] revalidated methods of aspect-based sentiment analysis, and Berrendorf et al. [47] revalidated methods for entity alignment. Both studies showed performance drops when the methods were reproduced in a **production-like evaluation or real-world setting**.

Sometimes, the original study's context changes over time. For instance, it was shown by Shrestha and Spezzano [369] that a fake news detection method could not be reproduced as the underlying writing style - in this particular case, the style of (fake) news - changed over time. Another example is given by Fröbe et al. [148], who revalidated anchor text as a ranking feature with substantially larger datasets. They showed that anchor texts are still effective for navigational queries but found differences between the term distributions of anchor texts and today's queries.

Implementation

What is more formally described by the method is translated by the implementation into operations that can be conducted *in silico* [132, 146]. By covering a broader spectrum of different computer science fields, Collberg and Proebsting [99] conducted a large-scale attempt to repeat over 600 experiments in ACM papers. Their

analysis showed that in one-third of the cases, they could retrieve and run the source code with reasonable effort. In other cases, **the source code was not retrievable**, the authors did not reply, or it took excessive time to run the experiments.

Slightly better outcomes were observed by Raff [337], who successfully reimplemented 162 out of 255 ML papers from scratch and evaluated the reproducibility with the help of 26 different features. If at least 75% of the original paper's claim could be successfully validated, the paper was considered as reproduced. The analysis showed that the following features impacted how well a paper could be reimplemented, including the rigor, readability, algorithm difficulty, pseudo code, primary topic, specification of hyperparameters, computational requirements, author's reply, and the number of equations and tables.

Kriegel et al. [234] showed that implementations of even simple algorithms (like k-nearest neighbors) could result in different orders of magnitude when evaluating the efficiency. Lin et al. [249] analyzed the results of OSIRRC and showed some variability in the retrieval performance between different implementations of the same method. As a follow-up study of Mühleisen et al. [301], Kamphuis et al. [213] compared different implementations of the BM25 method and showed that there were (no significant) differences between the effectiveness scores. While these differences did not significantly impact the ARP, they still might lead to more substantial differences for particular rankings, which can impact user behavior. To our knowledge, this has not been evaluated yet.

Both Drummond [115] and Crane [101] pointed out that the **availability of the source code should not be overestimated**. According to them, it only allows rerunning experiments and does not provide any insights about reproducibility in a different context or if it is even reproducible without the source code by the original experimenters. This critique is in line with the idea of shifting the focus of a reproducibility study from internal to external validity [149].

As part of the ECIR reproducibility track, Kusa et al. [237] pointed out that more than providing the code alone is needed, and **it can be critical not to report the versions of the software libraries**. Papariello et al. [319] provided an example of a failed reproducibility study despite an in-depth reimplementation study. Another example is given by Berrendorf et al. [46], who had no success with the reproduction and concluded that the original **implementation deviates from the descriptions in the paper**. On the other hand, several authors could successfully reimplement and consequently reproduce earlier work [128,169,263,270].

Method

The method component in the PRIMAD model describes the actual retrieval method, for instance, the mapping of query-document pairs to a ranking score [132,146]. Very often, it is the study's main focus. Even though the common retrieval pipeline has standardized processing steps, including the removal of stop words, lexical unit generation, and the final retrieval method, several methods exist for each particular processing step.

Ferro and Silvello [143] showed that these different methods for standardized processing steps impact the final ranking effectiveness to different extents. Furthermore, Silvello et al. [370] analyzed the reproducibility of language-agnostic statistical stemmers over test collections with different languages. An **underspecification of**

the retrieval pipeline in the final publication can lead to unwanted freedom of interpretation when an experiment is reproduced. For instance, Yu et al. [437] pointed out that the normalization of the tf-idf features had an impact on the retrieval performance in their reproducibility experiment — a detail that was not pointed out in the original study report. Likewise, Roy et al. [350] showed the impact on reproducibility when removing markup artifacts from web documents as part of the data preprocessing. Lin and Yang [253] showed that score ties have to be broken deterministically; otherwise, they can affect the reproducibility of the document ranking. As a solution, they favored external collection identifiers. Sometimes simplifications of the method result in a similar performance, and less complexity should be generally favored [380].

As shown by Kamphuis et al. [213], even for the same method (in this case, BM25), different implementations exist and can impact the final results. Thus, the concrete method should be described or referenced more clearly. On the other hand, too complex presentations, what Lipton and Steinhardt [255] refer to as **unnecessary "mathiness"**, can obfuscate clarity, making the method harder to reproduce.

At ECIR, several reproducibility studies exemplified the impact of changing the underlying method of particular processing steps. Oosterhuis and de Rijke [312] systematically validated the reproducibility of two different optimization algorithms for the task of online learning to rank. They showed that Dueling Bandit Gradient Descent did not reproduce well in noisy data environments or under the assumption of a non-cascading user model, being inferior to the Pairwise Differentiable Gradient Descent algorithm. Schliski et al. [364] reproduce node2vec - a graph embedding method and conclude that they cannot achieve structural equivalence to the skip-gram model as stated in the previous works. Li et al. [243] showed that preprocessing could influence the effectiveness of dense retrievers as well. Pradeep et al. [332] reproduced and improved in this context the effectiveness of a general cross-encoder reranking pipeline by varying, for instance, the loss function or the first- and second-stage ranking methods. Wang et al. [416] revalidated a method for systematic literature reviews with more recent datasets. They successfully reproduced the methods but failed to replicate them, presumably, due to the effects of **deviating document preprocessing**. Li et al. [243] studied the reproducibility of a pseudo-relevance feedback ranking method combined with a language model and concluded by the negative results that the method is not generalizable with a reranking method based on another Large Language Model. Bleeker and de Rijke [51] compared alternatives to loss functions in the context of image caption retrieval as part of a reproducibility experiment and concluded with negative results.

Actor

In the PRIMAD taxonomy, the actor describes the experimenter who operates the computer, implements the experiments, etc. [132,146], but in the broader sense, this person also decides about the study design and authors the corresponding publication. In this regard, the actor is obliged to scientific rigor [149].

According to Ivie and Thain [199], most issues of reproducibility in the computational sciences stem from finding compromises between the concrete instructions a computing machine requires and the more **abstract means of human communication** in computational research. Of course, a different actor does not have to

be another researcher but can also be a future self of the original experimenter, for whom the experimental setup should be properly documented.

As the actor component introduces human factors into the computational experiment, it is important to consider different forms of **cognitive bias**. Before publication, a study usually undergoes the peer-reviewing process. The reviewer could also be considered as another actor who is asked to confirm the validity of the claimed conclusions. For instance, as part of the review process, several forms of **bias towards prestige, affiliation, nationality, language, and gender can occur** [153, 240]. Even if the review process is blinded, it is not guaranteed that the reviewers are not biased towards specific contents, against interdisciplinary research, and towards positive outcomes. However, also the authors have to be aware of possible pre-assumptions that could influence the experimental outcomes like **confirmation bias** [152] or the **Dunning-Kruger effect** [235].

For many experiments, it is favorable to achieve actor-independence, i.e., the experiments should be repeatable by anyone interested in rerunning them, for instance, like it is implemented by EaaS platforms [190]. Nonetheless, the authors should feel responsible for **supporting follow-up research after publication** and helping others to reproduce it. However, authors often do not supply support upon request [99, 337]. Potthast et al. [328] conducted a large-scale reproducibility analysis of author identification methods with students. They systematically analyzed the ease of reproducing the original target study regarding several criteria. This study highlights that it is important to consider how the results are communicated and documented and whether the experimental artifacts are reconstructible.

Data

PRIMAD defines data as the component comprising the input data and the parameters required to run the experiments [132, 146]. Jones et al. [210], Ferro [130], and Fuhr [149] pointed out that it is **not always assessable what characterizes a test collection**, how it compares to other collections, and why particular methods perform well or not with different test collections. A recent analysis showed that test collections may only be suitable for some system types. Evaluating neural retrieval approaches based on document pools drawn from results of mostly keyword-based retrieval methods may result in **evaluation bias** [435]. However, some methods perform equally well irrespective of the pool depth, as shown by Zhang et al. [442] who successfully reproduced BERT-based passage score aggregation approach fine-tuned with both "shallow" and "deep" judgments that resulted in similar performance.

Voorhees et al. emphasized that a test collection is not reusable if the topics have too many relevant documents [412]. The collection cannot be reused as many unjudged relevant documents remain once the judgment budget is depleted. Likewise, precision-focused measures cannot be used to distinguish between retrieval systems when there is a disproportion between relevant and non-relevant documents. In their study, they validated the reusability of the test collections by leave-out-unique tests. The technique involves the evaluation of the performance before and after excluding a particular system and its contributions to the relevance pool. Evaluating a system by excluding its contributions from the pooled relevance judgments simulates a new system that did not participate in the original pooling procedures but is evaluated later on with the help of the test collection. The test collection

is considered reusable if there are no significant differences between the evaluation scores before and after excluding a particular system's contributions.

By the same technique, Tan et al. [386] analyzed the reusability of living lab test collections with negative results. They found that for 14 out of 41 systems, there are significant differences before and after excluding a system's contribution to the document pool. Furthermore, they showed that there are also time dependencies. Especially in the period from 8 pm to 4 am, there were more significant differences. They concluded that for this particular living lab, the test collection (consisting of tweets and the corresponding user feedback) is not reusable.

Faggioli and Ferro [123] reproduced the evaluation approach by Voorhees et al. [415] based on random partitions of the test collection and bootstrap ANOVA. In this way, the influence of the topic-system interactions on the evaluation of relative system comparisons could be reduced, allowing a more precise analysis of the system effects. In their reproducibility study, Faggioli and Ferro included the bootstrap ANOVA and the traditional ANOVA, showing that bootstrap ANOVA is more robust. Carterette [78] showed that **test collections can be "overused"**. Once a test collection gains popularity and is more frequently used, there is a higher probability that extreme performance values could be observed by chance alone and not due to the system's effectiveness. Consequently, the system's intrinsic effectiveness might be lower than the observed performance with a particular test collection.

Berrendorf et al. [47], and likewise, Rao et al. [339] showed that **different training and test data splits affect reproducibility** [339]. In this regard, Kapoor and Narayanan [216] reviewed the influence of **data leakage** on the reproducibility in the broader context of ML-based research. They outlined a taxonomy covering eight types of leakage that led to reproducibility issues across 17 different scientific fields. For instance, they pointed out the problems when there is no separation between training and test data, the illegitimate use of training features, or test sets that do not represent the actual distribution giving answers to the research question.

MacAvaney et al. [264] validated the reproducibility of experiments made with the AOL query logs, and they showed that it is critical to consider the snapshots of documents if the data collection is made from scraped web page documents as web content may change frequently. They illustrated that for certain web pages, the scraped content substantially differs when scraped years after the corresponding query was originally logged. As expected, **scraping snapshots of websites at different dates results in disjoint datasets**, leading to different experimental results. This circumstance can impact session-focused experiments, as the later scraped documents of a logged query may lack a topical fit. MacAvaney et al. proposed to scrape the documents' snapshots when the queries were logged with the help of the Internet Archive's WaybackMachine. When data collections and the corresponding indices are updated with new documents, it may impact the reproducibility of the ranking results [56]. Due to the resulting updates of the term statistics, probabilistic ranking methods could lead to different rankings. This is problematic for dynamically changing document collections, for instance, in live systems that receive frequent **index updates**.

On a more practical level, Ivie and Thain [199] criticized the **separation of code and data**. Likewise, the use of **private or sensitive** [102, 148] as well as **pay-walled** [133] data collections can hinder reproducers from reimplementing the experiments. With an anecdote, we emphasize that we could not conduct a

reproducibility experiment. A $3,000 paywall of the Gigaword corpus [458] kept us back from reproducing submissions by Benham et al. [43] when participating in CENTRE [133].

2.6.3 Proactive Solutions

In the following, we review possible solutions of preparing an experiment for reproducibility in a proactive way and align them to the six PRIMAD components.

Platform

Recently, several proactive reproducibility solutions were introduced for the platform. For instance, it is good practice to use **virtual machines** [329, 346] or **containerization** software [52, 145] to bundle the experiments with the platform.

Opposed to the shared tasks organized by TREC, EaaS infrastructures make it possible to submit the entire retrieval systems instead of submitting only the ranking results [190]. TIRA [329] is a commonly used **EaaS infrastructure** in shared tasks. It allows the submission of the entire retrieval system with the help of a virtual machine. Having the entire system in a virtual machine makes them reproducible and allows the task organizers to evaluate the systems in web-isolated environments. As an additional benefit or side-effect, it is possible to organize tasks with (sensitive) data that cannot be shared publicly and to prevent any leakage of the test data into the training procedures. Similarly, participants of the TREC Total Recall Track [346] were provided with virtual machines not only to submit their experiments but also to be provided with baseline methods.

A more lightweight method to archive the platform is made possible by Docker. Recently, a Docker-based toolkit was introduced at the OSIRRC workshop [93] in 2019. By defining interfaces and standardized commands for data ingestion, indexing, and ranking, the toolkit allows the integration of ad-hoc retrieval pipelines and makes them reproducible by containerization.

While it is beyond this work to draw any conclusions about the long-term preservability, we note that there are differences between (Docker-based) containers and virtual machines regarding the comprehensiveness of the underlying platform components. Archiving an experiment by a Dockerfile, which can be a single text-based file, does not guarantee that the required platform layers will still be available on the web in the future [311]. This can be mitigated by providing the compiled Docker image, including all the platform layers in public image libraries like the DockerHub. However, virtual machines are still more comprehensive as they bundle the entire operating system.

We note that there exist several other technical solutions to make the computational platforms reproducible. However, to our knowledge, these technologies have not been used for any IR experiment until now. Srivastava et al. introduced an EaaS platform based on Docker containers and a semantic workflow system [379], whereas the Singularity-Hub [377] validates the similarity between Docker containers. Both approaches have been used for biocomputational experiments. ROHub [318] is a digital library system for research objects that supports their **storage, lifecycle management, and preservation**.

Research Goal

The IR community has a long tradition of collaboratively working on the same research problem made possible by **conducting shared tasks**, for instance, as part of TREC [413]. In this setting, the researcher is implicitly forced to conduct the experiments according to several guidelines of good scientific practice. The researchers cannot choose the baseline or select data arbitrarily. Instead, it is a benchmark across the same dataset, and the submitted results are put into context and compared with other submissions. Furthermore, evaluating the system performance before formulating a hypothesis or research question is impossible, forcing the researcher to reason about the chosen approach. Liberman [112, 246] reffered to this setting as the ***common task framework***.

One of the first calls for more rigor in the computational sciences was made by Stodden, who introduced the *Reproducible Research Standard* [381] to encourage the release of the entire research compendium, including the research paper, the data, the experiment, the results of the experiment, and any auxiliary material. Several other **guidelines and manifestos** have been published since then in order to make researchers and practitioners aware of how to make the experimentation reproducible [100, 151, 304, 357].

Of course, not only are the researchers responsible for making research goals reproducible, but it is also how publishing authorities promote it. According to Stodden [382], journal policies can enforce reproducibility at submission time. **Preregistration** or **results-blind reviewing** [309] can enforce more scientific rigor during the review. In clinical trials, studies must be preregistered before any experiments are conducted. Even though preregistration has yet to be established as part of any IR conference, it offers a perspective towards emphasizing a study's research questions and scientific design, reducing the sole focus on performance gains.

Recently, the SIGIR community inaugurated the **ACM badging system**, allowing a paper to be evaluated for reproducibility after acceptance. Kelly and Ferro [134] surveyed the community about badging and concluded with mostly positive opinions about the procedures. The database community also applies reproducibility badging as part of SIGMOD [451] or PVLDB [479]. The ReScience initiative [348] is an open-access journal [481] dedicated to reproducibility experiments, which also explicitly invites to submit failed attempts. The web service *Papers with Code* [478] tracks openly available information from the arXiv, ACL anthology, and OpenReview and does not only link source code repositories to publications, but it also includes pointers to available reproducibility studies. Furthermore, the service hosts the *Reproducibility Challenge* [473] that is organized via the OpenReview platform and publishes selected studies in the ReScience journal.

Besides the publications in which the research goal is conventionally reported, the IR community developed several solutions in the form of services and platforms facilitating **resource management** to put the research goal into context with other findings and experimental outcomes. The DIRECT infrastructure [2, 3, 310] hosts experimental artifacts and enriches them with metadata according to a conceptual model. Armstrong et al. [12] introduced a solution for standardized evaluations by a central web-based service as a countermeasure to the lacking upwards trend of the overall retrieval performance as revealed by their meta-evaluations. The service allowed the upload of run files, and after evaluating them, they are put into context with other submissions for the same task or test collection. Yang and Fang [426] pro-

posed another more recent attempt to benchmark retrieval systems in a reproducible manner. They proposed a Docker-based service to evaluate a retrieval system over multiple test collections systematically. The authors implemented several standard retrieval methods into the system, which is extensible by encapsulating new retrieval systems in Docker containers.

Implementation

Proactively preparing the implementation for reproducibility can be achieved by integrating **good software development practices** into the implementation process [163]. This involves but is not limited to managing the experiment by configuration files [10, 423], logging [87], (release) versioning with version control software [338], data management (tracking the data provenance similar to control version software) [69,85], dependency management [237], open-source releases [50,126], test-driven developments that allow more stable software releases [121], or good documentation and communication including post-study support [337].

Configuration files provide a systematic way to manage (hyper-)parameters and configurations of the implemented experimental setup. It not only facilitates easier modifications of the experiments but also provides better access for external researchers with a high-level perspective on what could influence the experimental outcomes. Some reproducibility studies showed that a **proper specification of hyperparameters** is critical for reproducibility [47, 101, 300]. Hydra [423] is a software toolkit for setting up more general data science experiments by configuration files. A more task-specific example is the ELLIOT [10] framework allowing the specification of recommender systems experiments by configuration files. Bakshy et al. [27] developed a scripting language to systematically describe user experiments and have them checked automatically for validity [392].

There also exist toolkits like ReproZip [87] or *Whole Tale* [69] that automatically **log system calls and track the data provenance** throughout the execution of an experimental pipeline. With special regard to the source code, **version control** systems like Git [338] facilitate better transparency. Furthermore, reproducibility can be supported by making the source code open. As mentioned earlier, Voorhees et al. [414] introduced the idea of **Open Runs**, according to which run submissions to TREC should be backed by a public code repository. Fortunately, there is an increasing trend of publications accompanied by an **open-source code repository** [126], and a study by Bhattarai et al. [50] indicated that providing open-source code can have a positive effect on the citation rate. As pointed out by Kusa et al. [237], it is important to **explicitly refer to external and required dependencies**.

When implementing standard processing steps of retrieval pipelines, it is reasonable to use **established retrieval toolkits** that are commonly used by the community like Anserini [250,427,428], Terrier [266,267,314], Indri [383], or PISA [275]. In this manner, the corresponding implementations are less error-prone due to the community-based development process, and it is less time-consuming when integrating standard operations into the experiments. Furthermore, it allows a **better and fairer comparison** to other studies building upon the same toolkits.

Anserini is a Java-based retrieval toolkit built on the Lucene software library often used in industrial environments. In this regard, Anserini satisfies the IR community's request to prepare Lucene for academic experimentation [20]. The entire development process of Anserini keeps reproducibility as a primary requirement. The

corresponding GitHub project [461] contains several notebooks that support many standard experiments for established test collections that allow "off-the-shelve" use and continuous validation by regression tests. Pyserini [250] offers a Python-based interface to Anserini and facilitates in combination with the PyGaggle toolkit [331] the implementation of modern multi-stage retrieval pipelines covering sparse and consecutive dense retrieval operations [248]. Both Anserini [157, 332, 417, 429, 437] as well as Pyserini [243, 263] have been used for several reproducibility studies.

Similarly, there is the Java-based Terrier retrieval toolkit [266, 314] for which Python bindings are offered by Pyterrier [267]. Pyterrier follows a declarative programming style that adds an abstraction layer above the source code, making the experimental workflow more readable and easier to modify systematically. Like Pyterrier, it also allows the implementation of multi-stage ranking pipelines with deep language model-based rerankings by integrating specific packages and plugins.

Method

In order to avoid ambiguities between the (more abstract) method and the (more concrete) implementation, there exist some solutions to integrate the method more directly into the code execution environment as it is made possible by **"executable papers"**, which allow a tighter connection between the scholarly communication and the code implementations [58, 73]. For instance, PopperCI [205] is a **continuous integration service** that facilitates writing an article and hosting the corresponding experiments with a DevOps approach, making it possible to validate the reproducibility automatically. This results in a more integrated way of reporting the methodology and its implementation. CodaLab [455] is a similar platform for executable papers. Recently, there has been an increasing trend of documenting and implementing research with the help of **Jupyter notebooks** [228, 323], which allow the combination of executable code and detailed documentation by annotating the code snippets. Following the same idea, Bar and Wang propose a software package that allows running the experimental code directly from a LaTeX environment [32].

noWorkflow [305] automatically tracks the data provenance of software scripts, while YesWorkflow [291] introduces an **annotation language for scripting languages** in order to facilitate the experimental documentation with little overhead. Miksa and Rauber [292] proposed an **ontology to annotate workflow-based experiments** and provide solutions for system resource logging [341]. Snakemake [299] is a **configuration file-based workflow management** framework that allows defining reproducible pipelines with data in- and outputs combined with Python.

In publications, the technical details often do not contribute to a better understanding and may overstress the readers' cognitive resources. However, these details are essential for accurate reproductions. Recently, **model cards** were introduced by Mitchell et al. [294]. The general idea is to provide additional documentation for ML models in the form of metadata according to the proposed annotation framework. Piwowarski [325] introduced a software tool that can be used to **manage data and experimentation pipelines** in IR experiments by **source code annotations**.

Methodologically, **sources of randomness need to be identified** [199]. While some methods include intentional non-deterministic behavior, other sources of randomness are unintentionally part of the experiments, like issues related to concurrency or floating point operations [119]. If feasible, random seeds should be explicitly

reported. Otherwise, there exist ways to identify other sources of randomness [84]. In document rankings, **score ties should be broken deterministically** [253].

Regarding the evaluation of a method, **statistical significance testing** should be part of the analysis when comparing the method to a baseline. Recently, guidelines on significance testing with DL-based approaches were published by Ulmer et al. [401] or Dror et al. [114]. They reminded their readers that it is critical to test for statistical significance, especially if improvement may occur by chance alone without any reasoning behind the modifications of a neural network's architecture. When conducting multiple significance tests, Fuhr [149] emphasized that it is critical to **apply corrections in order to avoid the multiple comparisons problem**.

In a series of works, Ferro, Kim, and Sanderson [135, 139, 140, 141] proposed an approach for **improving the performance measurement accuracy**. Similar to Voorhees et al. [415], they split the document collection into shards or replicates — random partitions of documents. By combining a general linear mixed model with ANOVA testing, they showed that shards have a significant impact on the system effectiveness [139] and this circumstance is present across different datasets, highlighting the interactions between topics and shards [135, 140].

Actor

While in most cases, **actor-independence** should be the ultimate goal, like it can be made possible by EaaS platforms, it is not always possible to remove the original actor's influences entirely. As a way out, authors should feel responsible for **promoting their research after publication** and help others reproduce it [337]. Besides providing the corresponding author's **contact information** in the publication, it has now become good practice to include an **ORCID** [477] to avoid any ambiguities. Likewise, Git (or any other version control software) can help **trace the author's contributions** to an experiment.

Likewise, several initiatives have started attempts to increase the **awareness of the reproducibility** of younger researchers. Lucic et al. [261] successfully integrated **reproduction studies as part of a Master's program**. Based on specified learning outcomes, they let students reproduce state-of-the-art approaches from major Artificial Intelligence (AI) conferences and made the reproduction reports part of the earlier mentioned ReScience journal (cf. Subsection 2.6.3). TU Delft hosts a dedicated online database for reproducibility studies [480] made by students [434]. Similarly, Potthast et al. [328] conducted a systematic reproduction study with students. In other domains, reproducibility has also become part of teaching, for instance, in computer networks [424], social sciences [184], or psychology [344].

In the same way, researchers should be **sensitized to the different forms of cognitive bias**, which can occur for both the authors as well as reviewers [152, 153, 235, 240]. In addition, the **reviewing process could be made more diverse by making it public and more transparent**. The OpenReview platform [476] makes the reviews public and allows to have insights about what has been criticized and how it has been addressed even after the study is published.

Ultimately, **reproducibility should be understood as a community project** (as it can require continuous auditing [254]) and should reduce the burden of individual researchers [144]. The Anserini project exemplifies how this can be put into practice. The corresponding developers maintain rerunnable notebooks for regression tests that different experimenters can continuously validate.

Data

As the separation of code and data can cause one of the first reproducibility issues [199], it is critical to make the **data provenance** as well as the processing steps as transparent as possible. If feasible, **open data** should be used, and all of the data resulting from the experiments should be made **available for later requests** by others, or follow-up studies [96, 196].

Especially for deep neural network-based approaches, the **model's checkpoints** (i.e., learned weights and parameters) should be made available in the aftermath of an experiment. Not only to allow others to rerun the experiments but also to lower computational costs by avoiding retraining the model. Ma et al. [263] could show that it is feasible to reuse checkpoints and reproduce earlier outcomes. Similarly, Wang et al. [418] analyzed the reproducibility and replicability of TCT-ColBERT from a three-stage perspective, including the separate analyses of the training, inference, and evaluation. They concluded that it is more challenging to exactly reproduce the complete process, including the training, than building upon provided experimental artifacts like pre-trained models. They had no success when replicating the training with an independent reimplementation of the analyzed method. This study highlights once more the importance of artifact sharing.

Bösch [56] addressed the problem of an updated index and the resulting changes in the term statistics that could lead to irreproducible rankings, especially for probabilistic ranking methods. Based on the method by Rauber et al. [340] for making dynamic and changing data collections citable, Bösch proposed to assign **persistent identifiers and timestamps to queries** and enhance them with hashed result sets. When re-executing the query against the versioned database, the hashed result sets can be checked for integrity. As the method by Rauber et al. [340] requires a column-store database, Bösch reused the approach by Mühleisen et al. [301], translating the BM25 retrieval method to an SQL query.

On the other hand, it has to be considered that rerunning the original source code on the same dataset does not give any insights about how a method generalizes with other data [115] and overused test collections [78] could lead to improvements due to chance and not due to the method alone. As a compromise, it is good practice to include **more than one dataset** in the experimental evaluations.

Documenting the data according to common standards is made possible by **datasheets** [57, 154]. Gebru et al. [154] addressed the lack of a documentation standard for datasets by **compiling a catalog of guideline questions** that should be addressed by the curators when preparing the dataset for reproducibility. Besides reproducibility, datasheets can also address issues related to ethical concerns, as pointed out by Boyd [57].

There are different software tools and solutions that facilitate **better data management**. A non-extensive list includes the BEIR benchmarking toolkit [389] allowing evaluations over different IR tasks and datasets, the `datamaestro` toolkit by Piwowarski [325], or the data catalog and software package `ir_datasets` [265]. Lin et al. [251] proposed a **common index format** in order to make (Lucene-based) indices compatible with a variety of different retrieval toolkits. This allows for better and fairer comparisons of retrieval methods by benchmarking them based on a **shared and common preprocessing pipeline**.

2.7 Answers to the Research Questions

This section gives answers to the research question posed earlier. The existing body of literature was reviewed for both questions, and the references were aligned to the six components of the PRIMAD taxonomy. The following outlines how these issues and solutions relate to the more general causes for irreproducibility mentioned earlier in Section 2.5.

RQ1: What kinds of general reproducibility problems are there in computer science and particularly in IR research? As a starting point to find more general causes for irreproducibility, we grouped the answers given to Baker's questionnaire [474] into five different categories, including unethical actions (cf. Subsection 2.5.1), issues related to the scholarly communication (cf. Subsection 2.5.2), statistical and experimental flaws (cf. Subsection 2.5.3), the unavailability of the experimental setup (cf. Subsection 2.5.4), and the missing expertise (cf. Subsection 2.5.5). In order to provide an IR-specific review of the literature, we align known issues and outcomes of reactive reproducibility studies to the PRIMAD taxonomy, which covers the six components platform, research goal, implementation, method, actor, and data that can affect the reproducibility of a computational experiment.

Generally, the reviewed body of IR-specific literature in this chapter has a focus on problems and solutions related to statistical and experimental flaws, partly on issues related to scholarly communication, and on the unavailability of the experimental setup. The remaining two more general causes for irreproducibility (unethical actions and missing expertise) are mainly related to the actor component, which introduces the human factor of the experimenter to the PRIMAD taxonomy.

From the available body of IR-specific literature, it is not possible to estimate what kind of influence unethical actions like fraud [105], e.g., in the form of data falsification, have on the overall reproducibility of the research field. However, we note that the SIGIR community strictly discourages unethical and adheres to the *ACM Policy on Plagiarism, Misrepresentation, and Falsification* [449] as it is also underlined by Carterette's blog post about plagiarism [483].

Likewise, it is not possible to have a more concrete idea to which extent the missing expertise of a reproducer influences reproducibility, as most of the failures due to this cause presumably remain unpublished. However, the dedicated ECIR reproducibility track also contains failed reproducibility studies with negative outcomes. Furthermore, we also note that there is a current trend of making reproducibility projects part of curricula in order to sensitize students to the topic of reproducibility [261, 344, 424, 434].

One of the most striking issues of irreproducibility is the unavailability of the experimental setup, as it may also reveal other issues related to scholarly communication when the reproducer is forced to reimplement the experiment on the basis of what is described in a publication. The experimental setup usually covers the platform, implementation, and data components in the PRIMAD taxonomy. Through the literature review, we could identify the following platform-related issues that can harm reproducibility:

- **unavailability of the original hardware, kernel, and operating system**: for instance, an updated kernel or operating system in the reproduction attempt can lead to failures, e.g., [254],

- **unavailability of external platforms and corresponding resources**: failures due to dependencies on external platforms/resources that are not available at the time of the reproduction attempt, e.g., [173, 254],
- **high computational requirements and costs**: the reproducers cannot meet the hardware requirements, e.g., [337],
- **non-deterministic/random behavior of hardware components**, e.g., [137, 306].

The implementation describes how the method is translated into the source code, which leads to the machine instructions for the computing device. The literature review could identify the following issues:

- **no public open-source code repository**, e.g., [99],
- **missing code documentation**: even if the source code is available, it is not documented enough to rerun the experiments or not adequately prepared for rerunning the experiments, e.g., [99],
- **ambiguities between implementations of the same methods**: different implementations of the same method can lead to differences between the effectiveness scores, e.g., [213, 234],
- **overestimating the availability of source code**: in the best case, the source code allows rerunning experiments in the paper but does not provide any insights about the reproducibility in other contexts (cf. replicability/generalizability), e.g., [101, 115],
- **missing / insufficient dependency management**, e.g., [237].

The third PRIMAD component of the experimental setup is data, which often covers the IR test collection but also training data, model parameters, and others. We identified the following issues:

- **separation between code and data**, e.g, [199],
- **private / closed datasets**, e.g., [102, 147],
- **pay-walled datasets**, e.g., [133],
- **overused test collection**, e.g., [78],
- **data leakage**, e.g., [216],
- **training/test data splits**, e.g., [47, 339],
- **biased relevance judgments**: relevance labels can be biased towards a specific type of retrieval system that was used as part of the pooling; likewise, the distribution/proportion of positive and negative relevance labels has an impact, e.g., [386, 412, 435],
- **other data-related biases**, e.g., [53, 233, 308],

- **updated index statistics** of dynamically changing data collections, e.g., [56].

The research goal, as well as the method, are conventionally disseminated in publications. For both PRIMAD components, issues can occur related to the more general issues of statistical and experimental flaws and shortcomings in scholarly communication. For the research goal, we compiled the following list of issues:

- **no research questions/hypothesis**: the low entrance barrier of modern ML frameworks allows us to start experimenting without clearly formulated research questions or any hypothesis about the outcomes leading to post hoc formulated "assumptions" about the results, e.g., [149, 255],
- **illusionary improvements over weak baseline**: improvements over weak baselines that diminish when compared to stronger state-of-the-art baselines as shown by several systematic reproducibility benchmarks, e.g., [13,108,109,430],
- **leaderboard chasing**, e.g., [368],
- **overall evaluation criterion does not give answers to the research question**, e.g., [231],
- **missing statistical significance tests**, e.g., [114],
- **underpowered statistics and effect sizes**, e.g., [114, 149, 229, 401],
- **outcomes with low statistical power**, e.g., [229],
- **no correction method applied**: multiple comparisons problem, e.g., [150],
- **evaluation measures should be intervalscaled** when conducting significance test, e.g, [129],
- **change of context**: some research goals do not hold in a different context, e.g., in more real-world environments simulated by noise or due to temporal changes of the context, e.g., [47, 148, 369].

Besides the research goal and the corresponding evaluation, the evaluated method itself can also be affected by several issues that cover methodological aspects but also how the method is presented and communicated, including:

- **poor readability and insufficient documentation**, e.g., [337],
- **unnecessary complexity**, e.g., ("mathiness") [255],
- **deviations between methodological descriptions in the paper and the corresponding implementations**, e.g., [46],
- **technical and methodological underspecifications**: for instance, missing details about hyperparameter tuning or underspecifications of standard retrieval components, e.g., [47, 143, 350, 370, 437],
- **deviations from the constraints of the original settings**: changing some (sub-)components of the method leads to non-reproducible (or non-generalizable) outcomes [51, 243, 312, 364],

- **score ties** in the final ranking, e.g., [253].

Finally, the actor components can introduce additional obstacles to a successful reproduction:

- **compromises between concrete machine instructions and more abstract means of the human language**, e.g., [199],
- **no support after publication**, e.g., [99, 337],
- **cognitive biases**: "confirming" authors [152, 235] vs. "prejudiced" reviewers, e.g., [153, 240].

The reproducibility taxonomy by Potthast et al. (cf. Section 2.4) categorizes actions towards reproducibility into reactive, proactive, and supportive ones. Most of the listed issues were revealed in reactive attempts to reproduce earlier works by others. The second research question will be answered by reviewing how proactive and supportive actions address most of the issues resulting from the answer to RQ1.

RQ2: To what extent have reproducibility problems been addressed in IR research, and how are the countermeasures implemented? What kinds of open points are there? As a follow-up to RQ1, we review the body of IR-specific literature and, likewise, align the countermeasures, which can facilitate reproducibility, to the PRIMAD components. In general, a large variety of tools support researchers in making an experiment reproducible from the early beginning of the conceptualization. On the other hand, several countermeasures can only be realized at the methodological and organizational level and still require human intellect, and technological solutions can solve only some reproducibility issues. Yet, there exist some well-established software tools that mainly help to preserve the platform for future reuse, including:

- **virtualization or containerization**, e.g., [93, 346],
- **EaaS platforms**: TIRA, EvaluatIR, RISE, e.g., [12, 329, 426],
- **resource management**: hosting experimental artifacts and metadata like DIRECT, e.g, [2,3,310], improved lifecycle management of research objects by background logging software, e.g., [318].

The research goal of a study is conventionally reported in a publication. The peer-review process can be seen as an additional validation by the reviewers (who can also be seen as another instantiation of the actor component). In this sense, many of the actions towards reproducibility still require human reasoning and can be generally considered methodological and organizational:

- **shared tasks (common task framework)**: collaborative efforts to work on the same problem, e.g., [112, 246, 413],
- **research standards and guidelines** of good scientific and reproducible practice, e.g., [100, 151, 304, 357, 381],
- **journal policies** [382],

- **preregistration / results-blind reviewing**, e.g., [309],
- **systematic reproducibility analysis**: benchmarks that follow the same principled evaluation, e.g., [38, 99, 108, 109, 249, 254, 337, 421],
- **badging system and artifact evaluations**, e.g., [134],
- **conference tracks, journals, community challenges, and student programs** for reactive reproducibility studies, e.g., [9, 171, 348],
- **strong and tuned baselines**, e.g., [13, 300, 430],
- **methodological and scientific rigor**: for instance, formulating a hypothesis before starting the experiment, e.g., [149, 255].

As the implementation is mainly about how the method of an IR experiment is translated into software code, it benefits from good software development practices. Thus, many of the solutions that help to make the experiment more reproducible are not innovative but rather address how good practices can be enforced in computational research and include:

- integration of good **software development practices** in research projects, e.g., [163]
- **sharing open-source code**: a positive trend is observable, and sharing source code is correlated to the citation count, e.g., [50, 126]
- **configuration files**: making hyperparameters more explicit, e.g., [10, 423],
- **logging**, e.g, [87],
- **version control software**, e.g, [338],
- **data management**, e.g, [69, 85],
- **dependency management**, e.g, [237],
- **open-source releases**, e.g, [50, 126],
- **test-driven developments**, e.g, [121],
- **code documentation**, e.g, [337],
- **retrieval toolkits**: for standard retrieval operations, established toolkits should be used, e.g., [275, 314, 383, 428].

There exist some technical solutions for making a method more reproducible, but not least, the validation of a method also requires statistical evaluations, for which human interpretation and understanding are necessary (cf. [199]):

- **executable papers**, e.g., [32, 205, 228],
- **workflow description and logging**, e.g., [291, 292, 299, 305],
- **model cards**, e.g., [294],

- **identify sources of randomness and non-deterministic behavior** and report random seeds, e.g., [119, 199],
- **statistical evaluations** and corresponding corrections, e.g., [114, 149, 401],
- **improving performance measurement accuracy by replicates**, e.g., [135, 139, 140, 141, 415].

Suppose the actor is not essential for the outcome of a computational experiment. In that case, the actor's influence can be seen as noise, and this influence should be minimized, whereas actor-independence can be considered as the ultimate goal. On the other hand, it is not always feasible to remove the actor's influence, and every author and researcher should feel obliged to provide support even after work on a research subject is finished, resulting in the following solutions:

- **actor-independence** if feasible, e.g., [190],
- **support after publication**, e.g, [337],
- **explicit contributions** by ORCID or version control software, e.g., [338],
- **reproducibility project as part of the curricula** can help to sensitize students and future researchers for the topic, e.g., [184, 261, 328, 434],
- **open and transparent peer reviews**, e.g., [476],
- **sensitize authors/reviewers for cognitive biases**, e.g., [152, 153, 235, 240],
- **community involvement** in order to relieve individual researchers and understand reproducibility as something that needs to be continuously maintained, e.g., [254].

Finally, there are some proactive ways to prepare the data for future reuse. Generally, open data should be preferred. However, beyond using public data, also the data provenance and other data sources besides the actual test collection play an essential role:

- **archive data**, e.g., experimental artifacts [96, 196],
- **providing trained models**, e.g., [263, 418],
- **logging the data provenance**, e.g., [69],
- **using more than one test collection**, e.g., [149, 210],
- **no overuse of test collections**, e.g., [78],
- **datasheets**, e.g., [154],
- **data management software**, e.g., [265, 325, 389],
- **shared index**, e.g., [251],
- **data citation principles** for dynamically changing data collections, e.g., [56, 340].

Even though previous work has addressed reproducibility issues in many regards, there are still some open points left, and some of them will be addressed as part of this work. First, the analysis of the ECIR proceedings showed no agreement on how it can be quantified to which extent a reproduction is successful. In many experimental evaluations of reproduction studies, the authors consider reproduction as successful if the original is outperformed with statistical significance or if there are no statistical differences between the effectiveness scores. As a solution, Chapter 4 reviews reproducibility measures that can be used to quantify the degree to which an IR experiment has been reproduced.

Meta-evaluations reveal the reproducibility of particular retrieval methods by putting them into context with other approaches and systematically benchmarking them. Metadata can facilitate these meta-evaluations. While several more general solutions exist, like model cards or datasheets, there is no domain-specific "datasheet" or metadata schema for IR research. This gap is addressed in Chapter 3, which extends the PRIMAD taxonomy and shows how it can be used to annotate experiments to make them more transparent and reproducible. Similarly, there is no systematic way to conduct a reproducibility attempt under the consideration of how it relates to the original experiments. In this regard, Chapter 5 exemplifies how a reproducibility experiment can be classified in terms of the PRIMAD taxonomy and evaluated depending on which components have changed.

To our knowledge, there are very few user-oriented reproducibility studies, and most of the reactive reproducibility attempts focus on system-oriented aspects. We emphasize that the influence of the user is underrepresented when analyzing the reproducibility, and we propose to include user interactions as part of the evaluations, either by simulations or online experiments as outlined in Chapters 6, 7, and 8.

2.8 Conclusion

Through the literature review in this chapter, we provide an overview of the state of the art of reproducible IR research. Building upon the Cranfield paradigm, experimentation in IR has always been concerned with reproducibility. However, it had a more implicit nature for many years and recently got more attention. According to the Cranfield paradigm, the internal validity of a retrieval method is evaluated over different queries. Shared task conferences like TREC, CLEF, NTCIR, or FIRE made it possible to evaluate the external validity of a retrieval method over multiple reusable test collections with different contents, document types, or tasks. However, the IR community acknowledged the increasing reproducibility concerns in science and has developed several countermeasures in the last years.

In general, there is inconsistent use of the terminology around reproducibility and related concepts. In this chapter, we have included the ACM, NeurIPS, and Claerbout terminologies that partly overlap in their definitions. Throughout the rest of this work, we follow the ACM terminology when writing about reproducibility in more general terms, as it is the most commonly adopted terminology by the IR community. However, we note that PRIMAD answers the inconsistent and imprecise use of the terminology. Expressing the reproducibility in terms of PRIMAD gives a more precise description of what kinds of components with regard to the original experiments have been changed.

Furthermore, we addressed two research questions. In this context, we have adopted Potthast et al.'s taxonomy that distinguishes between reactive, proactive, and supportive actions towards reproducibility. By giving answers to RQ1, we reviewed reactive IR reproducibility studies and identified what kinds of issues and obstacles towards reproducibility can occur. Reactive reproducibility studies have been inaugurated as part of dedicated paper tracks by ECIR and, more recently, also by SIGIR. As our review of the ECIR reproducibility track shows, authors consider their reproductions to be (un)successful based on different criteria but also different levels of rigor. The comparisons to the original methods are often done by comparing average scores and determining p-values of paired t-tests. In this context, the significance testing is conducted to verify that the reimplemented systems outperform the original baseline method and thus confirm reproducibility.

All reproducibility issues can be aligned to PRIMAD, which confirms that the taxonomy is comprehensive enough to consider all essential components for the reproducibility of a system-oriented IR experiment. However, we note that even though there is a user-oriented interpretation of PRIMAD in the original reports, it does not contain an explicit user component, and instead, the user is subsumed by its data trace as part of the data component. Similarly, there are few user-oriented reproducibility studies. In general, user-related aspects are underrepresented in the evaluations. Even though some studies include additional evaluations like user judgments [49,271] or simulating noisy user click behavior [312,417], there is, to the best of our knowledge, only one very recent study by Roy et al. [351] that made the original user study the central objective of the reproducibility attempts.

The second research question adds up to the first one by giving answers to which extent and how the reproducibility issues have been addressed so far. Generally, different software tools exist that help the researcher conceptualize and implement reproducible computational experiments. Some tools partially relieve the experimenters from the documentation by automatically logging information about the experiment. However, several aspects can only be addressed at the methodological and organizational level and still require human reasoning and retrospection. In the IR community, there is an increasing interest in reproducibility, as can be seen from the steadily increasing amount of reproducibility studies in the ECIR proceedings. Likewise, the trend of integrating reproducibility projects into the curricula leads to more awareness of the topic of next-generation researchers.

Besides what has already been addressed, the answers to RQ2 also highlighted how this dissertation project addresses some of the open points. First, we will outline how reproducibility can be systematically analyzed and quantified in Chapters 3, 4, and 5. Second, we give ideas and directions towards reproducible experimentation, which also involves user-oriented aspects in Chapters 6, 7, and 8.

Chapter 3

PRIMAD-U

This chapter introduces PRIMAD-U as an extended version of the PRIMAD taxonomy [132, 146]. From the results of the literature review, we conclude that reproducibility research has mainly dealt with system-oriented IR experiments, and a shift towards more user-oriented evaluations is needed. With a special focus on the PRIMAD taxonomy, we conclude that the taxonomy is well-grounded and considers the key components that might affect the reproducibility of experimental outcomes. However, it is still a rather theoretical concept leaving the particular components underspecified without any practical application yet.

Given the outcomes of our literature review, we extend the taxonomy based on the answers to the research questions RQ1 and RQ2 and motivate these by the issues related to reproducibility and the corresponding solutions from the previous chapter. Besides extending the six conventional PRIMAD components with subcomponents and related concepts, we introduce an additional user component in this chapter. According to the original report of PRIMAD, there are separate definitions for the system- and user-oriented experiments, underlining its applicability to user-focused experiments. However, we favor a more holistic view of the IR experiment, which is illustrated in Figure 3.1.

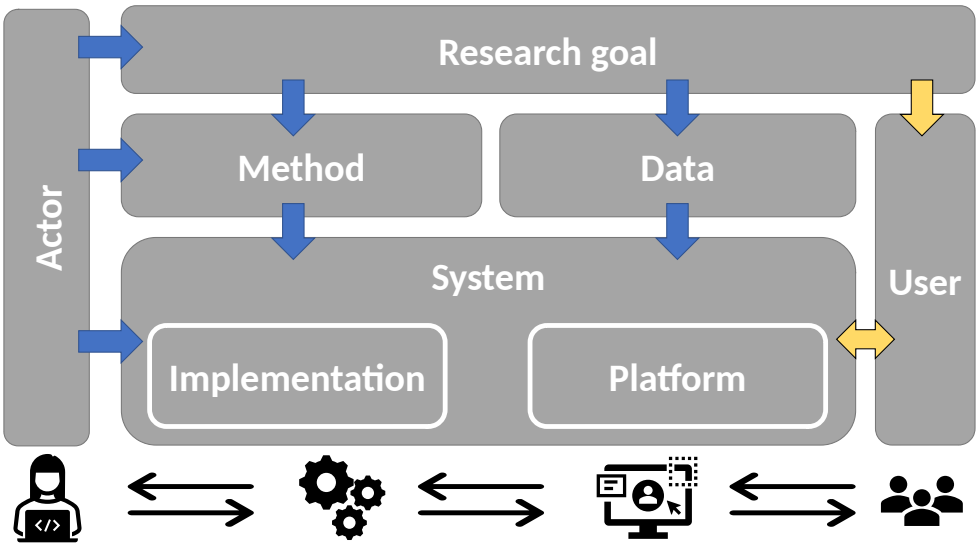

Figure 3.1: Conceptual view on the IR experiment based on PRIMAD-U.

In this figure, we see all the PRIMAD-U components put into context. At opposing ends are the **actor**, the experimenter who controls and designs the experiment, and the **user**, the recipient of the system outputs or rankings. The **actor** designs the experimental setup and hereby defines the **research goal**, which is often expressed by a research question that addresses the retrieval effectiveness in terms of a measure or the user behavior (*dependent variable*) in response to the system outputs based on the retrieval method (*independent variable*). The **actor** has complete control over the **method** and the retrieval system that comprises the **implementation** and the underlying **platform**. The **data** is a task-specific document collection, which is ingested by the retrieval system. Some research goals or questions can be data-specific, and generally, it can be seen as a *control variable* that remains static during experimentation.

In system-oriented IR experiments, the retrieval method (or system) is the evaluation focus and is considered the only source of variation. Its effectiveness is determined by several retrieval measures specific to the task or **research goal**. These evaluations and, likewise, the measures imply an abstract and static representation of the **user**. There are some measures that imply certain aspects of user behavior, like the effort of browsing through a ranking list. However, these evaluations remain limited to the interaction with a single result list for a pre-defined "static" query.

Contrary to the system-oriented approach, user-oriented experiments are the focus of Interactive Information Retrieval (IIR) experiments. In small-scale studies, the **actor** has more control and knowledge about the **user**, but these experiments are costly and generally considered irreproducible. Besides the different forms of how interactive retrieval experiments can be realized, for instance, as large-scale A/B experiments, user simulations are a viable solution to account for a more user-oriented evaluation in a cost-efficient way.

To this end, the **user** component and related concepts are motivated by earlier studies from these fields, including the *implicit user model of retrieval measures*, findings from *small-scale IIR experiments* but also other forms of user-oriented experiments (like large-scale A/B experiments), and *user simulations*. In sum, the contributions of this chapter include:

C2 Extension of the PRIMAD taxonomy: We extend the PRIMAD components based on the results of the literature review of the previous chapter. Instead of using two separate definitions for the system- and user-oriented experiment, we add an additional user component to the taxonomy in order to provide a more holistic view of the general IR experiment.

C3 Metadata annotation schema: Based on the extended components of the conventional six-dimensional PRIMAD taxonomy, we derive a metadata annotation schema for TREC run files, which was contributed to SIGIR [63].

The remainder is structured as follows. First, we introduce the PRIMAD-U taxonomy by describing the taxonomy trees for each of the components in Section 3.1. Afterward, we motivate and describe the metadata schema in Section 3.2.

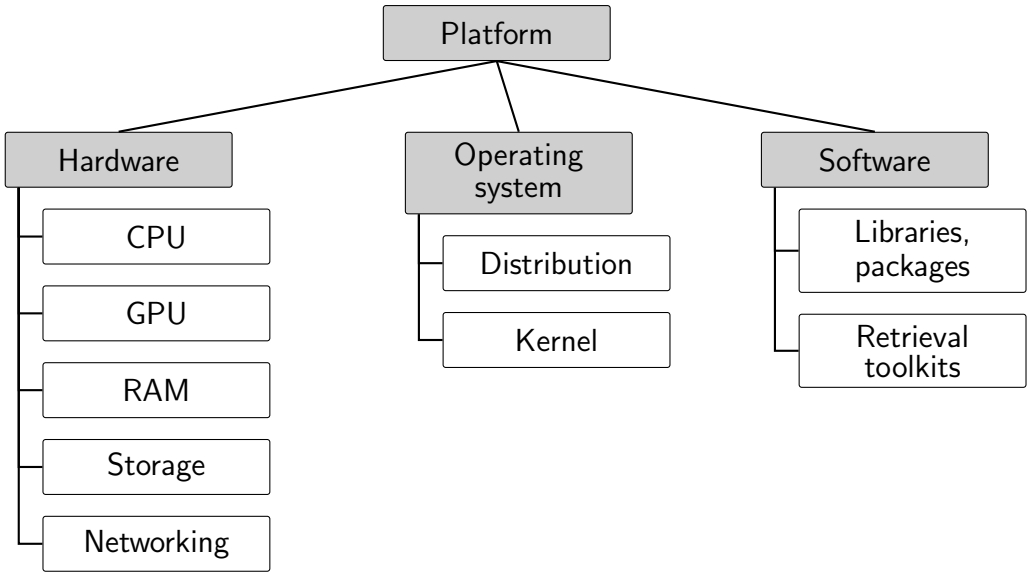

Figure 3.2: **Extended platform taxonomy:** the gray concepts were also (implicitly) mentioned by Ferro et al. [132], while the others were added by us.

3.1 Taxonomy

This section presents the PRIMAD-U taxonomy and the extended concepts for each component. At first, the taxonomies of the original PRIMAD components are introduced. Most of the extended concepts are inspired by the common pitfalls and solutions for reproducibility of the previous chapter. In addition, we extend the taxonomy by a user component. In this case, we include an additional literature review that spans findings from interactive studies and user simulation experiments to complement the user with related concepts.

3.1.1 Platform

Figure 3.2 illustrates the three subcomponents of the platform, including the *hardware*, the *operating system*, and the underlying *software* of the computational experiment. As it was outlined in Subsections 2.6.2 and 2.6.3, it can be critical if the underlying hardware of the original experiment is not available when rerunning the experiment. It is one of the most obvious solutions to make the physical computation device available for reuse purposes and to run experiments on the same machine. However, this is often impractical due to logistical inefficiencies [199].

Since the hardware is the least practical to share for future reuse and to have an idea about the requirements for rerunning the experiment, the hardware components should be documented as detailed as possible, for instance, by documenting the *central processing unit (CPU)* by its name, the architecture, the number of cores, the operation mode, and others. Of course, these can be limited to a single machine but could also cover multiple physical machines in the case of distributed computing. However, essential hardware components in IR experiments are the CPU, the *graphics processing unit (GPU)* (for modern DL methods), the (size of the) *random-access memory (RAM)*, and the *storage* size (i.e., required disk space), which mainly depends on the data components (cf. Subsection 3.1.6). With special regard to external platforms, the *networking* type and bandwidth can be critical. Especially in

user experiments, latencies caused by the delay of the system response can affect the outcomes of an experiment.

An updated operating system can affect the reproducibility [254], and to this end, the *distribution* and the *kernel* (version) must be considered. Research questions often do not deal with details of the hardware or the underlying infrastructure, and one of the primary goals should be the independency of the platform, which is the purpose of EaaS infrastructures that were introduced in Subsection 2.6.3. One of the prime examples of an EaaS infrastructure is TIRA [329], facilitating the submission of retrieval experiments in virtual machines. Thus, the EaaS solution makes it possible to provide the operating system, including its subcomponents — the distribution and the kernel — in a reproducible way.

With special regard to the software, the implementation's underlying dependencies should be considered, including software *libraries* and *packages*, which should be made explicit by their version numbers. We distinguish between the software subcomponent and the actual implementation (cf. Subsection 3.1.3) since the platform describes all layers below the implementation, including the software libraries upon which it builds. If domain-specific software libraries are used, they should be explicitly identified as *retrieval toolkits*. On the one hand, the software subcomponent could also be packaged within a virtual machine. On the other hand, a more lightweight alternative is made possible by containerization, as exemplified in the OSIRRC workshop [93].

3.1.2 Research Goal

The overall research goal (cf. Figure 3.3) of an IR experiment is usually communicated by the *publication*. Depending on the study's objective, the description of the research goal can be quite complex. Expressing it by a taxonomy may not be comprehensive enough as it may fail to put individual aspects into context. For this reason, the research goal should include a pointer to the publication. Thus, the taxonomy comprises several subcomponents related to metadata information about the publication, such as the *name* and the *year* of the study's publication *venue*, the *digital object identifier (DOI)*, the *arXiv*-ID, or any other unambiguous identifier that should be reported. If feasible, the *abstract*, *full text*, and *references* could be added as well, e.g., by the corresponding LaTeX code.

Nonetheless, some conventional information about the *experimental design* can be reported, such as the *task*, which is not limited to ad-hoc retrieval experiments but also includes other objectives such as question answering, multimedia, or cross-language retrieval, to name a few examples. An overview of what kinds of tasks have been conducted at CLEF is provided by Ferro and Peters [138]. Likewise, the study is sometimes accompanied by one or more explicit *research questions* to which the experiments provide answers. In this regard, the underlying null *hypothesis*, as well as the *(in-)dependent* and possible *confounding variables*, have to be made explicit.

Finally, the *evaluation* of the experiments provides the basis for answering the research questions. In IR experiments, the independent variables are often the chosen *measures*. Usually, the research questions define what kinds of measures are reasonable. However, over the years, the IR community has developed several evaluation measures [355], and there is a plethora of measures available, which sometimes leads to unreasonable use for the particular research question [445].

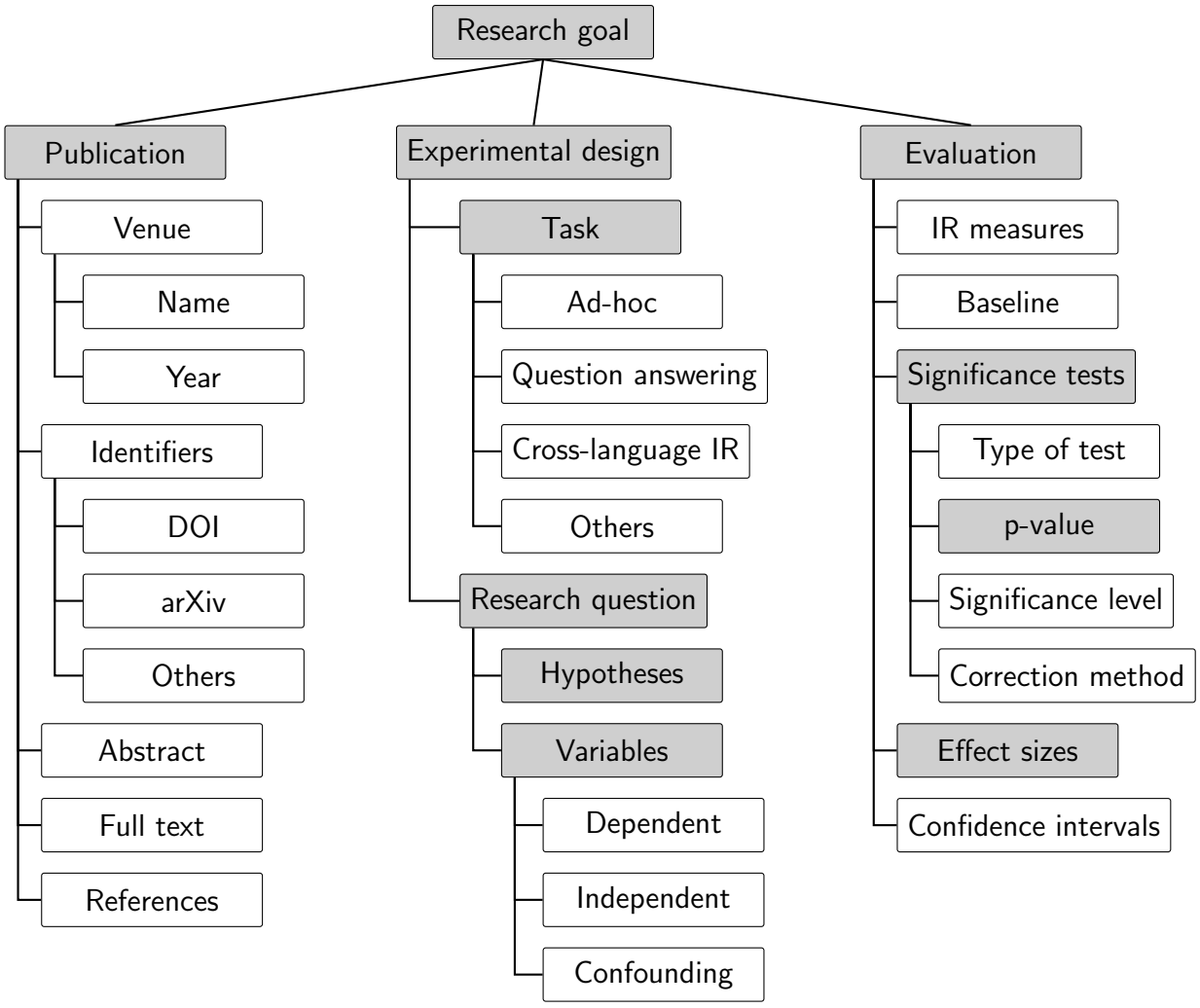

Figure 3.3: **Extended research goal taxonomy:** the gray concepts were also (implicitly) mentioned by Ferro et al. [132], while the others were added by us.

In a typical IR experiment, the retrieval method is compared to a *baseline* method that should be outperformed. In such a case, the null hypothesis assumes no significant difference exists between the retrieval effectiveness of the two compared methods. In order to reject the null hypotheses, a statistical significance test has to be conducted. Depending on the evaluation scenario, different *types of tests* and related to this different *test statistics* have to be considered, whereby the paired t-test is the most popular if the same test collection compares two methods [402]. In this context, we refer the reader to Ferrante et al. [129], who emphasize that measures should be interval-scaled for meaningful significance tests.

In order to make the statistical significance testing more transparent, the resulting *p-values*, as well as the *significance level*, should be documented. Furthermore, particular attention should be paid to the multiple comparisons problem. For instance, if multiple "versions" resulting from different parameterizations of the retrieval method are compared to the baseline. In such a case, a *correction method* has to be applied to rectify the alpha level. Likewise, including *effect sizes* and *confidence intervals* allows better conclusions about the research goal.

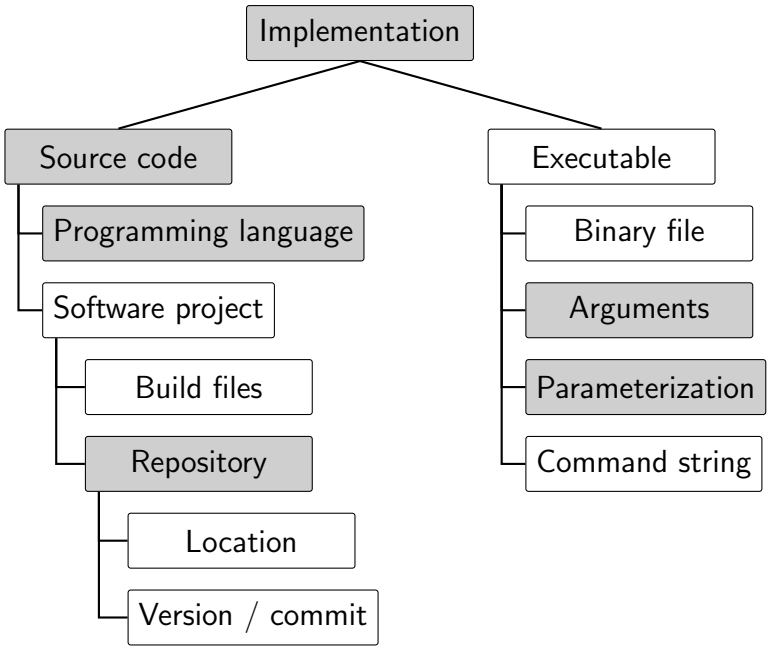

Figure 3.4: **Extended implementation taxonomy:** the gray concepts were also (implicitly) mentioned by Ferro et al. [132], while the others were added by us.

3.1.3 Implementation

The *source code* is a different means to communicate the experiments (cf. Figure 3.4). The actor implementing them should write clean and understandable code and annotate it with meaningful comments where necessary. Besides the actual source code, *build files* and an elaborated project structure can make the *software project* more assessable for others. Likewise, metadata information about the *programming language* can help to know what kind of proficiency is required to rerun the experiments. In general, reproducibility can be supported to a great extent if the actor follows good software development practices (cf. Subsection 2.6.3).

As already pointed out, Git, or version control software in general, helps make the development process more transparent [338]. In order to support reproducibility, it should be documented where the source code repository can be found, i.e., the *location* is favorably an open-access repository hosted on the web (example services include GitLab [471], GitHub [460], or Bitbucket [452]). In some cases, it may not be enough to have a pointer to the source code location if the software implementations are still under active development. As some of the critical implementations may change over time, the *software versioning* should be documented if there are release candidates or official releases. Otherwise, the *commit* hash can serve as a substitute.

However, an open-source implementation is not a hard requirement for reproducibility as results could also be reproduced with closed-source software that is available to the reproducers as a pre-compiled *binary file*. Furthermore, even if the code is hosted in a public repository, it does not mean that the experimental results are reproducible per se. A well-documented reproducibility protocol should also feature instructions on running the code. Besides setup instructions, it should be documented how the software has to be *executed*, for instance, by reporting the *command line string* that contains the parameters and arguments.

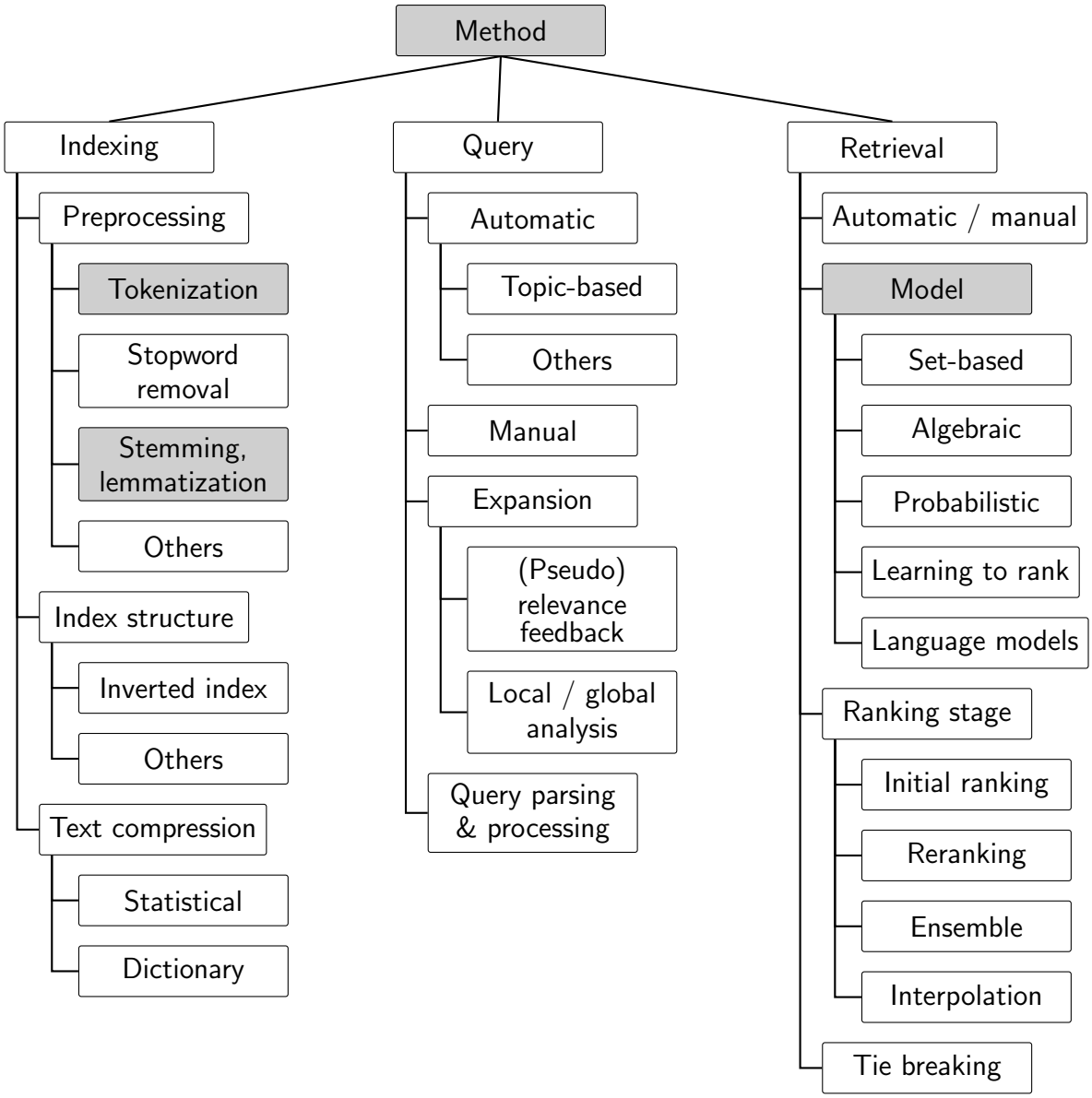

Figure 3.5: **Extended method taxonomy:** the gray concepts were also (implicitly) mentioned by Ferro et al. [132], while the others were added by us.

3.1.4 Method

The method describes the mapping of query-document pairs to a retrieval score, which is commonly made possible by a *retrieval model*. In the broader sense, the method does comprise not only the actual model but also the *indexing* and the *query processing*, all of which are included as subcomponents of the method in Figure 3.5.

The *preprocessing* pipeline as part of the indexing is usually composed of several standardized and established processing steps, including the *tokenization*, *stemming* or *lemmatization*, the *removal of stopwords*, and others, for which different algorithms or methodologies exist, but which are the primary focus of modern IR research. Nonetheless, it was shown by Ferro and Silvello [143] that it is not insignificant which methodology is used for the particular operations since they lead to different, sometimes significant, differences in retrieval effectiveness. Roy et al. [350] also analyzed the influence of markup artifacts on reproducibility. For this reason,

it is recommended to make them explicit and to use established retrieval toolkits and software libraries, as they facilitate reproducibility and comparability [250].

Besides the preprocessing of the index terms, different *index structures* exist, like those based on *inverted files* [446] or suffix arrays [22]. Related to the index structure, different techniques for *compressing* the index and texts can be classified into *statistical* and *dictionary-based* methodologies. We note that few reproducibility studies analyze index structures or related aspects [270]. However, proactive solutions exist to make the index files reusable [251].

The query is the user's conception and representation of the underlying information need and serves as the input to the method. In system-oriented experiments with test collections [355], it is often *automatically* extracted from the text of the *topic file* and thus represented as a keyword-based string. However, there are also more natural queries, for instance, as part of question-answering tasks [410]. Besides automatic runs, TREC also evaluates results from human queries, which are considered as *manual* runs [413]. Even though such an experiment can be repeated if the human-formulated queries are logged and provided for future reuse. However, it may not be easy to reproduce such an experiment with a different group of users as they might formulate other queries for the same topics.

Query expansions can result from a user's reformulation of an earlier query after having seen the first ranking or from query suggestions. The relevance feedback that is required can generally be classified into *explicit* [352] and *implicit* [15, 335] types. For instance, explicit relevance feedback [352] can be based on user judgments and implicit feedback can be based on interactive feedback data like clicks. Automatic retrieval methods model relevance feedback by *local analysis* [15] based on the contents of earlier retrieved documents, whereas *global analysis* [335] makes use of external resources like thesauri that are used for the query refinement. Similar to the preprocessing of indexed documents, the user's query should undergo a parsing with the same operations to derive a system's representation of the query that can be used for matches against index terms.

The actual *retrieval* is based on a model. Following the taxonomy by Baeza-Yates and Ribeiro-Neto [22], retrieval models can be classified into *set-based*, e.g., Boolean models, *algebraic*, e.g., the vector-space model [208, 354], or *probabilistic*, e.g., the BM25 model [345]. More modern, *learning to rank* approaches are either based on conventional ML methods [258] or neural networks [295]. More recently, Transformer-based approaches led to significant leaps in retrieval performance [252].

Since these effectiveness gains in the retrieval performance come at the cost of computational efficiency, multi-stage pipelines are composed of several low-cost lexical-based methods like the probabilistic BM25 model from which the initial result lists are reranked by more costly rerankers based on dense vector representations of the queries and documents. Consequently, multiple retrieval methods can be combined, but their stage in reranking pipelines should be made explicit, for instance, as *rerankings*, *interpolations*, or *ensembles*. Lastly, it is critical to *break score ties* once the rankings are retrieved to support better reproducibility. Lin and Yang [253] recommend breaking them with the help of external collections.

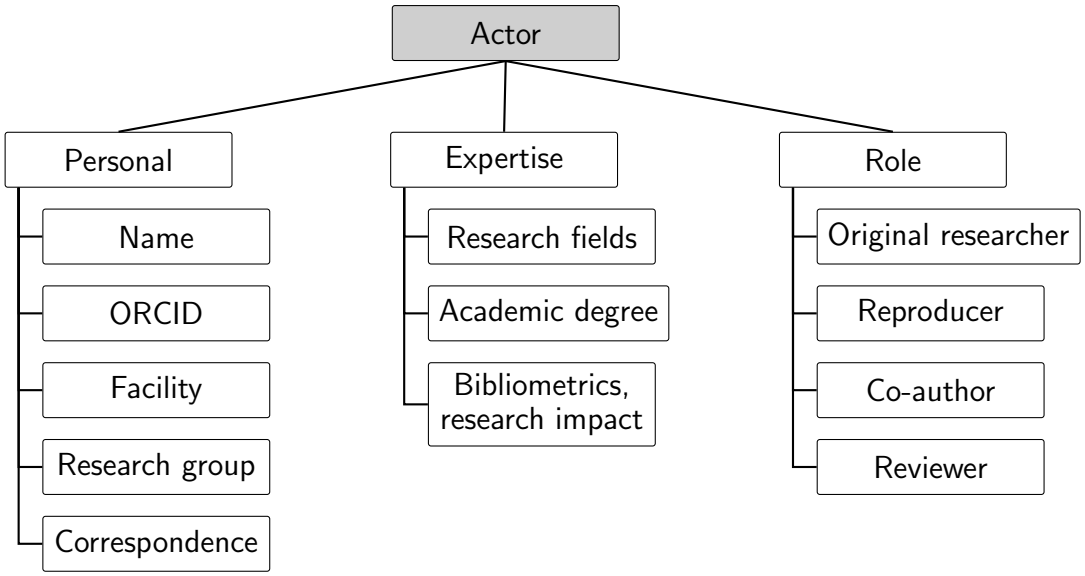

Figure 3.6: **Extended actor taxonomy:** the gray concepts were also (implicitly) mentioned by Ferro et al. [132], while the others were added by us.

3.1.5 Actor

The actor's attributes (cf. Figure 3.6) can be divided into *personal* information, the level of *expertise*, and the *role* in the experiment. First, the personal information should refer to the actors by their *names*. In addition, the *Open Researcher & Contributor ID (ORCID)* [168] should be used in publications as a unique identifier that avoids name ambiguities. Likewise, the *research facility*, e.g., the university, research department, or corporation, adds additional information about the actor's context. Very often, the corresponding *research group* has a name and an acronym used for tagging the TREC run files, for example.

Even though the actor's influence on the experiment should be minimized and, at best, eliminated [190], the original actor's support during a reproduction attempt by others is still a deciding key factor, as shown by Raff [337]. For this reason, the actor's *correspondence* should be given an e-mail address or other social media contacts, for example.

When evaluating the reproducibility of an earlier experiment, the reproducer's *expertise* must meet the required domain knowledge. While the publications should be written in a formal and understandable way and can be targeted at a general scientific audience, it cannot be expected that every reader will be able to reimplement the entire experiment. Specific reproductions require a certain level of expertise or familiarity with the technology, and it could help during a reproducibility study to know about the original actor's *research fields*.

Even though it is not often made explicit, certain reproducibility efforts make implicit assumptions about who should be able to reproduce the experiment as exemplified by larger-scale reproducibility attempts in student projects [261,328] or as part of the SIGIR Artifact Badging where the junior and senior members evaluate the software and methodology, respectively. Even though *bibliometric indicators* are only a weak proxy for the level of expertise in a research field, they can be used in combination with other attributes to estimate the actor's expertise better.

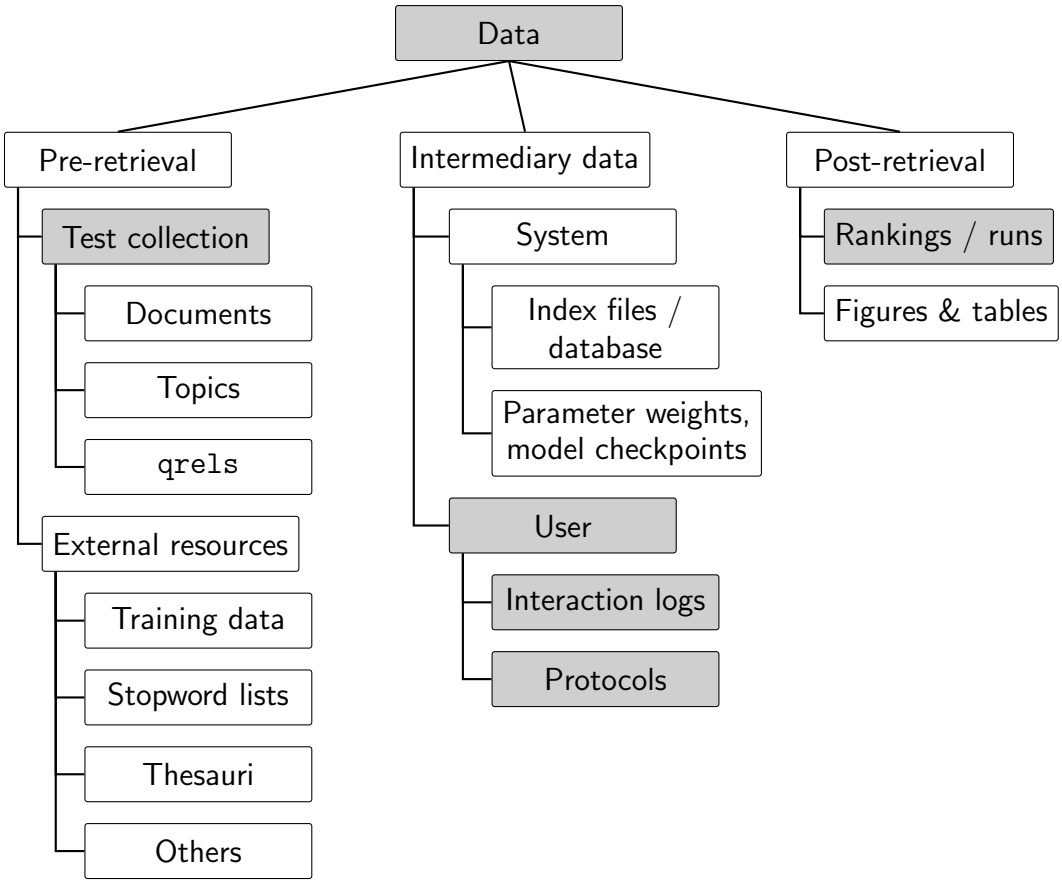

Figure 3.7: **Extended data taxonomy:** the gray concepts were also (implicitly) mentioned by Ferro et al. [132], while the others were added by us.

Lastly, the actor's roles in an experiment can be distinguished into the *original researcher* and the *reproducer*. However, the actor's scope is not limited to the original researcher who runs the software. Often the research design and the software implementations are a collaborative product, and they involve the work of multiple experimenters, and the *co-authors* should be considered as a special type of actor. Similarly, a different actor A′ can also be a future "version" of the original researcher who tries to rerun an earlier experimental setup. During the review process, the scientific work undergoes an additional validation by the *reviewer*, who can also be seen as a subtype of an actor.

3.1.6 Data

We distinguish the data subcomponents of Figure 3.7 into *pre-retrieval*, *intermediary*, and *post-retrieval* data. It should be considered that the data comprises not only the *test collection* in a retrieval experiment but also *intermediary* data outputs such as index files, the data artifacts, and *experimental results* after the experiment has been conducted.

Nonetheless, one of the most constituting data components is the IR test collection, which usually consists of a *document collection*, a set of *topics*, which express information needs usually by `title`, `description`, and `narrative` fields, and the corresponding *editorial relevance judgments* (`qrels`) made by domain experts.

Sometimes, the document collection — hosted separately from the topics and relevance judgments — may be hidden behind a paywall. For instance, the test collection of the TREC Common Core 2017 track [4] used the *The New York Times Annotated Corpus* made available by the Linguistic Data Consortium [486] after the payment of a license fee. In such a case, it should be explicitly documented where the single resources of the test collection can be found.

As mentioned in Subsection 2.6.3, `ir_datasets` [265] is a comprehensive data catalog that unifies the resources of IR test collections in a standardized way. It is an excellent resource to make sure that researchers build up their experimental pipelines on standardized data inputs. Each collection is associated with a unique identifier that could be used to document which collection is used in the experiments.

Besides the test collection, other *external data resources* can serve as additional input for an experiment. These include but are not limited to external *training data* resources, *stopword lists*, *thesauri*, or word embeddings.

Intermediary data outputs are those that emerge during the conduction of an experiment. Here, we distinguish between those data outputs by the *system* and those resulting from *users*. On the system side, there are the *index files*, but also *parameter weights* of the learned and adapted retrieval model, for instance, in the form of *model checkpoints* resulting from the training of DL methods for downstream ranking tasks. On the other hand, data outputs also emerge from the user's side as *interaction logs* or *interview protocols*.

The *post-retrieval* data includes the rankings in the run files, which are conventionally written to the disk as text files following the TREC format, which is composed of six columns (cf. Subsection 4.2.1). Even though they are not specific to IR research, post-retrieval data also includes results from experimental evaluations such as *figures* and *tables* for which the corresponding scripts or instructions should be provided as well [32].

3.1.7 User

By its original definitions, PRIMAD considers users to be part of the data component, represented by their data trace resulting from the interactions with the search system. The general motivation behind introducing an additional *user* component to the PRIMAD taxonomy is based on our critique that the user's influence is not sufficiently represented as a subcomponent of the data but requires a more explicit representation. Only by shifting the focus of the experimental evaluations towards more user-oriented aspects is it possible to assess the real-world impact and draw conclusions about the external validity of the system-oriented experiment.

The ultimate goal should be to answer whether the system-oriented effects and observations can be reproduced in user-oriented experiments, i.e., real-world environments. On the one hand, user-oriented IIR experiments allow us to answer research questions with real human subjects. In this case, compromises have to be found between small-scale user studies that give the experimenters more control over the human subjects and large-scale experiments in the form of A/B experiments with larger subject groups and a more substantial basis for statistical analysis [218, 232]. However, small-scale studies are generally considered to be not reproducible [386], and large-scale experiments can only be conducted at the cost of the control over

the users' context. Most strikingly, both types of user-oriented experiments can be costly to conduct and implement.

As a compromise, user simulations are generally considered a cost-efficient way to answer *what-if* questions when it is not feasible to conduct a user study. Of course, the generalizability of the conclusions drawn from simulated experiments strongly depends on the fidelity of the simulated user model. However, simulation experiments are generally considered reproducible as there is a more explicit understanding of user behavior. Thus, we see simulations as a way to analyze the external validity of an IR experiment, as it will be picked up in the later chapters.

To this end, we aim to provide an overview of what kinds of user attributes or aspects could be considered variables in user simulations that complement system-oriented IR experiments. The corresponding taxonomy tree of the user in Figure 3.8 is motivated by the cognitive models of information seeking and findings from user-oriented IIR experiments. In the following, we review how these concepts and studies contribute to the taxonomy tree, which comprises three subcomponents: the context of the user, the interface, and the interactions. Afterward, we outline how the taxonomy could complement user simulations in system-oriented IR experiments.

Cognitive Models of the Information Seeking Process

In system-oriented IR experiments, the information need usually has a well-defined scope as it is described in the topic files of a test collection. However, the general assumption in the information sciences is that the users themselves cannot always formulate or specify their information needs [97]. Taylor's four-layer model [388] considers the actual information need as visceral and inexpressible. Similarly, Belkin [37] pointed out that the information need is intangible and considered the users to be in an *Anomalous State of Knowledge* when beginning an information-seeking task.

In order to satisfy the information need and to dispose of the *Anomalous State of Knowledge*, the user is part of an information search process, which several cognitive models describe. In general, there exist more comprehensive cognitive models that try to describe the information search process as a whole by embedding the search activities into a larger context [120, 197, 236], and other more specific cognitive models mainly focusing on the interactions with the information system and objects during the search process [35, 72, 422].

All cognitive models treat the understanding and conception of the *information need* and the derived work and *search tasks* as dynamic as they evolve and may change during the user's search progress. For instance, Bates's model [35] uses the analogy to berrypicking, where the user selects fitting information objects and uses the extracted information, e.g., keywords, for the following queries. Depending on what has been found, the user's understanding of the information need is redefined, and the search progresses towards fulfilling the overall task.

Based on the analysis of the information-seeking behavior of social scientists, Ellis [120] proposed a behavioral model covering six stages, including 1) starting to search for information, 2) chaining by following chains of citations, 3) browsing as a form of semi-directed search, 4) differentiating the examined sources, 5) monitoring as an ongoing process to stay informed about state-of-the-art findings in a particular research field, and finally 6) extracting material of interest in a particular source. Wilson [422] embedded Ellis' model [120] into a more general model for information-seeking behavior. More specifically, it adds the user's context and related barriers.

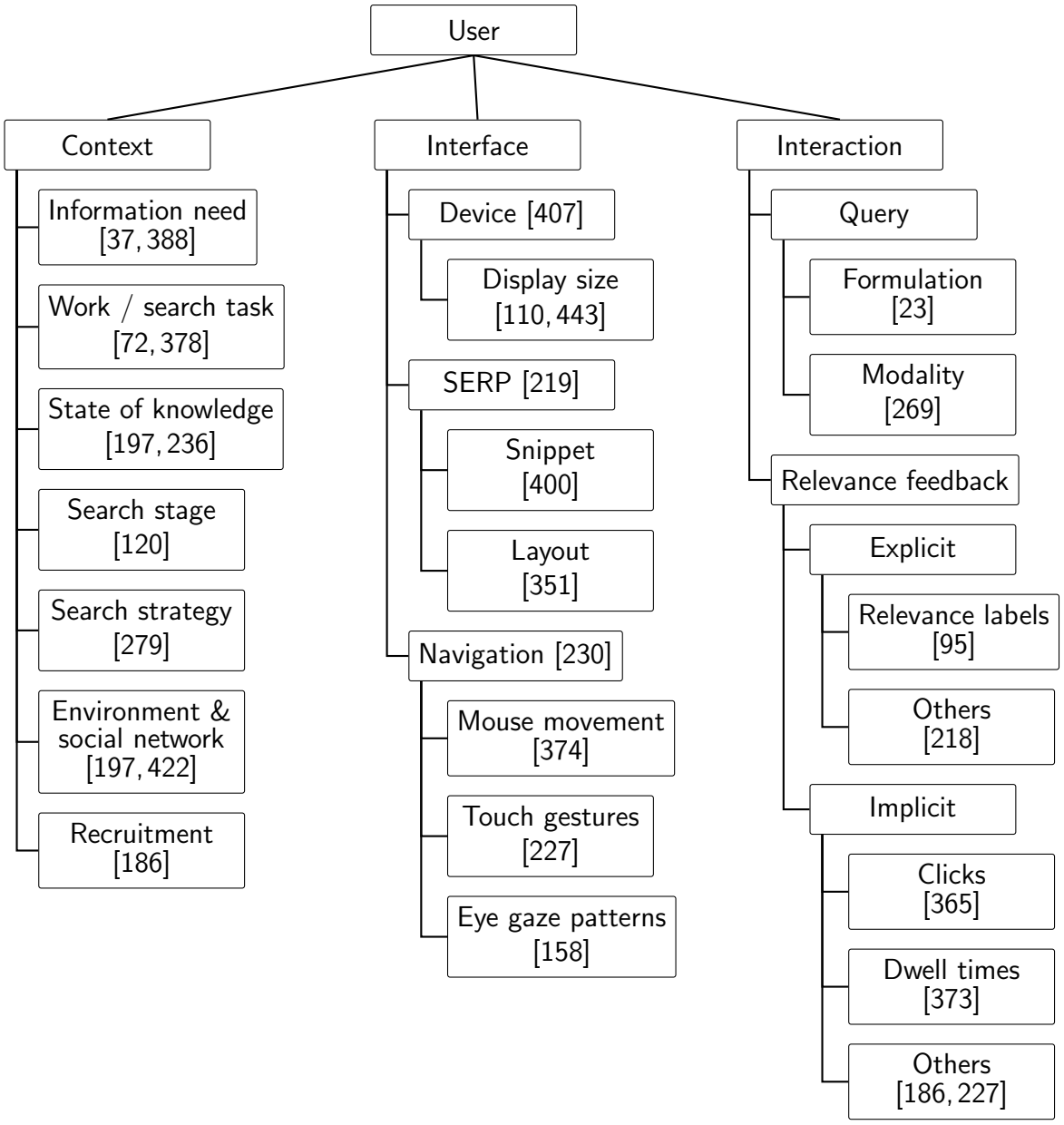

Figure 3.8: **User taxonomy**: an extension to PRIMAD [132].

Before the users can engage in the information-seeking process outlined by Ellis, they must overcome certain barriers that depend on the user's context, which comprises the role within a (work) *environment*.

Similarly, Kuhlthau's *Information Search Process Model* [236] includes different stages of thoughts and feelings that evolve during searching and result in different forms of how the user interacts with the systems and the provided information. In this regard, Marchionini [279] proposed a taxonomy of different search activities, which defines the user's interaction with the search results depending on the interface features and the chosen search strategy. Ingwersen's cognitive model provides a holistic view of the users and their interactions with the search system by considering the *search interface* and the *social context* [197]. Byström and Järvelin [72] highlighted that information seeking is influenced by *task complexity* and related

problems. They categorized task complexity into five levels and distinguished different information types based on empirical evidence.

In sum, all of these cognitive models consider that the user's context, including the understanding of the information need and the corresponding work or search tasks, as well as the acquired knowledge regarding the informational gap, throughout the search process, changes and leads to different kinds of interactions with the system and the provided information. Opposed to the system-oriented experiments where the information need is rather static, and only a single query is used, it should be pointed out that the real user's search behavior is an iterative and multi-staged process. It comprises multiple query formulations and interactions in response to the provided information objects that, in turn, strongly depend on how the results are presented by the interface and the individual user's preferences.

Finally, it should be considered that in many IIR experiments, the users are recruited [186], which superimposes an artificial context on the subjects. Once users know the experimental context, it may influence their behavior and, thus, the overall outcomes (cf. Hawthorne effect [268]). Furthermore, since there is no intrinsic motivation when simulating work tasks in small-scale user studies, participation rewards can also influence data quality.

In conclusion, the user's *context* is motivated by cognitive models that describe the search process. We note that specifying or assessing the context and related concepts in a user-oriented experiment is not always feasible. As such, these contextually "hidden variables" should be carefully considered as they can affect the reproducibility as confounders [232].

User-oriented IIR Experiments

In system-oriented IR benchmarks like those of TREC, the retrieval system is the target of the evaluations and very often the only variable of the experimental evaluations. At the same time, other components like the data collection and relevance judgments are kept fixed [23]. In this sense, Zobel [445] considered the numeric scores of system-oriented evaluations as proxies, which at best approximate the qualitative objective, and Zobel highlights that there is a gap between the measured effectiveness gains and the real-world impact on the user effectiveness. He pointed out that there is no guarantee that improving the system's effectiveness translates into benefits for the user experience. As a solution, IIR experiments involving users can complement system-oriented evaluations [218].

In this regard, several user studies compared system-oriented evaluations to outcomes of user studies [179, 180, 181, 396], i.e., in the studies it was analyzed how the user effectiveness (dependent variable) behaves in response to systematic changes of the system effectiveness (independent variable). For instance, Turpin and Hersh showed that statistically significant improvements over a baseline system in terms of Average Precision (AP) do not translate into comparable significant improvements of user effectiveness. Their experiments suggest that users can compensate for inferior results by browsing the ranking list, and they further assume that the improvements over the baseline system also depend on the exact query formulation as provided by the topic file of a test collection, which the users might not choose [396].

Another study by Turpin and Scholer [399] showed that the relationship between user effectiveness in web search tasks and system-oriented outcomes is limited. In a precision-oriented experiment, they compared users' time frames to find the first

relevant document in a ranking. The results did not indicate a significant relationship with the underlying system-oriented performance in terms of AP. Their recall-oriented experiments, where users were asked to gather as many relevant documents as possible within a set time frame, resulted in a weak relationship between the user and system performance. Similarly, Smith et al. [372] observed that users adapt their search behavior when they use a less effective search system, leading to overall similar search results as if a more effective system would have been used.

Besides these rather small-scale user studies, there are different implementations of how users can be involved in experimental evaluations. We refer the reader to Kelly's continuum [218] that spans different types of IR studies ranging from TREC-style studies with a strong system focus towards strongly human-focused studies.

Overall, it is still an open research question how the measured improvements over a baseline of system-oriented evaluations translate into benefits for user effectiveness. However, it has to be pointed out that the system effectiveness, and the underlying retrieval method, are not the only variables that possibly affect the outcomes of a user study. The previously introduced *contextual* aspects of the user also have to be considered as influencing factors, and cognitive and interaction-focused models help to define relevance from an individual user's perspective.

Another influential component in a user-oriented experiment is the *search interface*. Even though it cannot be considered a user property, it is an experimental component tailored for human interaction and reception, which is often only implicitly considered or fully neglected in system-oriented experiments. It basically constitutes what kinds of user interactions are possible. For example, web search results are presented to the user as a SERP, in which the *snippet* texts preview the document's content and strongly influence which search results users draw their attention to and subsequently consider relevant. Turpin et al. [400] let users make relevance judgments based on summaries of documents similar to snippet texts and used the newly generated relevance labels for system benchmarks. Their experiments revealed differences between the system evaluations based on judgments made either with the full text or the summary. While click decisions have to be seen in the context of their ranking position in the SERP, relevance annotators make judgments for every document in the pool, which means that it is not part of the annotation process to select the particular document from a SERP. Furthermore, clicks are often based on the attractiveness of the snippets, while annotators decide about the relevance after having screened the entire document. Likewise, the search *device* and the corresponding *screen size* impact how many results can be displayed per page. We note that this *selection process* in web search tasks is very different compared to the process of making editorial relevance judgments, where the domain experts decide about the relevance after having screened entire documents presented one by one and often in random order.

Our taxonomy distinguishes between explicit and implicit forms of relevance feedback. The editorial relevance judgments of test collections are an explicit and objective type of relevance feedback. However, explicit feedback can also be collected from IIR experiments in different forms like "think aloud" protocols, user interviews, or questionnaires. Recently, Gäde et al. [151] introduced a manifesto on how resources of interactive user studies can be prepared for future reuse, and they additionally propose the *User Study Exchange Format*, which is a specification of the corresponding data format.

User Simulations

The most prominent user model in system-oriented evaluations implies that the user formulates a single query for a given information need, scans the entire result list until a fixed rank, and judges the relevance of each item independent of any context knowledge, e.g., from previously seen results [31, 284]. However, depending on the IR measure, additional assumptions about the underlying user model are made as part of the evaluations. For instance, Normalized Discounted Cumulative Gain (nDCG) [202] discounts later items in the ranking by log-harmonic weights and, thus, simulates the user's persistence. Similarly, the Rank-Biased Precision (RBP) [298] also allows defining the user's persistence. In comparison, RBP's discount follows a geometric sequence and is a recall-independent measure. nDCG's normalization requires knowledge about the recall, which is a failing in modeling user satisfaction as the recall is unknown to real users according to Moffat et al. [298]. In this regard, RBP only measures the quality of search results as perceived by the user and does not require knowledge about all relevant documents, i.e., the recall.

Carterette introduced a coherent framework for model-based measures [77]. According to this framework, measures are composed of three underlying conceptual models: a browsing model, a model for document utility, and a utility accumulation model. Similarly, Moffat et al. [296] introduced the C/W/L framework to describe a family of parameterizable evaluation measures that account for the user browsing behavior by formalizing the conditional continuation probability of examining items in the ranking list. Both of these frameworks are able to describe conventional measures like nDCG, AP, or RBP but also allow for the analysis of derived variants. While these model-based measures allow for a principled system-oriented evaluation over different topics with certain assumptions about the user behavior, they are still a strong abstraction of how the user interacts with the search system, and the user behavior has a somewhat static notion.

By building upon the idea of extending the underlying user model of system-oriented experiments, simulations make it feasible to evaluate retrieval systems with regard to more *dynamic* user interactions. For instance, earlier seen retrieval results can be exploited for more diverse query formulations over multiple result pages, situational clicks, relevance decisions, and diverging browsing depths [78]. Simulated IR experiment date back to the early 1980s [384, 385], but more recently, several frameworks and user models were introduced [34, 79, 285, 286, 317, 391, 443]. Inspired by the user models of Baskaya et al. [34] and Thomas et al. [391], Maxwell and Azzopardi [285, 286] introduced the *Complex Searcher Model*. Carterette et al. [79] proposed the idea of *Dynamic Test Collections*, and Pääkönen et al. [317] introduced the *Common Interaction Model*. Besides, Zhang et al. [443] introduced another search simulation framework.

While these user models were used to answer different research questions, they all share several elements of the typical session-based search process that is illustrated in Figure 3.9. More specifically, it depicts nine stages of the simulated search process and shows how the subcomponent of the taxonomy in Figure 3.8 (the user's *context* and *interactions* with the *interface*) could be considered in the simulation stages.

The overall search process of the user can be described by different states or stages and the corresponding transition probabilities. At the beginning of a session, the simulated user is induced with an information need, for which the topic file of a test collection can be used (cf. ❶). However, beyond extracting pre-defined

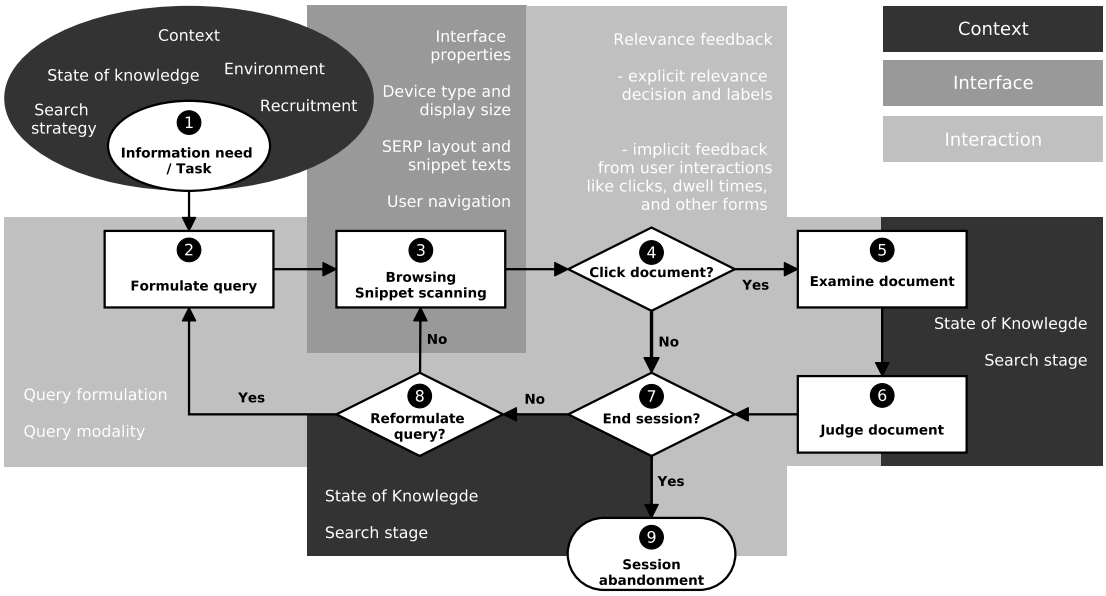

Figure 3.9: User simulation model abstracted and reproduced from [34, 79, 286, 317, 391, 443] and extended by the taxonomy components given in Figure 3.8.

`title` or `description` fields from the topic file, it could also be considered that the user's context influences the information need. For instance, the user might have wording problems as the information need is not always as explicit as in a topic's description, or the user might have prior knowledge about specific topics. In Chapter 6, we model user simulators with different knowledge states and analyze how the resulting queries influence the retrieval effectiveness.

The query formulation process results in a query string that serves as the input to a retrieval system (cf. ❷). It has been acknowledged in several studies that the implicit system-oriented user model does not adequately reflect the query formulation behavior of real users [23, 222]. Most apparently, users tend to formulate more than one query in a search task [23], and different users probably formulate different queries for the same information need [222]. Several principled rule-based methods for the query generation were introduced by Baskaya et al. [34] but also more sophisticated approaches based on language models exist [19, 211].

Upon the return of a ranking, the user starts examining the search results by scanning the snippets (cf. ❸). Depending on the attractiveness of a snippet text, the user may click (cf. ❹) and decide to examine the document text (cf. ❺), and finally decide upon the relevance of the document (cf. ❻).

Click decisions can be modeled by probabilities biased towards the level of the editorial relevance judgment [34, 185]. If session logs are available, click models are a more elaborated way to simulate click interactions. Most click models are based on pre-defined rules of how the underlying user iterates over the result list. The parameters of observable and hidden variables are estimated from session logs [88].

Most click models assume a simple layout of the SERP, i.e., *ten blue links*, or do not make it explicit, but it might be reasonable to account for the interface properties in the simulations. For instance, by including the effort that is required to search or navigate through the results with a particular device [407], how the display size may introduce bias [110, 443], or how the snippet texts might affect the click decisions and later query reformulations [400]. Chapter 7 picks up the idea of evaluating the system performance with the help of click feedback. We then analyze to what extent the relative system performance can be reproduced when editorial relevance judgments are unavailable.

Eventually, the user returns to the ranking list (cf. ❼) and either continues traversing the snippets, reformulates the query (cf. ❽), or abandons the session (cf. ❾). Stopping rules and decisions have been extensively studied by Maxwell and Azzopardi [284, 285, 286, 287, 288], while Pääkkönen et al. [316, 317] analyzed the overall effectiveness gains in simulated sessions for different user behaviors. The cognitive models outlined the characteristics of the search stages that emerge during the entire search process. As such, the user's knowledge state changes, which could also be modeled as part of user simulation, e.g., by the previously seen documents.

According to Carterette et al. [79], a (user) simulation model does not have to be an exact replica of the physical, real-world entity in every regard but instead can only deliver answers in the abstraction level for which it is modeled, and as such, it should be able to deliver answers at the abstraction level of the research goal. In this regard, several approaches can be used to validate the fidelity of the user simulations and to which degree they comply with real user interactions.

Labishetty and Zhai [238, 239] introduced the Tester-based approach to evaluate the fidelity of a user simulator. A Tester is based on heuristics or high-confidence assumptions about the relative performance of two or more retrieval methods. For instance, it has been shown in several studies that the BM25 ranking method is more effective than ranking solely based on the term frequency. Therefore, a user simulator should be able to identify the better-performing system, i.e., in the example, the simulations should decide on BM25 as the better-performing ranking method.

With a particular focus on query evaluation, Günther and Hagen [164] analyzed to which extent query suggestions can be used as query reformulations in simulated sessions and conclude with overall positive outcomes. However, they conclude that topical drifts can be problematic as more query suggestions are used. With a particular focus on conversational recommender systems, Zhang and Balog [441] introduced a simulation model based on natural language generation and understanding that shows a high correlation with human-based evaluations.

Maxwell and Azzopardi [286] developed a toolkit called `SIMIIR` that allows managing the user behavior with regard to the interactions outlined above by configuration files in a principled way. It was reused and extended in recent works by Câmara et al. [74], which decomposed more complex search tasks, while Zerhoudi et al. [438] introduced an updated version, which extends the framework by several features like Markov model-based interactions with the result list and more elaborated query reformulations. With a particular focus on reinforcement learning for recommendations, Huang et al. [192] emphasized that the presentation or popularity bias can negatively affect the simulation process when the training is based on logged user interactions. As a countermeasure, they introduced a debiasing method that is realized as an intermediate processing step before training the model. Another

recent research trend focuses not only on the simulation of the user but also on the simulation of IR test collections. Simulating a test collection is a feasible solution when it is critical to preserve the user's privacy or when the data collection cannot be shared due to business interest [174].

3.2 Metadata Annotations of TREC Run Files

This section describes the metadata annotation schema for TREC run files, which we introduce as `ir_metadata` [63]. Before describing it in Subsection 3.2.2, we shortly review recent trends regarding annotation frameworks in ML research. More examples and details about the metadata schema can also be found in Appendix B.

3.2.1 Recent Metadata Trends in ML Research

Recent trends in ML suggest that metadata information or protocols about the computational experiments are considered as a solution to make the experimental data more transparent and likewise easily reproducible. A non-exhaustive list of description frameworks includes ML reproducibility checklists [324], model cards [294], datasheets for datasets [57], data statements [40], FactSheets [14], and dataset nutrition labels [189].

As part of the NeurIPS conference, Pineau et al. [324] introduced ML reproducibility checklists that are presented twice in the form of a questionnaire to authors when submitting their papers to the peer-review and as a camera-ready version upon acceptance. The questionnaire covers more general aspects, such as the mathematical formalization in the paper, but also more detailed questions regarding the specification of hyperparameters and adequate statistical evaluations. In conclusion, Pineau et al. highlighted that these checklists could help the reviewers in their decision-making regarding the acceptance of the papers.

Model cards [294] are a reporting framework for ML models. Motivated by the issues related to systematic biases in ML models, Mitchell et al. aimed to address the need for a standardized description framework that defines the scope or application context of ML models. A model card should contain general details about the model, such as the intended use case, evaluation metrics, and ethical considerations. The authors proposed nine categories that may require individual descriptions for the particular ML model. The authors provided two examples of model cards for image and text classifiers, and they consider it a supplement to the datasheets [154].

Inspired by the documentation practices in the electronics industry, Gebru et al. [154] criticized that datasets for ML research lack comparable practices and that there is currently no standard for documenting ML datasets. They introduced datasheets for datasets and outlined a catalog with 57 questions that dataset curators should address to make the dataset characteristics more transparent to the consumers. They emphasized that the annotation workflow is not intended to be automated since the annotation quality would benefit from careful reflection during the maintenance process. By documenting datasets with datasheets, Boyd [57] envisioned mitigating potential biases, better conditions for reproducibility, and a more straightforward decision process when searching for the right dataset. In addition, Boyd emphasized that datasheet for datasets also increases the awareness of biases in the training data that could lead to ethical concerns.

Data statements [40] is a proposal for a documentation scheme of NLP datasets. The corresponding authors defined a vocabulary for data statements that does not only include annotators but also other actors involved in the creation process of the dataset, such as speakers (creators of the text contents), curators, and stakeholders. Furthermore, they proposed to document NLP datasets by different categories related to the properties of the text-based contents and the demographics of the actors to account for potential biases.

Inspired by documentation procedures from the industry, FactSheets by Arnold et al. [14] is a proposal for documenting AI services. It describes relevant characteristics of an AI service like use case, performance, security, and safety. In the corresponding publication, the authors provide motivating questions that can be used to describe some of these characteristics. The dataset nutrition label [189] is another proposal for documenting datasets by different labels, including metadata information, data provenance, and some statistical attributes.

Leipzig et al. [241] reviewed existing metadata formats for the computational sciences. As part of their evaluations, they had a particular focus on how metadata supports reproducibility and outlined five categories for metadata levels, including the (1) input, (2) tools, (3) statistical reports and notebooks, (4) pipelines, preservations, and binding, and (5) publication. We refer the reader to this work for a general overview of metadata formats for the computational sciences.

3.2.2 The `ir_metadata` Annotation Schema

In the following, we introduce `ir_metadata` — an annotation schema based on the conventional PRIMAD taxonomy that can be used to annotate experimental artifacts of IR experiments, i.e., TREC run files. To our knowledge, the PRIMAD taxonomy has not been put into practice, which could be partly explained by the rather abstract definitions. Yet, we think these abstract definitions allow enough flexibility to report details as they are required for reproducible experimentation. It is not reasonable to follow a strict annotation schema as some details do undeniably not affect the reproducibility, e.g., reporting a GPU model when it is not used in the experiments. On the other hand, it is simply not feasible to think of all subcomponents, which will be crucial for reproducible experimentation in the future. For this reason, the introduced metadata schema proposes a set of essential subcomponents that should be reported if feasible but also keeps extensibility in mind.

According to ISO 23081-1, a metadata schema requires a "logical plan showing the relationships between metadata elements". Similar to other metadata standards and protocols in IR research like those of the DIRECT platform [3], the USEF standard for user protocols of IIR experiments [151], or other more general standards from the computational sciences [241], we introduce a lightweight and extensible metadata schema for system-oriented IR experiments based on related conceptual components. More specifically, we define the metadata schema by the components of the extended PRIMAD model from the previous Section 3.1.

From a practical point of view, we propose to add the resulting metadata annotations to the beginning of run files — similar to a file header. Hereby, we avoid the separation of metadata annotations and run files, and no additional storage capacities or external databases for the metadata are required. Technically, these file annotations are compatible with the already existing evaluation infrastructure since

the `trec_eval` toolkit recently introduced the support of comments in the run and `qrels` files as it can be seen in the corresponding code repository [485]. Figure 3.10 shows an example of such an annotated run file.

We propose adding the metadata annotations as comments with YAML syntax for easy readability and extensibility. Using YAML, the annotations remain free of markup artifacts like those from XML-formatted data, making the annotations more human-readable. In addition, YAML is a recent and well-supported data-serialization language for which many well-curated parsing libraries exist. Its minimalistic syntax facilitates metadata extensions while being both human- and machine-readable. We did not explicitly decide against any existing metadata standard but preferred YAML because of its simplicity. In the future, it is worthwhile to implement the support of other existing standards by developing parsers for `ir_metadata` to other metadata formats or standards like the DCAT-US Schema [456], which would also contribute to a more sustainable use of annotated resources.

For more specific details about the YAML formatting and the encodings that are required by ISO 23081-1, we refer the reader to Appendix B that includes definitions for metadata annotations including *descriptions*, *encodings*, and *YAML types* for each metadata field. These definitions can also be found on the official project website of `ir_metadata` [484], where they are documented as checklists that IR practitioners can use as an annotation help. The website's source code is easy to maintain, and we aim to develop it collaboratively with the community in the future. Its source code is publicly hosted on GitHub [469]. It can be easily extended by pull requests, for instance, when the checklists need updates or if it is required to adapt the terminology for specific descriptors.

To lower the manual annotation effort for IR practitioners, we envisage the automatic annotations of run files. In this regard, we already implemented some first annotation features, which are described in the following Chapter 4. Likewise, it makes sense to automatically check the validity and integrity of the annotations to ease better consistency and validate the absence of errors. This feature could use the already implemented automatic annotation features and give feedback on crucial missing metadata fields. Similarly, the single metadata fields should be prioritized regarding their importance for reproducibility. For this purpose, RFC2119 [482] can be used to assign requirement levels. To promote the outreach of the schema, it makes sense to extend the existing retrieval toolkits like Pyterrier [267] or Pyserini [250] with compatibility features. In addition, the collaboration with shared task organizers can help to promote the schema as organizers can encourage participants to annotate their runs at submission time. For instance, as part of the TREC Deep Learning track, participants were already asked to provide some meta-information about the submitted run files [103].

Finally, it must be pointed out that the metadata annotations are not bound to run files but can be used versatilely. While the primary use-case outlined the annotation of TREC run files, the annotations can also be used to document IR experiments if no run files are generated in the experiments. In that case, the metadata information could be added as YAML files to the code repository or appended to the LaTeX code of the publication, to name a few other annotatable resources.

```
# ir_metadata.start
# schema-version: 0.1
# run-version: 1.0
# tag: bm25
# platform:
#     hardware:
#         cpu:
#             model: 'Intel Xeon Gold 6144 CPU @ 3.50GHz'
#         ram: '64 GB'
#     operating system:
#         distribution: 'Ubuntu 20.04.3 LTS'
#     software:
#         retrieval toolkit:
#             - 'anserini==0.3.0'
# research goal:
#     venue:
#         name: 'ECIR'
#     publication:
#         doi: 'https://doi.org/10.1007/978-3-030-15712-8_26'
#     evaluation:
#         significance test:
#             - name: 't-test'
#               correction method: 'bonferroni'
# implementation:
#     source:
#          lang:
#             - 'python'
#             - 'c'
#         repository: 'github.com/castorini/anserini'
#         commit: '9548cd6'
# method:
#     automatic: 'true'
#     indexing:
#         stemmer: 'lucene.PorterStemFilter'
#         stopwords: 'lucene.StandardAnalyzer'
#     retrieval:
#         - name: 'bm25'
#           method: 'lucene.BM25Similarity'
#           b: 0.4
#           k1: 0.9
# actor:
#     name: 'Jimmy Lin'
#     team: 'h2oloo'
#     role: 'experimenter'
# data:
#     test_collection:
#         name: 'The New York Times Annotated Corpus'
#         source: 'catalog.ldc.upenn.edu/LDC2008T19'
#         qrels: 'trec.nist.gov/data/core/qrels.txt'
#         topics: 'trec.nist.gov/data/core/core_nist.txt'
# ir_metadata.end
307  Q0  497476  1 0.9931  bm25
307  Q0  469928  2 0.9674  bm25
307  Q0  125806  3 0.9623  bm25
307  Q0  504815  4 0.9453  bm25
307  Q0  392547  5 0.9223  bm25
...
```

Figure 3.10: Annotation example of a run file.

3.3 Conclusion

This chapter introduced an extension to the existing PRIMAD taxonomy. Besides complementing the conventional six components with subcomponents and related aspects, we have introduced an additional user component to provide a more holistic view of the IR experiment. Given the extended six conventional PRIMAD components, we have outlined a metadata annotation schema, which can be used to annotate TREC run files. We do not claim to introduce a complete taxonomy but one comprehensive enough to allow for a more principled analysis of system-oriented reproducibility studies. Chapter 5 exclusively focuses on evaluating the reproducibility of system-oriented experiments and exploits the metadata schema for principled reproducibility analysis.

By introducing the user component, we outlined what user-related aspects could influence the outcome and the reproducibility of an IR experiment. While it is often not feasible to acquire knowledge about all user-related aspects, they should generally be considered as possible confounders in online experiments and controlled modifications in a user simulation experiment. In the future, it should be defined how metadata about the user can extend the schema. In the case of user simulations, the user behavior usually follows a user model that could be translated into descriptions that fit our metadata schema. Within the scope of this dissertation project, we consider user simulations as a tool to analyze the external validity of an experimental outcome. To this end, both Chapters 6 and 7 focus on two central aspects of the user interaction with the search system that are the *query formulation* and the relevance feedback given by the user in the form of *clicks on search results*.

Chapter 4

Reproducibility Measures

In this chapter, we review the general approach of a reactive reproducibility study and how the reproduction quality can be measured in system-oriented IR experiments. The proposed measures were developed throughout a series of cross-venue workshops [133, 136, 353, 375] and finally put into context as part of an evaluation framework that quantifies the reproducibility and replicability with different levels of specificity ranging from fine-grained comparisons of document rankings to more general comparisons of topic score distributions [60]. In this framework, the experimental setup is aligned with the ACM Policy on Artifact Review and Badging (cf. Chapter 2 and [448]), i.e., we consider an experiment to be *reproduced* if the results can be validated with a reimplementation and the same test collection and to be *replicated* if the results are validated with another test collection. The resulting contribution can be described as follows:

C4 Reproducibility framework for reactive reproducibility experiments and a corresponding software toolkit (cf. [60, 61])

The remainder is structured as follows. At first, we outline the general design of a reactive reproducibility attempt in an IR experiment. Second, we describe the different levels of reproducibility and the corresponding measures starting from the most specific level towards more general statistical comparisons. Finally, we describe `repro_eval`, which is an open-source software library [61] that implements the measures at the end of this chapter.

4.1 Setup of Reactive Reproducibility Studies

In our experimental setup, the target artifact of the *reproducibility* attempt is a run r that contains rankings for n_D topics derived from a test collection D that has also been used in the original experiment. If another test collection D' is used, we consider the revalidation as a *replicability* experiment. The corresponding reproduced or replicated run is denoted as r'.

Figure 4.1 illustrates the general procedure. We assume that the group of reproducers is provided with the original run r and the corresponding publication, which describes some or possibly all of the details required to rerun the experiment. However, the original experimental setup, i.e., the software implementation, is missing. Based on the descriptions in the publication, the reproducers reimplement the

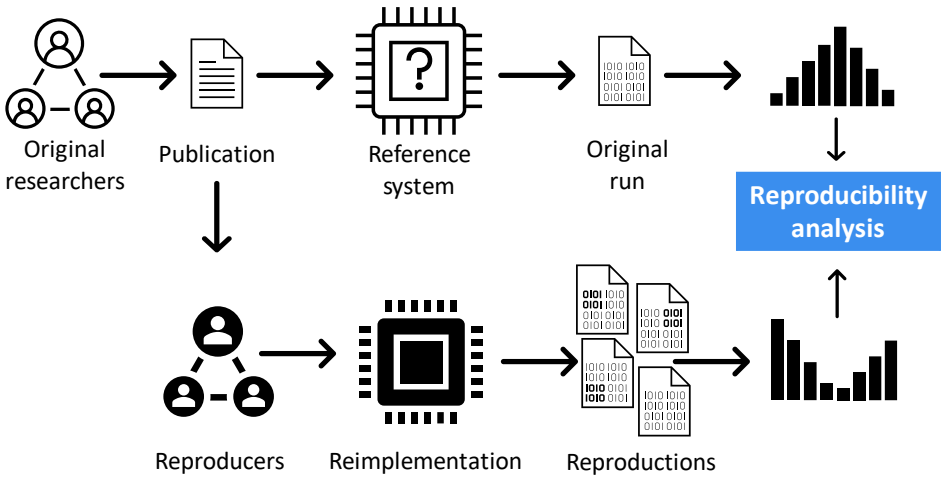

Figure 4.1: General approach of a reactive reproducibility study.

experimental setup as well as possible. However, some details, for instance, about hyperparameters, may be missing, and they try out variations leading to different versions of the reproduced run r'. How do they know which reproduction resembles the original reference the most? In this case, the degree of reproduction can be quantified by comparing r to r'. As the run files contain rankings for multiple queries, it is possible to compare the artifacts in several regards, which we consider as different levels of specificity. These different levels and the corresponding measures are described in the following section.

4.2 Levels of Reproducibility

Our proposed measures quantify the degree of reproducibility and replicability at increasing levels of specificity (cf. Figure 4.2). At the most specific level, it would be possible to determine the bitwise reproducibility of computational artifacts. However, as outlined in Subsection 4.2.1, this level of rigor may be too strict for most IR experiments. It is more reasonable to compare the correlation between the original and reproduced document rankings. Thus, we exploit Kendall's τ Union (KTU), which determines Kendall's τ with lists of ranks (cf. Subsection 4.2.2). In addition, the Rank-Biased Overlap (RBO) can be used to evaluate rankings with different sets of documents in both rankings and infinite lengths (cf. Subsection 4.2.2). The second level evaluates the effectiveness by the Root Mean Square Error (RMSE) between the topic scores of the original and reproduced run. We choose this measure since larger deviations are penalized more strongly due to the squaring of errors (cf. Subsection 4.2.3). The third level evaluates the overall effects. The Effect Ratio (ER) and Delta Relative Improvement (DRI) require a baseline and an advanced retrieval method. Since the measures evaluate relative effects, they can also be used for replicated experiments with a test collection that contains possibly different topics and documents (cf. Subsection 4.2.4). Finally, at the most general level, we

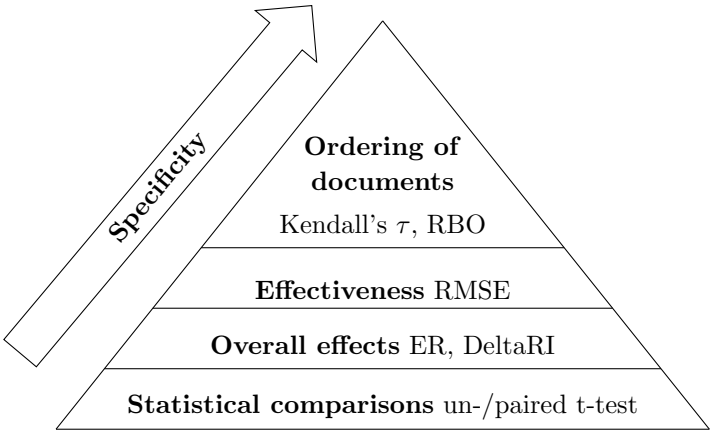

Figure 4.2: Reproducibility measures arranged regarding their level of specificity.

can compare the topic score distributions of the original and reproduced/replicated runs by (un-)paired t-tests. In this case, low p-values indicate a failed reproduction or replication (cf. Subsection 4.2.5). Sometimes, the reproducibility of an IR experiment involving multiple systems can be assessed by the relative ordering of two or more retrieval systems regarding their effectiveness. These *system rankings* (sometimes also referred to as *ranking of systems*) from the original and reproduced experiments are usually compared with the help of Kendall's τ (cf. Subsection 4.2.6).

4.2.1 Bitwise Reproducibility

In the computational sciences, researchers and experimenters are privileged to work with most digital entities, which come with certain merits and benefits. For instance, Potthast et al. [329] emphasize that the digital experimental setup can be made available for future reuse with little costs and without much overhead. Furthermore, it is easy to check if two digital entities, i.e., experimental artifacts, are identical.

At the most rigorous level, it is possible to verify a successful reproduction bit-by-bit. The verification of a bitwise reproduction can be easily implemented by determining hashes or checksums, for instance, by MD5 or SHA-based methods [199]. These kinds of validations also allow the automation of reproducibility checks through regression or unit tests like they are done according to the principles of test-driven software development. Considering the typical output of an IR experiment, a so-called run file, the premises for evaluating bitwise reproductions are fairly good as a commonly used data format provides a solid basis. Figure 4.3 illustrates a single line of the TREC run file format, which describes rankings for particular queries and is defined by six columns containing a topic number (`<qid>`), a wildcard entry (`<Q0>`), the document identifier (`<docid>`), the ranking position (`<rank>`), the ranking score (`<score>`), and a file-specific tag (`<tag>`) [413].

```
<qid> <Q0> <docid> <rank> <score> <tag>
```

Figure 4.3: TREC format

An identical relative ranking of documents could be reproduced with different scores that differ by their ranges or intervals. Determining and comparing checksums of the hashed run file would indicate a failed reproduction attempt. However, this level of rigor may be too strict. Depending on the use case, e.g., in a user experiment where the users do not care about document scores, the document ranking with different scores could be used as a completely valid reproduction of the system's outputs. In this regard, the requirement for "perfect" reproducibility in an IR experiment could be met by results without bitwise equality.

Drummond [115] goes as far as to question the importance of perfect reproduction as it can support internal validity at most but does not ensure any directions towards replicability or generalizability. Ivie and Thain [199] separate automatic verifications of the bits and data from the validation of reproduced statistics and phenomena that require domain-specific statistical tools and human interpretations. In this regard, the following measures fall into the category of the latter by proposing solutions for quantifying reproducibility beyond the very strict assessment of a bitwise identity.

4.2.2 Document Rankings

Kendall's τ [221] is a rank correlation coefficient that can be used to measure the similarity between two ranking lists with the same set of ranked items. For the j-th topic of a test collection, it is determined as:

$$\tau_j(r, r') = \frac{P - Q}{\sqrt{\big(P + Q + U\big)\big(P + Q + V\big)}}. \tag{4.1}$$

P is the total number of concordant pairs, Q the total number of discordant pairs, whereas concordant and discordant refer to item pairs ranked in the same or different order, respectively; and U and V are the numbers of ties in r and r', respectively. When comparing two document rankings, both must be permutations of the same set of documents. It ranges between -1 and 1, whereas a value of 1 corresponds to a perfect reproduction, 0 indicates no correlation, and -1 indicates a perfect inverse correlation between the rankings.

As pointed out by Ferro et al. [133,136] as part of CENTRE, it is very challenging to reimplement a perfectly reproduced document ranking with exactly the same set of documents. Thus, it may be too strict to determine Kendall's τ based on the rankings of documents. Instead, it can be determined in a slightly modified way by comparing lists of ranking positions made from a union set of both rankings.

Making lists with rank orders from a union set without duplicates makes it possible to determine the correlation of the relative ordering between two rankings, even with deviating sets of documents in both rankings. For a better illustration consider the following two rankings $r = [d_1, d_2, d_3, d_4, d_5]$ and $r' = [d_1, d_2, d_3, d_4, d_6]$, which result in a unified set of ranking positions as $[1, 2, 3, 4, 5, 6]$, whereas the corresponding single rank position lists are $r_p = [1, 2, 3, 4, 5]$ and $r'_p = [1, 2, 3, 4, 6]$ and $\tau_j(r_p, r'_p) = 1$. In this case, the second document ranking reproduced the first one with a different document at the last rank that is not in the first ranking or possibly at a lower rank than the cut-off. Consider another reproduced document ranking $r'' = [d_2, d_5, d_7, d_6, d_4]$, which has a partial overlap of documents with r but has a different order of those documents also contained in r. In that case, the union set of rank positions is $[1, 2, 3, 4, 5, 6, 7]$, and the reproduction yields $\tau_j(r_p, r''_p) = 0.2$. This

measure is referred to as Kendall's τ Union (KTU), and it is averaged over all topics n_D in a test collection D as follows:

$$\bar{\tau}(r, r') = \frac{1}{n_D} \sum_{j=1}^{n_D} \tau_j(r_p, r'_p). \tag{4.2}$$

An alternative rank correlation measure was introduced by Webber et al. [419] as the RBO. It can be used to compare ranking lists of infinite lengths with partial overlaps. It is based on a simple probabilistic user model and is defined as follows for the j-th topic:

$$\mathrm{RBO}_j(r, r') = (1 - \phi) \sum_{i=1}^{\infty} \phi^{i-1} \cdot A_i. \tag{4.3}$$

RBO is the weighted sum of the overlap A_i over increasing rank positions denoted by i. ϕ parameterizes the top-heaviness and, hereby, models the underlying user behavior — the higher ϕ, the higher the continuation probability and, thus, the user's persistence. Note that $1 - \phi$ models the probability of a stopping decision. RBO ranges between 0 and 1 and higher values indicate stronger correlations. Like KTU, it is averaged over all topics n_D in a test collection D as follows:

$$\overline{\mathrm{RBO}}(r, r') = \frac{1}{n_D} \sum_{j=1}^{n_D} \mathrm{RBO}_j(r, r'). \tag{4.4}$$

4.2.3 System Effectiveness

The reproduction quality of the system effectiveness is determined by comparing the topic score distributions of r and r' resulting from an IR evaluation measure M^D, e.g., P@10, nDCG, or AP, for (usually 50) different queries of a test collection. As part of the CENTRE workshop [133], the RMSE was proposed as a means to measure the closeness between the distributions represented by the vectors $M^D(r)$ and $M^D(r')$ both of length n_D:

$$\mathrm{RMSE}\left(M^D(r), M^D(r')\right) = \sqrt{\frac{1}{n_D} \sum_{j=1}^{n_D} \left(M_j^D(r) - M_j^D(r')\right)^2}. \tag{4.5}$$

Due to the squared difference between $M^D(r)$ and $M^D(r')$, the RMSE penalizes larger errors more severely. Compared to the similarity analysis of document rankings, the system effectiveness is evaluated at a more abstract level, i.e., with lower specificity. The system-oriented IR evaluation measures are usually determined by the relevance labels, meaning that equal scores can also be reproduced with different documents with the same relevance labels as those in the original ranking. Consequently, it is possible to have an RMSE of 0, while there are deviations between the document rankings.

4.2.4 Overall Effects

We note that both the comparison of the document rankings and the system effectiveness can only be determined for the reproduced experiment as they require

rankings derived from the same corpus and for the same topics. In contrast, reimplementations at the level of the overall effects can be determined for both reproducibility and replicability. This is made possible by evaluating the reproduction and replication of relative effects between the system effectiveness of a baseline run b and an outperforming method resulting in an advanced run a that were both used in the original experimental setup. At NTCIR-CENTRE, Sakai et al. [353] introduced the Effect Ratio (ER), for which the *per-topic-improvements* need to be determined and are defined for the j-th topic as:

$$\Delta M_j^D = M_j^D(a) - M_j^D(b) \ , \quad \Delta' M_j^D = M_j^D(a') - M_j^D(b') \tag{4.6}$$

where $\Delta' M_j^D$ denotes the per-topic-improvement for topic j in the reproduced experiment. In contrast, another test collection D' is used in the replicated experiment, and the corresponding per-topic-improvement is denoted as $\Delta' M_j^{D'}$. The ER is determined by the ratio between both per-topic improvements averaged over the topics of the test collections. For the reproduced experiment, it is defined as:

$$\mathrm{ER}\left(\Delta' M^D, \Delta M^D\right) = \frac{\overline{\Delta' M^D}}{\overline{\Delta M^D}} = \frac{\frac{1}{n_D}\sum_{j=1}^{n_D} \Delta' M_j^D}{\frac{1}{n_D}\sum_{j=1}^{n_D} \Delta M_j^D}. \tag{4.7}$$

We note that in a replicated experiment, it is unnecessary to have the same topics in D and D' as it depends on the ratio between the relative improvements of the outperforming method over the baseline — a perfect reproduction or replication results in an ER score of 1. Lower scores indicate a weaker improvement over the baseline. Vice versa, higher values indicate a greater improvement over the baseline than in the original experiment. It has to be pointed out that it can become critical to optimize a reimplementation with regard to the ER as it evaluates the relative improvements. ER does not take the reproduction of absolute scores into account, and it is theoretically feasible to optimize a reimplementation for a perfect score of 1 with possibly lower absolute scores of the baseline and the outperforming methods as long as the relative performance gains are the same as in the original experiment. As a solution, the difference between the baseline b and the advanced run a can be normalized by the baseline b. For the reproduced experiment, the Relative Improvement (RI) is defined as:

$$\mathrm{RI} = \frac{\overline{M^D(a)} - \overline{M^D(b)}}{\overline{M^D(b)}}, \qquad \mathrm{RI}' = \frac{\overline{M^D(a')} - \overline{M^D(b')}}{\overline{M^D(b')}}. \tag{4.8}$$

Finally, the Delta Relative Improvement (DRI) is determined by the difference between the RI of the original and reproduced/replicated experiment:

$$\mathrm{DRI}(\mathrm{RI}, \mathrm{RI}') = \mathrm{RI} - \mathrm{RI}'. \tag{4.9}$$

A perfect reproduction or replication yields a DRI score of 0 as there should not be any difference between the original and reimplemented RI. Negative DRI scores result from higher absolute scores of the reimplemented experiment, which could be considered a partial success since the reimplementations outperform the original results despite deviations. On the other hand, positive DRI scores indicate less effective reimplementations, which should rather be seen as an indication of a failed reproduction or replication.

4.2.5 Statistical Properties

At the most general level, the topic score distributions of the original run r and r' are compared by two-tailed t-tests. In this case, the general idea is to gain insights into the success of a reimplementation from the p-values. These comparisons are based on the assumption that a smaller p-value gives stronger evidence that the reimplementation has failed, while larger p-values should result from better reimplementations. In the reproduced experiment, a two-tailed paired t-test is preferred since the same test collection is used, i.e., the results are drawn from the same distribution as in the original experiment. If the experiment is replicated with another test collection, i.e., the runs are made from a different distribution and an unpaired t-test should be used. These statistical properties add another abstraction layer to the reproducibility levels since the low p-values indicate a deviation between the topic scores but do not show if the reimplementation performs better or worse.

4.2.6 System Rankings

According to Ferro [130], one of the major challenges of reproducible IR is the validation of meta-evaluation experiments like they are conducted as part of shared task efforts. Besides evaluating the reproducibility of the single system runs, it is of interest to evaluate the reproducibility of the relative system performance of all systems participating in a shared task. Similar to the validation of document rankings, Kendall's τ can be used to measure the correlation between system rankings as proposed by Voorhees [409]. In this context, she considers correlations above 0.9 as acceptable. Conventionally, system rankings are evaluated by leave-out-unique tests and Kendall's τ [386,412] in order to validate the reusability of a test collection. The general idea is to simulate the evaluation of a new system that did not participate in the pooling process by removing its contribution of unique documents from the pool. The system rankings are determined before and after excluding unique documents contributed by a single system and are finally evaluated by Kendall's τ. If the system ranking does not considerably change, i.e., $\tau > 0.9$, the test collection can be considered reusable. Consequently, the test collection qualifies as an evaluation tool to analyze how well a particular retrieval method or system rankings with multiple systems can be generalized with different data.

4.3 Software Toolkit

All of the measures depicted in Figure 4.2 and described in Subsections 4.2.2 to 4.2.5 have been implemented in an open-source software toolkit titled `repro_eval` [61]. It is a Python package that is integrated into the GitHub ecosystem [470], including automated unit tests and distribution through the Python Package Index.

It builds upon other open-source software packages like `numpy` [404], `scipy` [408], and `Pytrec_eval` [166] providing Python bindings to the commonly used evaluation toolkit `trec_eval`. Once downloaded and installed, it can be used with command line calls or via the API. The interface design is aligned to the two experimental types, reproducibility and replicability. Assuming that the reference run, the reimplemented files, as well as the corresponding relevance judgments are available, the `Evaluator` classes can determine the reproducibility and replicability measures.

The metadata schema introduced in the previous Chapter 3 is supported by `repro_eval==0.4.0` in different ways. On the one hand, we have implemented a `MetadataHandler` that reads the metadata from annotated run files and semi-automatically annotates run files if provided with a minimal set of the required information. On the other hand, we implemented the analysis of annotated run files by the `MetadataAnalyzer` and the `PrimadExperiment` that analyze the metadata information and align the reproducibility evaluations to PRIMAD. Depending on the deviating PRIMAD components, different reproducibility measures are part of the evaluations, and the `MetadataAnalyzer` identifies reasonable evaluations.

4.3.1 Automatic Annotations

In order to lower the manual annotation effort, the `MetadataHandler` can be used to annotate runs. Given a run file, it automatically compiles the available information and appends it to the metadata header of a run file. Figure 4.4 exemplifies how the metadata can be added to the run file in Python. When writing metadata to run files, the `MetadataHandler` fetches information regarding the platform. Likewise, if not specified otherwise, it assumes the run file to be in the root directory of a Git repository from which some of the information regarding the implementation can be extracted. Currently, it is possible to fetch information about the hardware, including the CPU model, the size of the random-access memory, information about the operating system and kernel, including the UNIX distribution and the version, respectively, and finally, information about the software project, including the location of the Git repository, the current commit, and the programming language determined by the source code files.

Nevertheless, it is impossible to determine the required information about some PRIMAD components automatically. Therefore, the `MetadataHandler` has to be provided with a template YAML file, in which the corresponding metadata should be added manually. Some information, specifically about the research goal, the method, or the actor, cannot be retrieved automatically and must be added by hand. Even though the entire metadata cannot be extracted automatically, the `MetadataHandler` reduces the manual annotation effort and avoids possible errors, contributing to a community-wide adoption.

```
1 from repro_eval.metadata import MetadataHandler
2
3 run_path ='./run.txt',
4 metadata_path ='./metadata.yaml'
5 metadata_handler = MetadataHandler(run_path, metadata_path)
6 metadata_handler.write_metadata()
```

Figure 4.4: Annotating runs with the `MetadataHandler`.

4.3.2 Analysis of Annotations

Given two or more run files with metadata annotations, the `MetadataAnalyzer` identifies similar PRIMAD components in the metadata and proposes several reasonable reproducibility evaluations. The code snippet in Figure 4.5 illustrates how

the `MetadataAnalyzer` can be used to scan an entire directory with annotated run files and automatically validates the type of reproducibility experiments. After it has been initialized with a reference run, the metadata of all annotated runs in the directory is compared to that of the reference metadata, and as a result, a list containing PRIMAD experiments and the corresponding run candidates is returned.

In our implementations, we distinguish the different experiment types by lower- and uppercase letters, e.g., parameter sweeps would result in ‘‘`priMad`’’ with the uppercase letter `M` that signifies the changes of the method. Provided with the experiment type and the reproduced runs, the `PrimadExperiment` evaluates the experiments and the corresponding runs with the help of the `repro_eval` measures. The reproducibility toolkit `repro_eval` follows the naming conventions introduced by the ACM Policy on Artifact Review and Badging [448], i.e., in the software library can be used to evaluate the reproducibility with the same test collection and the replicability with another test collection than in the original experiments.

Specific reproducibility measures can be determined depending on what kind of test collection is used to evaluate the reimplementations. For instance, if another collection than in the original experiment is used, only some of the measures can be determined. Consequently, the evaluations depend on the type of PRIMAD experiment from which the reproduced runs originate. Suppose the test collection is the same as in the original experiment. In that case, all reproducibility measures can be determined, e.g., as is the case of reproduced runs based on parameter sweeps.

```
from repro_eval.metadata import MetadataAnalyzer, PrimadExperiment

run_path ='./run.txt'
dir_path ='./runs/'
metadata_analyzer = MetadataAnalyzer(run_path)
experiments =  metadata_analyzer.analyze_directory(dir_path)
run_candidates = experiments.get('priMad')
primad_experiment = PrimadExperiment(primad='priMad',
                        rep_base=run_candidates,,...)
primad_experiment.evaluate()
```

Figure 4.5: Analyzing runs with the `MetadataAnalyzer`.

4.4 Conclusion

This chapter outlined the general procedure of a reactive reproducibility attempt and how the differences between the original results and those of reimplementations can be quantified. The corresponding measures can be used to quantify the degree of reproducibility at different levels of specificity, i.e., some measures are more sensitive to changes in the reimplemented rankings than others at more general levels. The measures can be used to compare different reimplemented run files and determine which is closer to the original reference. Additionally, we contributed the evaluation toolkit `repro_eval` as open-source software and reusable artifact to the community. The toolkit supports the metadata schema outlined in the previous chapter and can be extended by other reproducibility measures in the future.

Chapter 5

Reproducibility Evaluations

This chapter is about how reactive reproducibility studies of system-oriented IR experiments can be conducted in a principled way. The introduced approach builds upon the metadata annotation schema (cf. Chapter 3), the reproducibility measures, and the corresponding software toolkit `repro_eval` (cf. Chapter 4). The overall scenario follows that of reactive reproducibility attempts, where no software artifact of the original experiment is available but the system outputs, i.e., the TREC run files. For this purpose, we reimplemented the CCRF method by Grossman and Cormack (GC) based on the descriptions in the corresponding TREC notebooks [159, 160]. Our reimplementations and reproducibility protocols were originally submitted to the CENTRE workshop [66] and also as a dockerized version to the OSIRRC workshop [65]. These contributions were later on analyzed in a contribution to SIGIR [60]. In another later submission, we analyzed the web search-enhanced CCRF and contributed the results to CLEF [64].

The reimplementations are used to compile a dataset of runs with multiple reproduction candidates of the original run files, which, as a whole, simulate the results by a group of reproducers who try to reimplement the original experiments by trying out different configurations and parameterizations of the retrieval method. To make the dataset more diverse, i.e., to include results from another group of researchers, we also include regression experiments by Yu et al. (YXL) in the dataset.

Before introducing the principled evaluations based on the metadata annotations, we analyze our reimplementations as part of preliminary evaluations. In the corresponding section, we have a detailed look at the reimplementations of GC's submission to TREC Common Core 2017 and illustrate how comparisons of averaged retrieval measures can "hide" differences between the actual document rankings or topic score distributions of the reproduced and original runs. In addition, we look at the reproductions and replications of the web content-enhanced experiments that were originally contributed to TREC Common Core 2018 by GC. First, we analyze the robustness of the general workflow, and afterward, we have a closer look at the effect of the query formulation and the influence of the underlying web search engine.

Finally, we analyze the runs in the dataset by principled reproducibility evaluations. Based on the metadata annotations, the differences between the experiments from which the annotated run files originate can be expressed in terms of PRIMAD. The evaluations outline three selected variations regarding the original experiments and conclude with what can be learned about the CCRF method from the reproducibility experiments. In sum, our contributions are as follows:

C5 Reimplemenations of CCRF and, related to that, a curated dataset of reimplemented and annotated run files (cf. [63, 64, 65, 66])

C6 Principled reproducibility analysis of CCRF reimplementations based on how they relate to the original experiment in terms of PRIMAD (cf. [63]).

The remainder is structured as follows. First, we describe the target study of the reimplementations, which is based on CCRF. Afterward, we analyze the quality of our reimplementations that are also included in the dataset in a preliminary evaluation. Finally, in the experimental evaluations, we analyze the reproducibility and replicability at different levels as outlined in Chapter 4. In the end, we conclude by outlining the limitations of the entirely system-oriented focus of this chapter, which motivates the contributions of the following chapters.

5.1 Cross-Collection Relevance Feedback

The CCRF method recently gained interest in the IR community, especially as part of TREC Common Core, where existing topics were reused for building a new test collection [4]. The general workflow is illustrated in Figure 5.1. Dating back to 2017, GC's approach [159] inspired several follow-up studies. Reasons for the increased interest in this retrieval method are its simplicity and effectiveness, being the most effective automatic submission as part of TREC Common Core 2017 [4].

YXL [437] reproduced the approach by embedding it into a multi-stage ranking pipeline and documenting it in the Anserini toolkit. We (BFFMSSS [66]) reimplemented the approach as part of a dedicated reproducibility analysis. We used the reimplemented runs to simulate a researcher trying to reproduce the relevance transfer method [60]. As part of the TREC Common Core reiteration in 2018, YXL [436] reused the same method with another dataset, whereas GC [160] themselves also applied a modified version of the method in TREC Common Core 2018 that was also revalidated by us (BPS [64]).

The general workflow of CCRF is illustrated in Figure 5.1. The underlying retrieval method follows a point-wise learning-to-rank approach where each document is assigned a probability of being relevant [258]. CCRF is only possible if there is an overlap of topics in the target and source collections. For each topic, a relevance classifier is trained with the help of the relevance labels (`qrels`) and tf-idf features of relevant and non-relevant documents derived from a term-document matrix based on the source collection's vocabulary. The documents of the target collection are represented as tf-idf features that are also derived from the source collection's term-document matrix. The topic-specific relevance classifier assigns a relevance probability to each tf-idf feature of the target collection's documents that are ordered by decreasing probabilities in the final ranking.

GC introduced the outlined approach at TREC Common Core 2017 [159] either using the TREC Disks 4 & 5 (denoted as Robust04) or a combination of both Robust04 and the AQUAINT test collection (denoted as Robust05) as the source collection(s) to rank documents of the Annotated New York Times Corpus (denoted as Core17). Even though the approach is straightforward, it was the most effective automatic run submission at TREC Common Core 2017, ranking third behind two manual runs that were slightly more effective. Depending on the source corpora combination, the runs are either referred to as `WCrobust04` or `WCrobust0405`.

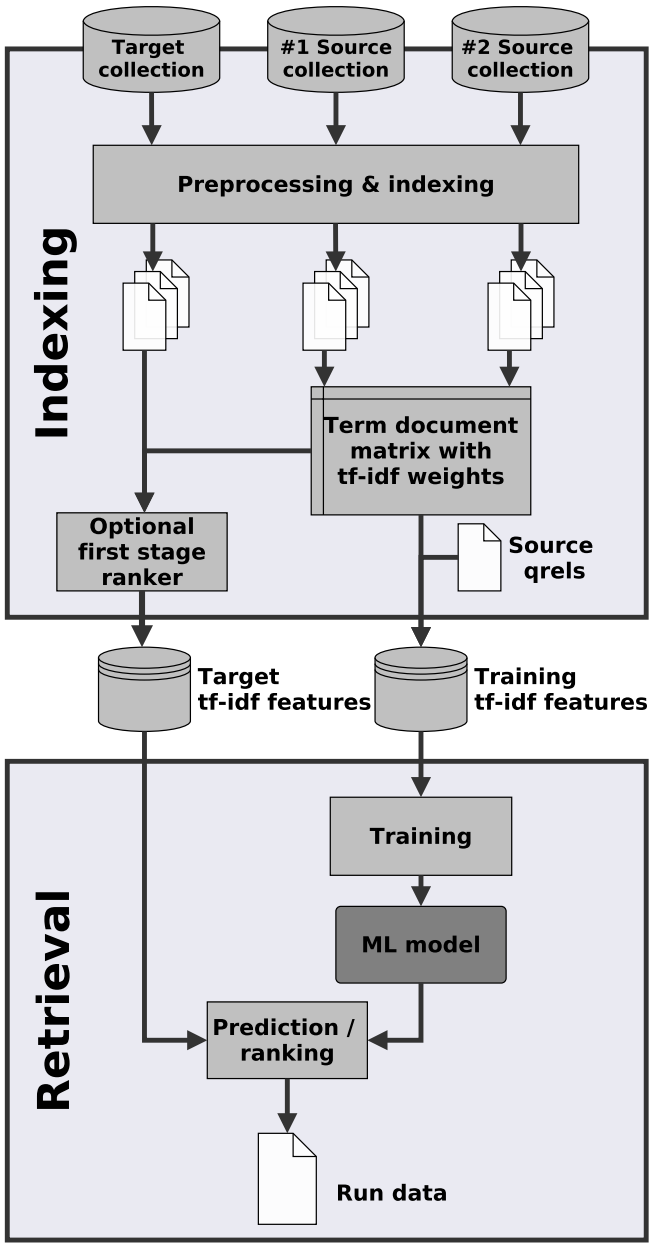

Figure 5.1: Cross-Collection Relevance Feedback [159].

YXL [437] reproduced the approach by introducing a first-stage ranker with a keyword-based method. Instead of ranking each document of the entire target collection with the topic-specific classifier, only a list of the first 10,000 documents retrieved by the keyword-based method is reranked. We reproduced the workflow as accurately as possible based on the description in the TREC notebook by GC [66].

As part of TREC Common Core 2018, YXL reused their reimplementation and submitted runs derived from the Washington Post Corpus v2 (denoted as Core18). GC also submitted the runs `uwmrg` and `uwmrgx` to TREC Common Core 2018 with a slightly modified workflow. Instead of using TREC test collections as the source collections, they scraped results from SERPs to train topic-specific relevance classifiers. This approach is robust, as shown in our regression experiments [64].

5.2 Reimplementation Details

In the following, we outline the details of our reimplementations. At first, we describe how the runs `WCrobust04` and `WCrobust0405` were reimplemented. Afterward, we describe how the reimplementations of the `uwmrg` and `uwmrgx` runs were made. As both methods share a fair amount of the same principles, the second reimplementation mainly builds upon the former, and we outline the adaptions.

5.2.1 `WCrobust04` and `WCrobust0405` Runs

Our reimplementation is based on the Python programming language. As the landscape of available Python packages offers a wide variety of different open and free software libraries, we had no problems finding the required software tools to reimplement the workflow. Besides the Python packages described below, we used the GNU tools `tar` and `gzip` to extract the document files out of the compressed data archives. For the markup removal and parsing of the raw text from the formatted document files, we relied on the `BeautifulSoup` package in combination with `lxml`. The raw text was normalized by excluding the punctuation, stop word removal, and stemming, which are implemented in the respective order with the `nltk` package.

According to the protocol by GC [159], the vocabularies of the source and target corpora have to be merged to determine the term-document matrix based on a unified vocabulary. This results in training samples that are augmented by the vocabulary of the corpus to be ranked. However, our reimplementations deviate from this approach. The term-document matrix, and consequently also the tf-idf weights of the training samples, are derived solely based on the source corpora. In their reproducibility study, YXL consider this augmentation step to be insignificant. In our experimental evaluation [66], we compared the resulting runs of augmented and non-augmented training samples and could confirm these assumptions. It is feasible to derive the tf-idf samples without merging the vocabularies of the source, and target corpora, as the differences in retrieval effectiveness are negligible.

Our implementation of the ML classifier builds upon the scikit-learn package [321]. More specifically, we make use of the `TfidfVectorizer` and the `LogisticRegression` classifier. YXL [437] pay special attention to the importance of the L2-normalization (of the tf-idf feature vectors). Even though the original report does not address this aspect, we had no issues concerning the normalization as the `TfidfVectorizer` uses the L2-norm as a default setting. Another detail about the tf-idf features, which caused performance drops in our initial reimplementations, was specifically related to the term frequency. As the original report remains unclear about how the term frequency is determined, we used the default settings of the `TfidfVectorizer`, which simply includes the raw term frequency *tf*. However, we achieved better performance scores and also a better reproduction when including the term frequency as $1 + log(tf)$.

Depending on the combinations of the source and target corpora, there are different overlaps between the topics in the corpora. For instance, Robust04 and Core17 have an overlap of 50 topics, i.e., all of the topics of Core17 are also judged for Robust04, whereas between Robust05 and Core17, only 33 topics overlap. This led to initial confusion on our side as the original protocol remained unclear about how to train the classifier when both Robust04 and Robust05 are used as source collec-

Table 5.1: Overview of the run dataset, including the original runs by Grossman and Cormack (GC), the reimplementations by Yu et al. (YXL), and our reimplementations as part of CENTRE and SIGIR (BFFMSSS), as well as CLEF (BPS).

Researchers	Method	Target collections	Runs
GC [159]	GC [159]	Core17	2
YXL [436, 437]		Robust04/05, Core17/18	327
BFFMSSS [60, 66]		Core17	100
GC [160]	GC [160]	Core18	2
BPS [64]		Robust04/05, Core17/18	32

tions. However, after clarification with the authors (GC) via mail correspondence, we learned that the tf-idf features are derived from both corpora were feasible and otherwise only from one source corpus, which means that for some topics of `WCrobust0405` only Robust04 was used for generating training samples.

The training features were stored in the SVMlight format to ensure compatibility with other ML frameworks. For each topic, a ranking with 10,000 entries was determined and written into a run file. The first implementation that was contributed to the CENTRE workshop is available on Bitbucket [453]. This version was also submitted to the OSIRRC workshop as a dockerized and easier-to-reproduce version [65]. As explained earlier, the workshop's organizers developed a software toolkit that allows for easy integration of custom ad-hoc retrieval pipelines in Docker containers. In cooperation with the community, these efforts resulted in a library of different Docker images that can be rerun on purpose [93]. An updated version of the reimplementations that fixes the issues outlined above is available on GitHub [467].

5.2.2 `uwmrg` and `uwmrgx` Runs

Our reimplementation of GC's submission to TREC Common Core 2018 mainly builds upon the source code described above. The main difference between these runs and those submitted to TREC Common Core 2017 is the composition of the training data. Each classifier is based on training data retrieved from texts of scraped SERPs, which, in turn, depend on the query of the related topic. In order to derive the first run variant `uwmrg`, the entire content of web pages corresponding to the URLs of the SERP was scraped, whereas the second run `uwmrgx` used only the snippets from SERPs instead of scraping complete web pages. During training, the class assignments of positive and negative features were based on a one-vs-rest principle. Depending on the topic, positive samples were retrieved with the corresponding `title` (and `description`), while scraped results of other topics served as negative samples. More details are provided in the publication [64] and in the corresponding GitHub repository [464].

5.3 Annotated Run Dataset

To demonstrate the potential of the reproducibility measures in combination with the metadata schema, we demonstrate the applicability by annotating and evaluating a run file dataset based on CCRF. Table 5.1 provides an overview of the run

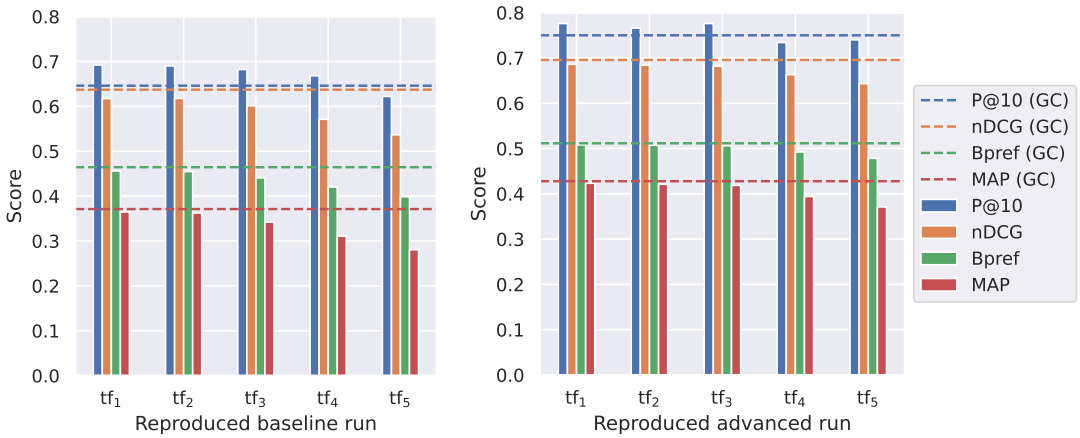

Figure 5.2: Average retrieval performance of the reproduced baseline run `WCrobust04` (left) and the advanced run `WCrobust0405` (right) with different tf-based threshold values (tf_1-tf_5) of the tf-idf features. The dashed lines correspond to the effectiveness of the original experiment by Grossman and Cormack (GC).

files and their underlying combinations (of retrieval methods and test collections) included in our dataset. While some of the runs were available from existing data archives, i.e., those runs submitted by GC to TREC Common Core in 2017/18, others were derived by us with Anserini's runbook [463] that belongs to the implementations of the reproducibility analysis by YXL [437]. More specifically, we used Java `v8`, Lucene `v7.6`, and Anserini `v0.3.0` at commit `9548cd6b`, which were also reported in the corresponding paper, to rerun the instructions of the runbook successfully on all four test collections. All of the runs were annotated by us as far as the respective information was publicly available. The annotated run data is hosted in an open-access data archive on Zenodo [489]. The corresponding metadata information is also provided in separate YAML files to demonstrate how run files can be annotated with the help of `repro_eval`.

5.4 Preliminary Reproducibility Evaluations

Before outlining how the metadata annotations can be used to conduct systematic reproducibility evaluations in a principled way, we focus on the quality of our reimplementations in this section. At first, we analyze the reproducibility of the `WCrobust04` and `WCrobust0405` runs based on the Core17 test collection. Afterward, we analyze some particular aspects of the reimplemented `uwmrg` and `uwmrgx` runs based on the Core18 test collection regarding the robustness and how the overall effect measures can be used as a visual analytics tool. Overall, we conclude from this analysis that our reimplementations are successful in terms of reproducing and replicating the retrieval effectiveness, i.e., the ARP over the 50 topics of the test collections. However, as some of the evaluations show, it is harder to reproduce the original results with a higher degree of rigor. For instance, the reimplementations mostly fail to reproduce the exact order of documents in the rankings, which can be critical in precision-oriented tasks, for instance, as part of experiments with real users. These outcomes show that depending on the use case, different levels of rigor regarding reproduction quality have to be considered.

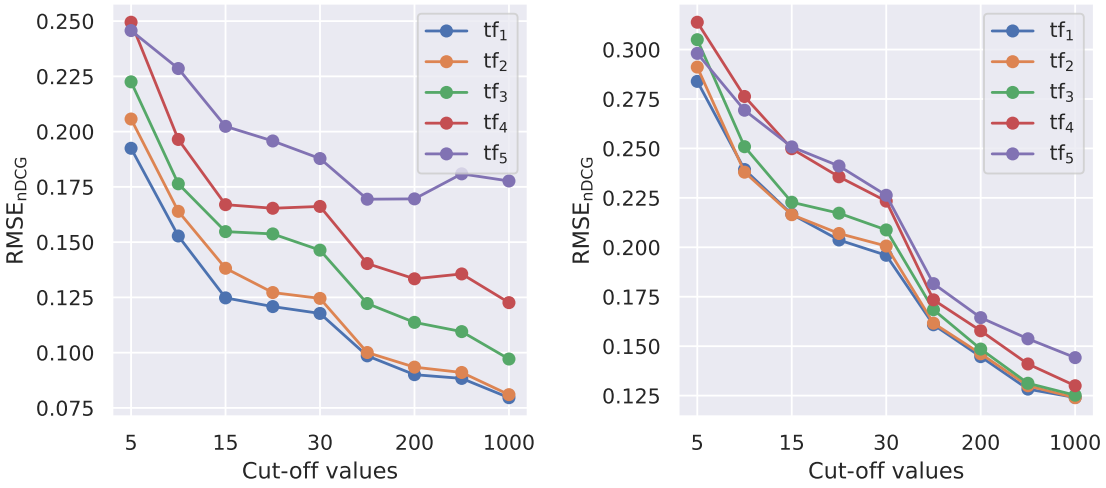

Figure 5.3: RMSE instantiated with nDCG between our reimplementations and the original runs. The error at different cut-off levels is shown for the reproduced baseline run `WCrobust04` (left) and the advanced run `WCrobust0405` (right) with different tf-based threshold values tf_1-tf_5.

5.4.1 Retrieval Effectiveness

From the literature review of the ECIR reproducibility track, we already concluded that it is common practice to compare reimplementations to the original results by the retrieval effectiveness, i.e., the performance expressed by a retrieval measure averaged over the topics of a test collection, to which we refer as the ARP in the following. Figure 5.2 compares the ARP of different reproduction candidates to the original results by four retrieval measures, including P@10, nDCG, Bpref, and AP. The bar plots correspond to the ARP scores of reimplementations with different tf-idf features in the training data, whereas the dashed line corresponds to the ARP scores from the original experiment. More specifically, we artificially shrink the size of the vocabulary by capping it with threshold values based on tf-weights. As we build up on the scikit-learn implementation of the tf-idf features [321], we use the `max_features` parameter that considers the top tf-idf features ordered by the term frequency across the corpus. We lower the parameter `max_features` from tf_1 to tf_5, i.e., decreasing `max_features` leads to fewer tf-idf features being considered for building the vocabulary.

First of all, we see that for both the baseline (`WCrobust04` on the left) and the advanced run (`WCrobust0405` on the right), the retrieval performance could be reproduced fairly well. While most of the reproduced ARP scores come close to the originals, the P@10 scores even outperform the original scores for some topics. Second, we see a drop in the performance as the vocabulary size shrinks, which allows us to modify the retrieval performance in a principled way. Overall, we consider our reimplementations to be a good basis for any further analysis.

In the following analysis, we have a look at how the effectiveness of the topic score distributions could be reproduced in terms of the RMSE measure (cf. Equation 4.5). Figure 5.3 shows the RMSE between the reimplemented and the original topic score distributions for the different tf-variants that were already compared in Figure 5.2. In this case, the RMSE is instantiated with nDCG and plotted over different cut-

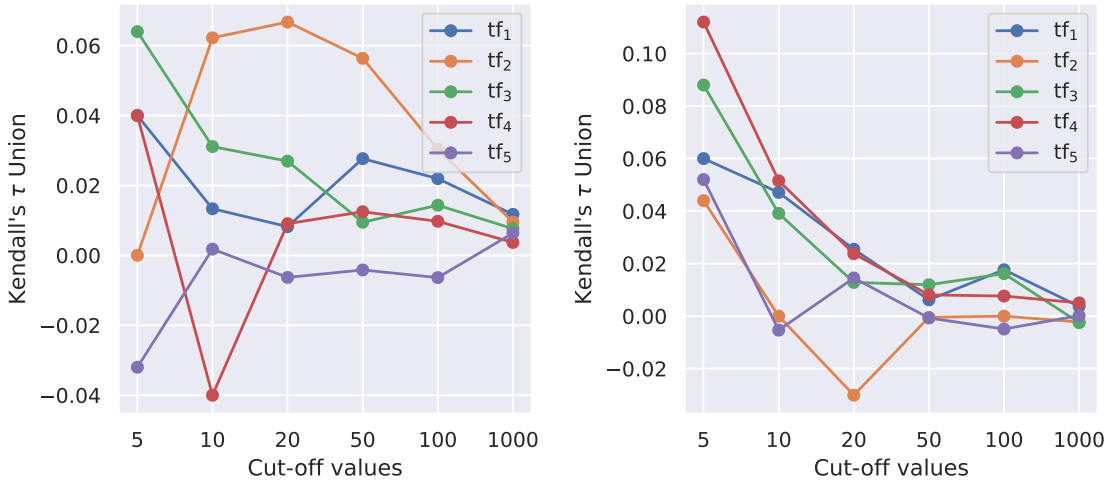

Figure 5.4: Kendall's τ Union of our reimplementations with regard to the document rankings of the original runs. The rank correlation at different cut-off levels is shown for the reproduced baseline run `WCrobust04` (left) and the advanced run `WCrobust0405` (right) with different tf-based threshold values tf_1-tf_5.

off levels. We see similar effects for both types of runs and draw the following conclusions. First, we see a higher error as the retrieval performance decreases. As expected, the RMSE increases with a larger "distance" between the topic scores related to the decrease of the retrieval performance. Second, we see that the RMSE decreases with higher cut-off levels, which can be explained by larger numbers of relevant documents in the ranking as more documents are considered. It means that low RMSE can also be achieved with different documents that comply with the relevance labels in the original ranking. Thus, the RMSE can be considered as a document-independent reproducibility measure. A comparison of the actual document rankings is presented in the next part.

5.4.2 Document Rankings

As outlined in Chapter 4, the most rigorous and (also reasonable) level for evaluating the reproducibility of IR experiments is the comparison of the actual document rankings. For this purpose, two rank correlation measures were introduced (cf. Equation 4.1 and 4.3). Figure 5.4 evaluates KTU over different cut-off values. These experiments show no correlation between the original and reimplemented rankings, neither for the baseline nor for the advanced run. Overall, the KTU tends to converge towards correlation scores of 0.0. Kendall's τ does not account for the rank-position of the overlaps in the rankings of the original and reproduced run, i.e., it is not top-heavy like RBO. Instead, it accounts for similarities or differences equally over all rank positions. As the number of compared documents in the rankings increases, there are more dissimilarities between the original and reimplemented rankings. Thus, we conclude that our reimplementations are quite dissimilar from the original results when comparing them by the document rankings.

Similar but slightly better results can be seen for the evaluations based on the RBO in Figure 5.5. The RBO is a "more forgiving" or less strict measure due to the parameterization of the underlying user model. In our experiments, we used

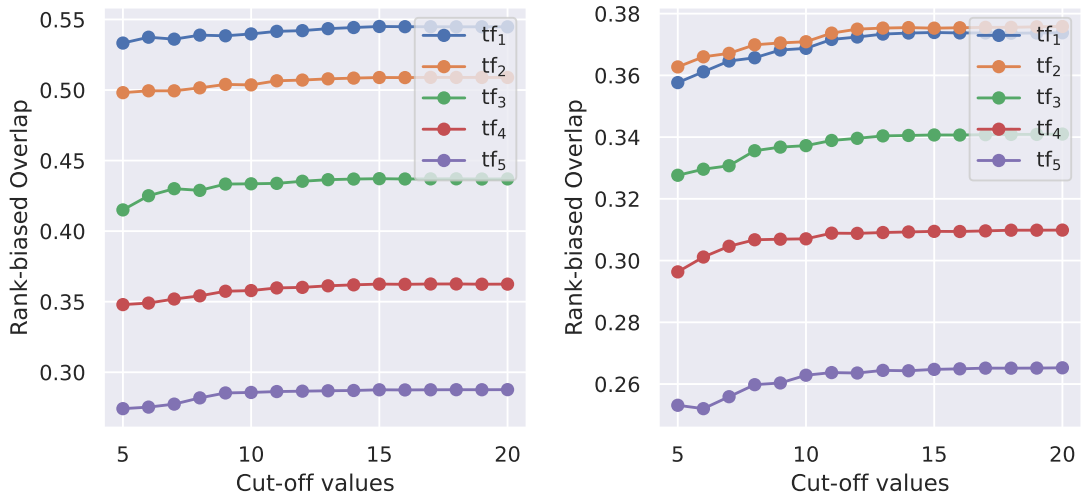

Figure 5.5: Rank-biased Overlap of our reimplementations with regard to the document rankings of the original runs. The rank correlation at different cut-off levels is shown for the reproduced baseline run `WCrobust04` (left) and the advanced run `WCrobust0405` (right) with different tf-based threshold values tf_1-tf_5.

$p = 0.8$, which puts more weight on the top-ranked document results. For this reason, Figure 5.5 only shows the scores of rankings with up to 20 documents as the RBO for higher cut-off levels does not change. For the baseline run, there is consistency between the RBO and the ARP scores. More effective runs also have a higher correlation in terms of the RBO. Likewise, most of the RBO scores of the advanced runs agree with the relative ordering according to the retrieval performance. In comparison, the RBO scores of the baseline are also higher than those of the advanced run.

5.4.3 Robustness of Web Search-Enhanced Classifiers

In the following, we focus on our reimplementations of GC's web content-enhanced run submissions to TREC Common Core 2018 [64]. By enriching the topic-specific training samples with text data from SERPs of web search engines and the linked web pages, we train topic-specific and cost-efficient classifiers that can be used to search test collections for relevant documents. We compare our reimplementations to the original results that were derived approximately two years before.

However, web content and especially SERPs are subject to several influences, and like the web content itself, they change frequently. Thus, it is worth investigating the robustness of the reimplementations on a more granular level. For this purpose, we retrieved training data from both web search engines for 12 days, starting on June 7th, 2020. Furthermore, we compare the influence of retrieving the web search results and, thus, the training data from the two different web search engines, Google and DuckDuckGo. In our experimental setup, we consider the inferior run `uwmrgx` as the baseline and the better-performing run `uwmrg` as the advanced version. More details about the two runs were outlined in Subsection 5.2.2. For all web search queries, concatenations of the topic's `title` and `description` were used.

Figure 5.6 shows the RBO and the intersections between the URLs scraped every second day compared to those scraped at the beginning of June 7th, 2020. As the

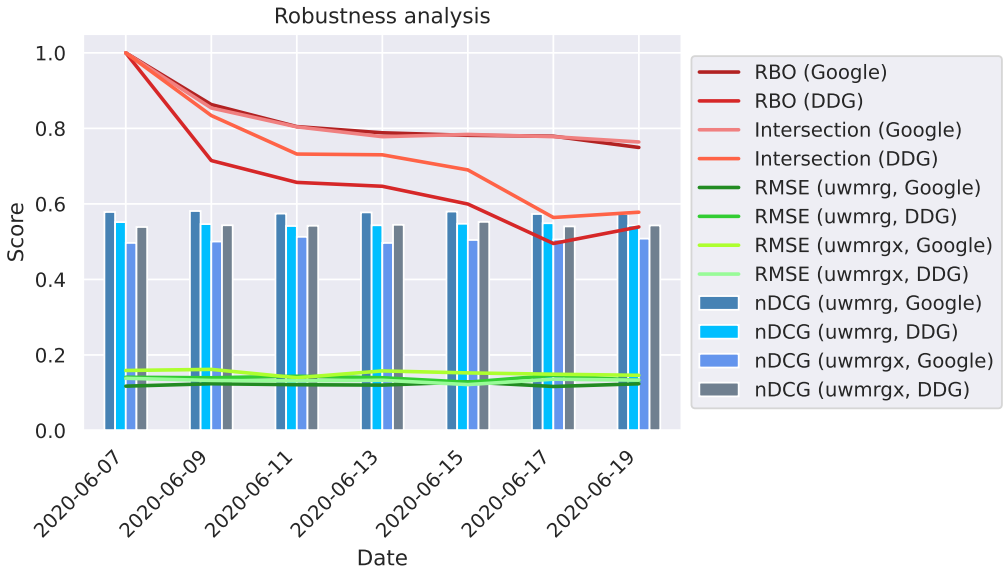

Figure 5.6: Robustness analysis over a period of twelve days. The bars plot the retrieval effectiveness in terms of nDCG, whereas the greenish plots show the corresponding error between the scores over the topics. The reddish plots show the rank correlation and the intersection between the entries of SERPs that were used to train the relevance classifiers at different dates.

URLs of the first date are used as the reference ranking, both plots of the RBO and the relative intersection start at 1.0 on the first date. Additionally, the figure includes the absolute nDCG scores and the RMSE scores of the reproduced baseline runs. While the RBO scores decrease over time, the nDCG and RMSE scores are robust with only slight variations. In combination, nDCG and RMSE show that the ARP can be reproduced at different dates and that it is possible to make estimates about the expected error between the score distributions.

We find a strong correlation between the RBO scores and the number of intersecting URLs in the search result lists (Pearson's $r = 0.9747$ and $p = 0.0002$), the lower the RBO, the fewer URLs are in both SERP lists from different days. While it is out of scope to reach any definitive conclusions, we see that the SERP's actual search results (and their URL orders) do not have to be the same as in the original experiment to reproduce the system performance and effectiveness. Under the consideration of this "bag of words" approach, we assume that the results can be reproduced with different web search results, having a similar vocabulary or tf-idf features that resemble those used to train the classifiers in the original experiments. In conclusion, the ARP can be reproduced independently of the actual web search results and dates. It is sufficient to have topical relatedness in the web search results to generate suitable tf-idf features that will yield the same effectiveness as in the original experiment.

5.4.4 Overall Effects

In the following, we evaluate the reimplementations by replacing the target collection and varying the query composition sent to the web search engines to retrieve the text-based training data. We evaluate the reimplementations over all four newswire

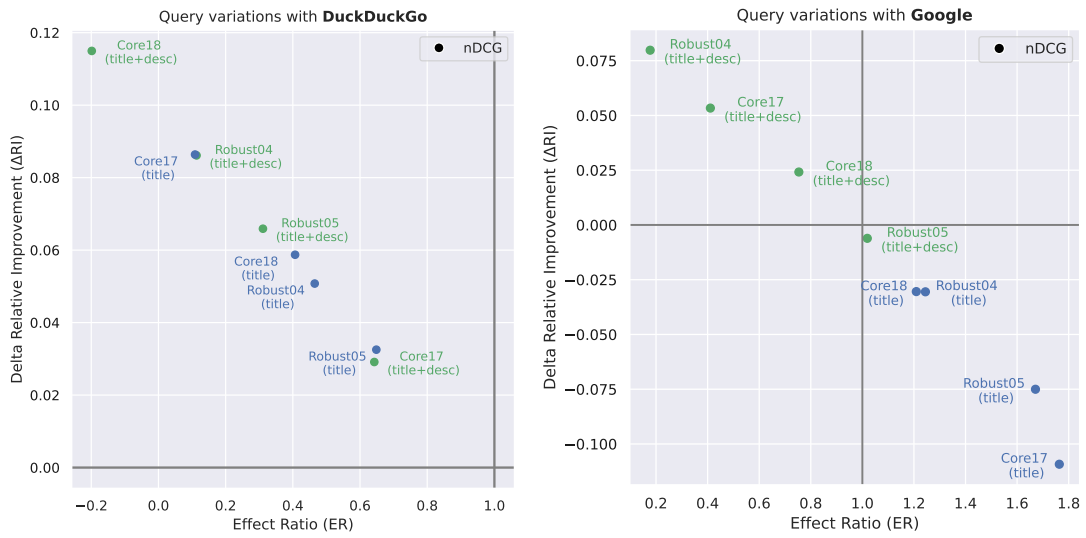

Figure 5.7: Comparison of the overall effects with training data either scraped from DuckDuckGo (left) or Google (right). Both plots show the Delta Relative Improvement over the Effect Ratio with different types of queries sent to the web search engines to retrieve the training data. The blue plots show the results of queries made from the topic's `title`, whereas the green plots correspond to results based on queries made from the topic's `title` and `description`.

test collections, including Robust04/05 and Core17/18. Furthermore, we include shorter queries solely based on the topic's `title` as well as longer queries composed of the topic's `title` and `description`.

When replacing the target collection, it is impossible to compare KTU, RBO, and RMSE since the runs contain different documents for possibly different topics. In this case, the experiment can be evaluated at the level of overall effects. Here, the ER and the DRI measure the effects between the baseline and advanced runs. As also pointed out in Chapter 4, perfectly replicated effects are equal to ER=1, whereas lower and higher scores than 1 indicate weaker or stronger effects, respectively, than in the original experiment. The DRI complements the ER by considering the absolute scores of the effects. In this case, perfect replication equals DRI=0. Likewise, lower and higher scores indicate weaker or stronger effects, respectively.

When combining both measures in the evaluations, it is helpful to illustrate the overall effects by plotting the DRI against ER. In general, it can be said that the closer a point to (ER=1, DRI=0), the better the replication. Figure 5.7 shows this visualization technique for runs based on training data from DuckDuckGo (left) or Google (right), whereas both measures are instantiated with nDCG.

The colors distinct runs with `title` queries (blue) from `title+description` queries (green). Comparing both search engines, the reproduced and replicated overall effects tend to be higher for training data retrieved with Google, as can be seen by the data points distributed over the second and fourth quadrants. In contrast, for DuckDuckGo, all data points are in the second quadrant. Especially the training data from Google retrieved with the `title` queries results in ER $>$ 1 across all test collections, and thus all `title` data points are in the fourth quadrant. In general, training data from Google with `title` queries results in stronger overall effects than in the original experiment.

Our additional analysis (cf. [64]) shows that this can be explained by lower replicability scores for the baseline runs. In contrast, the advanced runs resemble the original scores fairly well. For instance, the replicated advanced run `uwmrg` based on Google with `title` queries derived from Robust05 achieves a score of $\text{nDCG}_{\texttt{uwmrg}'}$=0.5865, while the corresponding replicated baseline run `uwmrgx` results in $\text{nDCG}_{\texttt{uwmrgx}'}$=0.5003. In reference to the original scores of $\text{nDCG}_{\texttt{uwmrgx}}$=0.5306 and $\text{nDCG}_{\texttt{uwmrg}}$=0.5822, ER=1.6712 indicates larger effects between the baseline and advanced version than in the original experiment.

Regarding the results based on training data from DuckDuckGo, there are weaker overall effects with ER < 1 for each combination of test collection and query type. In most cases, the baseline scores are higher than the corresponding counterparts based on Google results, whereas the advanced scores are lower than those from Google or the original experiments. For instance, replicated results derived from Core18 with `title+description` queries results in $\text{ER}_{\text{nDCG}}=-0.1985$. In this case, the baseline scores are higher than those of the advanced versions.

5.5 Principled Evaluations Based on PRIMAD

In the following, we analyze the annotated run files of the dataset introduced in Section 5.3. Having identified our reimplementations as reasonable candidates for a reproducibility analysis, we put them into context by comparing them to the results by YXL. Given the metadata annotations, we align the experiments to PRIMAD. While certain components are fixed, others are modified to gain new insights. In total, we analyze three different use cases that incrementally diverge from the original experimental setup by modifying the PRIMAD components. In the first experiment (cf. Subsection 5.5.1), only the method component is varied by principled parameter changes — a setting that complies with parameter sweeps as they are usually done in computational experiments (PRIM'AD). In the second experiment (cf. Subsection 5.5.2), we evaluate the reproducibility of the CCRF method in reference to the original submission made by GC and the corresponding reimplementations, which translate into keeping the data fixed, while other PRIMAD components are varied (P'R'I'M'A'D). Finally, we evaluate the replicability and generalizability in the third experiment (cf. Subsection 5.5.3) by varying all of the components (P'R'I'M'A'D').

5.5.1 PRIM'AD: Parameter Sweeps of the Method

After implementing a retrieval method, the actors usually improve the retrieval performance by finding optimal parameterizations. For instance, this can be realized with a systematic parameter analysis by tuning the implementations with grid search techniques. However, the following experiments only vary a single parameter. To this end, all of the PRIMAD components stay fixed except for the method (M').

In this experiment, we use the reimplementations by YXL [437]. In contrast to the original experiments, they introduced a multi-stage ranking pipeline to the CCRF method by retrieving the first ranking with BM25 and expanded methods (including RM3 and axiomatic reranking). The initially retrieved list is then reranked by a ML classifier. The first-stage ranking and the ML-based reranking are interpolated with parameterizable weights. YXL reimplemented the runs with different ML classifiers such as support vector machines, logistic regression, or gradient boosting

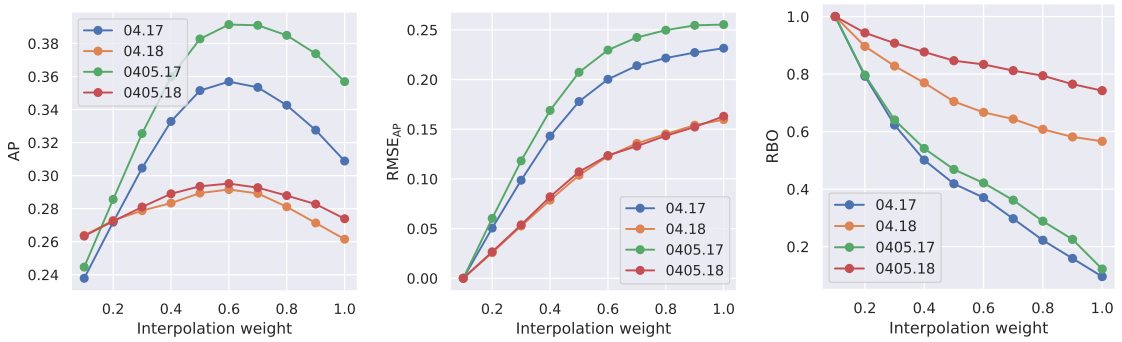

Figure 5.8: Parameter sweeps (of interpolation weights) evaluated by Average Precision (AP) and the corresponding error RMSE_{AP} and rank correlation RBO_{AP} based on reimplementations by Yu et al. (YXL) [437].

trees. Figure 5.8 evaluates the rerankings with different interpolation weights by the AP as well as the corresponding RMSE_{AP} and RBO_{AP}.

The evaluated runs are either made with only one source collection (Robust04) as training data and derived from two different target collections (Core17 and Core18), denoted as `04.17` and `04.18` or with two source collections as training data (Robust04 and Robust05), denoted as `0405.17` and `0405.18`. As the retrieval effectiveness in terms of AP shows, there is a sweet spot around an interpolation weight of 0.6, indicating that the runs of our regression tests reproduce those of YXL [437].

The RMSE quantifies the error between the topic score distributions of two runs [60]. This particular experiment determines the distributions of topic scores from the reranked runs with the lowest interpolation weight (0.1) and the other reranked runs. While the absolute retrieval performance decreases after it peaks around an interpolation weight of 0.6, the RMSE_{AP} monotonically increases, meaning that the topic score distributions more and more diverge from that of the reference (with an interpolation weight of 0.1), which can be attributed to the increasing influence of the ML-based reranker.

It can clearly be seen that the absolute AP scores, as well as the RMSE_{AP} scores, differ depending on the dataset (e.g., `04.17` vs. `04.18`), which indicates that the effectiveness of CCRF has a data dependency with regard to the combination of the source and target collections.

5.5.2 P'R'I'M'A'D: Reproducing the Experiments

Reactive actions towards reproducibility can be realized in the form of a reimplementation study where the original experiment is repeated based on the descriptions in the corresponding publication. Suppose the outputs or artifacts of the original experiments are available. In that case, as is the case for TREC runs, we can use these artifacts as points of reference to which we compare the reimplementations' outputs as it was outlined in the previous Chapter 4.

In the following experiment, we evaluate the reproducibility of the CCRF method by comparing the reimplementations of YXL and BFFMSSS to the original results by GC. We consider all of the PRIMAD components to be changed except for the data component D since the reimplementations are evaluated on the same test collection

Table 5.2: Reproducibility evaluation of Grossman and Cormack (GC) [159] compared to Yu et al. (YXL) [437] and Breuer et al. (BFFMSSS) [60].

Researchers	GC [159]	YXL [437]	BFFMSSS [60]
Baseline			
Average Precision	0.3711	0.4018	0.3612
Kendall's τ Union	1.0000	0.0086	0.0051
Rank-Biased Overlap	1.0000	0.1630	0.5747
Root Mean Square Error	0.0000	0.1911	0.1071
p-value	1.0000	0.1009	0.7885
Advanced			
Average Precision	0.4278	0.4487	0.4208
Kendall's τ Union	1.0000	0.0069	0.0111
Rank-Biased Overlap	1.0000	0.2231	0.6706
Root Mean Square Error	0.0000	0.2088	0.0712
p-value	1.0000	0.2785	0.8249
Overall effects			
Effect Ratio	1.0000	0.8267	1.0514
Delta Relative Improvement	0.0000	0.0362	-0.0123

(Core17) as in the original experiment. Please note that this aligns with the ACM terminology when treating the test collection as the original experimental setup.

Obviously, the reimplemented runs originate from three different groups of actors (A') who used different implementations (I') on different platforms (P'). In general, it is debatable if the other two components — the research goal and the method — also changed since, from a general point of view, all of the runs are made to rank documents of Core17 with the help of CCRF. However, as part of these evaluations, we differentiate between the research goals of the three reimplementations. Opposed to GC's participation in TREC Common Core, the reimplementations by YXL and us did not participate in a shared task but had the objective of reproducing the original experiment. Thus, they have a different research goal (R'). Similarly, we see differences in the method (M') because of YXL's reinterpretations of the workflow that make a two-stage ranking procedure out of the original approach by introducing BM25 as a first-stage retrieval method, followed by the actual CCRF, which, in turn, is also implemented by other ML approaches than logistic regression.

Table 5.2 shows an evaluation of the reproduction quality based on the measures that were introduced in Chapter 4. As the AP scores show, for both run types (baseline and advanced), the retrieval performance of the BFFMSSS reimplementations is slightly below the original results, while the YXL reimplementations outperform the results by GC. We assume that the additional first-stage-ranker based on BM25 already provides a good baseline with acceptable recall rates, which is of benefit for the ML-based reranker.

Regarding reproducibility, KTU shows that both reimplementations fail to reproduce the exact ordering of the documents in the rankings. Optimally, these scores should be close to 1.0, while the reported values show almost no correlation between

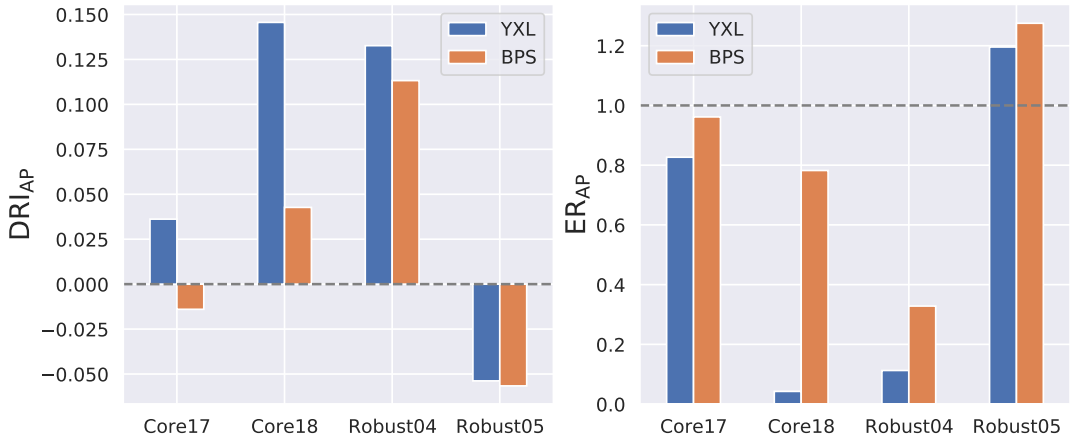

Figure 5.9: Evaluation of the overall effects based on Delta Relative Improvement (DRI) and Effect Ratio (ER) of AP. To evaluate how well the overall effects reproduce in a different setting, we used the relative differences by GC based on Core17 for the YXL runs (blue) as the reference, whereas the relative differences of the BPS runs (orange) were compared in reference to those by GC based on Core18.

the document rankings. However, in this regard, there is a higher similarity between the GC and reimplemented runs, especially for the BFFMSSS runs, in terms of the RBO, which can be partly explained by the measure's discount for lower ranks [419].

Likewise, the RMSE and the p-values indicate that, in comparison, the topic score distributions of the BFFMSSS runs are closer to the GC runs than the reimplementations by YXL. Both measures are determined by the topic score distributions of the AP scores. While low p-values would result from different distributions, higher p-values indicate more similar distributions. As mentioned earlier, the YXL runs already result in strong baseline scores that outperform the original AP scores, and as a result, the p-value is lower. Regarding the advanced run, both reimplementations achieve slightly higher p-values.

If a baseline and advanced run are available, it is possible to determine the ER and the DRI. Both measures quantify how well the reimplementations preserve the improvements of the advanced run over the baseline. As shown by the ER, the BFFMSSS runs are closer to the optimal value of 1.0, which again can be explained by the already strong baseline of YXL runs. As a result, the improvements of the advanced YXL runs are not as high as in the original experiments. Similarly, the DRI score, which also accounts for the absolute scores of the baseline runs, is closer to the optimal value.

5.5.3 P'R'I'M'A'D': Generalization with Other Data

Finally, we provide an outlook on how well CCRF generalizes with other data and with a method based on training data from web-search results. This setup translates into a setting where every PRIMAD component is changed regarding the original experiment. Similar to the evaluations in the previous Subsection 5.5.2, the first five PRIMAD components are different than in the original experiment. Especially, the method (M') differs as some runs do not rely on tf-idf features derived from a test collection but instead use scraped web search results as the "source collection".

Regarding the data (D'), we used all four test collections, i.e., Robust04, Robust05, Core17, and Core18, as target collections.

From both reimplementations of YXL and BPS, we evaluate runs derived from four target collections, including Core17/18 and Robust04/05. The YXL runs are evaluated in reference to the GC runs from Core17 [159], while the BPS runs are evaluated in reference to the GC runs from Core18 [160]. When reproducing a retrieval experiment on new data, it is impossible to evaluate some of the reproducibility measures since they depend on runs derived for the same topics or from the same pool of documents. Similar to the evaluations in Subsection 5.4.4, ER and DRI are used as proxies to quantify how well relative improvements can be reproduced or replicated. For both run types, Figure 5.9 illustrates these measures.

Both measures show that the reimplementations deviate from the original experiment except for the BPS runs of Core17 with a nearly optimal DRI value. Likewise, the corresponding ER score is below but close to 1.0. The YXL runs of Core18 have the largest DRI deviation and also the lowest ER score. This complies with the results of Subsection 5.5.2, which already showed that CCRF does not generalize well with this particular combination of the Robust corpora and Core18. The experiments with Robust04 also show that CCRF does not generalize with this particular dataset. The positive DRI and low ER scores show that the baseline scores are higher than in the original experiment, while the reimplemented advanced runs do not provide a similar improvement. Both experiments on Robust05 have comparable ER scores above 1.0, which shows that the generalization was more successful than the experiments with the other test collections. However, it has to be considered that there are lower absolute scores, as shown by the negative DRI scores.

Overall, the reproducibility of CCRF strongly depends on the combinations of the datasets. While it is out of this study's scope to draw any conclusions about this circumstance, we assume this can be attributed to a higher overlap of vocabulary terms in documents with relevance assessments in the respective corpora.

5.6 Conclusion

This chapter has introduced principled reproducibility (and replicability) evaluations for system-oriented IR experiments. Based on the reimplementations of the CCRF method by GC introduced in Sections 5.1 and 5.2, we have compiled an annotated dataset of runs (cf. Section 5.3). Our preliminary reproducibility evaluations in Section 5.4 analyzed the quality of our reimplementations before using the corresponding run files for the principled reproducibility evaluations based on PRIMAD in Section 5.5. The outlined setups incrementally diverge from the original experiment and cover three typical scenarios described in terms of PRIMAD.

It has to be pointed out that the contributions of this chapter have an entirely system-oriented focus. Referring back to the experimental results from Subsection 5.4.2, we note that even though the retrieval effectiveness was reproduced fairly well, there were tremendous differences between the document rankings with KTU scores that did not indicate any correlation between the document rankings at all.

In order to provide an anecdotal example of what these results might imply for the user experience, we have picked them up in a reproducibility experiment based on the lexical retrieval method BM25. In Figure 5.10, we have determined the P@10 scores as well as differences in terms of KTU and RMSE between runs with

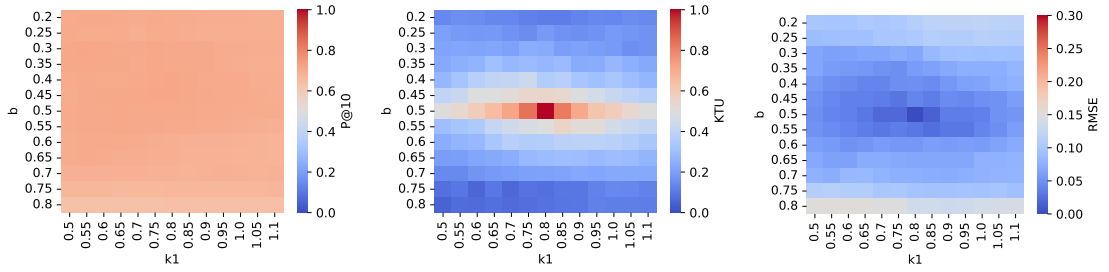

Figure 5.10: P@10, Kendall's τ Union, and RMSE scores of rankings based on BM25 and the TREC-COVID collection. All of the scores have been averaged over 50 queries and the reproducibility measures were determined in reference to the center of the heatmaps, i.e., the run with parameters $b = 0.5; k_1 = 0.8$.

variations of the parameters b and k_1. All of the runs were determined based on the TREC-COVID test collection [411] and contain ten documents for each query. The P@10, KTU, and RMSE scores have been averaged over 50 `title` queries, whereas the reproducibility measures were determined in reference to the parameter setting of $b = 0.5; k_1 = 0.8$ that is in the center of the heatmaps.

Even though these parameter variations result in negligible differences in the retrieval performance, as seen from the P@10 scores, there are low correlations between the document rankings regarding KTU with only subtle changes in the parameterization. Although the retrieval performance might be nearly the same as indicated by the low RMSE scores, the list of top-ranked documents is completely different for some topics.

It is still an open research question what kinds of implications these deviations would have in a user-oriented experiment. While it is fair to say that a reproduced ranking with a completely different document order is a serious problem when repeating a user experiment, it was shown, on the other hand, that users compensate for worse ranking results [396]. In this sense, the P@10 scores show that the ranking still contains documents with the same average relevance and possibly the same information, which means that users could possibly gain the same knowledge as in the original experiment and still close their information gap when adapting their browsing behavior or reading documents in a different order.

These examples highlight once more the importance of considering the user as part of the experiment and the corresponding evaluations. In general, little attention is paid regarding the implications for users in reproducibility experiments, as could also be seen from our literature review of the ECIR reproducibility track. To this end, the following Chapters 6, 7, and 8 shift the focus towards more user-focused evaluations, either by simulation or real-world experimentation.

Chapter 6

Simulated User Query Variants

In this chapter, we validate simulations of User Query Variants (UQV) with the help of TREC test collections. Besides, we introduce a simple yet effective method with better reproductions of real queries than the conventional simulation methods. Our evaluation framework validates the simulations in reference to real UQV regarding the retrieval effectiveness, reproducibility of topic score distributions, shared task utility, effort and effect, and query term similarity. In sum, our contribution **C7** covers a **method for query simulations** based on IR test collections and a corresponding **evaluation framework**.

Our experiments include more general query simulators making query formulations based on topic texts, and opposed to that, simulations of known-item searchers who are familiar with the vocabulary of relevant documents in the document collection. As a compromise between these two types of simulators, we introduce a simulation method that allows parameterizing the query reformulation behavior and thus better reproduces the outcomes of real queries. More specifically, our research questions are as follows:

RQ3 *How do real user queries relate to simulated queries made from topic texts and known-items in terms of retrieval effectiveness?*

RQ4 *To which degree do simulated queries reproduce real queries provided that only resources of the test collection are considered for the query simulation?*

RQ5 *How well does the introduced query simulation method generalize in a cross-collection setting where the query simulations are based on a source collection and used to rank documents of a target collection?*

Overall, we conclude that the general query simulators based on the topic text's vocabulary are suitable estimates for lower-bound effectiveness. At the same time, the simulated know-item searchers are upper-bound effectiveness estimates within the range of the analyzed queries. Whereas the retrieval effectiveness and statistical properties of the topic score distributions and economic aspects are close to that of real queries, it is still challenging to simulate exact term matches and later query reformulations. Our cross-collection experiments show that most of the findings can be replicated if the query simulations are based on the test collection of Core17 and used to rank the test collection of Robust04. Most of the contents in this chapter are based on our results that were contributed to ECIR [62].

The remainder of this chapter is structured as follows. The next Section 6.1 puts our contributions into context with other existing work. In Section 6.2, we review the existing methods for simulating queries from test collections and present our simulation approach. Section 6.3 introduces the validation framework, which is used in Section 6.4 for the experimental evaluations, followed by the replicability analysis with another test collection in Section 6.5. Finally, we give answers to our research questions in Section 6.6 and conclude in Section 6.7.

6.1 Query Simulations and User Query Variants

As pointed out in Chapter 3, more specifically in Subsection 3.1.7, system-oriented IR evaluations are limited to a rather abstract understanding of how the real user behaves. Under the Cranfield paradigm, the simplified understanding of users is limited to a single query and the examination of the result list in its entirety [31]. However, real search behavior is more complex: searching is normally an iterative process with query reformulations. Furthermore, not every search result is examined but rather picked out after judging its snippet text.

In order to compensate for this shortcoming, it is common practice to include (logged) user interactions in the evaluation process. Industrial research is often supported by large datasets of user interactions that, unfortunately, cannot be shared publicly, e.g., due to privacy concerns [102]. As a solution, simulating user interactions provides a cost-efficient and reproducible way to support system-oriented experiments with more realistic directives when no interaction logs are available or when it is simply not feasible to conduct a user-oriented study [31].

Carterette et al. [79] addressed the lack of user interaction data available to academic research by introducing the concept of Dynamic Test Collections. Their framework expands test collections with simulated interactions comprising the entire sequence of interactions, including the simulation of queries, clicks, dwell times, and session abandonment. More recently, similar frameworks were introduced by Maxwell and Azzopardi [286] as Complex Searcher Model, by Pääkkönen et al. [317] as Common Interaction Model, or by Zhang et al. [443]. Our work can be seen in the light of Dynamic Test Collections and the related simulation frameworks, but with a special focus on simulating UQV.

Whereas previous work on simulating interactions either sought to cover complete interaction sequences [79,285,443], click interactions [88] (that are in the focus of the next Chapter 7), or stopping rules [285,317], work on simulating queries is underrepresented as also pointed out by Günther and Hagen [164]. Very few attempts have been made towards query simulations, and it has not been investigated if these can reproduce properties of real queries. Especially, it has not been validated yet to which degree query simulators reproduce real user queries when they are based on the resources of TREC test collections.

As opposed to previous work in this regard, it is not our primary goal to generate the most effective queries but rather to validate simulated queries since the query formulation is one of the first user interactions with the search system, and as such, it is a critical component for any subsequent simulated interactions like clicks and others. This study aims to answer how queries can be simulated and evaluated by using TREC test collections and the corresponding resources. Most of the current methods for query simulations follow a two-stage approach, including the *term can-*

didate generation and the *query modification strategy*. Usually, the term candidates are derived from a language model. Jordan et al. [211] introduced Controlled Query Generation (CQG) that exploited the relative entropy of a language model for query term generation. Azzopardi et al. [18, 19] applied CQG when generating queries for known-item search. Similarly, Berendsen et al. [45] used annotations to group documents, and Huurnik et al. [195] simulated queries for purchased items. When query term candidates are available, there exist some commonly used query modification strategies [16,33,211,222], which were also applied in follow-up studies [285,286,405], following principled query reformulation patterns (cf. Table 6.1).

If large-scale user logs are available, different approaches proposed learning to rewrite queries [176], model syntactic and semantic changes between query reformulations [177], or replace old query terms with new phrases with the help of the point-wise mutual information [209]. In contrast to these examples, the query simulations analyzed in this study do not rely on large-scale user logs but use test collections and related resources, i.e., topics and relevance judgments.

As part of follow-up studies related to the TREC Session Track, Guan et al. [162, 425] improved session search results by introducing the Query Change Model (QCM) according to which the session search is modeled as a Markov Decision Process that considers transitions between states, i.e., queries and other interactions, to improve search results for query reformulations. Van Gysel et al. [167] found that QCM is especially effective for longer sessions while being on par with term-frequency-based approaches for shorter sessions. Our query simulation method is inspired by QCM but generates queries instead of improving retrieval results throughout a session.

Simulated UQV contribute to more diverse and more realistic user-oriented directives as part of the system evaluations. Besides the actual simulation of session search, applications for simulated queries are manifold when real user queries are unavailable. For instance, Bailey et al. [23] claim that a larger variance of the retrieval effectiveness can be expected from query variations than from system variations, and they highlight that system evaluations can be improved when multiple query variants for the same information need are considered. Besides, UQV can enhance the pooling process [297], make rank fusion approaches possible [44], are used for query performance prediction [124], or assist users with query suggestions that improve the recall [406].

6.2 Query Generation Techniques

In the following, we recapture the conventional query generation techniques that rely on a two-stage process, including the generation of term candidates (cf. Subsection 6.2.1) and the modification strategy (cf. Subsection 6.2.2). Afterward, we describe our new approach, giving control over the query reformulation behavior (cf. Subsection 6.2.3).

6.2.1 Term Candidate Generation

Simulating queries based on topics of test collections most likely complies with exploitation search tasks [257], where users normally have a very concrete understanding of their information needs but are not necessarily familiar with the documents in the collection. After real users have read the topic, they will likely in-

Table 6.1: Query modification strategies by Baskaya et al. [33]. The corresponding terms are either taken from the concatenated topic text or based on a language model made from relevant documents.

Strategy	Query modifications
S1	$q_1 = \{t_1\}; q_2 = \{t_2\}; q_3 = \{t_3\}; ...$
S2	$q_1 = \{t_1, t_2\}; q_2 = \{t_1, t_3\}; q_3 = \{t_1, t_4\}; ...$
S2'	$q_1 = \{t_1, t_2, t_3\}; q_2 = \{t_1, t_2, t_4\}; q_3 = \{t_1, t_2, t_5\}; ...$
S3	$q_1 = \{t_1\}; q_2 = \{t_1, t_2\}; q_3 = \{t_1, t_2, t_3\}; ...$
S3'	$q_1 = \{t_1, t_2, t_3\}; q_2 = \{t_1, t_2, t_3, t_4\}; q_3 = \{t_1, t_2, t_3, t_4, t_5\}; ...$

clude key terms of the topic texts when formulating queries. As a simplified implementation, the TREC Topic Searcher (TTS) considers only terms of the sequence $T_{\text{topic}} = \{t_1, ..., t_{n_{\text{topic}}}\}$, composed of the topic's title, description, and narrative, where $t_1, ..., t_{n_{\text{topic}}}$ is the term sequence in the concatenated text with n_{topic} terms.

Opposed to the TTS, we simulate a Known-Item Searcher (KIS) for upper bound effectiveness estimates. Here, we assume the simulated users to be familiar with the document collection. When reading the topics, they recall key terms of the relevant documents in the collection and use these as their query terms. In this case, the sequence with n_{rel} term candidates $T_{\text{rel}} = \{t_1, ..., t_{n_{\text{rel}}}\}$ is derived with the help of a language model based on CQG by Jordan et al. [211] according to:

$$P(t|D_{\text{rel}}) = (1 - \lambda)P_{\text{topic}}(t|D_{\text{rel}}) + \lambda P_{\text{background}}(t) \tag{6.1}$$

where the topic model $P_{\text{topic}}(t|D_{\text{rel}})$ is made from the relevant documents D_{rel} for a given topic, while the background model $P_{\text{background}}(t)$ is derived from the vocabulary of the entire corpus. λ is used to model the influence of the background model, and it is set to 0.4 to be consistent with previous work [107,211]. In this case, $t_1, ..., t_{n_{\text{rel}}}$ are ordered by the decreasing term probabilities of the language model.

6.2.2 Query Modification Strategy

We make use of the query modification strategies proposed by Baskaya et al. [33], that were also used in previous simulation studies [211, 285, 286, 317, 405]. Table 6.1 shows these query modification strategies, which are used in combination with the term candidates of T_{topic} and T_{rel}: the strategy S1 outputs single term queries following the ordering of term candidates; S2 keeps the first candidate term fixed and composes query strings by replacing the second term for reformulations; S2′ is similar to S2, but keeps two candidate terms fixed; S3 starts with a single term query and incrementally adds query terms for reformulations; S3′ is similar to S3, but starts with two candidate terms. In total, we analyze ten different query simulators that result from the two-term candidate generators that are combined with five query modification strategies, denoted as $\text{TTS}_{\text{S1-S3}'}$ and $\text{KIS}_{\text{S1-S3}'}$, respectively.

Table 6.2: Controlled query reformulation strategies and corresponding weights according to Equation 6.3. The chosen weights are based on optimal parameters determined by Yang et al. [425] and were adapted to comply with the described reformulation behaviors.

Strategy	α	β	ϵ	δ	The user simulator ...
S4	2.2	0.2	0.05	0.6	... prefers title and topic terms, and keeps previous query terms;
S4'	2.2	0.2	0.25	0.1	... mainly keeps previous query terms, and tends to include other terms;
S4"	0.2	0.2	0.025	0.5	... sticks to topic terms, with more variations between reformulations.

6.2.3 Controlled Query Reformulations

Compared to the previous query simulators, this approach adds a scoring stage for the generated query string candidates. These candidates are generated by considering every possible combination of n-grams from a term set. The corresponding terms are either taken from T_{rel} or $T_{\text{topic+rel}} = (T_{\text{topic}} \cap T_{\text{rel}}) \cup (T_{\text{rel}} \setminus T_{\text{topic}})_k$, where $(T_{\text{topic}} \cap T_{\text{rel}})$ contains topic terms in T_{rel} and $(T_{\text{rel}} \setminus T_{\text{topic}})_k$ denotes the top k terms of T_{rel} that are not in the topic text. In this regard, k models the user's vocabulary and domain knowledge. Having a set of different query string candidates, we rank the queries by:

$$\text{score}(q) = \frac{\sum_{t \in q} \Theta(t)}{|q|} \tag{6.2}$$

where $|q|$ denotes the query length and $\Theta(t)$ is a term-dependent score inspired by QCM [162, 425] and is defined as follows:

$$\Theta(t) = \begin{cases} \alpha(1 - P(t|D_{\text{rel}})), & t \in q_{\text{title}} \\ 1 - \beta P(t|D_{\text{rel}}), & t \in +\Delta q \wedge t \in T_{\text{topic}} \\ \epsilon \ \text{idf}(t), & t \in +\Delta q \wedge t \notin T_{\text{topic}} \\ -\delta P(t|D_{\text{rel}}), & t \in -\Delta q \end{cases} \tag{6.3}$$

where q_{title} is the set of topic title terms, and $+/-\Delta q$ denotes added or removed terms of a query reformulation that is made in reference to the previously simulated query. α gives weight to query terms in the query string candidate that are also contained in the query based on the topic title q_{title}. β controls how much weight is given to terms (in the query reformulation candidate) that were added to the previous query ($+\Delta q$) and that are also contained in the concatenated topic text T_{topic}. Similarly, ϵ controls how much weight is given to added query terms. In contrast, these terms are not contained in T_{topic}. Finally, δ controls how much weight is given to terms that were removed from the previous query formulation.

To generate the first query of a simulator q_1, we use the topic title as the reference for which the first formulation is determined with the help of Equation 6.2.

Subsequent reformulations are determined with regard to the previously generated query. To find the parameters of S4-S4″, we initially set them to the optimal parameters reported by Yang et al. [425]. Afterward, we adapted single parameters to reflect the query reformulation behavior described in the following.

In our experiments, we analyze 3-, 4-, and 5-gram query candidates and three different parameterizations of the simulators, defined in Table 6.2 and described as follows. First, we analyze strategy S4, which tends to prefer topic terms and mainly keeps terms of previous queries. Second, we analyze the strategy S4′, which mainly keeps terms of previous queries, but tends to include terms that are not in the topic text. Finally, we analyze the strategy S4″, which tends to stick to the topic terms but does not necessarily keep terms of previous query formulations. In sum, we analyze six different instantiations of these simulators, which are either based on T_{rel} (denoted as $\text{KIS}_{\text{S4-S4}''}$) or based on $T_{\text{topic+rel}}$ with $k = 4$ (denoted as $\text{TTS}_{\text{S4-S4}''}$).

The reasoning behind choosing these three parameterizations can be described as follows. The first strategy S4 represents a user who formulates queries close to the topic text and who varies little of the previously formulated query to try out modifications of the initial query in a principled way. Similarly, the underlying user of strategy S4′ also varies the query strings more principled and incrementally. However, in contrast to the simulated user of strategy S4, the S4′ user prefers terms not mentioned in the topic text, corresponding to a user who tries to include previous knowledge about the topic in the query formulations. Finally, the third strategy S4″ corresponds to a user whose vocabulary is mainly based on the topic text, like the user of S4. However, the user of S4″ tries to explore the information space by a stronger variation between query reformulations.

6.3 Validation Framework

In the following, we outline our evaluation framework used to validate the simulations in reference to real queries in different aspects. It includes evaluating the retrieval effectiveness, shared task utility, effort and effect, and query term similarity between simulated and real queries.

6.3.1 Retrieval Effectiveness

As shown by Tague and Nelson, simulated queries fall behind real queries in terms of retrieval effectiveness [384]. For this reason, we evaluate the retrieval effectiveness as it is common practice in system-oriented IR experiments. The retrieval effectiveness is determined by the average of a measure over all topics in a test collection. Beyond comparing the averaged means of different queries, we propose a more in-depth analysis of the topic score distributions for which we use some of the reproducibility measures that were introduced in Chapter 4. More specifically, we use RMSE (cf. Equation 4.5) to measure the closeness between the topic score distributions, with low errors indicating similar distributions of the retrieval effectiveness over the topics. Additionally, low p-values of paired t-tests (cf. Subsection 4.2.5) indicate a higher probability of different retrieval effectiveness.

6.3.2 Shared Task Utility

According to Huurnik et al. [195], the ARP of the simulated queries alone is not an appropriate indicator of how well the simulations resemble the real queries since useful query simulators should identify the best system. As proposed by them, we analyze how well the simulated queries reproduce system rankings, i.e., systems ordered by their relative retrieval effectiveness, by comparing them with the help of Kendall's τ — an approach that is common practice in meta-evaluations of shared tasks [409] (cf. Subsection 4.2.6) and that will also be picked up in the following Chapter 7. We compare the simulated and real queries by determining how well simulated queries can reproduce the system rankings with different parameterizations (and different retrieval effectiveness). More specifically, we determine Kendall's τ for the i-th query formulation of n_D topics in a test collection D as:

$$\tau_i = \frac{\sum_{j=1}^{n_D} \tau(s_{i,j}, s'_{i,j})}{n_D} \tag{6.4}$$

where $s_{i,j}$ denotes the reference system ranking for the i-th query formulation of the j-th topic, $s'_{i,j}$ the corresponding reproduced system ranking, and $\tau(s_{i,j}, s'_{i,j})$ denotes the rank correlation between the two system rankings according to Kendall's τ. In our experiments, we determine $s_{i,j}$ as the reference by the query formulations of real users and compare it to $s'_{i,j}$ resulting from the query formulations generated by our simulators.

6.3.3 Effort and Effect

In order to account for a more user-oriented evaluation, we simulate sessions and evaluate them with regard to the effort (number of queries) that has to be made and the resulting effects (cumulated gain). First, we simulate sessions using ten simulated queries and an increasing number of documents per query and evaluate the results by the Session-Based Discounted Cumulated Gain (sDCG), which discounts the cumulated gain document- and also query-wise by the logarithm and the corresponding base bq as well as by the query position i in a session [203]:

$$\text{sDCG} = \sum_{i \in \{1,\ldots,n_j\}} \frac{\text{DCG}_{q_i}}{1 + \log_{bq}(i)} \tag{6.5}$$

where n_j denotes the number of available queries for a topic j. In our experiments, we evaluate sDCG at different ranks, i.e., the simulated users browse a fixed depth of ranks before reformulating a query. Furthermore, we simulate memoryless users, i.e., the users also take previously seen documents into account.

In addition, we evaluate the simulation quality from another, more economical point of view. Azzopardi [16] applied economic theory to the retrieval process and demonstrated that for a pre-defined level of cumulated gain, query reformulations could be compensated by browsing depth (or vice versa browsing depth by more query reformulations). Furthermore, he illustrates this relationship with isoquants, a visualization technique used in microeconomics. Thus, we evaluate the closeness between isoquants of the simulated and real queries with the help of the Mean Squared Logarithmic Error (MSLE) as follows:

$$\mathrm{MSLE}(\bar{B}, \bar{B}') = \frac{1}{n_j} \sum_{i=1}^{n_j} (\log(\bar{B}_i + 1) - \log(\bar{B}_i{}' + 1))^2 \tag{6.6}$$

where $\bar{B}_i$ denotes the average browsing depth, i.e., the number of documents that have to be sighted, in order to achieve a pre-defined level of gain with i queries for each topic in a test collection. $\bar{B}$ is a vector that contains n_j entries with the corresponding average browsing depths, and n_j is the total number of available queries for each topic. $\bar{B}'$ is the corresponding vector of the simulated queries.

6.3.4 Query Term Similarity

This study's primary goal is not to simulate query strings with exact term matches. Instead, simulated UQV should result in diverse query strings for a given information need. Nonetheless, it is worth analyzing the term overlap between the simulated and real queries. As Liu et al. [256] or Mackenzie and Moffat [269] propose, we determine the Jaccard similarity between the sets of unique terms made from the query reformulations. When compared with the other evaluations, the term similarities add more insights about the simulated UQV. For instance, if it is possible to simulate query reformulations that adequately relate to the properties of real queries but with other terms. We determine the Jaccard similarity over n_D topics by:

$$\begin{aligned} J_j(Q, Q') &= \frac{|Q \cap Q'|}{|Q \cup Q'|} \\ \bar{J}(Q, Q') &= \frac{1}{n_D} \sum_{j=1}^{n_D} J_j(Q, Q') \end{aligned} \tag{6.7}$$

where $J_j(Q, Q')$ denotes the Jaccard similarity for the j-th topic between the query term sets Q and Q' resulting from the unique query terms of all available query formulations for the corresponding topic. If there are different numbers of query formulations available for a particular topic, we restrict both term sets to the number of query formulations available from both query generators in order to avoid a biased Jaccard similarity that would result from one query set having substantially more query terms. With our query simulators, we can generate an arbitrary number of query reformulations. However, some users (of the dataset described in the following subsection) formulated just a few queries for some topics or a different number of queries per topic in general; thus, we compare the sets with unique query terms resulting from an equal number of real and simulated queries. As an alternative, the similarity can be determined by comparing an equal number of terms instead of an equal number of queries. However, in our evaluations, we want to account for the differences that can occur when simulated users tend to generate longer query strings over query reformulations (cf. S3 and S3′ in Table 6.1).

6.3.5 Datasets and Implementation Details

In our experimental setup, we use the UQV dataset [487] provided by Benham and Culpepper [41]. Given the topic texts, eight users formulated up to ten query variants for each topic. As can be seen in Figure 6.1, each of the eight users formulated at least one query for each topic, and the fifth user (denoted as UQV_5) formulated

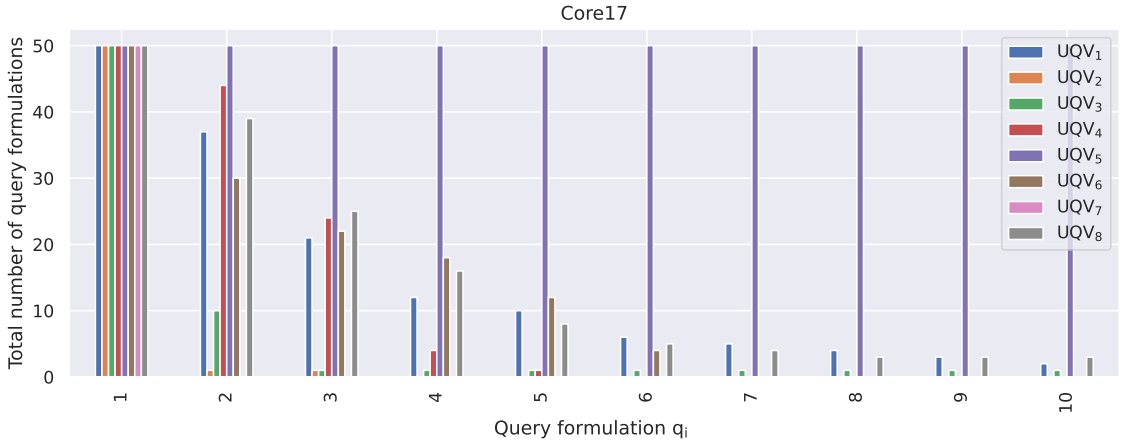

Figure 6.1: Query distribution in the dataset by Benham and Culpepper [41] over the topics of the Core17 test collection. The bar plots show for how many topics queries were made for the i-th formulation. There is a single user, to which we refer as UQV_5, who consistently formulated ten queries for each topic.

ten queries for each topic. More details about the query collection process are provided by Benham et al. [42]. Accordingly, we evaluate the system runs with The New York Times Annotated Corpus and the topics of TREC Common Core 2017 [4]. As part of our experiments, we exploit the interactive search possibilities of the Pyserini toolkit [250]. We index the Core17 test collection with the help of Anserini [428] and the default indexing options as provided in the regression guide [462]. Unless stated otherwise, all results are retrieved with the BM25 method and Anserini's default parameters ($b = 0.4$, $k = 0.9$). We evaluate the results with the `repro_eval` toolkit [61] featuring bindings to `trec_eval` measures, which was already introduced in Chapter 4. We made the source code of the experiments and the simulated queries publicly available in a GitHub repository [465].

6.4 Experimental Validation

The following section presents the experimental results that are validated according to the framework introduced in the previous section, including the retrieval effectiveness (cf. Subsection 6.4.1), the shared task utility (cf. Subsection 6.4.2), tradeoffs between effort and effect (cf. Subsection 6.4.3), and the query term similarities (cf. Subsection 6.4.4).

6.4.1 Retrieval Effectiveness

Regarding **RQ3**, we validate the retrieval effectiveness of real (UQV) and simulated (TTS/KIS) queries. Table 6.3 shows the ARP, including nDCG and AP scores that are determined by averaging results with 1000 documents per topic and P@10 scores over *all* queries, the *first*, or the *best* query of a topic.[1] First of all, our assumptions are confirmed. The retrieval effectiveness of real queries ranges between that of the $TTS_{S1\text{-}S3'}$ and $KIS_{S1\text{-}S3'}$ simulators. Especially, the effectiveness of the $TTS_{S1\text{-}S3'}$

[1] S1 and S3, as well as S2 and S3′, do not differ when averaging over the first queries.

Table 6.3: Retrieval effectiveness over q queries. Besides averaging the retrieval effectiveness over all available queries, we also evaluated the effectiveness of the first and the most effective queries for each topic.

	All queries				First queries				Best queries			
	q	nDCG	P@10	AP	q	nDCG	P@10	AP	q	nDCG	P@10	AP
UQV_1	150	.3787	.4507	.1581	50	.4293	.5040	.2003	50	.4969	.6320	.2429
UQV_2	52	.4221	.5058	.2020	50	.4096	.4880	.1894	50	.4103	.4900	.1896
UQV_3	68	.3922	.4353	.1780	50	.3979	.4560	.1813	50	.4117	.4800	.1878
UQV_4	123	.4126	.4894	.1888	50	.4469	.5220	.2099	50	.5146	.6300	.2644
UQV_5	500	.3922	.4330	.1649	50	.4447	.4920	.2043	50	.5353	.7240	.2807
UQV_6	136	.4030	.4713	.1843	50	.4488	.5080	.2197	50	.4980	.5980	.2515
UQV_7	50	.4980	.5720	.2418	50	.4980	.5720	.2418	50	.4980	.5720	.2418
UQV_8	156	.3814	.4545	.1645	50	.4046	.4500	.1799	50	.4556	.5620	.2193
TTS_{S1}	500	.0479	.0306	.0127	50	.1705	.1280	.0541	50	.3066	.2360	.0971
TTS_{S2}	500	.1964	.1716	.0688	50	.3592	.3900	.1604	50	.4391	.5100	.2097
$TTS_{S2'}$	500	.3387	.3426	.1413	50	.3895	.4020	.1821	50	.4639	.5940	.2283
TTS_{S3}	500	.3323	.3632	.1388	50	.1705	.1280	.0541	50	.4776	.6080	.2383
$TTS_{S3'}$	500	.3499	.3874	.1474	50	.3592	.3900	.1604	50	.4709	.6060	.2311
TTS_{S4}	500	.4493	.5168	.2088	50	.4409	.4920	.2072	50	.5945	.7620	.3282
$TTS_{S4'}$	500	.4788	.5626	.2288	50	.4976	.5940	.2429	50	.6207	.8040	.3554
$TTS_{S4''}$	500	.3780	.4224	.1644	50	.4393	.4860	.2065	50	.5812	.7680	.3222
KIS_{S1}	500	.1334	.1044	.0314	50	.2836	.2040	.0813	50	.4087	.4400	.1492
KIS_{S2}	500	.3969	.3972	.1615	50	.5096	.5400	.2535	50	.5988	.7460	.3429
$KIS_{S2'}$	500	.5114	.5666	.2507	50	.5474	.6220	.2870	50	.6336	.7980	.3762
KIS_{S3}	500	.5598	.6336	.3009	50	.2836	.2040	.0813	50	.6907	.8620	.4299
$KIS_{S3'}$	500	.5941	.6882	.3285	50	.5096	.5400	.2535	50	.6922	.8620	.4337
KIS_{S4}	500	.5216	.5976	.2604	50	.5146	.5960	.2630	50	.6461	.8200	.3902
$KIS_{S4'}$	500	.5008	.5888	.2416	50	.5033	.5980	.2400	50	.6269	.8080	.3703
$KIS_{S4''}$	500	.4859	.5584	.2293	50	.5191	.6020	.2644	50	.6401	.8360	.3781

queries stays below that of real queries. For instance, the average nDCG scores of the UQV queries range between 0.3787 and 0.4980, whereas the maximum score of the $TTS_{S1\text{-}S3'}$ queries is 0.3499 and the nDCG scores of $KIS_{S2'\text{-}S3'}$ lie above those of the UQV. Similarly, the nDCG scores averaged over the first UQV queries reach 0.3979 at a minimum, whereas the maximum score of the $TTS_{S1\text{-}S3'}$ queries is 0.3895. When averaging over the best queries, most nDCG scores of TTS fall into the range of real queries, but there is also a higher probability of finding a good-performing query since more TTS than UQV queries are available. Except for single-term queries (S1), all KIS scores outperform the UQV queries when averaging over the best queries. With regard to the simulated queries based on the $TTS_{S4\text{-}S4''}$ approach, most of the nDCG, P@10, and AP scores fall into the range of the real queries, while $KIS_{S4\text{-}S4''}$ queries outperform UQV queries. Thus, we have a specific focus on $TTS_{S4\text{-}S4''}$.

Figure 6.2 shows the $RMSE_{nDCG}$ between queries with conventional query modification strategies ($TTS_{S1\text{-}S3'}$/$KIS_{S1\text{-}S3'}$) and the real queries (UQV). Especially for

the TTS queries, the strategy S2′ has the lowest RMSE scores and acceptable scores for the KIS queries. In the following experiments, we primarily use the strategy S2′ for both the TTS and KIS queries since their term length complies with the typical length of real queries [201] and they serve as estimates of lower and upper bound retrieval effectiveness.

Additionally, we evaluate the $\text{TTS}_{\text{S4-S4}''}$ queries with the help of the RMSE and simulations in reference to the ten queries per topic of UQV_5. For each query reformulation, 100 documents are retrieved and contribute to the final ranking list of a topic if a previous query has not retrieved them. Suppose the second query reformulation retrieves 80 previously unseen documents. In that case, the queries would be appended to the first 100 results and placed a rank 101 to 180 in the final ranking list. Figure 6.3 shows the RMSE instantiated with P@1000, nDCG, and AP, along with an increasing number of documents retrieved with ten queries. For all measures, the error increases when more documents per query are retrieved. With regard to P@1000 and nDCG, the $\text{TTS}_{\text{S2}'}$ and $\text{KIS}_{\text{S2}'}$ queries have the largest error, while $\text{KIS}_{\text{S2}'}$ has a lower RMSE_{AP} than $\text{TTS}_{\text{S4}'}$. For all measures, the $\text{TTS}_{\text{S4-S4}''}$ queries have the lowest error, which means they are the best approximation of UQV_5 among all analyzed query simulations.

Finally, we compare the topic score distributions of the simulated queries and all UQV queries by paired t-tests.[2] Since some users formulated no more than one query per topic, we limit our evaluations to the first query of each simulator. It means that each of the p-values shown in Figure 6.4 is determined by t-tests with nDCG score distributions that result from 50 UQV and 50 simulated queries. The $\text{TTS}_{\text{S2}'}$ queries have the highest p-values when compared with $\text{UQV}_{\{2,3,8\}}$. These results align with the ARP scores reported in Table 6.3. The nDCG scores of UQV_2 (0.4096), UQV_3 (0.3979), and UQV_8 (0.4046) are the most similar to the nDCG score of TTS_{S3} (0.3895) in comparison to other simulators. In contrast, the p-values of $\text{KIS}_{\text{S2}'}$ queries are low for all UQV queries, which complies with the ARP scores in Table 6.3. The $\text{KIS}_{\text{S2}'}$ scores averaged over the first queries are substantially higher compared to the UQV scores (e.g., $\text{nDCG}(\text{KIS}_{\text{S2}'})=0.5474$ compared to the best UQV query with $\text{nDCG}(\text{UQV}_7)=0.4980$). The $\text{UQV}_{\{1,4,5,6,8\}}$ queries have comparably higher p-values with the $\text{TTS}_{\{\text{S4,S4}''\}}$ queries which align with similar ARP scores. Interestingly, the t-test with UQV_7 and $\text{TTS}_{\text{S4}'}$ results in the highest overall p-value of 0.9901 and similarly high p-values with $\text{KIS}_{\text{S4-S4}''}$. This circumstance lets us assume that the corresponding user of the UQV_7 queries diverged from the terms in the topic texts and had some prior knowledge about adequate queries for at least some of the topics. In sum, not only can the ARP be reproduced with the simulated $\text{TTS}_{\text{S4-S4}''}$ and $\text{KIS}_{\text{S4-S4}''}$ queries, but also statistical properties of the topic score distributions.

6.4.2 Shared Task Utility

Regarding **RQ4**, we validate to what degree the simulated queries reproduce properties of the real queries in several ways. First, we evaluate if the simulated queries can preserve the relative system orderings. To be consistent with Huurnik et al., we evaluate five systems with the same parameterizations ($\mu = 50, 250, 500, 1250, 2500, 5000$)

[2] Applying the Bonferroni correction adjusts the alpha level to $\alpha = \frac{0.05}{64} \approx 0.0008$ (considering eight users and eight query simulators for an alpha level of 0.05).

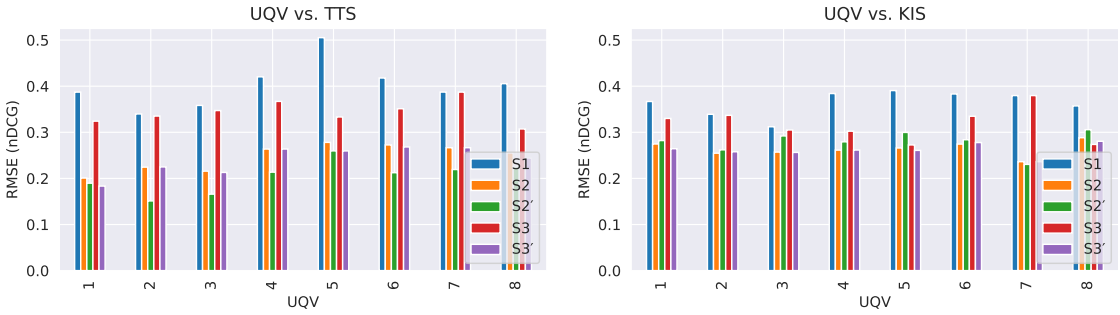

Figure 6.2: RMSE between the topic scores resulting from the simulated and real UQV queries. The left plot shows the error of the $\text{TTS}_{\text{S1-S3}'}$ queries, and the right plot shows the error of the $\text{KIS}_{\text{S1-S3}'}$ queries regarding queries made by eight users.

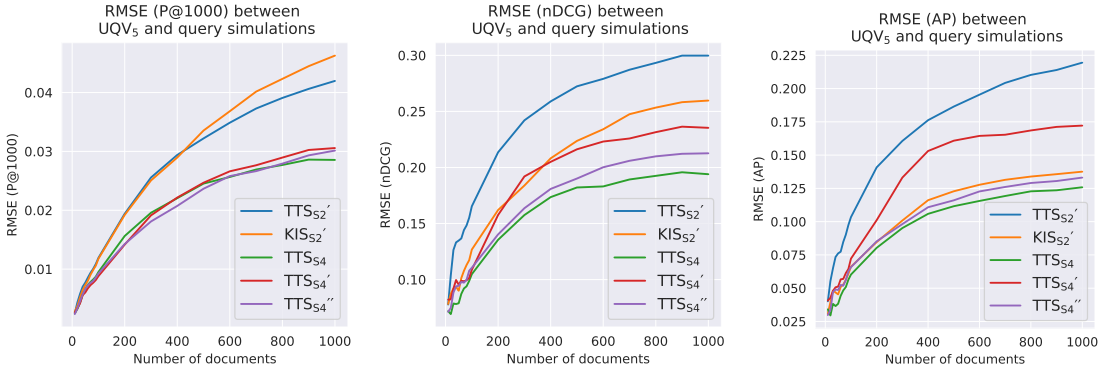

Figure 6.3: RMSE instantiated with P@1000, nDCG, and AP over an increasing number of documents per query w.r.t. the fifth user in the dataset (UQV_5).

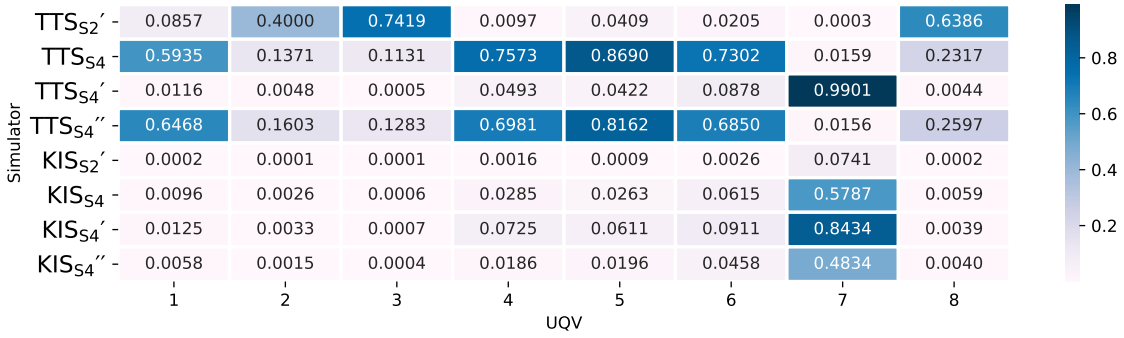

Figure 6.4: p-values of paired t-tests between UQV and simulated queries. The corresponding topic scores are based on nDCG for the first query of each topic that was generated by a simulator or formulated by one of the eight users.

of the Query Likelihood Model with Dirichlet Smoothing (QLD) [439]. However, other retrieval methods and variations could also be reasonable, e.g., different interpolation weights between a reasonable and inferior ranking. For each query formulation q_i, we determine the correlation by Kendall's τ averaged over all topics (cf. Figure 6.5) in comparison to the UQV_5 queries. The $\text{TTS}_{\text{S2}'}$ queries do not preserve the relative system ordering. Especially for the first five query reformulations, there is a low correlation with the relative system orderings of the real queries. Interestingly, the $\text{KIS}_{\text{S2}'}$ queries result in acceptable Kendall's τ scores [409], while

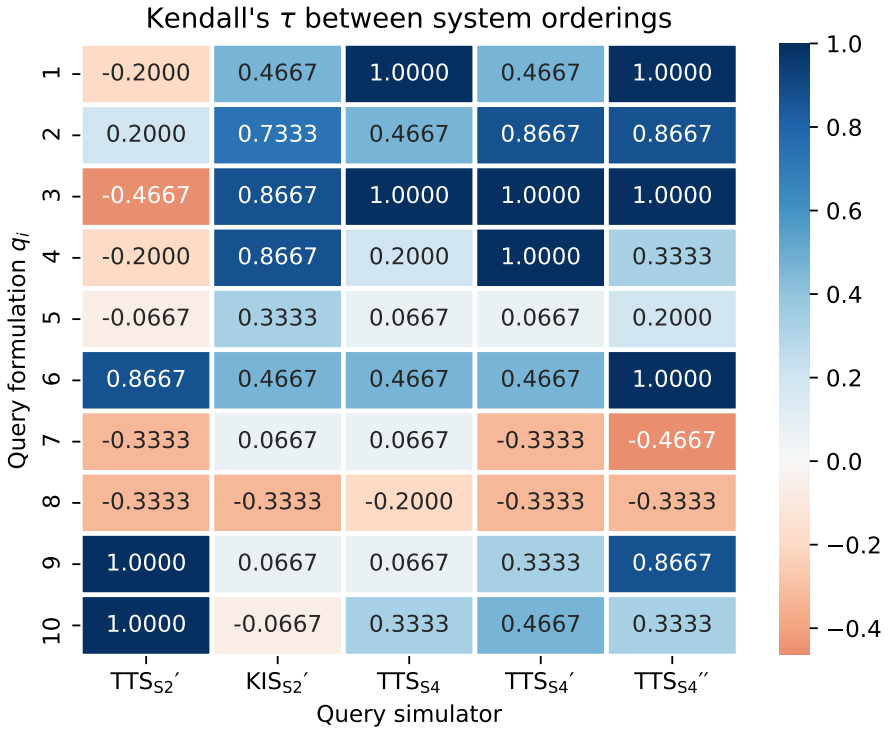

Figure 6.5: Kendall's τ between the system rankings resulting from the i-th query of the simulators in reference to the system ranking of the real user (UQV_5).

the scores beyond the sixth query formulation show low correlations. Similarly, the $TTS_{S4\text{-}S4''}$ queries correlate with the system orderings of UQV_5 queries fairly well, even reaching the maximum score of 1.0. Beyond the sixth query reformulation, the correlation falls off. While it is out of this study's scope to reach any definitive conclusions, we assume that this is related to query drifts - an issue that is also known from term expansions as part of pseudo-relevance feedback [106, 352].

6.4.3 Effort and Effect

Since most of the experiments validate single queries only, we simulate search sessions and evaluate these by sDCG (instantiated with b=2, bq=4). We compare sessions with 3, 5, or 10 queries and an increasing number of documents per query. Figure 6.6 compares the queries of UQV_5 (made by a single user [42]) to ten simulated queries of $TTS_{S2'}$, $KIS_{S2'}$, and $TTS_{S4\text{-}S4''}$.

As expected, the cumulative gain increases faster when more queries per session are used. For instance, when using five instead of three queries, the results of two additional queries are included in the session, i.e., the first three queries are the same, and additional queries can potentially contribute more relevant information. Thus, users can either formulate more queries or look at more documents to increase their information and knowledge gain.

Likewise, the $TTS_{S2'}$ and $KIS_{S2'}$ queries deliver lower and upper bound limits, respectively. In between, there are the cumulative gains by the UQV_5 and $TTS_{S4\text{-}S4''}$ queries. These results show that it is possible to fine-tune and reproduce the cumulative gain close to that of real queries, in this particular case with $TTS_{S4''}$.

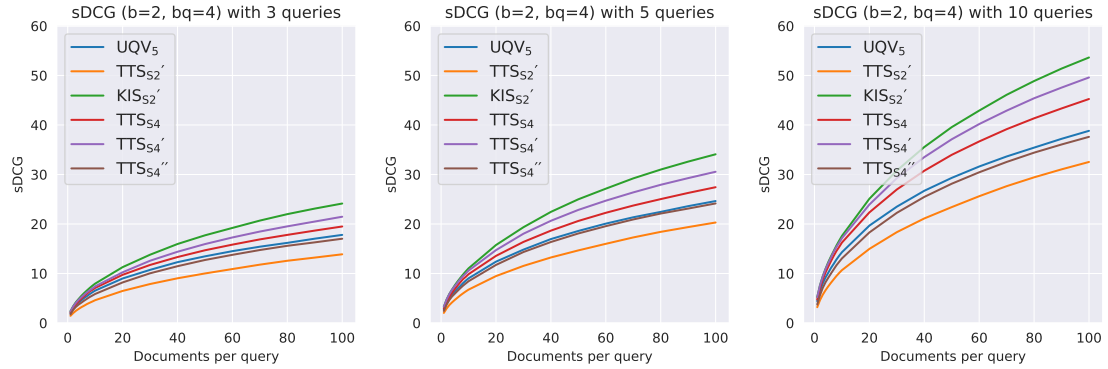

Figure 6.6: Simulations with 3, 5, or 10 queries per session over an increasing number of documents per query evaluated by sDCG (instantiated with $b = 2$ and $bq = 4$).

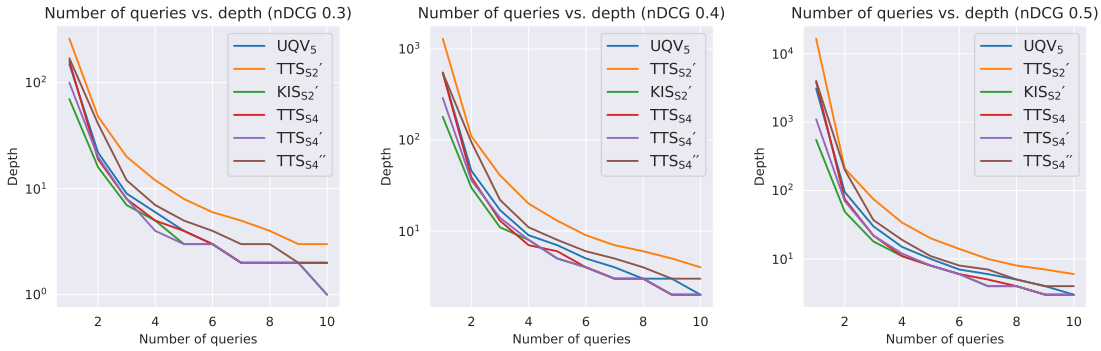

Figure 6.7: Number of queries vs. browsing depth: the plots show isoquants with fixed nDCG levels of 0.3, 0.4, and 0.5.

Table 6.4: MSLE between the isoquants for pre-defined nDCG levels (cf. Figure 6.7).

nDCG	0.3	0.4	0.5
$\text{TTS}_{\text{S2}'}$	0.3371	0.4279	0.6568
$\text{KIS}_{\text{S2}'}$	0.0949	0.1837	0.3987
TTS_{S4}	0.0059	0.0323	0.0444
$\text{TTS}_{\text{S4}'}$	0.0509	0.0758	0.1550
$\text{TTS}_{\text{S4}''}$	0.0713	0.0807	0.0791

Figure 6.7 shows the isoquants and illustrates how many documents have to be examined by a simulated user to reach pre-defined levels of nDCG (0.3, 0.4, 0.5). More queries compensate browsing depth, and as expected, the least documents have to be examined with $\text{KIS}_{\text{S2}'}$ queries and the most with $\text{TTS}_{\text{S2}'}$ queries. The $\text{TTS}_{\text{S2}'}$ isoquants lie above the others, which the poorer retrieval effectiveness can explain (cf. Subsection 6.4.1). As also shown by the MSLE in Table 6.4, the TTS_{S4} isoquants have the lowest error for all values of nDCG. Overall, the results show a better approximation of the UQV_5 isoquants with the $\text{TTS}_{\text{S4-S4}''}$ strategies and that it is possible to reproduce economic properties of the real queries by parameterizing the query reformulation behavior.

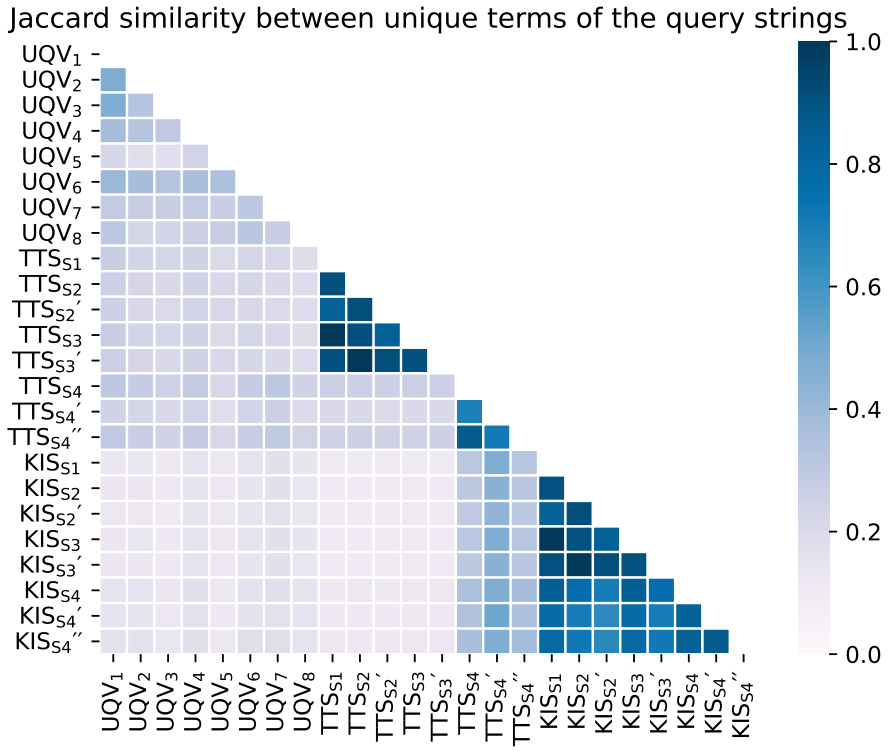

Figure 6.8: Jaccard similarity between the unique terms resulting from an equal number of query reformulations for both of the compared query generators.

6.4.4 Query Term Similarities

Figure 6.8 shows the Jaccard similarities between the sets of unique terms resulting from the query strings and is determined according to Equation 6.7. More specifically, only normalized unique terms are compared. Depending on the number of available queries for a specific topic, we include an equal number of simulated queries to avoid low Jaccard similarities when fewer than ten UQV are available. As the results show, the highest similarities are between the simulated queries. While the similarities between conventional strategies S1 to S3′ and the strategies S4 to S4″ are rather low for the TTS queries, there are higher similarities for the KIS queries. Compared to UQV and TTS queries, the KIS queries have the lowest similarities, which indicates that descriptive terms of relevant documents are very different from those used in real queries and the topic texts. Interestingly, the $\text{UQV}_{\{2,3,8\}}$ queries do not have a remarkably high Jaccard similarity with $\text{TTS}_{\text{S2}'}$ queries, despite the high p-values that are shown in Figure 6.4. This shows that it is possible to simulate UQV with different query terms than in the real queries but with comparable statistical properties as indicated by the p-values even with the rather naive approach of $\text{TTS}_{\text{S2}'}$. There are slightly higher similarities between KIS queries and the $\text{TTS}_{\text{S4-S4}''}$ queries. In particular, there is a higher similarity between $\text{TTS}_{\text{S4}'}$ and the KIS queries since the simulator is parameterized to diverge from the topic terms. Overall, we conclude that the analyzed simulation methods do not result in query strings that exactly match the terms of real queries in the UQV dataset by Benham and Culpepper [41].

6.5 Replicability Experiments

This section addresses the question of how well the introduced approaches for the UQV simulations generalize with another dataset. As the simulated queries in the earlier experiments are derived from the same test collection that was also used to determine the rankings, it is fair to criticize that the evaluations do not provide any insights about the generalizability of the introduced approach. To this end, we analyze if the experimental results hold if they are based on another test collection.

As the simulated queries are made for the Core17 topics, they can only be evaluated if the other test collection also has relevance judgments for these topics. As already pointed out in Chapter 5, the Robust04 and Core17 test collection have an overlap of 50 topics as for Core17 existing topics were reused. It means that we can evaluate 50 out of the 249 topics of Robust04 for our simulated queries and can apply a cross-collection query simulation approach that simulates queries based on a source collection to rank documents of the target collection.

In Appendix C, Tables C.1 and C.2, as well as the Figures C.1 to C.5, replicate the experimental outcomes of the previous sections. Overall, we consider most of the experiments to be successfully replicated, which indicates that the query simulations seem to be viable UQV that can be reused in other contexts, for instance, as part of rank fusion approaches or as part of the pooling process to make the document pool more diverse when existing topics are reused for constructing a new test collection.

Table C.1 replicates Table 6.3 and confirms that the approach introduced in Subsection 6.2.3 delivers better approximations of the retrieval effectiveness of real UQV indeed. Similarly, Figure C.1, which replicates Figure 6.2, shows an overall lower RMSE for the S2 strategy and confirms the outcomes made in Subsection 6.4.1. Figure C.2 replicates Figure 6.3 and similarly shows that the queries of TTS_{S4} and $\text{TTS}_{\text{S4}''}$ also result in lower RMSE scores over an increasing amount of documents per query. Likewise, the experiment based on the p-values of paired t-tests between UQV and the simulated queries is successfully replicated in Figure C.3, confirming that the S4 strategies have a higher similarity with real UQV as in Figure 6.4.

The economic aspects of Subsection 6.4.3 are also successfully replicated. For the sDCG measure shown in Figure 6.6 and replicated by Figure C.4, the TTS and KIS with the conventional S2′ modification strategy can be used as lower- and upper-bound effectiveness estimates, respectively. Figure C.6 replicates Figure 6.7 as bar plots. In order to keep the resource use low and reduce the computation time, we excluded the evaluation of single queries as they require substantially more documents to reach a pre-defined level of cumulative gain. In addition, Table C.2 shows lower errors for the TTS_{S4} and $\text{TTS}_{\text{S4}'}$ strategies by replicating Table 6.4.

Finally, the system rankings replicated in C.5 slightly differ from those outcomes in Figure 6.5. While there is an overall higher correlation between all of the simulated and the real queries, there is no obvious preference for a simulator that better replicates the system rankings resulting from the real UQV. There are positive and sometimes strong correlations up to the sixth query formulation for the $\text{KIS}_{\text{S2}'}$, TTS_{S4}, and $\text{TTS}_{\text{S4}'}$, while the $\text{TTS}_{\text{S2}'}$ and $\text{TTS}_{\text{S4}''}$ queries already result in negative correlations before the sixth reformulation. However, the evaluation approach itself could also be improved by considering different types of systems in the ranking, i.e., comparing system rankings beyond different parameterizations of the QLD method.

6.6 Answers to the Research Questions

In the following, we summarize our main findings of the previous sections and give answers to our three research questions.

RQ3: How do real user queries relate to simulated queries made from topic texts and known-items in terms of retrieval effectiveness? It is possible to use the $\text{TTS}_{\text{S1-S3}'}$ and $\text{KIS}_{\text{S1-S3}'}$ queries, which follow conventional simulation methods, as lower and upper bound estimates between which the retrieval effectiveness of real user query variants ($\text{UQV}_{\text{1-8}}$) ranges. Simulations based on our new method ($\text{TTS}_{\text{S4}''}$) provide better approximations of real query effectiveness, and the parameterization allows the simulation of different query formulation behaviors and retrieval effectiveness better resembling real queries.

RQ4: To which degree do simulated queries reproduce real queries provided that only resources of the test collection are considered for the query simulation? Our experiments show that the simulated $\text{TTS}_{\text{S4-S4}''}$ queries reproduce the real UQV reasonably well in several regards. Beyond a similar ARP, they also reproduce statistical properties of the topic score distributions as shown by the RMSE and p-values. In addition, it is shown that the simulated queries also reproduce economic aspects of the real queries as evaluated with the sDCG experiments and the isoquants that compare tradeoffs between the number of query reformulations and the browsing depth for a fixed level of gain. Furthermore, when evaluating the shared task utility, the queries of our new parameterized simulation approach preserve the relative system orderings up to the fifth reformulation, while the correlations fall off for later reformulations. We assume that this is related to topic drifts, and further analysis in this direction is required. Finally, even though it is not the primary goal to simulate exact term matches with UQV, the analysis of the query term similarity shows that there is only a slight overlap between terms of simulated and real queries and a more dedicated approach is required to reproduce exact term matches.

RQ5: How well does the introduced query simulation method generalize in a cross-collection setting where the query simulations are based on a source collection and used to rank documents of a target collection? Our replication analysis on another test collection shows that the introduced simulation approach results in UQV simulations that have similar properties of real UQV even when the runs are based on a different test collection. In essence, we can apply a cross-collection query simulation approach for which we use Core17 as the source collection to simulate queries and rank documents of the target collection Robust04. This approach is a promising method of simulating viable UQV that can, for instance, be used as part of a pooling process when existing topics are used to compile a new IR test collection. This would make the document pool more diverse, as it was already highlighted in earlier works [269, 297].

6.7 Conclusion

This chapter presented an evaluation framework and a new method for simulated UQV. Our experiments showed that the retrieval effectiveness of real queries ranges between that of simulated queries from conventional methods based on topic texts and known-items. To better approximate user queries, we introduced a simulation

method that allows parameterizing the query reformulation behavior and thus better reproduces real queries from specific users. The UQV dataset with real user queries allowed us to validate simulated sessions to the querying behavior of real users. Unfortunately, the dataset contains only a single user who formulated ten queries for all 50 topics of the test collection. However, some evaluations did not require ten queries per topic and could be evaluated for all eight users in the dataset. While there is enough data to demonstrate the potential of our simulations, the generated queries should be validated with a larger dataset in the future. For instance, the query logs of the TREC Session track [81] or the UQV dataset by Bailey et al. [24] are suitable candidates for large-scale evaluations. Nevertheless, our replicability analysis based on another target collection showed that the query simulation method could be generalized if the test collections share the same topics.

Similar to earlier experiments in the previous chapter, the experimental setup of this chapter can also be aligned to the taxonomy introduced in Chapter 3. However, in contrast to the earlier system-oriented experiments, the focus of this chapter was different variants of the users' queries. Obviously, the conventional PRIMAD components are insufficient to describe the experimental setup in its entirety, and a user component is required to describe the experiments sufficiently. In this regard, most of the evaluations can be denoted by PRIMAD-U', where the system-oriented components mainly stay fixed, and the variation of the user behavior is analyzed. In addition, the replicability experiments can be denoted by PRIMAD'-U', i.e., there was a change in the underlying test collection.

Simulating users with domain knowledge required a test collection with editorial relevance labels in our setting. Consequently, the setup was still closely aligned to Cranfield-style experiments, which gave us complete control over the six conventional PRIMAD components, i.e., all these components were defined and implemented by us, and they are fully transparent and reproducible. Similarly, it was possible to define different user profiles in a controlled manner. On the one hand, the user variation (U') was based on the strategy of how reformulations were made. On the other hand, different states of knowledge could be defined by controlling the underlying vocabulary of the simulated users. For example, better domain knowledge was based on key terms of relevant documents. The other categories of the user taxonomy (cf. Subsection 3.1.7) were not analyzed or implicitly included in the experiments. For instance, it was assumed that every search result was clicked. Furthermore, only some evaluations implied that the simulated users did not click on earlier seen documents or stopped after a certain number of documents were seen.

One important limitation of the experimental setup is the evaluation of unjudged documents in the rankings. As some UQV were not part of the original pooling process, they might bring up many unjudged documents that were treated as non-relevant in our evaluations. Future work should analyze to which extent this circumstance has an influence on the experimental results and how it affects our overall conclusions. Another limitation, which is specific to the overall simulation approach, is the exclusion of relevance feedback from search results. Users normally include terms of documents or snippets they consider relevant in their query reformulations [118, 371]. We leave it for future work to complement and analyze simulations in this regard. Likewise, the experiments neglect click simulations that are analyzed in the following chapter.

Chapter 7

Click-Based System Evaluations

In the previous chapter, we looked at simulated user interactions in the form of queries, which express the user's information need and serve as the text-based input to the retrieval system. In this chapter, we look at another kind of simulated user behavior: the perception of relevance and the interaction with system outputs, i.e., retrieved result lists. Particularly, we analyze how click models can be used to reproduce system rankings based on the relative effectiveness, and we contribute **click model-based evaluations (C8)** of IR experiments.

As an alternative to explicit editorial relevance labels, click signals are a more implicit type of feedback, providing a cost-efficient but different and sometimes biased perspective on relevance. Having enough user interaction data available, click models, which embed implicit user models based on pre-defined rules, can be parameterized from historical sessions to simulate click interactions. However, it is still little studied how click models can be validated for reliable user simulations when click data is available in moderate amounts.

To this end, our experiments compare different click models and their reliability as more session log data becomes available. We ground our methodology on the click model's estimates about a system ranking compared to a reference ranking for which the relative effectiveness is known. Specifically, we compare two types of different system rankings, one including lexical-based systems and another based on interpolated systems with different weights. In addition, we compare the Document-Based Click-Through Rate Model (DCTR) to the Dependent Click Model (DCM) and Simplified Dynamic Bayesian Network Model (SDBN), which embed the continuation probability and the notion of satisfying clicks.

The retrieval effectiveness of the systems is either determined by log-likelihood or the outcomes of simulated interleaving experiments. In both cases, we determine the correlation of the click model's estimates to a reference ranking by Kendall's τ. More precisely, we give answers to the following research questions:

RQ6 *Can click models reproduce system rankings?*

RQ7 *Do continuation and satisfaction probabilities in click models improve the simulation quality?*

RQ8 *How does the type of system ranking impact the outcomes of simulated interleaving experiments?*

Our results show that simple click models (based on the attractiveness assumption, as is the case for DCTR) can reliably determine the relative system effectiveness with already 20 logged sessions for 50 queries. In contrast, more complex click models like DCM and SDBN require more session data for reliable estimates of the effectiveness, but they are a better choice in simulated interleaving experiments when enough session data is available. While it is easier for click models to distinguish between more diverse systems, it is harder to reproduce the system ranking based on the same retrieval algorithm with different interpolation weights.

The remainder of this chapter is structured as follows. Section 7.1 motivates our analysis and covers the related work. Section 7.2 outlines the methodology and the experimental setup, whereas Section 7.3 presents the experimental evaluations, and Section 7.4 gives answers to our research questions. Finally, Section 7.5 concludes this chapter.

7.1 Motivation

One of the primary goals in IR evaluation is to find the best-performing system, i.e., to identify the ranking of retrieval systems based on effectiveness, to which we refer as the *system ranking* in the following. As already outlined in Chapter 2, the corresponding evaluation benchmarks are conducted according to the Cranfield paradigm [95], for which the creation of the underlying test collections comes at a high cost and is usually only feasible as part of larger community efforts like shared tasks at CENTRE, NTCIR, or TREC [355].

A completely different approach to collecting relevance feedback for IR systems is made possible by online experimentation [186, 232]. In this case, user interaction feedback is used to estimate the relevance of the search results. Large-scale web search companies can rely on an abundance of such data but cannot share it due to privacy concerns and business interests [102]. For this reason, few datasets covering the document collection, user interaction data, and the corresponding SERP exist.

In order to make experimental evaluations in small- to mid-scale user data environments possible, the living lab paradigm, which is the focus of Chapter 8, offers a viable solution. As part of this paradigm, small and domain-specific search services open their infrastructures for researchers who are able to evaluate their IR systems in online experiments with user interaction data.

Usually, living lab experiments are based on interleavings. The general idea is to combine ranking lists of two or more retrieval systems, show the interleaved ranking to users, and let them decide on the better-performing system by their click decision based on their relative preference. While there is a need to validate system-oriented experiments in the real world, the corresponding experiments risk harming the user experience. Thus, it is a desideratum to keep the online time of experimental systems short while having insights into the systems' usefulness.

User interactions like clicks are alternative relevance signals that could be used for estimating the system performance from a different perspective. With enough click logs available, click models can be parameterized and used afterward to simulate user interactions. When evaluating highly experimental systems, these click models can potentially replace real users in living labs. For this reason, it is highly interesting to know how much log data is required to use the click model in a reliable way.

We propose an evaluation approach in which the click model has to decide about the relative system effectiveness, i.e., the system ranking. According to this method, the click models are provided with a *reference system ranking*, for which the relative system performance is known in advance with high confidence. Based on its click decisions and the simulated click data, the model produces a *click model system ranking*, which can be compared to the reference system ranking. If the click model returns the correct system ranking, it can be considered a suitable user simulator that generates meaningful click data.

In contrast to explicit editorial relevance judgments of test collections, click signals, or user interactions in general, are a more implicit form of relevance feedback [206], which is often used to improve the quality of search results [1]. In general, it is controversially discussed how user interactions like clicks can reflect topical relevance. While several studies suggest that improvements in the measured system effectiveness do not directly translate into better user effectiveness [179, 214, 396, 397, 398, 399], some works showed that user and system metrics correlate under certain constraints [86, 204, 206, 356, 444].

We see our contributions aside from these discussions. We acknowledge that the decisions behind clicking on a search result and annotating a document with a positive relevance label are fundamentally different. While click decisions have to be seen in the context of their ranking position in the SERP, relevance annotators make judgments for every document in the pool, which means that it is not part of the annotation process to select a particular document from a SERP. Furthermore, clicks are often based on the attractiveness of the snippets, while annotators decide about the relevance after having screened the entire document. Judging the document's relevance by the snippet impacts the relevance, e.g., as shown by Turpin et al. [400], who included summaries of documents into the judgments process and show deviations of system rankings.

Chen et al. [86] investigated the correlation between user satisfaction either based on offline metrics or interaction signals (including clicks) and showed that both reflect user satisfaction but from different perspectives. Similarly, we think that it is beyond our contributions' scope to draw conclusions about how clicks correlate with topical relevance judgments. Instead, we see click-based evaluations as an alternative to the conventional Cranfield paradigm or as a proxy when topical relevance judgments are unavailable or it is not feasible to make them. Especially domain-specific search services, unlike large web search engine providers, cannot afford to create dedicated test collections with editorial relevance labels but also have lower user traffic rates. From an economical business perspective, it is, furthermore, of interest to reduce the online time of user experiments in order to increase the rate at which new experiments can be conducted [116, 224].

It is common practice to reuse historical session logs to evaluate new ranking methods before exposing them to real users [244], either to avoid harming the user experience or to reduce online time in order to increase the rate at which new experiments can be conducted [116, 224]. Once candidate systems are identified, they can be deployed in interleaving experiments like it is often done in living lab environments [155, 200, 361, 366] and as an alternative to A/B experiments, which only deliver meaningful results with large amounts of user data and which are in the focus of the next Chapter 8. The general idea is to combine ranking lists of two or more retrieval systems and let users decide on the better-performing system by their click de-

cisions based on relative preference. There exist different interleaving strategies like probabilistic interleaving [187], multileaving [367], preference-based balanced [175], or temporal interleaving [334] but the Team Draft Interleaving (TDI) [336] is one of the more common methods that is also studied in our experiments.

While earlier click models mainly differ by the pre-defined rules that make assumptions about the underlying user behavior [88, 161], several improved models were introduced, accounting for clicks on multiple result pages, and aggregated search [89, 90], embedding time awareness by accounting for dwell times and timestamps between click sequences [259], or omitting pre-defined rules by replacing them with neural vector states learned from user logs [55], or embedding global and local click models into a framework for better personalization [440]. Click models can be distinguished by the parameter estimation wich is either done by the maximum likelihood estimation (MLE) or the expectation-maximization (EM) algorithm, which has been improved for more efficiency [223] and online retraining [280]. Suppose both clicks and editorial relevance judgments are available. In that case, it is possible to turn click models into information retrieval metrics [91] or to make new relevance labels for previously unjudged documents [82, 315].

The quality of click models is often evaluated by the log-likelihood and perplexity [274], but also other reliability measures exist [278]. In previous work, click models have mainly been evaluated on private web search datasets, e.g., from Yahoo! [80,245,315,326] or Yandex [89,161], in which the SERPs were anonymized, and the underlying web corpus was not provided or under closure. To our knowledge, we are the first to evaluate simulated interleaving experiments with a completely open and transparent experimental setup.

7.2 Methodology and Evaluation Setup

In this section, we describe the system rankings (and the constituting systems) that have to be estimated by the click models (cf. Subsection 7.2.1). Second, we recapture and compare the three click models(cf. Subsection 7.2.2). Third, we describe our experimental setup, including the dataset (cf. Subsection 7.2.3) and the evaluation measures (cf. Subsection 7.2.4). Finally, we provide implementation and hardware details (cf. Subsection 7.2.5).

7.2.1 Experimental Systems

In our experiments, we include two types of system rankings, and selecting them is motivated by the Tester-based approach by Labhishetty and Zhai [238, 239]. According to them, a user simulator, or the click model in our case, can be validated by how well it can distinguish the retrieval effectiveness of methods for which we know the relative system effectiveness with high confidence or based on heuristics. For instance, by experience, we know that BM25 is more effective than ranking documents by the term frequency. In this regard, the first system ranking is based on **Lexical Retrieval Methods (LRM)** and is defined by:

$$\text{DFR}\chi^2 \succ \text{BM25} \succ \text{Tf} \succ \text{Dl} \succ \text{Null}.$$

More precisely, the LRM system ranking comprises the following five methods (in decreasing order of hypothesized effectiveness), including (1) the DFR χ^2 model [7],

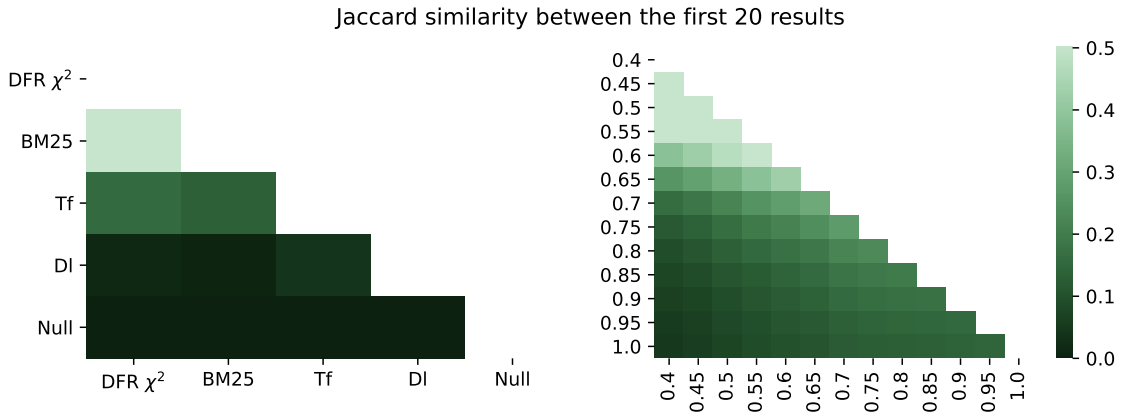

Figure 7.1: Jaccard similarity between the first 20 documents of the systems in the **LRM** (left) and **IRM** (right) rankings averaged over the top 50 queries in TripClick.

which is a (free from parameters) DFR method based on Pearson's χ^2 divergence, (2) the BM25 [345] method, (3) the term frequency (Tf) of the query terms in the document, (4) the query-agnostic method based on the document length (Dl), and (5) a method that assigns score values of zero (Null).

In contrast, the second system ranking is composed of **Interpolated Retrieval Methods (IRM)** based on combinations of a reasonable and a less effective retrieval method, giving more control over the effectiveness by weighting the influence of the less effective retrieval method. In our experiments, we combine the DFR ranking method with the ranking criterion based on the document length (Dl) as follows:

$$\text{score}(d, q) = \rho \cdot \text{score}_{\text{Dl}}(d, q) + (1 - \rho) \cdot \text{score}_{\text{DFR}}(d, q). \tag{7.1}$$

We set $\rho = \{0.4, 0.45, ..., 1.0\}$ and exclude interpolations with $\rho < 0.4$ to cover a similar score range of the Jaccard similarity for the LRM and IRM rankings as shown in Figure 7.1. By increasing ρ, we deteriorate the ranking results systematically but more subtly, which better simulates incremental and less invasive changes to an existing search platform in an online experiment. In total, the IRM ranking covers 13 different systems and is defined by:

$$\text{IRM}_{\rho=0.4} \succ \text{IRM}_{\rho=0.45} \succ \cdots \succ \text{IRM}_{\rho=1.0}.$$

When comparing the LRM and IRM rankings, the LRM ranking has more diverse document rankings, as shown in Figure 7.1. The heatmaps compare the first 20 results of the document rankings for the 50 most frequent queries of the dataset described in Subsection 7.2.3 between the combinations of the different systems by the Jaccard similarity. Except for the comparison of DFR and BM25, most of the LRM combinations are quite dissimilar. In comparison, the IRM systems with different interpolation weights cover a similar score range. However, they have a more gradual transition of the Jaccard similarity over the different combinations of weight pairs. The LRM ranking includes fewer but has more distinct systems. In contrast, the IRM ranking is based on more similar document rankings but also more systems, which means that changing the rank position of a single system would result in less severe changes in Kendall's τ as compared to changes in the LRM ranking.

7.2.2 Click Models

All of the analyzed click models are based on probabilistic modeling of the underlying user behavior [88]. All of them can only estimate the click probability of query-document pairs that were available during the parameter optimization. For all three click models, the parameters are derived from observable variables, e.g., via the MLE algorithm. Given a document ranking, a click model estimates the probability $P(C_d = 1 \mid \mathbf{C}_{<r})$ of a click C_d on the document d considering earlier clicks $\mathbf{C}_{<r}$ before the rank r by:

$$P(C_d = 1 \mid \mathbf{C}_{<r}) = P(C_d = 1 \mid E_r = 1) \cdot P(E_r = 1) = \alpha_{dq}\varepsilon_r \tag{7.2}$$

where the probability $P(C_d = 1 \mid E_r = 1)$ depends on the probability $P(E_r)$ that the document is examined. Thus, the click probability of a document d can be decomposed into the *attractiveness* α_{dq} of the query-document pair (d, q) and the *examination* probability ε_r. For all click models, the attractiveness is given by:

$$\alpha_{dq} = \frac{1}{|\mathcal{S}_{dq}|} \sum_{s \in \mathcal{S}_{dq}} c_d^{(s)}. \tag{7.3}$$

DCTR [104] determines the click probability by the ratio of clicks on a document d and how often it has been shown to users for a query q, i.e., α_{dq} is determined over all available sessions where q and d occur. The examination probability of DCTR for the document at the next rank $(r + 1)$ is defined as:

$$\varepsilon_{r+1} = 1. \tag{7.4}$$

Consequently, DCTR does not consider the context of previously seen documents. In comparison, both click models DCM [165] and SDBN [83] extend the *cascade model* [104] and determine the attractiveness by considering sessions with documents before the last-clicked document at rank l in a particular session, assuming that the user continued the search after having clicked unsatisfying results and documents beyond l were not observed by the user. The set of sessions is defined as:

$$\mathcal{S}_{dq} = \{s_q : d \in s_q, r \leq l\}. \tag{7.5}$$

In order to account for the satisfaction of clicks, the DCM [165] introduces the continuation probability λ_r determined by the ratio between the total number of sessions with clicks at rank r that were not the last clicks in a session (denoted as $\mathcal{I}(r \neq l)$) and the total number of sessions in which rank r was logged $|\mathcal{S}_r|$. The continuation probability λ_r is defined as:

$$\lambda_r = \frac{1}{|\mathcal{S}_r|} \sum_{s \in \mathcal{S}_r} \mathcal{I}(r \neq l). \tag{7.6}$$

The examination probability ε_{r+1} of DCM is then defined as:

$$\varepsilon_{r+1} = c_r^{(s)}\lambda_r + \left(1 - c_r^{(s)}\right) \frac{(1 - \alpha_{dq})\,\varepsilon_r}{1 - \alpha_{dq}\varepsilon_r} \tag{7.7}$$

where $c_r^{(s)}$ denotes the probability of a click being observed at rank r in a session s. Similarly, the SDBN [83] model embeds the satisfaction probability by the

parameter σ_{dq} but instead, it accounts for the total number of sessions with the last clicks (denoted as $\mathcal{I}\left(r_d^{(s)} = l\right)$) in reference to the total number of sessions $\mathcal{S}'_{dq}$ in which the document d is clicked at a rank before or equal to l. The satisfaction probability σ_{dq} is defined as:

$$\sigma_{dq} = \frac{1}{\left|\mathcal{S}'_{dq}\right|} \sum_{s \in \mathcal{S}'_{dq}} \mathcal{I}\left(r^{(s)} = l\right) \tag{7.8}$$

where the corresponding set of sessions $\mathcal{S}'_{dq}$ is defined by:

$$\mathcal{S}'_{dq} = \left\{ s_q : d \in s_q, r \leq l, c_d^{(s)} = 1 \right\}. \tag{7.9}$$

The examination probability ε_{r+1} of SDBN is then defined as:

$$\varepsilon_{r+1} = c_r^{(s)} \left(1 - \sigma_{dq}\right) + \left(1 - c_r^{(s)}\right) \frac{\left(1 - \alpha_{dq}\right) \varepsilon_r}{1 - \alpha_{dq} \varepsilon_r}. \tag{7.10}$$

Table 7.1 provides a toy example with five sessions, for which we assume that the same ranking was logged for a single query q. The following illustrates how the continuation and satisfaction probabilities can be determined by the clicks represented by the filled circles. For instance, we can determine the continuation probability of the second rank λ_2 by the sessions s_1, s_3, and s_4 at which the rank r_2 was clicked. For two of these three sessions, the click at the second rank was followed by additional clicks at the lower ranks, indicating that the users continue to browse through the ranking after seeing the document at rank r_2. Accordingly, the continuation probability is determined by this ratio, i.e., $\lambda_2 = \frac{2}{3}$.

Similarly, we can determine the satisfaction probability at the second rank σ_{d_2q}. For one out of the three sessions (s_4), it was also the last click in the session. Accordingly, the satisfaction probability is determined by this ratio, i.e., $\sigma_{d_2q} = \frac{1}{3}$. Note that the continuation and satisfaction probabilities are complementary when comparing them for a single query, i.e., $\lambda_r = 1 - \sigma_{dq}$. The two click models DCM and SDBN differ if they are compared over multiple queries as the continuation probability of DCM depends only on the rank r and is determined over all queries. In contrast, σ_{dq} of SDBN is specific to the query-document pair.

Suppose no clicks at a rank have been logged. In that case, it is impossible to determine the continuation and satisfaction probabilities (cf. r_4), and as a workaround, default probabilities can be used, or it is likewise possible to estimate values from the probability distribution.

7.2.3 Dataset

For our experiments, it is a fundamental requirement to have open data. Nowadays, several datasets are available for the general research community, but a large fraction of them is not suitable for our experiments. As pointed out before, previous work about click models was done in cooperation with large web search companies like Yahoo! [80,245,315,326] or Yandex [89,161] and used entirely private or semi-public datasets. For example, a popular dataset for the training of click models was made publicly available by Yandex as part of the *Personalized Web Search Challenge* [472]. A similar dataset is publicly provided by Yahoo! as the *L18 - Anonymized Yahoo! Search Logs with Relevance Judgments* [488]. However, the web search results in

Table 7.1: Toy example of the continuation λ_r and satisfaction σ_{dq} probabilities for five logged sessions. The filled circles correspond to clicks.

r_i \ s_i	s_1	s_2	s_3	s_4	s_5	λ_r	σ_{dq}
r_1	○	○	○	●	○	$\frac{1}{1} = 1.0$	$\frac{0}{1} = 0.00$
r_2	●	○	●	●	○	$\frac{2}{3} = 0.\overline{6}$	$\frac{1}{3} = 0.\overline{3}$
r_3	○	●	●	○	●	$\frac{1}{3} = 0.\overline{3}$	$\frac{2}{3} = 0.\overline{6}$
r_4	○	○	○	○	○	-	-
r_5	●	○	○	○	●	$\frac{0}{2} = 0.0$	$\frac{2}{2} = 1.0$

both datasets are anonymized, and no document collection of the entire corpus is provided, which is critical for our experiments as we want to build a custom index and retrieval pipelines as defined above.

ORCAS [102] is a companion dataset to MSMARCO that provides click-document pairs, and both the query, as well as the document are available in a clear text version. However, the DCM and SDBN click models do not only require triples containing the query, the documents, and the corresponding clicks but also the context of other documents in the SERP that were seen but not clicked. For this reason, ORCAS is unusable for our experiments.

Instead, our experiments use the recently introduced TripClick [342] dataset of the biomedical search engine Trip. The dataset contains documents and user interaction logs covering a period of seven years, from 2013 to 2020. It was shown that the annotation coverage for the top results is low [188], and additional topical relevance judgments called TripJudge were provided [6]. As the DCM and SDBN click models also require the context of other documents in the SERP that were seen but not clicked, we can only use data logs with information about the entire SERP, which are available from 13th August 2016. Furthermore, we restrict the sessions to the 50 most frequent queries to make sure that at least 100 logged sessions are available for each query. We note that the Trip database has professional and non-professional users alike, and head queries are a very particular query type.

7.2.4 Evaluation Measures

In the following, we introduce the measures of our experimental evaluations, including the log-likelihood, the outcome of interleaving experiments, and the rank correlation measure Kendall's τ.

Log-Likelihood

Log-likelihood is a common evaluation measure of click models, and it was shown that better scores correlate with a higher fidelity of simulated clicks [274]. We determine it over a run R with $|\mathcal{Q}|$ queries and ranking length n as follows:

$$\mathcal{LL}(R) = \sum_{q \in \mathcal{Q}} \sum_{r=1}^{n} \log P\left(C_d = c_d \mid \mathbf{C}_{<r}\right) \tag{7.11}$$

where $P\left(C_d = c_d \mid \mathbf{C}_{<r}\right)$ denotes the click probability of a particular click model for a document d at rank r given the ranking of a retrieval method for a query q and the list of previous clicks $\mathbf{C}_{<r}$ before rank r of the examined document. In our experiments, we use the TripClick data logs that contain SERPs with 20 entries ($n = 20$) and $|\mathcal{Q}| \in [1, 50]$. Unlike previous work, we do not use log-likelihood to evaluate the click model itself but to distinguish between the ranking quality of retrieval systems. Assuming that a well-performing retrieval method delivers attractive rankings that result in clicks, the system maximizes the click probabilities, and thus log-likelihood, over every result in a ranking list.

Outcome of Interleaving Experiments

When simulating the interleaving experiments, we compare two systems with the help of the TDI method as introduced by Radlinski et al. [336]. The underlying idea of the interleaving algorithm is inspired by the team selection in a team-sports match. In this analogy, the systems can be seen as team captains and documents as players. Based on random selection, one of the systems goes first and contributes its top-ranked document to the interleaved ranking. Afterwards, the systems take turns contributing their next highest-ranked documents that are not part of interleaved ranking yet. Consequently, the TDI algorithm produces ranking lists that can be decomposed into two sets containing those documents D_{exp} contributed by the experimental system and those documents D_{base} of the competing baseline. An experimental system wins if it contributes the document with the highest click probability to the interleaved ranking, i.e., we determine the rank of the document with the highest click probability by:

$$r = \underset{k \in \{1,\ldots,n\}}{\arg\max} P\left(C_k \mid \mathbf{C}_{<k}\right). \tag{7.12}$$

A *win* is assigned to the experimental system if $d_r \in D_{\text{exp}}$. Otherwise, the experimental system looses, i.e., $d_r \in D_{\text{base}}$, and a *loss* is assigned. Suppose the click probabilities of the interleaving are indifferent from those of a ranking with unknown documents. In that case, the click model cannot decide on a better system, and a *tie* is assigned. Finally, the outcome is determined over multiple queries $\mathcal{Q}$ and is defined as:

$$\text{Outcome} = \frac{\text{Wins}}{\text{Wins} + \text{Losses}}. \tag{7.13}$$

A clear *winner* achieves an outcome of 1.0, whereas 0.5 means that the experimental system is on par with the baseline, and any outcome below 0.5 indicates an inferior experimental system.

Rank Correlation

As is common practice when comparing relative system rankings, we use Kendall's τ. As a rule of thumb, Voorhees considers correlations with $\tau > 0.9$ acceptable [409].

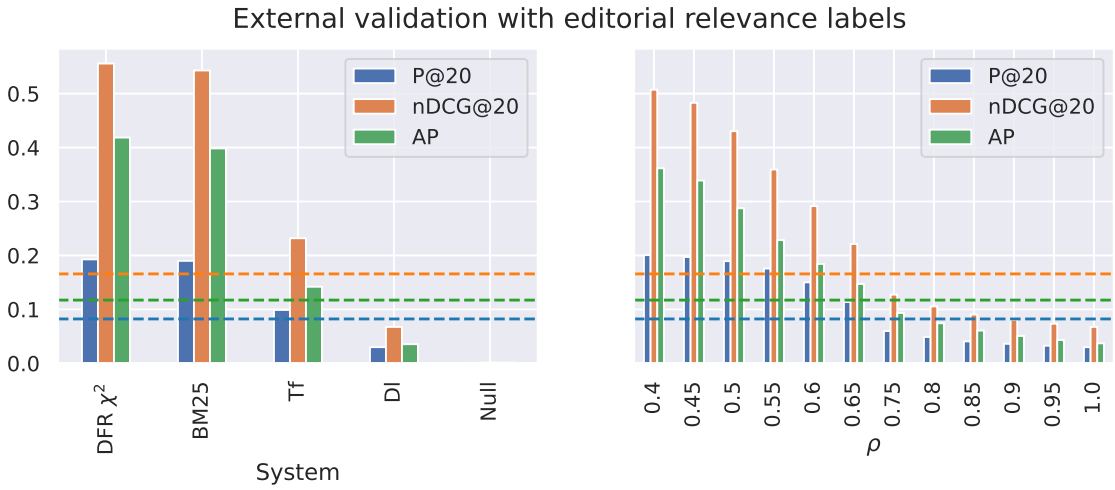

Figure 7.2: LRM (left) & IRM (right) system rankings evaluated by editorial relevance judgments. The dashed lines correspond to the baseline system.

We evaluate the system rankings resulting from the click model-based evaluations in reference to the LRM & IRM rankings, for which the relative orderings are motivated by the Tester-based approach (cf. Subsection 7.2.1). In order to strengthen the reasoning behind the hypothesized system rankings, we evaluate them with the help of editorial relevance judgments. For this purpose, we use the previously mentioned TripJudge relevance labels [6]. The results in Figure 7.2 show that the system-oriented experiment gives evidence to the hypothesized relative orderings of the system effectiveness. We can control the retrieval effectiveness for both types of system constellations by choosing an entirely different ranking method or increasing the interpolation weight towards the inferior ranking criterion. We consider these system-oriented experiments as another perspective of the system effectiveness as a form of additional validation, strengthening the reasoning behind the chosen reference system rankings. For consistency, we keep the baseline system for the interleaving experiments fixed for both types of rankings, i.e., the IRM system with $\alpha = 0.7$ that is indicated by the dashed line in the plots.

7.2.5 Implementation Details

We implement the experiments with the help of the Pyterrier retrieval toolkit [267] (the Python interface to the Java-based retrieval toolkit Terrier [314]) and the dataset library `ir_datasets` [265], which features bindings to the TripClick dataset. We filter and select the session logs with the help of the NoSQL database MongoDB. When implementing the click models, we rely on the PyClick [466] library. In addition, we provide the required parsers to ingest the session logs from our database into the PyClick framework. All experiments are run on moderate hardware, i.e., with an *Intel Xeon Gold 6144* CPU and 64 GB of RAM on *Ubuntu 18.04 LTS*.

7.3 Experimental Evaluations

In the following, we present the experimental evaluations. In order to determine the performance of click models over an increasing amount of click data and queries,

we randomly sample an increasing number of logged sessions, which are used to parameterize the click model. For each query $q \in \mathcal{Q}$, we randomly sample s sessions ten times, i.e., we let the click model adapt to the given data sample (with s sessions for $|\mathcal{Q}|$ queries) and evaluate the system rankings over ten trials.

In the first experiment in Subsection 7.3.1, the system rankings are determined by log-likelihood based on the click probabilities, whereas in the second experiment in Subsection 7.3.2, the living labs are simulated, and the system rankings are based on the outcome of the corresponding interleaving experiments. Finally, each system ranking that results from either the log-likelihood or the outcome measure is compared to the reference system rankings with the help of Kendall's τ.

7.3.1 Log-Likelihood Evaluations

We determine the log-likelihood for all combinations resulting from the two system rankings and the three click models and evaluate them over an increasing amount of click log data that is used for parameterizing the click models. Figure 7.3 shows the log-likelihood over the number of sessions with either 5 or 50 queries. Unsurprisingly, the log-likelihood increases as more sessions are used to parameterize the click models. As more click logs are available, the click models *becomes familiar* with relevant documents, and consequently, there is a higher click probability.

There are apparent differences between DCTR and the other two click models. In the case of the DCTR-based log-likelihood, the ranking order of documents is irrelevant as the click model does not account for the ranking position. Consequently, there is no rank-biased discount of the documents' attractiveness, leading to an overall higher log-likelihood of DCTR. In contrast, the document order affects the click probabilities of the DCM and SDBN click models, leading to an overall lower log-likelihood, which can be explained by the examination probabilities of these click models that are a rank-biased discount of the documents' attractiveness.

As can be seen from the LRM ranking (in the upper half of Figure 7.3), the *Null* system has a constant log-likelihood and is an estimate for lower bound performance. For the other systems, the log-likelihood increases as more sessions are considered, whereas the DFR and BM25 methods are quite distinct from the simple ranking criteria based on the term frequency (Tf) and document length (Dl). In the lower half of Figure 7.3, the IRM system rankings based on the log-likelihood align with the earlier system-oriented evaluations, i.e., the overall log-likelihood is lower (the retrieval system performs worse) when the interpolation parameter ρ gives more weight to the inferior ranking criterion.

By evaluating the log-likelihood with 50 queries, we see a steeper increase in the log-likelihood as more (possibly earlier clicked) documents are retrieved. Once enough click data is available, there are consistent click probabilities, as seen by the plateau-like shape of the log-likelihood plots with 50 queries. Any additional sessions with new click data only provide redundant relevance information and only affect the click probabilities to a negligible extent. In comparison, the log-likelihood averaged over fewer queries is noisier, as can be seen by the larger confidence intervals but increases over the sessions. By the example of the DCTR model, we see that the log-likelihood also increases as more queries are considered.

Overall, these preliminary evaluations suggest that either more queries or more sessions are required to distinguish between the single ranking systems. To this

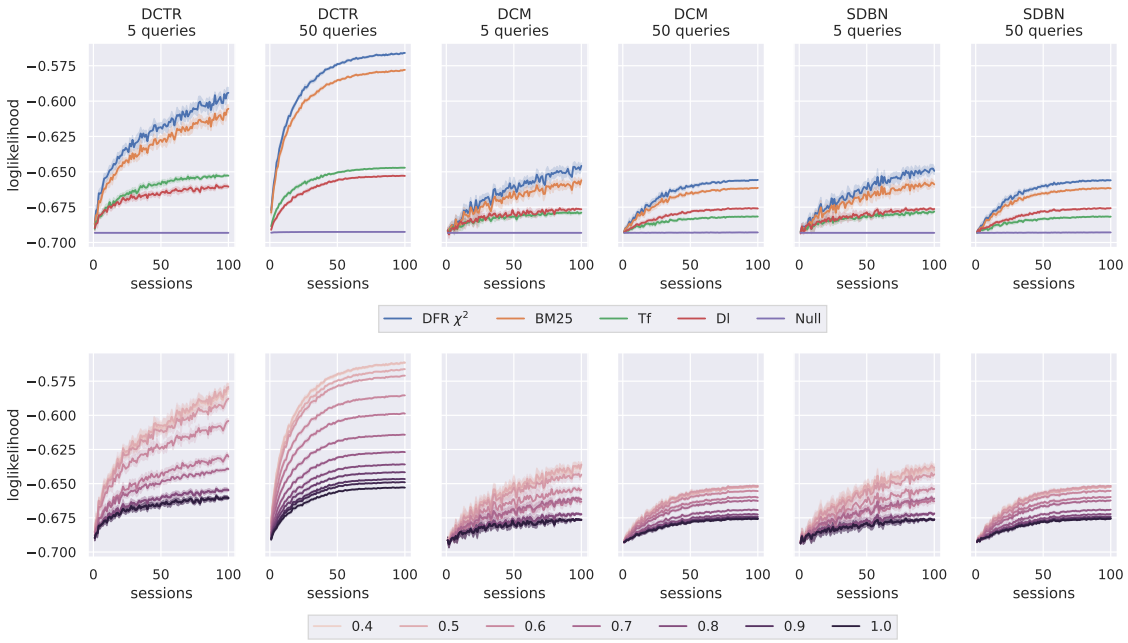

Figure 7.3: Log-likelihood of the system rankings (LRM in the first and IRM in the second row) evaluated by the three click models and compared by 5 and 50 queries.

end, we conduct a more extensive analysis with an increasing number of queries and sessions. Figure 7.4 compares Kendall's τ scores over different combinations of queries and log sessions for all three click models and the two system rankings. The heatmaps show the rank correlation in terms of Kendall's τ for the different combinations of queries (ranging from 3 to 50) and sessions (ranging from 1 to 20). The greener the corresponding patch, the higher the correlation between the reference and the click model system ranking.

The first heatmap based on DCTR and the LRM ranking shows a diagonal transition from the upper left corner to the lower right corner — the heatmap gets more greenish as more queries and sessions are used to evaluate the click model. In comparison, the IRM heatmap of the DCTR model has an overall darker appearance, which means that in comparison to the LRM ranking, less log data and queries are required to determine the correct system ranking.

Evaluating 50 queries with a DCTR model based on 20 session logs for each query is already enough to reproduce the LRM system ranking with a perfect correlation of $\tau = 1.0$. In contrast, the DCM and SDBN click models require more session logs to reliably reproduce the correct system orderings, resulting in lower correlation scores of $\tau_{\text{DCM}} = 0.4267$ and $\tau_{\text{SDBN}} = 0.5867$ on average with the same amount of queries and corresponding sessions. This can also be seen by the overall lighter heatmaps, which indicate low correlations between the system rankings.

In general, the IRM system ranking also results in higher Kendall's τ scores with fewer queries and sessions for DCM and SDBN, which suggests that it is easier for the click models to distinguish between systems that rely on the same retrieval method by the log-likelihood. We assume that the smaller document pool can explain this (cf. Figure 6.8), i.e., there are fewer document candidates by which the method can be compared, and less click data is required for meaningful parameterization.

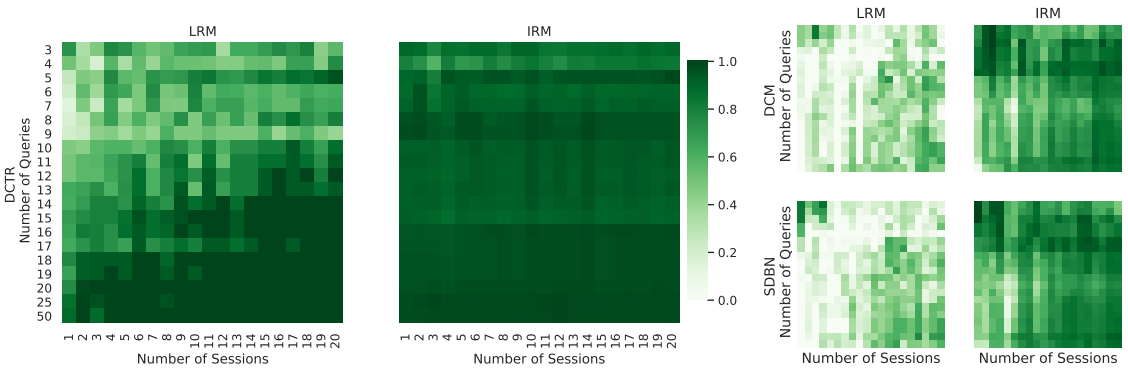

Figure 7.4: Kendall's τ of the LRM and IRM rankings for different numbers of queries and logged sessions, compared for the click models DCTR, DCM, and SDBN.

We conclude that evaluating the relative system performance by the log-likelihood is a viable solution under the assumption that good-performing systems maximize the click-through rate only by the attractiveness of the ranking list. In comparison, DCTR is more robust and results in more reliable estimations when less log data is available. For instance, the LRM system rankings result in Kendall's τ scores of 1.0 with 50 queries and click data from 20 sessions for each query, while the log-likelihood based on DCM and SDBN scores is below 0.6 when evaluated with the same amount of queries and click data. Overall, log-likelihood is lower when evaluated with the DCM and SDBN click models due to the examination probability discounting the attractiveness.

7.3.2 Simulated Interleaving Experiments

In the interleaving experiments, we determine the system ordering by the outcome measure (cf. Equation 7.13) for which the highest click probability is used as the winning criterion (cf. Equation 7.12). For each interleaving, the experimental ranking is interleaved with the baseline, which is consistent for both types of system rankings for the sake of better comparability and is set to $\text{IRM}_{\alpha=0.7}$.

Figure 7.5 compares the outcomes for 50 queries with 100 session logs over ten trials for each experiment. Most strikingly, all of the click models can reproduce the correct orderings of the LRM system ranking, whereas, for the IRM system rankings, the relative ordering cannot be reproduced. However, all click models can differentiate between systems that out- or underperform the baseline. Our analysis showed that often the *winning* queries, i.e., those queries for which the experimental system wins, directly turn into losing queries as soon as the bad ranking criterion is assigned a higher weight than that of the baseline system.

For better illustration, an in-depth analysis of the *winning* and *losing* queries is given in Figure 7.6. More specifically, the Jaccard similarity is shown for the *winning* (lower triangle) and for the *losing* (upper) queries over different interpolation weights, whereas winning and losing queries are those for which the experimental system is either assigned a *win* or a *loss*, respectively.

There are higher query similarities between those systems with an interpolation weight, either below or above the baseline system. However, there is a low overall similarity when comparing the winning/losing queries of system combinations with

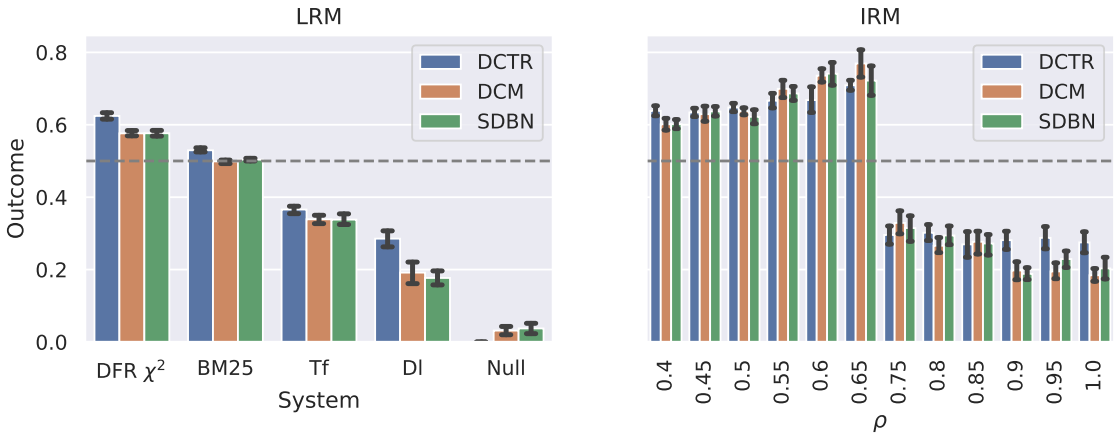

Figure 7.5: Outcome measures of interleaving experiments with click models based on 50 queries and 100 session logs. The dashed line corresponds to the baseline that is consistent for both types of system rankings.

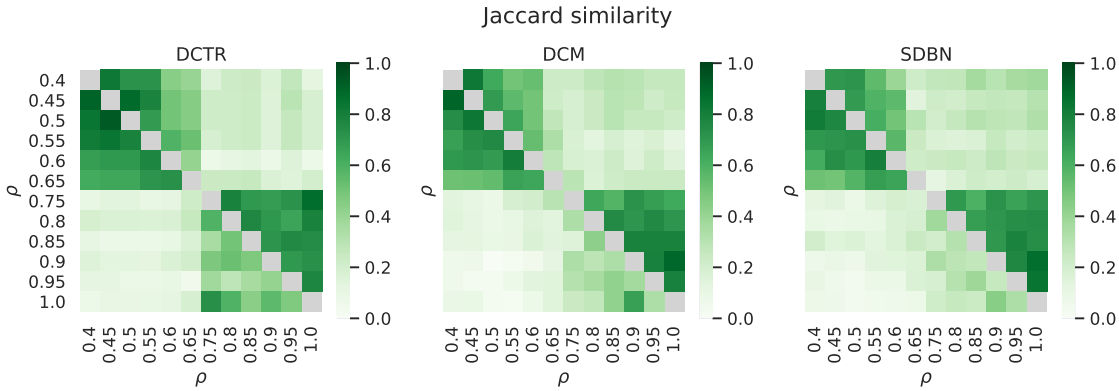

Figure 7.6: Jaccard similarity between the *winning* (lower triangle) and *losing queries* (upper triangle) of the simulated interleaving experiments.

lower and higher interpolation weights (compare the light green areas in the lower left and upper right of the heatmap). This circumstance is independent of the click model, as the three heatmaps show similar results.

Regarding the IRM system ranking, the winning queries, i.e., those queries for which the experimental system wins, turn into losing queries as soon as the bad ranking criterion is assigned a higher weight than that of the baseline system. Queries resulting in *ties* (in that case, no or an equal number of clicks are made for both interleaved systems) barely change as the click models cannot decide on a better system with unseen documents. These experimental results show that it can be problematic to compare systems with a small document pool with fewer document candidates and low click-through rates.

Finally, Figure 7.7 shows Kendall's τ of the system rankings derived from the interleaving experiments resulting from click models parameterized over an increasing number of sessions. As can be seen by the light stripes in the heatmap, it is not possible to reproduce the correct ordering of IRM systems for any of the click models. Most of the rank correlations of the IRM rankings stay below 0.6, which aligns with our earlier observations.

When comparing the click models with the LRM system rankings, we see that the DCTR model results in comparably higher correlations when less log data is available. For instance, the patches in the heatmap have a darker green when using ten or fewer session logs per query for the DCTR model. However, the DCTR experiments show that the correlation scores do not stabilize even if more sessions are used for the parameterization, and once a certain amount of log data is used for the parameterization of the click models, DCM and SDBN deliver more robust correlation scores. For a better understanding and analysis, we determine the relative error between the cumulated and the ideal Kendall's τ score as

$$\delta\tau = \frac{\Delta\tau}{\tau_{ideal}} = \frac{\tau_{ideal} - \tau_{sum}}{\tau_{ideal}} = 1 - \frac{\tau_{sum}}{\tau_{ideal}} = 1 - \frac{\sum_{s=1}^{|\mathcal{S}|} \tau_s}{\sum_{s=1}^{|\mathcal{S}|} 1} = 1 - \frac{\sum_{s=1}^{|\mathcal{S}|} \tau_s}{|\mathcal{S}|} \tag{7.14}$$

where τ_{ideal} is considered as the sum of the ideal rank correlation up to the amount of considered sessions $|\mathcal{S}|$, and *ideal* refers to a perfect rank correlation of 1. Accordingly, $\Delta\tau$ describes the difference between the actual sum of rank correlations and the ideal sum. A good performing user simulator or click model gives a low $\delta\tau$ score or minimizes it as it gets more session data for an adequate parameterization.

Figure 7.8 shows $\delta\tau$ for the click models in combination with both types of system rankings over an increasing amount of session logs. These results confirm that once enough session data is available, the DCM and SDBN click models can better distinguish between the relative system performance in these particular simulated interleaving experiments.

Regarding the LRM system ranking, there are higher errors for DCM and SDBN when only a few sessions are available, and the DCTR is a better choice when considering the lower error rates. However, it can be that with an increasing amount of click data, the error for both DCM and SDBN decreases while the error of the DCTR model evens out and does not decrease as more sessions are used for parameterizing the click models.

In comparison, it is generally more challenging for the click models to distinguish between the IRM system ranking based on interpolations. The experiments with 100 sessions result in considerably higher errors (higher $\delta\tau$ scores), but still, the DCM and SDBN give slightly better estimates than the DCTR. In this case, the $\delta\tau$ scores even out, while the scores of the DCTR still increase as more session logs become available. Similar to the earlier results, it is better to use DCTR when less log data is available. However, once enough logged clicks are available for the parameterization, the DCM and SDBN are less error-prone and more reliable.

7.4 Answers to the Research Questions

In the following, we summarize our main findings of the previous Section 7.3 and give answers to our three research questions.

RQ6: Can click models reproduce system rankings? All click models can reproduce the system rankings if enough click logs are available, which is fundamental to our proposed methodology. We defined the simulation quality by how well the click model's click probabilities can reproduce the correct system ranking that is known in advance. The simulation quality improves depending on how much session

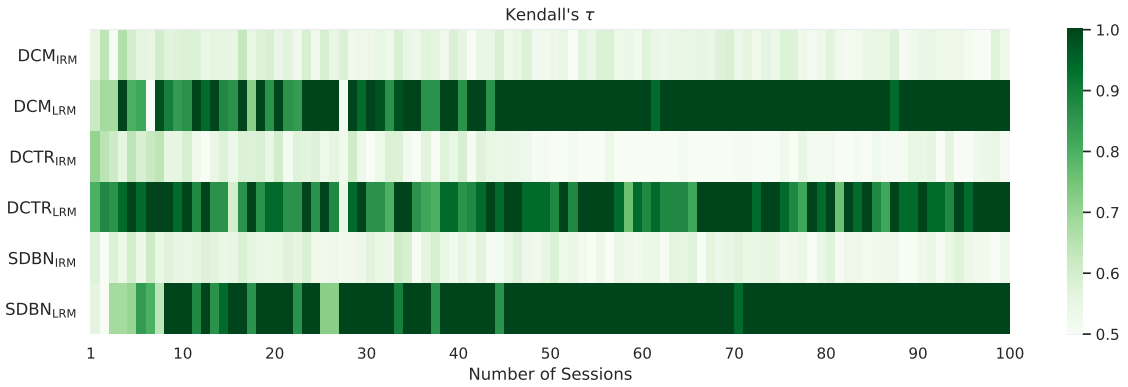

Figure 7.7: Kendall's τ of the LRM and IRM rankings based on simulated interleavings, compared for the click models DCTR, DCM, and SDBN.

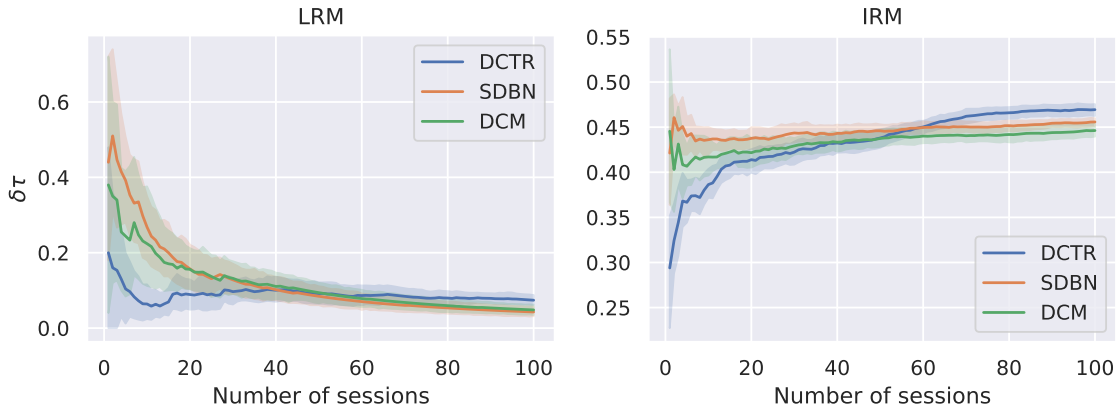

Figure 7.8: $\delta\tau$ over an increasing number of sessions for the LRM and IRM rankings based on interleavings, compared for the click models DCTR, DCM, and SDBN.

data is available to parameterize the click model. In this regard, our evaluations showed that DCTR could distinguish reliably between the LRM systems by the log-likelihood with already 20 logged sessions if 50 queries are used. In comparison, the IRM ranking can be reproduced with fewer data, which can be explained by a smaller pool of documents for which interaction data has to be logged.

RQ7: Do continuation and satisfaction probabilities in click models improve the simulation quality? Our experiments showed that using the DCM and SDBN for the log-likelihood in an interactive data-sparse setting is not recommended. Our evaluations showed that the DCM and SDBN result in overall lower scores in comparison to the DCTR model, which can be explained by the rank-biased discount of the attractiveness due to the examination probability. However, this is not critical when many session logs are available. For instance, if we can use 100 sessions per query, it is enough for adequate parameterization. However, compared to the DCTR, more than 20 sessions per query are needed to let the DCM and SDBN reproduce the correct system ranking. On the other hand, the DCM and SDBN system rankings are a better choice when simulating the interleaving like they are implemented in living labs. In this case, our experiments showed that the estimates of the LRM system ranking are more robust, and the continuation and satisfaction probabilities of DCM and SDBN can indeed improve the simulation quality.

RQ8: How does the type of system ranking impact the outcomes of simulated interleaving experiments? While all models can determine the correct ordering of the LRM system ranking reasonably well in the simulated interleaving experiments, it is impossible to reproduce the correct IRM ranking. However, one can still distinguish between better and worse-performing IRM systems and separate them from the baseline. Our analysis showed that it is generally harder to reproduce the IRM ranking as there are deciding queries that either let the IRM system win or lose against the baseline system, depending on the interpolation weight. Once the interpolation parameter gives a higher weight to the bad ranking criterion, most of the queries, which formerly let the system win against the baseline, are the deciding queries that let the system lose against the baseline. This finding is critical for search platform operators, as different parameterizations of the same retrieval method may result in measurable differences in system-oriented experiments, while they are not reproducible in click model-based simulations.

7.5 Conclusion

Instead of risking exposing poor results to real users, it is possible to evaluate experimental systems with simulated click interactions in pre-assessments. However, it is often unclear when a click model can be reliably used to simulate real user behavior by generating meaningful clicks. To this end, we introduced an evaluation approach for validating a click model's simulation quality in this chapter. Our evaluation methodology aims at letting the click model decide about the relative system performance that is known with high confidence or based on some reasonable heuristics. The click model's system ranking is compared to the reference system ranking, and the rank correlation based on Kendall's τ is an indicator of the simulation quality.

In our experiments, we compared two different types of system rankings. The first ranking was composed of different lexical retrieval methods. In contrast, the second ranking was composed of a single ranking approach with different interpolations between a reasonable and less effective retrieval method. While these retrieval methods are rather simple compared to other state-of-the-art approaches, they are reasonable candidates to validate the general plausibility of our approach.

Our experimental results have shown how the DCTR, DCM, and SDBN click models can be used in combination with the log-likelihood and the outcomes of simulated interleaving experiments for the assessments of retrieval methods and how much session data is required for reliable performance estimates. Overall, it is possible to reproduce the system rankings in simulations, confirming our methodology's general plausibility. Regarding the evaluations based on the log-likelihood, the DCTR click model is a better choice if only a few sessions are logged. For example, our experiments showed that the DCTR could perfectly reproduce the system ranking with 20 logged sessions for 50 queries, while the DCM and SDBN could not. However, as more session logs become available, the DCM and SDBN click models are equally well-suited for this type of evaluation.

This leaves the question of how the interpretation of the examination probabilities of the DCM and SDBN models is of benefit for the user simulations. For a better understanding, we simulated living lab experiments and let the click models decide about the preference for one of two competing systems in interleavings. The corresponding system rankings were based on the outcome measure and showed

that, once again, DCTR is a better choice when only a small amount of session data is available. However, as more session logs are available, the DCM and SDBN gave better, i.e., more robust, estimates about the system rankings. DCM and SDBN not only better approximate real user behavior, but they are also more reliable click models in simulated interleaving experiments.

When comparing the DCM to the SDBN model, no substantial differences in our experiments were observable. The rank-biased discount of the DCM model is determined by a rank-dependent continuation probability, which is determined over all available sessions, while the SDBN introduces an additional satisfaction probability specific to the query-document pair. We conclude that for the underlying TripClick dataset the consideration of the satisfaction probability did not make that much of a difference in comparison to the continuation probability.

Conceptually, the experiments of this chapter can be aligned with the PRIMAD-U taxonomy. More specifically, the experimental setup can be denoted by PRIM'AD-U'. Different methods (M') were used to compose system rankings evaluated by different types of user interaction behavior (U') based on click models. The queries did not define the user variation compared to the previous chapter. Instead, the query strings stayed fixed, and a more interaction-focused type of relevance was implied in the experiments. In contrast to editorial relevance judgments, the click-based relevance indicators are less explicit but better simulate how relevance would be signaled in real search sessions.

We think that click signal-based evaluations are a promising alternative when a curated test collection is unavailable. Instead, click models can evaluate the relative system performance when editorial relevance judgments are missing. For instance, click models could be used in a pre-assessment, similar to the idea of pseudo-relevance judgments [376], to identify more promising systems for online experiments. Especially for small- and mid-scale search platforms that often partnered with living labs in the past, it would be a viable solution to use click signals instead of curating a costly test collection.

Lastly, click data is biased [420]. To a certain extent, the click models address the bias that would emerge from using single clicks as relevance indicators, i.e., the probabilistic models grasp the behavior and preferences of the average user. However, there are other biases related to the click signals. For instance, there is the position or system bias introduced by the unknown production system of the Trip database that we could not remove from the session logs.

As part of the future work, it needs to be re-evaluated how the introduced evaluation approach applies to system rankings, including retrieval methods with a higher retrieval performance, i.e., methods based on Large Language Models [188], and it may be necessary to address the bias by counterfactual methods and propensity estimation [80,193,207]. Likewise, it should be analyzed to which extent these kinds of evaluations are insightful pre-assessments of the real system performance by deploying them in living labs [155,361,366], which are the focus of the next chapter.

Chapter 8

Living Lab Experiments

This chapter introduces the living lab infrastructure STELLA, which offers a platform for user-oriented experimentation. Within the scope of this dissertation project, we see it as a way to analyze the ecological validity of IR experiments in real-world environments. The platform design keeps the principles of technical reproducibility and transparency in mind. It facilitates the transition of a system-oriented experiment into the living lab environment, as outlined in the following. The infrastructure was tailored explicitly for shared task collaborations and served as the backbone of the Living Labs for Academic Search (LiLAS) at CLEF in 2021. To this end, the lab was the first testbed for evaluating the infrastructure. One of the substantial improvements over earlier living lab attempts is the possibility of submitting the entire experimental system instead of submitting precomputed results only, which addresses several of the earlier shortcomings. Furthermore, we summarize the findings from the experimental evaluations of the lab and conclude with their implications for reproducible experimentation with the STELLA platform in the future. The contributions of this chapter are as follows:

C9 Living lab infrastructure for reproducible experimentation (cf. [67])

C10 Evaluations of the shared task that was organized as part of CLEF and served as a testbed for the infrastructure (cf. [362])

The chapter's contents are mainly based on the contributions to the ISI conference [67], which describe the infrastructure from a technological point of view, and the experimental evaluations have been published in the overview report of the CLEF lab [361, 362]. Besides, other aspects mentioned throughout the following sections also can be found in [68, 363].

The remainder is structured as follows. First, in Section 8.1, we recapture the living lab paradigm for user-oriented IR experiments. Then, in Section 8.2, we introduce our living lab platform, the STELLA infrastructure. Afterward, we summarize the shared task at CLEF and give a brief overview of its organization in Section 8.3, which is followed by Section 8.4 that provides the corresponding evaluations of the experimental ranking and recommender systems. Finally, we outline future directions in Section 8.5.

Figure 8.1: An illustration of the living lab paradigm for shared task experiments.

8.1 Living Labs for Real-World Experimentation

Figure 8.1, which can be seen as a simplified version of Figure 3.1, illustrates the principle behind the living lab paradigm. Within the scope of shared tasks, the participants contribute their experimental systems or sometimes only the precomputed outputs to the living lab platform, which can be considered a *broker* between participants on the one side and the connected search services on the other side. Users can then be provided with the experimental results upon request, and their interactions will be logged in order to evaluate or improve the experimental systems.

One of the earlier works that mentioned the idea of a "living laboratory" was made by Kelly et al. [220] and dates back to 2009. The idea was picked up by Azzopardi and Balog [17], who made the first proposal for a living lab architecture in 2011. In 2013, a workshop dedicated to living labs discussed several requirements and extensions of the living lab paradigm [29] followed by the first implementation of the living lab architecture for ad-hoc IR experiments in 2014 [30]. Finally, the first living lab for ad-hoc retrieval was held at CLEF in 2015 and was continued in a second iteration in 2016 [366]. The same organizers were also involved in the Open Search track at TREC in 2016 and 2017 [200]. NEWSREEL was the first living lab for real-time news recommendations and ran from 2014 until 2017 [70, 191].

More recent living lab implementations are not specifically tailored for shared tasks but have a domain-specific focus. Some recent examples include APONE [281, 282] and arXivDigest [155]. APONE is a living lab platform designed for A/B tests focusing on evaluating user interfaces. As it builds upon the PlanOut language [27], it allows designing the experiments by scripting them. arXivDigest is a recommendation service for research articles based on personalized email updates on recent publications from arXiv's computer science repositories. After registration, an interest profile helps to find adequate recommendations, and feedback is provided with the help of clicked URLs in the personalized mail. Besides arXivDigest, Beel et al. [36] also provide a living lab platform for scholarly recommendations.

8.2 The Living Lab Infrastructure STELLA

Figure 8.2 provides an overview of the infrastructure's design principles. The following describes this figure from left to right so that it can be "read" alongside the descriptions in the text. Participants of the shared task (*experimenters*) provide systems with retrieval and recommendation algorithms in the form of micro-services that can be deployed on purpose in a reproducible way. The infrastructure builds upon Docker and containerization to make this possible. An additional central component is Git and the integration of the web service GitHub, facilitating the experimental components' software versioning, transparency, and reproducibility.

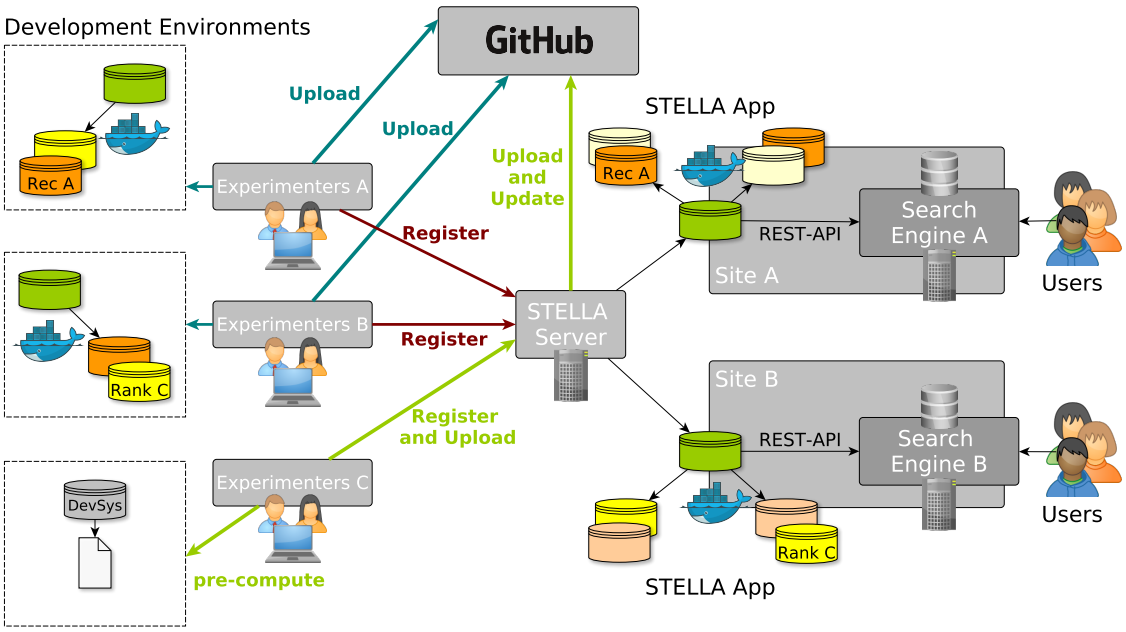

Figure 8.2: Overview of STELLA based on the paradigm illustrated in Figure 8.1.

Once the systems are implemented, the experimenters prepare them with Docker containers. More specifically, they prepare a dockerizable source code repository that builds upon a provided template provided by us (the organizers). After having registered at a central administration server (*STELLA server*), each dockerized system can be integrated into the *STELLA app* that is referred to as a *Multi-Container Application (MCA)*. Multiple systems from possibly different experimenters are combined into the MCA, which means that the administrators at the search services do not have to set up individual systems but rather can rely on complete replicas of all submitted systems once the MCA is running.

Search service providers are referred to as *sites*. They provide access to data collections and, most importantly, user interaction data. Each site deploys an instance of the MCA on its backend servers. Queries from users will then be conducted from the search interfaces to the MCA. As the REST-API provides access to all features, the MCA can easily be integrated into existing services with the only requirement of redirecting the user traffic and making the corresponding API calls for retrieving experimental results and sending back the user feedback data. Upon request, experimental rankings and recommendations are returned by the MCA, and with the help of the corresponding session identifiers, the sites can send back the logged user interactions. Eventually, the MCA sends the user interaction data to the central server, where it is stored and can be used for further analysis, training, and optimization of the experimental systems.

In the following, we outline how experimenters can contribute their systems for participation either as dockerized micro-services or simply by their ranking outputs (run files). Afterward, we describe the MCA and its interaction with the central STELLA server.

8.2.1 The Micro-Services

In earlier living labs, participants did not contribute the entire system for participation, but only its precomputed results for particular head queries [30, 366]. The precomputed results were stored on a central server that could be queried whenever users entered one of the head queries into the search interface. While lightweight, this approach suffered from three major drawbacks.

First, as the central server was hosted on the web and not on the site's servers, network latencies possibly occurred and impacted the user experience, causing frustration and early session abandonments. Second, results could only be retrieved for the selected head queries, which per definition, occur more often but which are also a particular type of query. To this end, the experiments were restricted to these queries, and sessions with query reformulations without head queries could not be logged. Third, outdated precomputed results led to biased evaluations. Especially in environments with frequent index updates like e-commerce platforms, outdated ranking results can be critical as they are an obstacle to the final purchase decision.

In order to address these shortcomings, we integrated Docker, or more generally the containerization technology, into the infrastructure to make the submission of the entire retrieval system possible. However, our infrastructure makes it possible to submit precomputed results for selected head queries for better comparability to the older implementations. In the following, we outline both ways of submitting a system for participation, either in its entirety as a dockerized container or by its precomputed outputs for selected queries and target items.

Precomputed Results

As described above, the contribution of precomputed results to the STELLA platform is inspired by how contributions were made to former living lab attempts [200]. Primarily, we integrated this submission feature to evaluate our new infrastructure design for two reasons. First, experiments based on precomputed results serve as the baseline for evaluating our new infrastructure design proposal. Second, it makes participation possible for those experimenters unfamiliar with Docker. Instead of submitting the entire retrieval or recommendation service, only its results are contributed for experimentation, as illustrated in Figure 8.3.

The precomputed results are restricted to a specific set of queries extracted from the query logs' top k results. In cooperation with the sites, we reused existing logs and determined the most frequent queries, which were provided to the participants after registration. In order to make sure that the submitted results are compatible with our infrastructure, we asked the participants to submit them in the TREC run file syntax already shown in Figure 4.3. It means that each line in the submitted file contains the numeric query identifier (`<qid>`), a string identifying the document (`<docid>`), an increasing rank number (`<rank>`), the corresponding score (`<score>`) and the tag chosen by the experimenters (`<tag>`).

The files must be uploaded to the central server either with the help of HTTP requests and the REST-API or in a more convenient way by the user interface. Then, the infrastructure service will automatically prepare the uploaded files for integration into the MCA after the submissions have been checked for validity and consistency. In this regard, the infrastructure service reduces the experimenters' technical workload and lowers participation barriers. Hence, the experimenters only

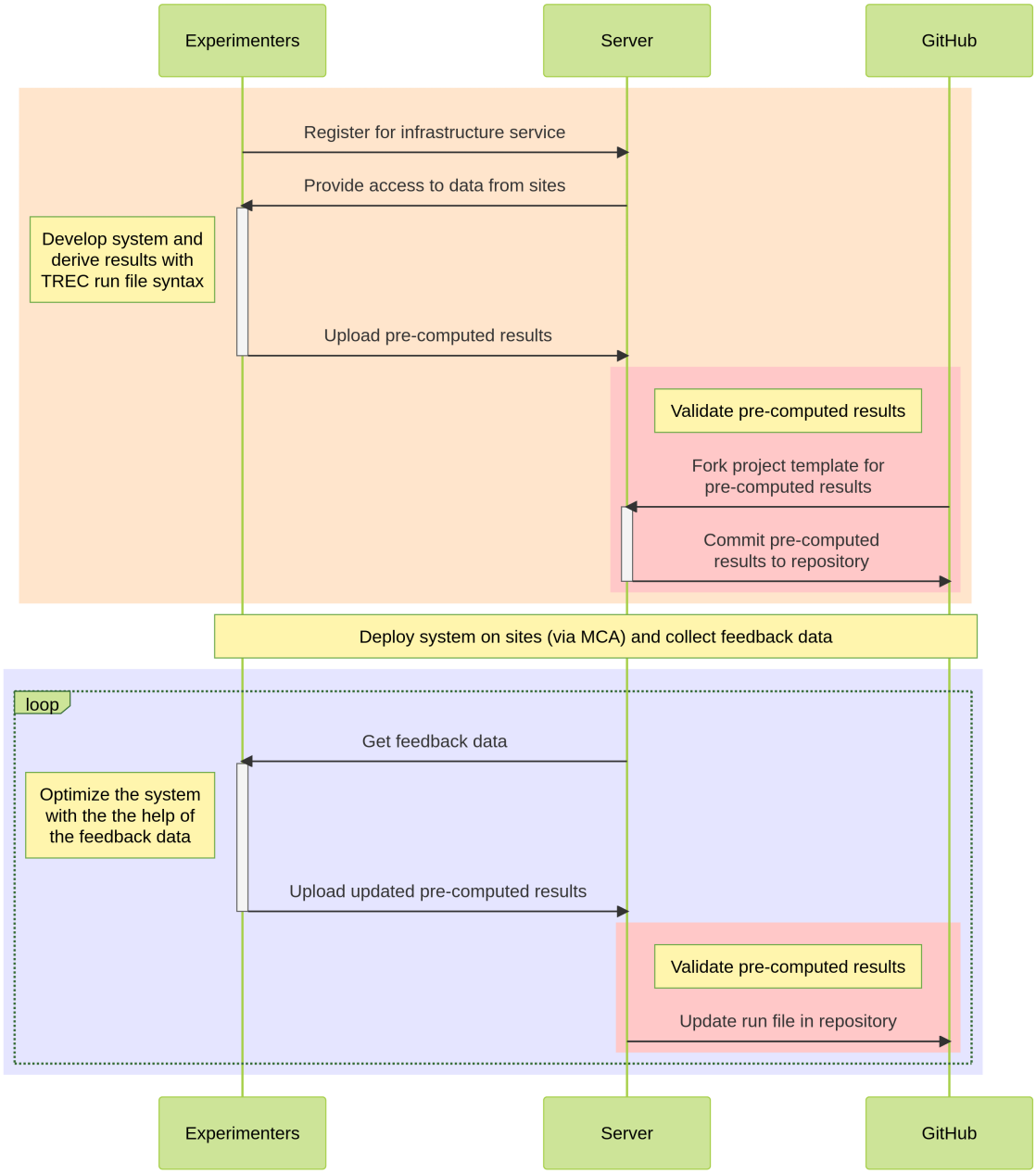

Figure 8.3: Sequence of submitting precomputed results to the infrastructure.

have to upload their precomputed results, and the STELLA service prepares them for experimentation. More specifically, a new micro-service is automatically set up and integrated into the MCA (cf. to the red boxes in Figure 8.3).

Container Systems

Alternatively, experimenters can submit the entire IR or RecSys application as a countermeasure to the drawbacks of the precomputed results. Compared to precomputed results, these micro-services are more comprehensive, i.e., their responses are not limited to preselected queries (or target items) when rankings or recommendations can be provided for arbitrary requests. In this case, the applications are contributed as individual micro-services packaged in Docker containers. Compared

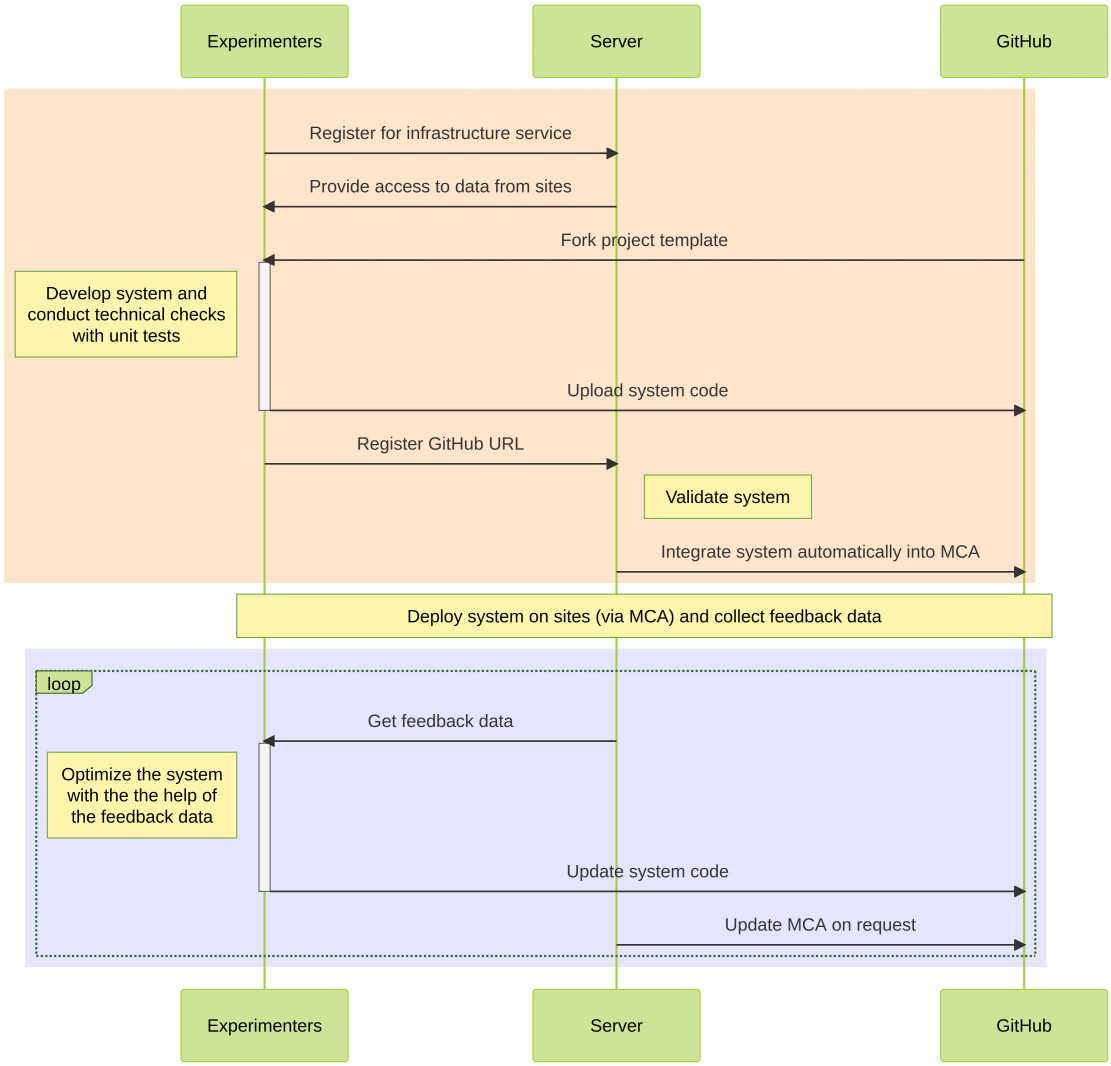

Figure 8.4: Sequence of submitting containerized systems to the infrastructure.

to the outlined sequence of Figure 8.3, submitting Docker containers to STELLA requires interaction with the STELLA server and the GitHub web service. Figure 8.4 gives a sequence diagram of the submission process.

After registration and downloading the sites' data collection, the experimenters can start developing the experimental systems. To ease participation and compatibility between experimental systems and the MCA, a project template in the form of a minimal Docker application is provided to the experimenters by us (the organizers). Of course, these project templates must be adapted, i.e., experimenters integrate their applications. As a starting point, the project template provides a minimal Python application based on the Flask web framework. However, it is not a hard requirement for experimenters to use Python at all. They can modify the template as they see fit, with the only requirement that is Docker containers responding to defined REST endpoints, which are used by the MCA to index documents and retrieve rankings or recommendations.

Having finished the developments, the experimenters upload the source code and the Dockerfiles into the public GitHub repository and register the corresponding URL at the STELLA server. Once the system is validated, i.e., it has passed

several unit tests, it can be integrated into the MCA, which is deployed at the sites. Once enough data is available, experimenters can start the feedback loops that are illustrated as blue boxes in Figure 8.3 and 8.4. Clicks and other interactions can be used as points of reference for improving the ranking and recommender systems. The modified system (outputs) can then be re-submitted and re-evaluated, and the resulting interactions can be compared to those of the previous evaluation phase.

8.2.2 The MCA and the Central Server

Once the experimental systems have passed technical unit tests and sanity checks for selected queries and target items, they are ready to be deployed and evaluated. In order to reduce the deployment efforts for the site administrators, the single experimental systems are bundled into an MCA, which serves as the gateway to the infrastructure for the sites. Using Docker as a packaging tool ensures that all systems are set up as intended, i.e., Docker makes the systems more reproducible.

The MCA handles the query distribution among the experimental systems on a least-served basis and sends user feedback data to the central server at scheduled intervals. After the REST-API of the MCA is connected to the search interface, the user traffic can be redirected to the MCA, returning the experimental results.

The results of single experimental systems are interleaved with those from the baseline system based on the TDI algorithm, which has already been used as part of the experiments in Chapter 7, leading to the following two benefits. First, we prevent users from subpar retrieval results that, in the worst case, affect the site's reputation. Second, interleaved results can be used to infer statistically significant results with fewer user data as compared to conventional A/B tests [336].

The sites can implement logging tools for the user interactions, for which several solutions were recently introduced [48, 242, 289, 359]. STELLA expects a minimal set of JSON-formatted feedback data, but the sites can freely add any additional feedback information and interactions to the data payload, for instance, logged clicks on site-specific SERP elements.

Technically, the MCA is built with `docker-compose` [457] – a build automation tool for multi-container Docker applications. With the help of a YAML file, it is possible to define multiple micro-services and combine them into one application. Since each micro-service is defined by a separate text-based entry in the YAML file, the infrastructure service can automatically add them once the individual systems pass the validation based on unit tests.

At the center of the infrastructure is the STELLA server, which handles the authentication of users, validates submissions by experimenters, and automates the build process of the MCA. Furthermore, it offers a RESTful API used by the MCA to post feedback data. Likewise, the experimenters can use this API to retrieve feedback data and the corresponding rankings and recommendations from the database.

After having registered, users can log in and submit new systems by adding pointers to the corresponding GitHub URL or by uploading their precomputed system results. Once systems' outputs have been exposed to site users, experimenters can visit the dashboard service and are provided with visual analytics tools to have first impressions of how the system performs compared to the baseline.

The dashboard shows the number of impressions (counter how often system results have been shown to the user) and the total click numbers. In addition, we

provided experimenters with derived measures like wins, losses, and ties [366] as STELLA interleaves the systems' outputs by the TDI algorithm. The open-source implementation of the STELLA infrastructure is available on GitHub [468].

8.3 Shared Task Organization

In 2021, we organized a shared task at CLEF [361], which served as the first testbed for the STELLA infrastructure described above. In the following, we give a short overview of how LiLAS was organized before we analyze the lab's experimental results in Section 8.4. The shared task was divided into two rounds. The first round took place in March, and the second round ran from mid-April until the end of May. In between, participants optimized their systems based on the feedback data.

In cooperation with our project partners at *GESIS - Leibniz Institute for the Social Sciences* and *ZB MED - Information Centre for Life Sciences*, we were able to deploy the infrastructure on their backend servers and connect the infrastructure to their search services called GESIS Search and LIVIVO, respectively. GESIS Search serves different types of information from the social sciences, most notably, literature in different languages and research data, questions and variables, and others. LIVIVO serves literature and other resources in multiple languages covering medicine, health, environment, agriculture, and nutrition. Combining the two broader scientific fields gave us an optimal setting for cross-domain validations.

LiLAS offered the possibility to participate in two different tasks. The first task was dedicated to ad-hoc ranking experiments that were exclusively deployed on the LIVIVO platform. Given a query by the platform users, the experimental system should return a ranked list with the most relevant documents. The second task was dedicated to research dataset recommendations that were exclusively deployed on the GESIS Search platform, i.e., given a target item, which was a text-based publication, the experimental RecSys had to return recommendations for appropriate research datasets. For both tasks, the participants could choose to submit their contributions either as precomputed results or as a dockerized container.

Both institutes provided datasets and the corresponding top k queries or target items to the LiLAS participants. For LIVIVO, the most frequent queries were provided with candidate documents of the productive system that could have been used for a reranking task. Due to license restrictions, only a subset of LIVIVO's entire document collection could be provided to the participants, which was sufficient for developing dockerized container system as the indexing of the entire document collection could be conducted on LIVIVO's servers later on. However, the precomputed results were restricted to this subset of documents, which could result in a biased comparison during interleaving with the productive system.

For GESIS Search, a selection of target items for the recommendations, which are the equivalence to the most frequent queries, were provided to the participants. However, as research dataset recommendations were introduced as a new feature to the search platform as part of LiLAS, there was no productive system that could provide candidates for recommendations, but the platform organizers determined them by matching documents with the datasets' abstracts via cosine similarity between the tf-idf representations of the texts. More details about the organization can also be found in the corresponding overview reports [361, 362].

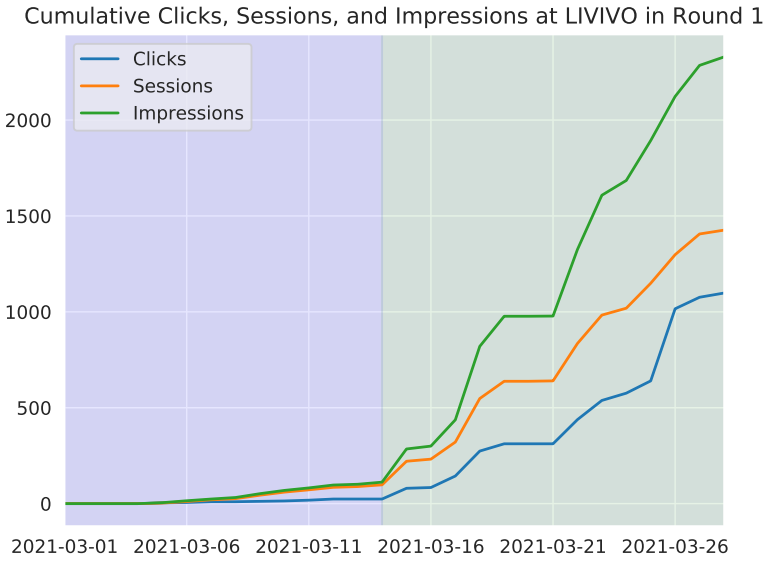

Figure 8.5: Cumulative sum of logged session data (including clicks (blue), sessions (orange), and impressions (orange)) at LIVIVO before (left blue area) and after (right green area) the first fully dockerized system went online in the first round.

8.4 Experimental Evaluations

In the following, we compare precomputed results with containerized systems and describe the evaluations, covering general statistics and the system benchmark.

8.4.1 Precomputed Results vs. Containerized Systems

As pointed out earlier, a major difference between STELLA and earlier living lab implementations is the possibility of submitting the entire experimental system. In the following, we compare the volume of logged data at LIVIVO during the first round, which can be seen in Figure 8.5. Only precomputed results were submitted for experimentation during the first two weeks (blue region). Consequently, the amount of logged user feedback data at LIVIVO was comparatively low due to systems with precomputed results for preselected head queries. However, once the first containerized systems were deployed, substantially more user traffic data could be redirected to our infrastructure. For the latter half of March (green region), the cumulative sums of logged sessions, impressions, and clicks steeply increased. This can be explained by a larger number of queries for which rankings could be retrieved; likewise, more user interactions could be logged. It can be seen that the number of impressions was higher than the number of sessions. It means that multiple results of an experimental system were shown in a single session since our infrastructure can provide experimental rankings throughout a user session with query reformulations. However, not every impression received clicks, as the number of clicks stayed below the number of impressions. Overall, we consider integrating entire experimental systems into the STELLA infrastructure as successful.

Table 8.1: Number of sessions, impressions, clicks and CTR.

Round	Site	Sessions	Impressions	Clicks	CTR
1	LIVIVO	2852	4658	2452	0.5264
	GESIS	4568	8390	152	0.0181
2	LIVIVO	12962	25830	11562	0.4476
	GESIS	6576	12068	250	0.0207

8.4.2 Evaluation of the Shared Task

This subsection covers the evaluation of the shared task. First, we provide some general statistics about the volume of logged data and the distribution of the click data. Afterward, we introduce the evaluation measures for the system analysis, presented at the end of this part.

General Statistics

As mentioned before, the lab was split into two rounds. Table 8.1 provides an overview of the traffic logged in both rounds. In sum, substantially more sessions, impressions, and clicks were logged in the second round, not only due to a slightly longer period but also because more systems were contributed as containerized submissions. In the first round, the deployed experiments at LIVIVO were mainly based on precomputed results, meaning their responses were restricted to the preselected head queries. LIVIVO started the second round with dockerized systems, which delivered results for arbitrary queries, and thus, more session data was logged.

GESIS started both rounds with most experiments based on entire systems in Docker containers. In comparison to LIVIVO, more sessions and impressions were logged in the first round, but fewer recommendations were clicked. Similarly, there are fewer clicks in the second round compared to LIVIVO, which is also reflected by the CTR that is determined by the ratio between clicks and impressions. As mentioned before, GESIS introduced the recommendations of research datasets as a new service, and presumably, users needed to be made aware of this new feature. During the first two weeks of the first round, the amount of logged data at LIVIVO is comparatively low due to systems with precomputed results for preselected head queries. After that, the first type B systems were deployed, and increasingly more user traffic could be redirected to our infrastructure.

In general, we observed many skewed data distributions in our experiments. For instance, the logged impressions follow a power-law-like distribution for both rankings and recommendations, as shown in Figure D.1. Most of the impressions could be attributed to a few top k queries for the rankings or target items for the recommendations. Note that unbalanced query distributions can lead to unfair comparisons if a system is evaluated more often with a hard query, for which the system returns poor retrieval results. At the same time, it may perform better with less frequent queries.

Another critical aspect to be considered as part of the system evaluations is the position bias inherent in the logged data. Click decisions were biased towards the top ranks of the result lists, as shown in Figure D.2. The rankings and recommendations

were displayed to users as vertical lists for both use cases. Note that GESIS restricted the recommendations to the first six recommended datasets, and no pagination over the following recommended items was possible. LIVIVO showed its users ten results per page; as can be seen from the logged data, users rarely clicked results beyond the fifth page.

In addition to "simple" clicks on ranked items, we logged specific SERP elements that were clicked at LIVIVO. Table 8.3 provides an overview of which elements were logged, and Figure D.3 shows the CTR of these elements also follows a power-law-like distribution. The number of clicks was the highest for the *details* button, and the *title* and *full text* click options followed it. In comparison, the other four logged elements received substantially fewer clicks.

As LIVIVO offers a search service for life sciences, the COVID-19 pandemic influenced the query distributions: the most frequent and the fifth most frequent queries were "covid19" and "covid", respectively. Likewise, multilingualism had to be considered as LIVIVO offers results in multiple languages. Three of the ten queries were German queries ("demenz", "pflege", "schlaganfall"); others were domain-specific or could be interpreted as English queries. In both rounds, interaction data was logged for 11,822 unique queries with an average length of 2.9840 terms (which is also typical for web search queries), and each session had 1.9340 queries on average.

Evaluation Measures

Our logging infrastructure allowed us to track search sessions and the corresponding interactions made by users. Each session comprised a specific site user, multiple queries (or target items), and the corresponding results and feedback data in the form of user interactions, primarily logged as clicks with timestamps. Given this session log data, we could determine several measures and evaluation criteria listed in Table 8.2. Like previous living lab initiatives, we designed our user-oriented experiments with interleaved result lists based on TDI. As explained in Chapter 7, it is inspired by the team selection in a team-sports match and combines ranking lists of two competing retrieval systems. Compared to other interleaving methods, TDI is less biased [336]. Consequently, we could also determine the wins, losses, ties, as well as derived outcomes for relative comparisons of the experimental and baseline systems [366]. We refactored the same implementation for the highest degree of comparability [454].

In addition, to the "conventional" living lab evaluations by Schuth et al. [366], we adapt the proposal by Gingstad et al. [155] and evaluate the *Reward* as a weighted sum of clicks on different elements in a SERP. Figure 8.6 shows the corresponding elements (highlighted by red boxes) for a search result as it would have been displayed to the LIVIVO users. While the outcome measure treats all clicks equally, we argue that it is reasonable to weigh the different elements as they have different implications throughout the search session. For instance, ordering an item can be considered equivalent to a purchase decision in an e-commerce setting, whereas clicking the *details* button can be a weaker indicator of relevance. Table 8.3 shows the weights we assigned intellectually to the seven SERP elements logged in the LIVIVO experiments. Alternatively, the weights can also be determined by existing click logs in the future.

Table 8.2: Measures and evaluation criteria in the living lab.

Measure	Description
Win [366]	If the results of experimental system in the interleaved ranking received more clicks than those of the baseline system, a win is assigned to it.
Loss [366]	In case the experimental system received less clicks than the baseline system it is assigned a loss.
Tie [366]	If both the experimental system and the baseline receive an equal number of clicks or no clicks at all, a tie is assigned to the experimental system.
Outcome [366]	$\frac{\text{Wins}}{\text{Wins+Losses}}$; Ratio between the total number of wins and the sum of wins and losses.
Sessions	Number of sessions in which the system participated.
Impressions [366]	Number of times results of the system have been shown in experiments.
Clicks	Number of total clicks a system has received.
Click-through rate	$\frac{\text{Clicks}}{\text{Impressions}}$; The click-through rate (CTR) is the ratio between the total number of clicks and the total number of impressions.
Reward [155]	$\sum_{s \epsilon S} w_s c_s$; Weighted sum of the clicks on SERP elements: S denotes the set of all elements on a SERP, w_s denotes the corresponding weight of the SERP element s that was clicked, and c_s denotes the total number of clicks on the SERP element s.
nReward [155]	$\frac{\text{Reward}_{\text{exp}}}{\text{Reward}_{\text{exp}}+\text{Reward}_{\text{base}}}$; sum of all weighted clicks on experimental results ($\text{Reward}_{\text{exp}}$) normalized by the total Reward given by $\text{Reward}_{\text{exp}}$ + $\text{Reward}_{\text{base}}$.

System Analysis

In our living lab experiments, we evaluated a total of eleven different systems, out of which nine were experimental systems primarily based on lexical retrieval methods that differ by their configurations and preprocessing pipelines. Table 8.4 provides an overview of the systems that participated in both rounds.

Tran et al. [393] contributed precomputed rankings from two systems. Their first system ($\text{LJM}^{\text{Rank}}_{\text{Precom}}$ [393]) was based on Elasticsearch and used a language model with Jelinek-Mercer smoothing [439] as a ranking method. Similarly, their second precomputed submission ($\text{BM25}^{\text{Rank}}_{\text{Precom}}$ [393]) was based on Elasticsearch and uses the default BM25 ranking method but has a modified preprocessing pipeline, which tried to address the multilingualism of the LIVIVO platform. Another submission of precomputed rankings was made by Keller and Munz [217]. As part of their submission ($\text{BM25}^{\text{Rank}}_{\text{Precom}}$ [217]), they precomputed the results with the BM25 implementation of Solr, which is similar to other BM25 submissions as both Solr and Elasticsearch build upon the Lucene library.

Table 8.3: Weights assigned to the SERP elements that are shown in Figure 8.6.

SERP element	Bookmark	Order	Full Text	In Stock	More Links	Title	Details
Weight w_s	10	10	8	8	2	1	1

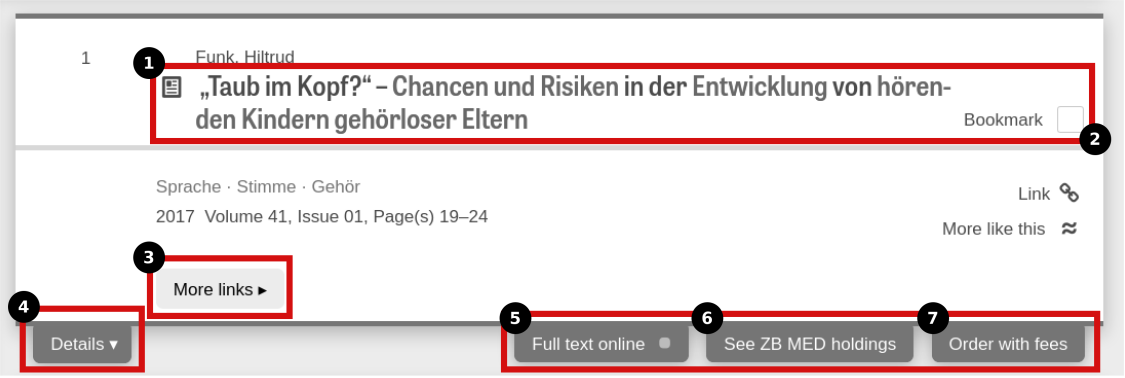

Figure 8.6: Example with highlighted elements of LIVIVO's SERP including ❶ title, ❷ bookmark, ❸ more links, ❹ details, ❺ full text, ❻ in stock, and ❼ order.

In the second round, Tran et al. reused their implementations for a dockerized submission ($\text{DFR}^{\text{Rank}}_{\text{Docker}}$ [393]) but changed the ranking method to DFR [8]. Likewise, they combined it with the modified preprocessing ($\text{DFR}^{\dagger\text{Rank}}_{\text{Docker}}$ [393]), which is denoted by the dagger symbol in Table 8.4. In both rounds, we integrated a Docker container ($\text{BM25}^{\text{Rank}}_{\text{Docker}}$ [361]) based on Pyserini's BM25 implementation (also based on Lucene) and let it participate for a few weeks in both rounds. All experimental rankings were interleaved with the productive baseline system $\text{LIVIVO}_{\text{Baseline}}$.

We evaluated a total of four different recommender systems for research datasets. All of the systems were based on the lexical ranking methods and matched the target item's title against the abstracts of the research datasets. Keller and Munz reused their BM25-based approach for the recommendation task ($\text{BM25}^{\text{Rec}}_{\text{Precom}}$ [217]) and Tavakolpoursaleh and Schaible [387] contributed a dockerized tf-idf-based submission ($\text{TFIDF}^{\text{Rec}}_{\text{Docker}}$ [387]). We also evaluated the tf-idf-based candidates that were precompiled for the participants ($\text{TFIDF}^{\text{Rec}}_{\text{Precom}}$ [361]). Compared to $\text{TFIDF}^{\text{Rec}}_{\text{Docker}}$ [387], the tf-idf matching was not based on Pyterrier but uses a custom implementation. All experimental recommendations were interleaved with the baseline system $\text{GESIS}_{\text{Baseline}}$, based on Pyserini's BM25 implementation.

Table 8.5 compares the system effectiveness, and the corresponding logged interactions and sessions during both rounds. At LIVIVO, none of the experimental systems could outperform the baseline systems regarding the outcome measure. Note that the baseline systems' reported outcomes result from comparisons against all experimental systems. The systems with precomputed rankings received a total number of 32 clicks for four weeks at LIVIVO. Since interaction data was sparse in the first round, we only received enough data for our dockerized BM25-based system ($\text{BM25}^{\text{Rank}}_{\text{Docker}}$ [361]) to conduct meaningful significance tests. The reported p-value results from a Wilcoxon signed-rank test shows a significant difference between the experimental and baseline system.

The lower half of Table 8.5 shows the results of the second round. As part of their BM25-based submission ($\text{BM25}^{\text{Rec}}_{\text{Precom}}$ [217]), Keller and Munz precomputed

Table 8.4: System overview of LiLAS.

Task	Type	Method	Round 1	Round 2
Ranking	Precomputed	$\text{LJM}^{\text{Rank}}_{\text{Precom}}$ [393]	●	●
		$\text{BM25}^{\text{Rank}}_{\text{Precom}}$ [393]	●	●
		$\text{BM25}^{\text{Rank}}_{\text{Precom}}$ [217]	●	●
	Docker	$\text{DFR}^{\text{Rank}}_{\text{Docker}}$ [393]	○	●
		$\text{DFR}^{\dagger\text{Rank}}_{\text{Docker}}$ [393]	○	●
		$\text{BM25}^{\text{Rank}}_{\text{Docker}}$ [361]	◑	◑
		$\text{LIVIVO}_{\text{Baseline}}$	●	●
Recommendation	Precomputed	$\text{BM25}^{\text{Rec}}_{\text{Precom}}$ [217]	○	●
		$\text{TFIDF}^{\text{Rec}}_{\text{Precom}}$ [361]	●	○
	Docker	$\text{TFIDF}^{\text{Rec}}_{\text{Docker}}$ [387]	●	●
		$\text{GESIS}_{\text{Baseline}}$	●	●

recommendations for the entire volume of publications at GESIS. Their submission replaced the system with the precompiled candidates ($\text{TFIDF}^{\text{Rec}}_{\text{Precom}}$ [361]) in the second round and achieved a higher CTR compared to the other recommender systems. Likewise, it achieved an outcome of 0.62, which indicates that it might outperform the baseline recommendations by the $\text{GESIS}_{\text{Baseline}}$ system. Unfortunately, we could not conduct any meaningful significance test due to the low volume of click data.

At LIVIVO, the systems with precomputed rankings received a comparable amount of clicks similar to the first round. In sum, all three systems received a total number of 35 clicks for five weeks, slightly more than in the first round, but it has to be considered that the experimental sessions had to be distributed over more systems. Even though click data was sparse and interpretations have to be made carefully, the relative effectiveness order of these three systems was preserved in the second round (e.g., in terms of the outcome, the total number of clicks, or CTR).

No experimental dockerized system could outperform the $\text{LIVIVO}_{\text{Baseline}}$ system in the second round. Both systems $\text{DFR}^{\text{Rank}}_{\text{Docker}}$ [393] and $\text{DFR}^{\dagger\text{Rank}}_{\text{Docker}}$ [393] achieved significantly lower outcome scores as the baseline. However, the second system had substantially lower outcome and CTR scores. Both systems shared a fair amount of the same methodological approach and only differed in processing the input text. In this case, the system effectiveness did not benefit from the modified preprocessing.

Our dockerized system $\text{BM25}^{\text{Rank}}_{\text{Docker}}$ [361] did not participate in the entire second round since we took it offline as soon as the other two systems were available to let them participate in more sessions by reducing the number of experimental systems from three to two. Despite having participated in comparatively fewer experiments than in the first round (1260 sessions vs. 243 sessions), our system $\text{BM25}^{\text{Rank}}_{\text{Docker}}$ [361] achieved comparable outcome and CTR scores in both rounds. This circumstance raises the question of how long systems must be online to deliver reliable estimates of effectiveness (cf. Chapter 7).

Figure 8.7 provides an overview of how the outcome score evolves over aggregated sessions for different systems and rounds. As the results show, the outcome tends to

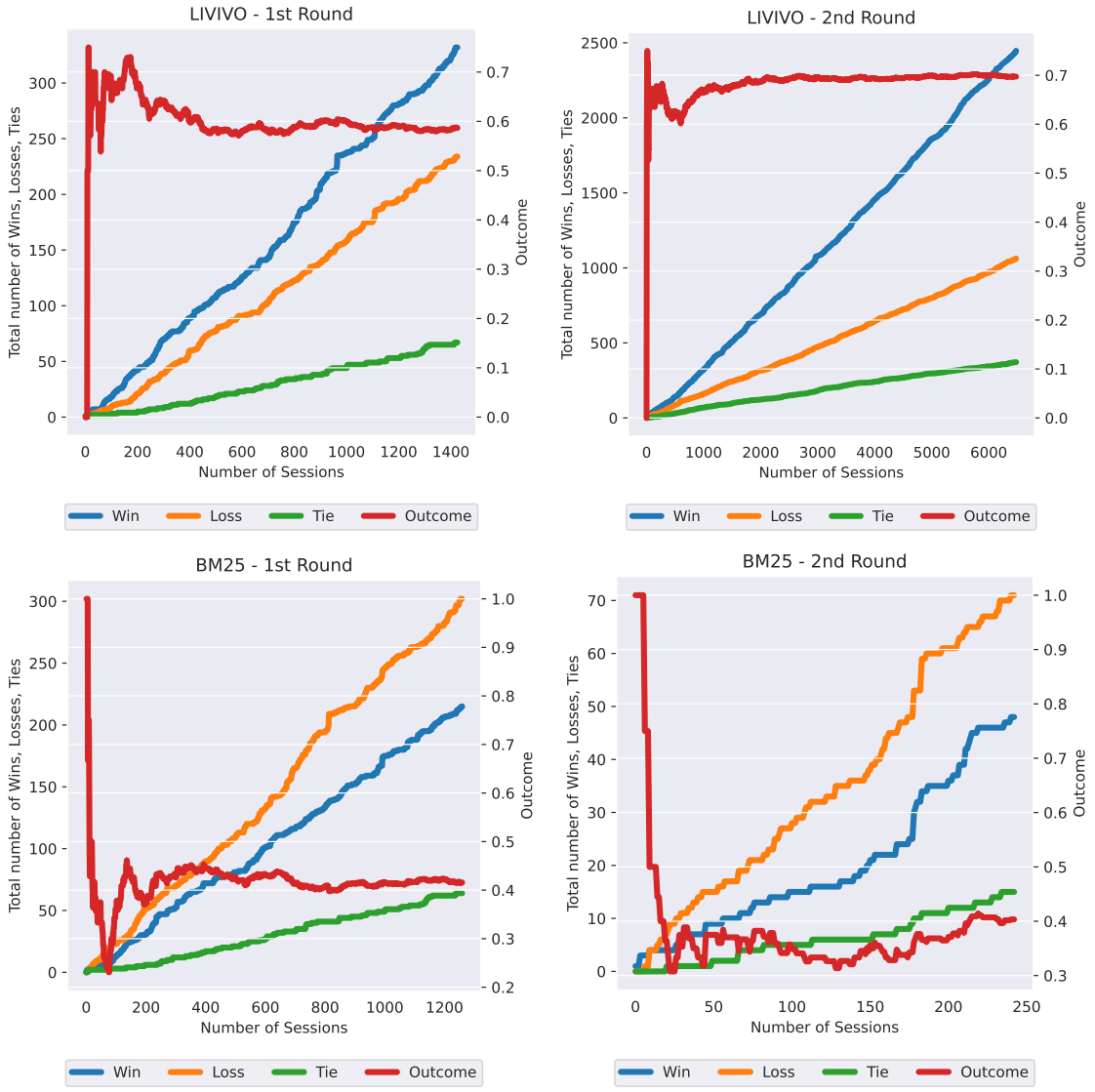

Figure 8.7: Outcome (red), wins (blue), losses (orange), and ties (green) over an increasing number of sessions in round 1 (left) and 2 (right). The top row corresponds to the $\text{LIVIVO}_{\text{Baseline}}$ system competing against all experimental systems. The bottom row corresponds to $\text{BM25}^{\text{Rank}}_{\text{Docker}}$ [361] competing against $\text{LIVIVO}_{\text{Baseline}}$.

stabilize after a certain number of sessions. Putting these results into context with the findings of Chapter 7, we conclude that it is indeed possible to determine the relative system effectiveness once a substantial amount of click data is logged. In the future, it should be analyzed what factors impact how many clicks and sessions have to be logged to make reliable estimates about the system effectiveness and the benefits for users.

One of the primary differences between these evaluations and those of Chapter 7 is the query type included in the experiments. While the click model-based evaluations focused on the top k queries (to have enough data available for parameterizing the click models), the living lab experiments included the entire query distribution. It means that also less frequent queries were included for determining the outcome. Future work should analyze how the query type affects the reliability of the effectiveness estimates. For instance, a system could result in good effectiveness over

Table 8.5: Outcomes of rounds 1 and 2. Significant differences based on Wilcoxon signed-rank tests are denoted by an asterisk symbol (∗).

System	Win	Loss	Tie	Outcome	Sessions	Impressions	Clicks	CTR
Round 1								
$\text{GESIS}_{\text{Baseline}}$	36	36	1	0.50	2284	4195	37	0.0088
$\text{TFIDF}^{\text{Rec}}_{\text{Docker}}$ [387]	26	28	1	0.48	1968	3675	28	0.0076
$\text{TFIDF}^{\text{Rec}}_{\text{Precom}}$ [361]	10	8	0	0.56	316	520	11	0.0212
$\text{LIVIVO}_{\text{Baseline}}$	332	234	67	0.59	1426	2329	677	0.2907
$\text{BM25}^{\text{Rank}}_{\text{Docker}}$ [361]	215	302	64	0.42*	1260	2135	517	0.2422
$\text{LJM}^{\text{Rank}}_{\text{Precom}}$ [393]	4	8	1	0.33	45	55	10	0.1818
$\text{BM25}^{\text{Rank}}_{\text{Precom}}$ [217]	6	10	1	0.38	64	77	8	0.1039
$\text{BM25}^{\text{Rank}}_{\text{Precom}}$ [393]	9	12	1	0.43	57	62	14	0.2258
Round 2								
$\text{GESIS}_{\text{Baseline}}$	51	68	2	0.43	3288	6034	53	0.0088
$\text{TFIDF}^{\text{Rec}}_{\text{Docker}}$ [387]	26	25	1	0.51	1529	2937	27	0.0092
$\text{BM25}^{\text{Rec}}_{\text{Precom}}$ [217]	42	26	1	0.62	1759	3097	45	0.0145
$\text{LIVIVO}_{\text{Baseline}}$	2447	1063	372	0.70	6481	12915	3791	0.2935
$\text{BM25}^{\text{Rank}}_{\text{Docker}}$ [361]	48	71	15	0.40	243	434	112	0.2581
$\text{DFR}^{\text{Rank}}_{\text{Docker}}$ [393]	707	1042	218	0.40*	3131	6274	1273	0.2029
$\text{DFR}^{\dagger\text{Rank}}_{\text{Docker}}$ [393]	291	1308	135	0.18*	2948	6026	570	0.0946
$\text{LJM}^{\text{Rank}}_{\text{Precom}}$ [393]	6	13	0	0.32	61	69	10	0.1449
$\text{BM25}^{\text{Rank}}_{\text{Precom}}$ [217]	4	7	1	0.36	36	42	5	0.1190
$\text{BM25}^{\text{Rank}}_{\text{Precom}}$ [393]	7	6	3	0.54	62	70	20	0.2857

a uniform query distribution while failing to deliver good results for more frequent queries. Consequently, it would not be a good candidate for industrial use.

Previous studies showed that a system is more likely to win if its documents are ranked at higher positions [200]. As part of our experimental evaluations, we also could confirm this circumstance. We also determined the Spearman correlation between an interleaving outcome (1: win, -1: loss, 0: tie) and the highest-ranked position of a document contributed by an experimental system. At both sites, we see a weak but significant correlation (LIVIVO: $\rho = -0.0883$, $p = 1.3535e-09$; GESIS: $\rho = -0.3480$, $p = 4.7422e-07$).

One area for improvement of the previous measures derived from interleaving experiments was the simplified interpretation of click interactions. As outlined earlier, weighting clicks makes it possible to account for the meaning of the corresponding

Table 8.6: Experimental systems of round 2 and the corresponding number of clicks on SERP elements, total number of clicks, and the Reward score.

System	Bookmark	Details	Full Text	In Stock	More Links	Order	Title	Total Clicks	nReward
$BM25^{Rank}_{Docker}$ [361]	182	341	176	55	62	28	263	1107	0.4367
$LIVIVO_{Baseline}$	180	443	228	154	57	29	329	1420	0.5633
DFR^{Rank}_{Docker} [393]	63	832	481	107	105	54	638	2280	0.4045
$LIVIVO_{Baseline}$	56	1066	646	295	129	85	858	3135	0.5955
$DFR\dagger^{Rank}_{Docker}$ [393]	23	355	257	23	28	21	285	992	0.2143
$LIVIVO_{Baseline}$	69	1190	762	301	119	82	934	3457	0.7857
LJM^{Rank}_{Precom} [393]	1	13	16	0	2	0	10	42	0.4242
$LIVIVO_{Baseline}$	1	24	7	14	1	0	20	67	0.5758
$BM25^{Rank}_{Precom}$ [217]	2	11	2	2	1	0	6	24	0.3430
$LIVIVO_{Baseline}$	0	13	6	7	0	1	9	36	0.6570
$BM25^{Rank}_{Precom}$ [393]	11	21	9	3	1	1	16	62	0.5496
$LIVIVO_{Baseline}$	8	13	7	5	2	1	6	42	0.4504
All experimental systems	282	1573	941	190	199	104	1218	4507	0.3485
$LIVIVO_{Baseline}$	314	2749	1656	776	308	198	2156	8157	0.6515

SERP elements. Table 8.6 shows each system's total number of clicks on SERP elements and the nReward resulting from the weighting scheme given in Figure 8.3. We compare the total number of clicks of those experiments in which the experimental and baseline systems delivered results. As can be seen, comparing systems by clicks on different SERP elements provides a more diverse analysis. Some systems achieve higher numbers of clicks (and CTRs) for some SERP elements compared to the baseline system. For instance, $BM25^{Rank}_{Docker}$ [361] or DFR^{Rank}_{Docker} [393] got more clicks on the *bookmark* element than the baseline system, while both systems achieve lower numbers of total clicks.

Similar to the previous evaluations, none of the systems could outperform the baseline system by the nReward measure. However, compared to the outcome scores, there is a more balanced ratio between the nReward scores that also accounts for the meaning of specific clicks. Likewise, it accounts for clicks even if the experimental system did not win in the interleaving experiment. Table 8.6 compares the total number of clicks over multiple sessions. While the win, loss, tie, and outcome only measure if there have been more clicks in a single experiment, the nReward also considers those clicks that were made in experiments in which the experimental system did not necessarily win.

8.5 Conclusion

This chapter introduced our living lab platform STELLA and the corresponding IR and RecSys experiments of LiLAS. Within the scope of this dissertation project, we consider living lab experiments as a solution for analyzing the ecological validity of findings from system-oriented experiments in real-world user environments. A key component of the underlying infrastructure is the integration of experimental ranking and recommendation systems as micro-services that are implemented with the help of Docker. LiLAS was the first testbed to use this evaluation service, and it exemplified some of the benefits of the new infrastructure design. First, completely dockerized systems can overcome the restrictions of results limited to filtered (top k) queries or target items. Significantly more data and click interactions can be logged if the experimental systems can return results on purpose for arbitrary requests of rankings and recommendations. Consequently, this allows more data aggregation in a shorter time, providing a solid basis for statistical significance tests.

Furthermore, the deployment effort for site providers and organizers is considerably reduced. Once the systems are properly described with the corresponding Dockerfile, they can be rebuilt on purpose, precisely as intended by the participants and developers. Likewise, the entire infrastructure service can be migrated with minimal costs due to the use of Docker. We note that even though dockerizing a system requires additional development time, the effort will pay off. If the systems are properly adapted to the required interface and the source code is available in a public repository, the research community can rely on these artifacts that make the experiments transparent and reproducible.

In this regard, we address the reproducibility of these living lab experiments mainly from a technological point of view. It is possible to repeat the experiments in the future with reduced efforts since the participating systems are openly available and should be reconstructible with the help of the corresponding Dockerfiles and the STELLA infrastructure. Future work should investigate how feasible it is to rely on the Dockerfiles for long-term preservation. Since experimental systems are rebuilt each time with the help of the Dockerfile, updates of the underlying dependencies might be a threat to reproducibility. An intuitive solution is integrating pre-built Docker images that allow longer reproducibility.

Apart from the technical questions, the reproducibility of the actual experimental results has to be investigated. Our experimental setup allowed us to answer questions regarding the reproducibility of the experimental results over time and across different domains (e.g., life vs. social sciences). However, some limitations of our experiment have to be considered. No test collection with editorial relevance labels was available to test any of the systems in a typical system-oriented experiment. In the future, it should be analyzed how such an editorial labeling process could be integrated into living lab experiments. Then it is possible to compare system-oriented and living lab experiments. Similarly, it is reasonable to keep several pairs of systems fixed throughout the experimental round (evaluation phases) for better and more systematic comparisons, e.g., a baseline and an improved method that is known to outperform the baseline in terms of system-oriented measures like P@10 or AP. With the help of STELLA, it can be analyzed if these effectiveness gains also transfer over to the real world, similar to the experiments by Turpin and Hersh [396].

Most of the evaluation measures were made for interleaving experiments that also depend on the results of the baseline system and not solely on those of an experimental system. We have not investigated yet if the experimental results follow a transitive relation: if the experimental system A outperforms the baseline system B, denoted as $A \succ B$, and the baseline system B outperforms another experimental system C ($B \succ C$), can we conclude that system A would also outperform system C ($A \succ C$)? The evaluations showed that click results are heavily biased towards the first ranks. In the future, the click position bias should be addressed by weighting clicks on top-ranked documents differently, e.g., with the help of existing search session logs like the Sowiport User Search Sessions Data Set [290].

Likewise, clicks were context-dependent, i.e., depending on the entire result list, and single-click decisions must be interpreted concerning other results of the SERP. Previously seen results and further evaluations in these directions would require counterfactual reasoning. Nonetheless, the second round illustrated how our infrastructure service could be used for incremental developments and component-wise analysis of experimental systems. The two experimental systems by Tran et al. $\text{DFR}^{\text{Rank}}_{\text{Docker}}$ [393] and $\text{DFR}\dagger^{\text{Rank}}_{\text{Docker}}$ [393] followed a similar approach and only differ by the preprocessing component that was not of any benefit.

In addition to established outcome measures of interleaving experiments (win, loss, tie, outcome), we also accounted for the meaning of clicks on different SERP elements. In this context, we implemented the Reward measure as the weighted sum of clicks on different elements corresponding to a specific result. Even though most of the experimental systems could not outperform the baseline systems in terms of the overall scores, we saw some differences in the system effectiveness, which allowed us to assess a system's merits more thoroughly when the evaluations were based on different SERP elements. Overall, our lab is a successful advancement over previous living labs as we were able to exemplify the benefits of fully dockerized systems, delivering results for arbitrary queries and also confirming previous findings.

As the evaluations showed, there were many skewed distributions in the logged data, like, e.g., the power law-like distribution of clicks that could be attributed to the position bias. In the future, these biases must be considered in the evaluations, but also when reusing the logged session data for new evaluations as they might be biased towards the systems that participated in the original experiments. In order to make participation in the shared task more attractive, it might be helpful to provide participants with open and more transparent baseline systems they can build upon. Some of the precomputed experimental rankings and recommendations seemed to deliver promising results; however, the evaluations needed to be interpreted with care due to the sparsity of the available click data, which could be addressed by continuous evaluations freed from the time limits of rounds.

Finally, the living lab experiments can be conceptually aligned with the PRIMAD-U taxonomy. The experimental setups implied variation for each of the taxonomy's components. More generally, the experimentation platform stayed fixed as it was the STELLA infrastructure. On a more granular level, the participants could choose the platform (P') that could be reproduced by containerization. The research goal (R') was twofold. On the one hand, ranking systems were analyzed in the life sciences. On the other hand, recommender systems were analyzed in the social sciences. Of course, there were different implementations when a method changed (M'). Nevertheless, implementation details (I') also varied even if the same groups

contributed systems with the same retrieval methods. The evaluations were made as part of a shared task, so the participants implied actor variance (A'). As two different academic search engine providers were involved in the experiments, there were consequently different datasets (D'). Finally, user variation (U') was naturally given as the user base of academic search engine providers is diverse.

In the future, it should be analyzed if the logged data is usable for the simulation approaches outlined in the earlier chapters. In order to implement the controlled query reformulations of Chapter 6, relevance labels are required. We did not curate a test collection with editorial labels. However, it could be a possible solution to derive pseudo-relevance labels with the help of click models, as outlined in Chapter 7. Overall, the contributions of the three chapters lay the foundation for dynamic evaluation environments, where the fidelity of simulations can be validated by aligning them to real user behavior, and reasonable simulation models help to identify promising systems for online experiments.

Chapter 9

Discussion and Conclusion

This chapter summarizes and discusses the contributions of this dissertation project. Afterward, we outline ideas for future work, building upon the outlined contributions. Finally, we conclude at the end of this chapter.

9.1 Discussion

In the following section, we discuss the contributions of the previous chapters and put them into context. The section is aligned with the different levels of validity to which our contributions were made. Related to the reproducibility analysis, there is also the question of validity. Reproducing an experimental outcome under the same conditions as in the original experiment gives evidence of internal validity. If former findings also generalize under the condition of a different experimental setup, there is strong evidence for external validity.

At the level of internal validity, we review the results of the Chapters 2, 3, 4, 5 that primarily dealt with the evaluation of system-oriented IR experiments. With a particular focus on the influence of the user, we have lowered the abstraction of user interactions to validate the external validity of an IR experiment beyond the Cranfield approach that implies a strong simplification of the user for the sake of reproducibility. As an alternative to IR experiments with real users like they are known from small-scale IIR or large-scale online experiments, we have considered user simulations as a more reproducible way to validate the external validity. In Chapters 6 and 7, we had a particular focus on the simulation of *query variants* and *click interactions*, respectively. Finally, we outlined how the ecological validity — a subtype of the external validity — can be evaluated in living lab experiments with real users in Chapter 8. With a focus on the key findings, we discuss the limitations of the results and the contributions to reproducible IR research.

9.1.1 Internal Validity

Our literature review in Chapter 2 answered the research question about factors possibly affecting reproducibility and what kinds of countermeasures have already been implemented. Furthermore, we highlighted open points that motivated the dissertation's contributions. Reasons for failed reproducibility attempts range from mundane reasons like a missing experimental setup to more profound reasons when

the original findings cannot be reproduced in a slightly different experimental context. We have reviewed the existing problems and solutions regarding reproducible IR research and aligned these to the PRIMAD taxonomy, which defines six influential components of an IR experiment that could affect reproducibility. In addition, we put these results into a broader context by referring to the more general causes of irreproducibility, also pointed out in a Nature survey across different research fields.

Besides the answers to our research questions, we concluded that the PRIMAD taxonomy basically considers nearly all relevant components a reproducible experiment requires. However, it is still outlined at a very abstract level. Furthermore, it has separate definitions for system- and user-oriented IR experiments, which confirms its applicability but undervalues the role of users as they are mainly considered by their data-trace in the taxonomy. To this end, we favor a more holistic view on the IR experiment by extending the taxonomy and introducing PRIMAD-U in Chapter 3. Our taxonomy extends the original taxonomy by an additional user component but also makes it more specific by adding subcomponents. Based on the outcomes of the literature survey, we have extended each PRIMAD-U component with several subcomponents and related aspects that could affect the reproducibility of an IR experiment. By the examples of earlier studies that, on the one hand, pointed out factors that cause reproducibility issues but, on the other hand, also solutions to prepare an experiment for reproducibility in advance, we have aligned these findings to the components of the PRIMAD-U taxonomy. Regarding the introduced user component, we have outlined how the related subcomponents can be considered as part of user simulations, as earlier studies gave evidence for their influence on the experimental outcomes.

Regarding the conventional six PRIMAD components, we have developed the metadata annotation schema `ir_metadata`, which can be used to annotate the experimental artifacts of IR experiments, e.g., in the form of TREC run files. By no means do we claim to provide a complete taxonomy, but we emphasize its extensibility and point out that it considers essential components that influenced the reproducibility of earlier works. Given that there is currently no metadata standard for run files, we think it contributes to better reproducibility practices that also help make an experiment more transparent and comparable in the aftermath. In the future, a community-wide adoption should be enforced as the annotations reveal their full potential as more experimental results — run files — are annotated. Having a large amount of annotated run files, the effort of systematic meta-evaluations could be reduced to a minimum. Instead of conducting tedious literature research to find baseline methods or candidates for benchmark studies, it would be feasible to formulate a structured query that expresses the properties of the particular PRIMAD components that should be the same or different. Likewise, such a metadata service could be used to find adequate baselines for experimentation. However, the annotation itself requires effort. While some annotations could be automated in the future, some metadata fields will still require manual labeling. As a solution, it makes sense to cooperate with shared task organizers who can enforce the annotations at submission time. In this regard, the TREC Deep Learning track successfully demonstrated how participants could be motivated to provide more metadata than conventional information, like the name of the approach.

In Chapter 4, we framed the typical approach of a reactive reproducibility attempt and how reproducibility measures can help to quantify the degree of system-

oriented IR experiments. As another practical contribution, the evaluation toolkit `repro_eval` compiles all of these measures and provides them to the research community as an open-source software library. It is still an open question when we consider an experiment successfully reproduced, i.e., when a reimplementation is "good enough" to be considered a reasonable reproduction of the original reference. At the current state, the reproducibility measures provide feedback about the reproduction quality by relative comparisons to other reimplementations. They can be used to evaluate which reimplementation delivers results more similar to those of the original software implementations. However, it is still part of the future work to give an answer when something can be considered as successfully reproduced solely based on the score of a reproducibility measure.

From a practical point of view, this dissertation contributed resources in the form of reimplementations based on the principle of CCRF, which were used to compile a dataset of annotated run files. Building upon the reproducibility measures and the annotated dataset, we have demonstrated how principled reproducibility evaluations can be conducted based on similar or different PRIMAD components of experimental setup components. Identifying suitable evaluation protocols based on the metadata annotations is a challenging task, and it was included in our evaluations by prototypical implementations. For instance, we assumed that the run files in the same directory implicitly provided reasonable comparisons. However, identifying combinations of runs, which result in reasonable comparisons, can be challenging when having a large variety of metadata indexed in a database. Future work should explore how such run combinations for meaningful comparisons can be automatically identified.

Overall, our contributions to evaluating the reproducibility at the level of internal validity are made available in a distilled form by the reusable artifacts for the reactive evaluations and proactive annotations of IR experiments. `repro_eval` is an evaluation toolkit for measuring the degree of reproducibility as part of reactive reproductions, being extensible by implementations of other reproducibility measures. `ir_metadata` allows IR practitioners to describe and annotate the artifacts of IR experiments in a proactive way, which supports the reuse in reactive reproducibility studies or as part of meta-evaluations.

9.1.2 External Validity

Within the scope of this dissertation, we focused on the influence of the user when evaluating reproducibility at the level of external validity. Other work considers a retrieval experiment to be generalized when observing similar effects to the original experiment with another test collection. However, we mainly focused on the question of to which extent the results of system-oriented experiments can be successfully validated under the variation of user behavior.

While user experiments provide answers to the question of how the outcomes of system-oriented experiments translate into the real world, they are generally not considered reproducible as little is known about the users, and their interactions depend on individual preferences. As an alternative to experiments with real users, we consider the simulation of their interactions as a viable solution to account for their influence in the experiments without sacrificing control over the behavior. As

part of Chapters 6 and 7, we analyze the simulated user behavior in the form of clicks and queries and their implications for reproducibility.

In Chapter 6, we analyzed the differences in the retrieval performance that results from UQV, i.e., how different query formulations for the same underlying information need affect the experimental outcomes. We have compared different user models for the query simulations covering naive approaches based on the topic texts of the test collections, proficient searchers with background knowledge of the topics and the documents in the collections, as well as a parameterizable query simulation approach that gives better estimates of real user querying behavior.

The experimental outcomes have shown clear differences between the user models, which also impact the reproducibility and the external validity of an IR experiment. There are different ways to formulate a query for a given information need, depending on the knowledge and familiarity with the topic. As a result, users formulate different queries that likewise result in different retrieval outcomes. Our query simulation method is parameterizable and allows us to define different querying strategies, which had a higher similarity with real UQV than other conventional query simulation approaches. The replicability study showed that most of the findings could be successfully revalidated with another test collection, implying that the approach could also be considered a way to simulate UQV when topics are reused for another test collection.

In Chapter 7, we have analyzed how feasible it is to reproduce the relative ranking of systems with the help of click signals collected from search sessions of a medical database, which is similar to the search platform that took part in our living lab experiments. In our analysis, we have included click models based on different user models. Compared to a simple CTR-based click model that solely considers the attractiveness of the search results, we evaluated two more complex click models, which also embed satisfaction and continuation probabilities. Our experiments have shown that click signals can be used to determine the relative performance ordering of systems if enough click data is available.

While user simulations give complete control of how a user should behave in an IR experiment, the conclusions drawn heavily depend on the fidelity of the user models. Therefore, it may suffice to focus on simulating the aspects that are the focus of the analysis. However, the simplifications a user model implies should be considered, and the generalizability of the conclusions drawn from the simulations should not be overestimated. For instance, some of our session simulations in Chapter 6 focused on the query aspects and did not consider any simulated interaction with the result list, i.e., the simulated user scanned the entire result list similar to the implicit user of Cranfield-style evaluations.

In Chapter 7, the click models allow a more elaborated simulation of the interaction with the result lists. However, it has to be considered that the analyzed click models can only be used for known queries. To have enough click data available, we focused on the most frequent queries sent by the search service users. However, provided only with the query and the corresponding clicks, we do not know if the queries originate from the same information need. While some queries are easily identifiable, like know-item queries, others are more ambiguous. In this regard, the click models imply the same underlying information need for the logged queries.

Future work should analyze other user interactions but also consider their interplay. For instance, the click decisions often depend on the attractiveness of the

results, that in turn depends on the presentation of the SERP. Besides the layout, snippet generation is a key component influencing how the single results attract the user's attention. Future simulation experiments should analyze how simulated users interact with the results (make their click decisions) under the consideration of different snippet generation algorithms. The corresponding user models could be based on a language model that depending on their comprehensiveness, represent different knowledge states of the user. Throughout the progress of the search sessions, these user models could be extended by the already-seen terms of the snippet texts. Furthermore, the interplay of different simulation stages requires more in-depth investigations. For instance, the query simulation method could be combined with parameterized click models for more realistic interactions with result lists.

Overall, our contributions to evaluating the reproducibility at the level of external validity include simulations of UQV and the analysis of how the query variants impact retrieval effectiveness. The retrieval results were analyzed regarding different aspects, and most outcomes could be generalized with another test collection. In addition to the simulation of queries, we have analyzed how click models that simulate a different user behavior can be used to reproduce the ranking of systems. Our results showed that clicks could be used as alternative relevance signals but also pointed out limitations. The next subsection describes another contribution of how external validity can be evaluated. More specifically, it summarizes our contributions to evaluating ecological validity, which can be seen as a subtype of external validity based on the validation in a real-world environment.

9.1.3 Ecological Validity

In Chapter 8, we have described how living lab experiments provide a solution for analyzing the ecological validity of IR experiments. Besides outlining the architecture of the living lab platform STELLA, we also included the corresponding evaluations of the shared task at the CLEF conference that served as the first testbed for the infrastructure. In contrast to IIR studies, the living lab experiments offer access to a substantially larger user base at the cost of knowledge and control over the users.

By using Docker — or, more generally, the concept of containerization — as a core technology, our infrastructure STELLA implements technical reproducibility by making the contributed experimental systems reusable. Besides, it reduces the deployment effort and ensures the systems run as intended when deployed on the search services' backend servers.

Our experimental evaluations showed that containerizing experimental systems are a key technology to collecting larger amounts of user feedback data. Compared to earlier living lab implementations, we did not restrict the experimental rankings to the most frequent queries but let the systems participate in tremendously more experiments as they could deliver experimental results for arbitrary queries.

Unfortunately, the retrieval methods of our living lab experiments were not diverse and did not outperform the baseline system. In our experiments, we deployed traditional lexical retrieval methods. In the future, evaluating more effective retrieval methods based on Large Language Models and Transformers would be interesting. However, more demanding retrieval and recommender approaches come at a cost, as they require hardware resources only some platform providers can afford.

In the future, it is reasonable to implement comprehensive evaluation life cycles that analyze the system-oriented results with test collections, afterward, estimate the experimental results of online experiments by simulated user interactions, and finally evaluate them in real-world online experiments. In order to draw better conclusions about how system-oriented outcomes relate to user-oriented outcomes, a domain-specific test collection with editorial relevance judgments should be curated as it is a key resource to compare outcomes of system- and user-oriented experiments.

Having access to both system- and user-oriented experimentation environments, the simulations can be improved by comparing the user models to the real-world outcomes. In addition, a higher fidelity of the user model regarding the real-world reference allows better estimates that help to identify low-performing retrieval approaches that could harm the user experience and, likewise, help to reduce the time of online experiments and avoid ethical considerations related to user experiments.

Overall, our contributions to evaluating the reproducibility at the level of ecological validity include the introduction of a living lab infrastructure based on the concept of containerization and micro-services and the corresponding evaluations of the shared task that served as the first testbed for the infrastructure.

9.2 Future Work

In the following, we recap the ideas for future work mentioned earlier in the text by making them explicit and describing how they could be implemented.

- **Dissemination and extension of the metadata schema:** The metadata schema `ir_metadata` can be used as a pro- and reactive resource supporting the reproducibility of IR experiments. On the one hand, it can be used to prepare an experiment for reproducibility by making the underlying experimental setup of the run file more transparent when annotating it. On the other, it helps to analyze run files as part of meta-evaluations or reactive reproducibility studies, as outlined in our experiments. However, the usefulness of the metadata annotations for meta-evaluations heavily depends on the amount of annotated experimental data. In this regard, the adoption by the community should be enforced, and IR practitioners should be motivated to annotate their experimental data. In the future, collaboration with shared task organizers could help inform participants of annotated runs' benefits. Likewise, it would be possible to host an additional web service that provides public access to the metadata, which could be consolidated when conducting meta-evaluations or searching for an adequate baseline. In addition, the schema and taxonomy should be extended depending on the use cases.

- **Evaluation and extension of the reproducibility measures:** The reproducibility measures provide a starting point for evaluating a system-oriented IR experiment at different levels of specificity. Right now, it is possible to conclude the reproduction quality concerning other reimplementations, i.e., it is possible to say which reimplementation is closer to the original. However, it remains an open question when an experiment can be considered sufficiently reproduced. As our experiments showed, the reproducibility measures at the document level are quite "sensitive" to only slight modifications of the retrieval

method. It implies that users are exposed to different documents when using the reimplemented ranking as part of a reproduced user experiment. In this sense, it is an open question if such a user experiment has to be considered a failed reproduction per se or if other aspects of the user experience can still be reproduced. For example, earlier studies showed that users compensate for poorer retrieval results with different search behavior, and it is possible to gain the same knowledge about a topic with different search results containing the same information as the original experiment. Likewise, it makes sense to extend the framework of measures. For instance, in addition to the p-values at the most general level, it makes sense to consider effect sizes. We contribute the implementations of the measures as part of the evaluation toolkit `repro_eval`, which is provided as open-source software and extensible with new implementations of other reproducibility measures.

- **Simulation and validation of other user interactions:** Besides query variants and clicks, other forms of simulated user interactions and user-related aspects should be considered. For instance, the analyzed click models only implicitly consider the user interface. However, earlier user experiments have shown that the SERP layout also influences the search process. Therefore, in addition to the layout of the SERP elements, the content of the snippets and documents could be analyzed. For example, the reading time or the level of expertise required to understand the contents of a search item would result in cognitive strains that could be modeled as costs. Likewise, the knowledge gain could be modeled by the document's relevance or the terms shown to users.

- **Validation of comprehensive user interaction sequences:** In the earlier chapters, the simulated user interactions were analyzed in isolation. For instance, the query simulations were not combined with click interactions, whereas the analysis of click models did not vary the query formulations of single topics. In the future, the individual stages of the search process should be integrated into more comprehensive and connected interaction sequences. Furthermore, the interplay and dependencies between individual user interactions motivate future research. For instance, how does the query formulation impact the click decisions, or how do previously seen search results influence the query reformulations? As a starting point, existing frameworks and toolkits can be reused. However, also new approaches should be considered, like a UQV-aware click model. In our experiments, we used click models parameterized for a single query formulation. However, it should be investigated if a click model could be adapted to an underlying information need for which different query formulations can be made.

- **Development of a domain-specific test collection:** A key resource to compare system- with user-oriented experiments is a domain- and platform-specific test collection with editorial relevance labels. The search platforms, integrated into the living lab infrastructure, have domain-specific users and data. For instance, in our experiments, search platforms from the social and life sciences took part in the living lab experiments. As a starting point, existing test collections like the TREC Precision Medicine datasets or the TripClick dataset could be reused as they also contain medical documents

with relevance labels. However, there may be a mismatch between the topics of the judged documents and those queries sent by the users in our living lab experiments, and a dedicated test collection should be favored for the sake of better comparability. First of all, combining click data and editorial relevance labels for a single test collection would be a valuable and reusable contribution to the community. Second, some simulations of this thesis could be integrated into the evaluations, like the method for the query simulations, which requires relevance labels to generate query variants.

- **Integrated evaluation life cycle of IR experiments from system- to user-oriented evaluations:** Ultimately, the evaluation of an IR experiment should cover all stages from the Cranfield-style system-oriented evaluations, the validation by simulated user interactions, and finally, the living lab experiments with real users. On the one hand, the user simulations could be improved by evaluating and optimizing their fidelity regarding the outcomes of the living lab experiments. On the other hand, system-oriented evaluations combined with user simulations can be done offline and serve as a preassessment of the retrieval methods that will be deployed later in an online experiment. Depending on the fidelity of the user simulation, the online time of an experiment can be reduced. However, it is likewise possible to identify low-performing retrieval methods that could harm the user experience and exclude them from real-world experiments in advance. Furthermore, such an integrated evaluation life cycle lays the foundation for drawing better conclusions about how changing a retrieval method and observing differences in the system effectiveness carries over to the user experience, i.e., how the user effectiveness reproduces these effects.

Some of these ideas for future work will be addressed in two DFG-funded projects. The STELLA project is continued in a follow-up, which will continue developing and extending the living lab infrastructure. Furthermore, the RESIRE project envisages the reproduction and simulation of IIR experiments and directly takes up some of the outlined ideas.

9.3 Conclusion

This thesis dealt with reproducible IR research and made contributions to how reproducibility can be evaluated at different levels of validity. For the evaluation of the internal validity, we provided a solution for principled reproducibility evaluations of system-oriented IR experiments based on the PRIMAD model. Beyond the scope of the original context, we considered the reproducibility under the variation of the user's influence as a way to draw conclusions about the external validity. As an alternative to online experiments, we considered user simulations a more controllable and reproducible way to validate an IR experiment by accounting for the user variability. Finally, our living lab experiments showed how technically reproducible retrieval systems could be deployed in online experiments to analyze the ecological validity — a subtype of external validity — with real users.

In conclusion, the reproducibility of computational and IR experiments cannot be taken for granted, and ensuring the reproducibility of earlier experiments is an

ongoing challenge. Even if the experimental setup is made available for future reuse, the reproducibility of earlier findings requires constant revalidation, considering the pace of technological advances. However, it is possible to establish a culture of reproducibility, raise awareness to avoid common pitfalls, and prepare experiments proactively to make IR research more reproducible.

In this regard, we have contributed reusable artifacts to the community. First, the evaluation toolkit `repro_eval` provides help for measuring the degree of reproducibility as part of reactive reimplementation attempts. The toolkit is also extensible and provides a starting point for implementing other reproducibility measures. Second, our metadata schema `ir_metadata` allows IR practitioners to describe and annotate the artifacts of IR experiments in a proactive way, which supports the reuse in reactive reproducibility studies or as part of meta-evaluations. The underlying schema is based on our extended version of the PRIMAD taxonomy. Similar to the evaluation toolkit, it is extensible and can be expanded with new subcomponents.

Beyond technical preservations of the experimental setup that ensure rerunning the experiments in a reproducible way, a different experimental setting questions the validity beyond the scope of the original context. Future work should not only consider the system-oriented reproducibility and focus on preparing the experimental setup for possible reuse but also analyze which results are valid under different conditions, i.e., answer whether the original findings can be reproduced in a different experimental context. With special regards to IR experiments, such an influential component of an experiment is the user.

We consider the user's influence in an IR experiment as one of the most influential components that could lead to different conclusions from those drawn from the original experiments, i.e., it questions if the measured system improvements can be reproduced in a user experiment and if they lead to similar improvements of the user effectiveness. In our opinion, every system-oriented experiment with implications for the user should ideally be validated by a real-world user experiment. In the end, the user is the recipient of the retrieved results. However, due to the large effort required to run user experiments alongside system-oriented evaluations with test collections, we think that user simulations allow us to consider the user behavior in a more cost-efficient way and as a more realistic directive compared to the abstract user model of Cranfield-style experiments. Our simulation experiments provided some first ideas of how user simulations can accompany system-oriented experiments. In addition, we outlined how the living lab experiments can support the validation of experiments with real users. In the future, a stronger connection between the evaluation stages should be pursued, and their interrelationship should be analyzed.

Bibliography

[1] Agichtein, E., Brill, E., and Dumais, S. T. Improving Web Search Ranking by Incorporating User Behavior Information. In *SIGIR* (2006), ACM, pp. 19–26.

[2] Agosti, M., Buccio, E. D., Ferro, N., Masiero, I., Peruzzo, S., and Silvello, G. DIRECTions: Design and Specification of an IR Evaluation Infrastructure. In *CLEF* (2012), vol. 7488 of *Lecture Notes in Computer Science*, Springer, pp. 88–99.

[3] Agosti, M., Nunzio, G. M. D., and Ferro, N. Scientific Data of an Evaluation Campaign: Do We Properly Deal with Them? In *CLEF* (2006), vol. 4730 of *Lecture Notes in Computer Science*, Springer, pp. 11–20.

[4] Allan, J., Harman, D., Kanoulas, E., Li, D., Gysel, C. V., and Voorhees, E. M. TREC 2017 Common Core Track Overview. In *TREC* (2017), vol. 500–324 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[5] Althammer, S., Hofstätter, S., and Hanbury, A. Cross-Domain Retrieval in the Legal and Patent Domains: A Reproducibility Study. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 3–17.

[6] Althammer, S., Hofstätter, S., Verberne, S., and Hanbury, A. TripJudge: A Relevance Judgement Test Collection for TripClick Health Retrieval. In *CIKM* (2022), ACM, pp. 3801–3805.

[7] Amati, G. Frequentist and Bayesian Approach to Information Retrieval. In *ECIR* (2006), vol. 3936 of *Lecture Notes in Computer Science*, Springer, pp. 13–24.

[8] Amati, G., and van Rijsbergen, C. J. Probabilistic Models of Information Retrieval Based on Measuring the Divergence from Randomness. *ACM Trans. Inf. Syst. 20*, 4 (2002), 357–389.

[9] Amigó, E., Castells, P., Gonzalo, J., Carterette, B., Culpepper, J. S., and Kazai, G., Eds. *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. ACM, 2022. https://doi.org/10.1145/3477495.

[10] Anelli, V. W., Bellogín, A., Ferrara, A., Malitesta, D., Merra, F. A., Pomo, C., Donini, F. M., and Noia, T. D. Elliot: A Comprehensive and Rigorous Framework for Reproducible Recommender Systems Evaluation. In *SIGIR* (2021), ACM, pp. 2405–2414.

[11] Arguello, J., Diaz, F., Lin, J., and Trotman, A. SIGIR 2015 Workshop on Reproducibility, Inexplicability, and Generalizability of Results (RIGOR). In *SIGIR* (2015), ACM, pp. 1147–1148.

[12] Armstrong, T. G., Moffat, A., Webber, W., and Zobel, J. EvaluatIR: An Online Tool for Evaluating and Comparing IR Systems. In *SIGIR* (2009), ACM, p. 833.

[13] Armstrong, T. G., Moffat, A., Webber, W., and Zobel, J. Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998. In *CIKM* (2009), ACM, pp. 601–610.

[14] Arnold, M., Bellamy, R. K. E., Hind, M., Houde, S., Mehta, S., Mojsilovic, A., Nair, R., Ramamurthy, K. N., Olteanu, A., Piorkowski, D., Reimer, D., Richards, J. T., Tsay, J., and Varshney, K. R. FactSheets: Increasing Trust in AI Services through Supplier's Declarations of Conformity. *IBM J. Res. Dev. 63*, 4/5 (2019), 6:1–6:13.

[15] Attar, R., and Fraenkel, A. S. Local Feedback in Full-Text Retrieval Systems. *Journal of The ACM 24*, 3 (1977), 397–417.

[16] Azzopardi, L. The Economics in Interactive Information Retrieval. In *SIGIR* (2011), ACM, pp. 15–24.

[17] Azzopardi, L., and Balog, K. Towards a Living Lab for Information Retrieval Research and Development - A Proposal for a Living Lab for Product Search Tasks. In *CLEF* (2011), vol. 6941 of *Lecture Notes in Computer Science*, Springer, pp. 26–37.

[18] Azzopardi, L., and de Rijke, M. Automatic Construction of Known-Item Finding Test Beds. In *SIGIR* (2006), ACM, pp. 603–604.

[19] Azzopardi, L., de Rijke, M., and Balog, K. Building Simulated Queries for Known-Item Topics: An Analysis Using Six European Languages. In *SIGIR* (2007), ACM, pp. 455–462.

[20] Azzopardi, L., Moshfeghi, Y., Halvey, M., Alkhawaldeh, R. S., Balog, K., Buccio, E. D., Ceccarelli, D., Fernández-Luna, J. M., Hull, C., Mannix, J., and Palchowdhury, S. Lucene4IR: Developing Information Retrieval Evaluation Resources Using Lucene. *SIGIR Forum 50*, 2 (2016), 58–75.

[21] Azzopardi, L., Stein, B., Fuhr, N., Mayr, P., Hauff, C., and Hiemstra, D., Eds. *Advances in Information Retrieval - 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14-18, 2019, Proceedings, Part I*, vol. 11437 of *Lecture Notes in Computer Science*. Springer, 2019. https://doi.org/10.1007/978-3-030-15712-8.

[22] Baeza-Yates, R. A., and Ribeiro-Neto, B. A. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[23] Bailey, P., Moffat, A., Scholer, F., and Thomas, P. User Variability and IR System Evaluation. In *SIGIR* (2015), ACM, pp. 625–634.

[24] Bailey, P., Moffat, A., Scholer, F., and Thomas, P. UQV100: A Test Collection with Query Variability. In *SIGIR* (2016), ACM, pp. 725–728.

[25] Baker, M. First Results from Psychology's Largest Reproducibility Test. *Nature 30*, 10.1038 (2015). https://www.nature.com/articles/nature.2015.17433.

[26] Baker, M. 1,500 Scientists Lift the Lid on Reproducibility. *Nature News 533*, 7604 (2016), 452. https://www.nature.com/articles/533452a.

[27] Bakshy, E., Eckles, D., and Bernstein, M. S. Designing and Deploying Online Field Experiments. In *WWW* (2014), ACM, pp. 283–292.

[28] Balazs, A. International Vocabulary of Metrology-Basic and General Concepts and Associated Terms. *Chemistry International 25* (2008).

[29] Balog, K., Elsweiler, D., Kanoulas, E., Kelly, L., and Smucker, M. D. CIKM 2013 Workshop on Living Labs for Information Retrieval Evaluation. In *CIKM* (2013), ACM, pp. 2557–2558.

[30] Balog, K., Kelly, L., and Schuth, A. Head First: Living Labs for Ad-Hoc Search Evaluation. In *CIKM* (2014), ACM, pp. 1815–1818.

[31] Balog, K., Maxwell, D., Thomas, P., and Zhang, S. Report on the 1st Simulation for Information Retrieval Workshop (Sim4IR 2021) at SIGIR 2021. *SIGIR Forum 55*, 2 (2021), 10:1–10:16.

[32] Bar, H., and Wang, H. Reproducible Science with LATEX. *Journal of Data Science 19*, 1 (2021), 111–125.

[33] Baskaya, F., Keskustalo, H., and Järvelin, K. Time Drives Interaction: Simulating Sessions in Diverse Searching Environments. In *SIGIR* (2012), ACM, pp. 105–114.

[34] Baskaya, F., Keskustalo, H., and Järvelin, K. Modeling Behavioral Factors in Interactive Information Retrieval. In *CIKM* (2013), ACM, pp. 2297–2302.

[35] Bates, M. J. The Design of Browsing and Berrypicking Techniques for the Online Search Interface. *Online review* (1989).

[36] Beel, J., Collins, A., Kopp, O., Dietz, L. W., and Knoth, P. Online Evaluations for Everyone: Mr. DLib's Living Lab for Scholarly Recommendations. In *ECIR (2)* (2019), vol. 11438 of *Lecture Notes in Computer Science*, Springer, pp. 213–219.

[37] Belkin, N. J. Anomalous States of Knowledge as a Basis for Information Retrieval. *Canadian Journal of Information Science 5*, 1 (1980), 133–143.

[38] Belz, A., Agarwal, S., Shimorina, A., and Reiter, E. A Systematic Review of Reproducibility Research in Natural Language Processing. In *EACL* (2021), Association for Computational Linguistics, pp. 381–393.

[39] Belz, A., Shimorina, A., Agarwal, S., and Reiter, E. The ReproGen Shared Task on Reproducibility of Human Evaluations in NLG: Overview and Results. In *INLG* (2021), Association for Computational Linguistics, pp. 249–258.

[40] Bender, E. M., and Friedman, B. Data Statements for Natural Language Processing: Toward Mitigating System Bias and Enabling Better Science. *Transactions of the Association for Computational Linguistics 6* (2018), 587–604.

[41] Benham, R., and Culpepper, J. S. Risk-Reward Trade-Offs in Rank Fusion. In *ADCS* (2017), ACM, pp. 1:1–1:8.

[42] Benham, R., Gallagher, L., Mackenzie, J. M., Damessie, T. T., Chen, R.-C., Scholer, F., Moffat, A., and Culpepper, J. S. RMIT at the 2017 TREC CORE Track. In *TREC* (2017), vol. 500–324 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[43] Benham, R., Gallagher, L., Mackenzie, J. M., Liu, B., Lu, X., Scholer, F., Culpepper, J. S., and Moffat, A. RMIT at the 2018 TREC CORE Track. In *TREC* (2018), vol. 500–331 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[44] Benham, R., Mackenzie, J. M., Moffat, A., and Culpepper, J. S. Boosting Search Performance Using Query Variations. *ACM Trans. Inf. Syst. 37*, 4 (2019), 41:1–41:25.

[45] Berendsen, R., Tsagkias, M., de Rijke, M., and Meij, E. Generating Pseudo Test Collections for Learning to Rank Scientific Articles. In *CLEF* (2012), vol. 7488 of *Lecture Notes in Computer Science*, Springer, pp. 42–53.

[46] Berrendorf, M., Faerman, E., Melnychuk, V., Tresp, V., and Seidl, T. Knowledge Graph Entity Alignment with Graph Convolutional Networks: Lessons Learned. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 3–11.

[47] Berrendorf, M., Wacker, L., and Faerman, E. A Critical Assessment of State-of-the-Art in Entity Alignment. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 18–32.

[48] Bhattacharya, N., and Gwizdka, J. YASBIL: Yet Another Search Behaviour (and) Interaction Logger. In *SIGIR* (2021), ACM, pp. 2585–2589.

[49] Bhattacharya, P., Hiware, K., Rajgaria, S., Pochhi, N., Ghosh, K., and Ghosh, S. A Comparative Study of Summarization Algorithms Applied to Legal Case Judgments. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 413–428.

[50] Bhattarai, P., Ghassemi, M. M., and Alhanai, T. Open-Source Code Repository Attributes Predict Impact of Computer Science Research. In *JCDL* (2022), ACM, p. 16.

[51] Bleeker, M. J. R., and de Rijke, M. Do Lessons from Metric Learning Generalize to Image-Caption Retrieval? In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 535–551.

[52] Boettiger, C. An Introduction to Docker for Reproducible Research. *ACM SIGOPS Oper. Syst. Rev. 49*, 1 (2015), 71–79.

[53] Boratto, L., Fenu, G., and Marras, M. The Effect of Algorithmic Bias on Recommender Systems for Massive Open Online Courses. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 457–472.

[54] Boratto, L., Fenu, G., Marras, M., and Medda, G. Consumer Fairness in Recommender Systems: Contextualizing Definitions and Mitigations. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 552–566.

[55] Borisov, A., Markov, I., de Rijke, M., and Serdyukov, P. A Neural Click Model for Web Search. In *WWW* (2016), ACM, pp. 531–541.

[56] Bösch, H. Reproducible Ranking Lists for Retrieval from Evolving Document Collections: How Column-Store Technology Enhances the Capability of Inverted Indices, 2017. *Master's Thesis*.

[57] Boyd, K. L. Datasheets for Datasets Help ML Engineers Notice and Understand Ethical Issues in Training Data. *Proc. ACM Hum. Comput. Interact. 5*, CSCW2 (2021), 438:1–438:27.

[58] Brammer, G. R., Crosby, R. W., Matthews, S. J., and Williams, T. L. Paper Mâché: Creating Dynamic Reproducible Science. In *ICCS* (2011), vol. 4 of *Procedia Computer Science*, Elsevier, pp. 658–667.

[59] Breuer, T. Reproducible Online Search Experiments. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 597–601.

[60] Breuer, T., Ferro, N., Fuhr, N., Maistro, M., Sakai, T., Schaer, P., and Soboroff, I. How to Measure the Reproducibility of System-Oriented IR Experiments. In *SIGIR* (2020), ACM, pp. 349–358.

[61] Breuer, T., Ferro, N., Maistro, M., and Schaer, P. Repro_eval: A Python Interface to Reproducibility Measures of System-oriented IR Experiments. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 481–486.

[62] Breuer, T., Fuhr, N., and Schaer, P. Validating Simulations of User Query Variants. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 80–94.

[63] Breuer, T., Keller, J., and Schaer, P. Ir_metadata: An Extensible Metadata Schema for IR Experiments. In *SIGIR* (2022), ACM, pp. 3078–3089.

[64] Breuer, T., Pest, M., and Schaer, P. Evaluating Elements of Web-Based Data Enrichment for Pseudo-Relevance Feedback Retrieval. In *CLEF* (2021), vol. 12880 of *Lecture Notes in Computer Science*, Springer, pp. 53–64.

[65] Breuer, T., and Schaer, P. Dockerizing Automatic Routing Runs for the Open-Source IR Replicability Challenge (OSIRRC 2019). In *OSIRRC@SIGIR* (2019), vol. 2409 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 31–35.

[66] Breuer, T., and Schaer, P. Replicability and Reproducibility of Automatic Routing Runs. In *CLEF (Working Notes)* (2019), vol. 2380 of *CEUR Workshop Proceedings*, CEUR-WS.org.

[67] Breuer, T., and Schaer, P. A Living Lab Architecture for Reproducible Shared Task Experimentation. In *ISI* (2021), Werner Hülsbusch, pp. 348–362.

[68] Breuer, T., Schaer, P., Tavakolpoursaleh, N., Schaible, J., Wolff, B., and Müller, B. STELLA: Towards a Framework for the Reproducibility of Online Search Experiments. In *OSIRRC@SIGIR* (2019), vol. 2409 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 8–11.

[69] Brinckman, A., Chard, K., Gaffney, N., Hategan, M., Jones, M. B., Kowalik, K., Kulasekaran, S., Ludäscher, B., Mecum, B. D., Nabrzyski, J., Stodden, V., Taylor, I. J., Turk, M. J., and Turner, K. Computing Environments for Reproducibility: Capturing the "Whole Tale". *Future Gener. Comput. Syst. 94* (2019), 854–867.

[70] Brodt, T., and Hopfgartner, F. Shedding Light on a Living Lab: The CLEF NEWSREEL Open Recommendation Platform. In *IIiX* (2014), ACM, pp. 223–226.

[71] Brunswik, E. *Perception and the Representative Design of Psychological Experiments*. Univ of California Press, 1956.

[72] Byström, K., and Järvelin, K. Task Complexity Affects Information Seeking and Use. *Inf. Process. Manag. 31*, 2 (1995), 191–213.

[73] Callahan, S. P., Freire, J., Santos, E., Scheidegger, C. E., Silva, C. T., and Vo, H. T. VisTrails: Visualization Meets Data Management. In *SIGMOD Conference* (2006), ACM, pp. 745–747.

[74] Câmara, A., Maxwell, D., and Hauff, C. Searching, Learning, and Subtopic Ordering: A Simulation-Based Analysis. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 142–156.

[75] Camerer, C. F., Dreber, A., Forsell, E., Ho, T.-H., Huber, J., Johannesson, M., Kirchler, M., Almenberg, J., Altmejd, A., Chan, T., Heikensten, E., Holzmeister, F., Imai, T., Isaksson, S., Nave, G., Pfeiffer, T., Razen, M., and Wu, H. Evaluating Replicability of Laboratory Experiments in Economics. *Science (New York, N.Y.) 351*, 6280 (2016), 1433–1436.

[76] Camerer, C. F., Dreber, A., Holzmeister, F., Ho, T.-H., Huber, J., Johannesson, M., Kirchler, M., Nave, G., Nosek, B. A., Pfeiffer, T., et al. Evaluating the Replicability of Social Science Experiments in Nature and Science between 2010 and 2015. *Nature Human Behaviour 2*, 9 (2018), 637–644.

[77] Carterette, B. System Effectiveness, User Models, and User Utility: A Conceptual Framework for Investigation. In *SIGIR* (2011), ACM, pp. 903–912.

[78] Carterette, B. The Best Published Result Is Random: Sequential Testing and Its Effect on Reported Effectiveness. In *SIGIR* (2015), ACM, pp. 747–750.

[79] Carterette, B., Bah, A., and Zengin, M. Dynamic Test Collections for Retrieval Evaluation. In *ICTIR* (2015), ACM, pp. 91–100.

[80] Carterette, B., and Chandar, P. Offline Comparative Evaluation with Incremental, Minimally-Invasive Online Feedback. In *SIGIR* (2018), ACM, pp. 705–714.

[81] Carterette, B., Clough, P. D., Hall, M. M., Kanoulas, E., and Sanderson, M. Evaluating Retrieval over Sessions: The TREC Session Track 2011-2014. In *SIGIR* (2016), ACM, pp. 685–688.

[82] Carterette, B., and Jones, R. Evaluating Search Engines by Modeling the Relationship between Relevance and Clicks. In *NIPS* (2007), Curran Associates, Inc., pp. 217–224.

[83] Chapelle, O., and Zhang, Y. A Dynamic Bayesian Network Click Model for Web Search Ranking. In *WWW* (2009), ACM, pp. 1–10.

[84] Chapp, D., Johnston, T., and Taufer, M. On the Need for Reproducible Numerical Accuracy through Intelligent Runtime Selection of Reduction Algorithms at the Extreme Scale. In *CLUSTER* (2015), IEEE Computer Society, pp. 166–175.

[85] Chard, K., Gaffney, N., Jones, M. B., Kowalik, K., Ludäscher, B., Nabrzyski, J., Stodden, V., Taylor, I. J., Turk, M. J., and Willis, C. Implementing Computational Reproducibility in the Whole Tale Environment. In *P-RECS@HPDC* (2019), ACM, pp. 17–22.

[86] Chen, Y., Zhou, K., Liu, Y., Zhang, M., and Ma, S. Meta-Evaluation of Online and Offline Web Search Evaluation Metrics. In *SIGIR* (2017), ACM, pp. 15–24.

[87] Chirigati, F., Rampin, R., Shasha, D. E., and Freire, J. ReproZip: Computational Reproducibility with Ease. In *SIGMOD Conference* (2016), ACM, pp. 2085–2088.

[88] Chuklin, A., Markov, I., and de Rijke, M. *Click Models for Web Search*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2015.

[89] Chuklin, A., Schuth, A., Hofmann, K., Serdyukov, P., and de Rijke, M. Evaluating Aggregated Search Using Interleaving. In *CIKM* (2013), ACM, pp. 669–678.

[90] Chuklin, A., Schuth, A., Zhou, K., and de Rijke, M. A Comparative Analysis of Interleaving Methods for Aggregated Search. *ACM Trans. Inf. Syst. 33*, 2 (2015), 5:1–5:38.

[91] Chuklin, A., Serdyukov, P., and de Rijke, M. Click Model-Based Information Retrieval Metrics. In *SIGIR* (2013), ACM, pp. 493–502.

[92] Claerbout, J. F., and Karrenbach, M. Electronic Documents Give Reproducible Research a New Meaning. In *SEG Technical Program Expanded Abstracts 1992*. Society of Exploration Geophysicists, 1992, pp. 601–604.

[93] Clancy, R., Ferro, N., Hauff, C., Lin, J., Sakai, T., and Wu, Z. Z. The SIGIR 2019 Open-Source IR Replicability Challenge (OSIRRC 2019). In *SIGIR* (2019), ACM, pp. 1432–1434.

[94] Cleverdon, C. W. The Cranfield Tests on Index Language Devices. In *Aslib Proceedings* (1967), vol. 19, pp. 173–192.

[95] Cleverdon, C. W. The Significance of the Cranfield Tests on Index Languages. In *SIGIR* (1991), ACM, pp. 3–12.

[96] Clyburne-Sherin, A., and Fei, X. Preparing Code and Data for Computational Reproducibility. In *JCDL* (2019), IEEE, pp. 449–450.

[97] Cole, C. A Theory of Information Need for Information Retrieval That Connects Information to Knowledge. *J. Assoc. Inf. Sci. Technol. 62*, 7 (2011), 1216–1231.

[98] Collberg, C., Proebsting, T., and Warren, A. M. Repeatability and Benefaction in Computer Systems Research. *University of Arizona TR 14*, 4 (2015).

[99] Collberg, C. S., and Proebsting, T. A. Repeatability in Computer Systems Research. *Commun. ACM 59*, 3 (2016), 62–69.

[100] Community, T. T. W. The Turing Way: A Handbook for Reproducible, Ethical and Collaborative Research. Zenodo, July 2022. https://doi.org/10.5281/zenodo.6909298.

[101] Crane, M. Questionable Answers in Question Answering Research: Reproducibility and Variability of Published Results. *Transactions of the Association for Computational Linguistics 6* (2018), 241–252.

[102] Craswell, N., Campos, D., Mitra, B., Yilmaz, E., and Billerbeck, B. ORCAS: 20 Million Clicked Query-Document Pairs for Analyzing Search. In *CIKM* (2020), ACM, pp. 2983–2989.

[103] Craswell, N., Mitra, B., Yilmaz, E., and Campos, D. Overview of the TREC 2020 Deep Learning Track. In *TREC* (2020), vol. 1266 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[104] Craswell, N., Zoeter, O., Taylor, M. J., and Ramsey, B. An Experimental Comparison of Click Position-Bias Models. In *WSDM* (2008), ACM, pp. 87–94.

[105] Crocker, J., and M. Lynne Cooper. Addressing Scientific Fraud. *Science (New York, N.Y.) 334*, 6060 (2011), 1182–1182. https://www.science.org/doi/abs/10.1126/science.1216775.

[106] Croft, W. B., and Harper, D. J. Using Probabilistic Models of Document Retrieval without Relevance Information. *J. Documentation 35*, 4 (1979), 285–295.

[107] Cronen-Townsend, S., Zhou, Y., and Croft, W. B. Predicting Query Performance. In *SIGIR* (2002), ACM, pp. 299–306.

[108] Dacrema, M. F., Boglio, S., Cremonesi, P., and Jannach, D. A Troubling Analysis of Reproducibility and Progress in Recommender Systems Research. *ACM Trans. Inf. Syst. 39*, 2 (2021), 20:1–20:49.

[109] Dacrema, M. F., Cremonesi, P., and Jannach, D. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *RecSys* (2019), ACM, pp. 101–109.

[110] Demaree, D., Jarodzka, H., Brand-Gruwel, S., and Kammerer, Y. The Influence of Device Type on Querying Behavior and Learning Outcomes in a Searching as Learning Task with a Laptop or Smartphone. In *CHIIR* (2020), ACM, pp. 373–377.

[111] Dmitriev, P. A., Gupta, S., Kim, D. W., and Vaz, G. J. A Dirty Dozen: Twelve Common Metric Interpretation Pitfalls in Online Controlled Experiments. In *KDD* (2017), ACM, pp. 1427–1436.

[112] Donoho, D. 50 Years of Data Science. *Journal of Computational and Graphical Statistics 26*, 4 (Oct. 2017), 745–766. https://doi.org/10.1080/10618600.2017.1384734.

[113] Donoho, D. L., Maleki, A., Rahman, I. U., Shahram, M., and Stodden, V. 15 Years of Reproducible Research in Computational Harmonic Analysis.

[114] Dror, R., Shlomov, S., and Reichart, R. Deep Dominance - How to Properly Compare Deep Neural Models. In *ACL (1)* (2019), Association for Computational Linguistics, pp. 2773–2785.

[115] Drummond, C. Replicability Is Not Reproducibility: Nor Is It Good Science. In *Proceedings of the Evaluation Methods for Machine Learning Workshop at the 26th ICML, Montreal, Canada,2009* (Collection / Collection : NRC Publications Archive / Archives des publications du CNRC, 2009), Evaluation Methods for Machine Learning Workshop, the 26th ICML, June 14-18, 2009, Montreal, Canada.

[116] Drutsa, A., Gusev, G., Kharitonov, E., Kulemyakin, D., Serdyukov, P., and Yashkov, I. Effective Online Evaluation for Web Search. In *SIGIR* (2019), ACM, pp. 1399–1400.

[117] Dür, A., Rauber, A., and Filzmoser, P. Reproducing a Neural Question Answering Architecture Applied to the SQuAD Benchmark Dataset: Challenges and Lessons Learned. In *ECIR* (2018), vol. 10772 of *Lecture Notes in Computer Science*, Springer, pp. 102–113.

[118] Eickhoff, C., Teevan, J., White, R., and Dumais, S. T. Lessons from the Journey: A Query Log Analysis of within-Session Learning. In *WSDM* (2014), ACM, pp. 223–232.

[119] Einarsson, B. *Accuracy and Reliability in Scientific Computing.* SIAM, 2005.

[120] Ellis, D. A Behavioural Approach to Information Retrieval System Design. *J. Documentation 45*, 3 (1989), 171–212.

[121] Erdogmus, H., Morisio, M., and Torchiano, M. On the Effectiveness of the Test-First Approach to Programming. *IEEE Trans. Software Eng. 31*, 3 (2005), 226–237.

[122] Errington, T. M., Denis, A., Perfito, N., Iorns, E., and Nosek, B. A. Reproducibility in Cancer Biology: Challenges for Assessing Replicability in Preclinical Cancer Biology. *eLife 10* (Dec. 2021), e67995. https://doi.org/10.7554/eLife.67995.

[123] Faggioli, G., and Ferro, N. System Effect Estimation by Sharding: A Comparison between ANOVA Approaches to Detect Significant Differences. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 33–46.

[124] Faggioli, G., Zendel, O., Culpepper, J. S., Ferro, N., and Scholer, F. An Enhanced Evaluation Framework for Query Performance Prediction. In *ECIR (1)* (2021), vol. 12656 of *Lecture Notes in Computer Science*, Springer, pp. 115–129.

[125] Fanelli, D. How Many Scientists Fabricate and Falsify Research? A Systematic Review and Meta-Analysis of Survey Data. *PLOS ONE 4*, 5 (May 2009), 1–11. https://doi.org/10.1371/journal.pone.0005738.

[126] Färber, M. Analyzing the GitHub Repositories of Research Papers. In *JCDL* (2020), ACM, pp. 491–492.

[127] Feitelson, D. G. From Repeatability to Reproducibility and Corroboration. *ACM SIGOPS Oper. Syst. Rev. 49*, 1 (2015), 3–11.

[128] Fernández-Pichel, M., Losada, D. E., Pichel, J. C., and Elsweiler, D. Reliability Prediction for Health-Related Content: A Replicability Study. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 47–61.

[129] Ferrante, M., Ferro, N., and Fuhr, N. Towards Meaningful Statements in IR Evaluation: Mapping Evaluation Measures to Interval Scales. *IEEE Access 9* (2021), 136182–136216.

[130] Ferro, N. Reproducibility Challenges in Information Retrieval Evaluation. *ACM J. Data Inf. Qual. 8*, 2 (2017), 8:1–8:4.

[131] Ferro, N., Crestani, F., Moens, M.-F., Mothe, J., Silvestri, F., Nunzio, G. M. D., Hauff, C., and Silvello, G., Eds. *Advances in Information Retrieval - 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20-23, 2016. Proceedings*, vol. 9626 of *Lecture Notes in Computer Science*. Springer, 2016. https://doi.org/10.1007/978-3-319-30671-1.

[132] Ferro, N., Fuhr, N., Järvelin, K., Kando, N., Lippold, M., and Zobel, J. Increasing Reproducibility in IR: Findings from the Dagstuhl Seminar on "Reproducibility of Data-Oriented Experiments in e-Science". *SIGIR Forum 50*, 1 (2016), 68–82.

[133] Ferro, N., Fuhr, N., Maistro, M., Sakai, T., and Soboroff, I. Overview of CENTRE@CLEF 2019: Sequel in the Systematic Reproducibility Realm. In *CLEF* (2019), vol. 11696 of *Lecture Notes in Computer Science*, Springer, pp. 287–300.

[134] Ferro, N., and Kelly, D. SIGIR Initiative to Implement ACM Artifact Review and Badging. *SIGIR Forum 52*, 1 (2018), 4–10.

[135] Ferro, N., Kim, Y., and Sanderson, M. Using Collection Shards to Study Retrieval Performance Effect Sizes. *ACM Trans. Inf. Syst. 37*, 3 (2019), 30:1–30:40.

[136] Ferro, N., Maistro, M., Sakai, T., and Soboroff, I. Overview of CENTRE@CLEF 2018: A First Tale in the Systematic Reproducibility Realm. In *CLEF* (2018), vol. 11018 of *Lecture Notes in Computer Science*, Springer, pp. 239–246.

[137] Ferro, N., Marchesin, S., Purpura, A., and Silvello, G. A Docker-Based Replicability Study of a Neural Information Retrieval Model. In *OSIRRC@SIGIR* (2019), vol. 2409 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 37–43.

[138] Ferro, N., and Peters, C., Eds. *Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF*, vol. 41 of *The Information Retrieval Series*. Springer, 2019.

[139] Ferro, N., and Sanderson, M. Sub-Corpora Impact on System Effectiveness. In *SIGIR* (2017), ACM, pp. 901–904.

[140] Ferro, N., and Sanderson, M. Improving the Accuracy of System Performance Estimation by Using Shards. In *SIGIR* (2019), ACM, pp. 805–814.

[141] Ferro, N., and Sanderson, M. How Do You Test a Test?: A Multifaceted Examination of Significance Tests. In *WSDM* (2022), ACM, pp. 280–288.

[142] Ferro, N., and Silvello, G. Rank-Biased Precision Reloaded: Reproducibility and Generalization. In *ECIR* (2015), vol. 9022 of *Lecture Notes in Computer Science*, pp. 768–780.

[143] Ferro, N., and Silvello, G. A General Linear Mixed Models Approach to Study System Component Effects. In *SIGIR* (2016), ACM, pp. 25–34.

[144] Fomel, S. Reproducible Research as a Community Effort: Lessons from the Madagascar Project. *Computing in Science & Engineering 17*, 1 (2015), 20–26.

[145] Forde, J., Head, T., Holdgraf, C., Panda, Y., Nalvarete, G., Ragan-Kelley, B., and Sundell, E. Reproducible Research Environments with Repo2docker. In *Workshop on Reproducibility in Machine Learning RML@ICML 2018* (2018).

[146] Freire, J., Fuhr, N., and Rauber, A. Reproducibility of Data-Oriented Experiments in e-Science (Dagstuhl Seminar 16041). *Dagstuhl Reports 6*, 1 (2016), 108–159.

[147] Fröbe, M., Bittner, J. P., Potthast, M., and Hagen, M. The Effect of Content-Equivalent near-Duplicates on the Evaluation of Search Engines. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 12–19.

[148] Fröbe, M., Günther, S., Probst, M., Potthast, M., and Hagen, M. The Power of Anchor Text in the Neural Retrieval Era. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 567–583.

[149] Fuhr, N. Some Common Mistakes in IR Evaluation, and How They Can Be Avoided. *SIGIR Forum 51*, 3 (2017), 32–41.

[150] Fuhr, N. Proof by Experimentation? Towards Better IR Research. In *SIGIR* (2020), ACM, p. 2.

[151] Gäde, M., Koolen, M., Hall, M. M., Bogers, T., and Petras, V. A Manifesto on Resource Re-Use in Interactive Information Retrieval. In *CHIIR* (2021), ACM, pp. 141–149.

[152] García, J. A., Rodríguez-Sánchez, R., and Fdez-Valdivia, J. Confirmatory Bias in Peer Review. *Scientometrics 123*, 1 (2020), 517–533.

[153] García, J. A., Rodríguez-Sánchez, R., and Fernández-Valdivia, J. Bias and Effort in Peer Review. *J. Assoc. Inf. Sci. Technol. 66*, 10 (2015), 2020–2030.

[154] Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H. M., III, H. D., and Crawford, K. Datasheets for Datasets. *Commun. ACM 64*, 12 (2021), 86–92.

[155] Gingstad, K., Jekteberg, Ø., and Balog, K. ArXivDigest: A Living Lab for Personalized Scientific Literature Recommendation. In *CIKM* (2020), ACM, pp. 3393–3396.

[156] Gorp, P. V., and Mazanek, S. SHARE: A Web Portal for Creating and Sharing Executable Research Papers. In *ICCS* (2011), vol. 4 of *Procedia Computer Science*, Elsevier, pp. 589–597.

[157] Grand, A., Muir, R., Ferenczi, J., and Lin, J. From MAXSCORE to Block-Max Wand: The Story of How Lucene Significantly Improved Query Evaluation Performance. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 20–27.

[158] Granka, L. A., Joachims, T., and Gay, G. Eye-Tracking Analysis of User Behavior in WWW Search. In *SIGIR* (2004), ACM, pp. 478–479.

[159] Grossman, M. R., and Cormack, G. V. MRG_UWaterloo and WaterlooCormack Participation in the TREC 2017 Common Core Track. In *TREC* (2017), vol. 500–324 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[160] Grossman, M. R., and Cormack, G. V. MRG_UWaterloo Participation in the TREC 2018 Common Core Track. In *TREC* (2018), vol. 500–331 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[161] Grotov, A., Chuklin, A., Markov, I., Stout, L., Xumara, F., and de Rijke, M. A Comparative Study of Click Models for Web Search. In *CLEF* (2015), vol. 9283 of *Lecture Notes in Computer Science*, Springer, pp. 78–90.

[162] Guan, D., Zhang, S., and Yang, H. Utilizing Query Change for Session Search. In *SIGIR* (2013), ACM, pp. 453–462.

[163] Guervós, J. J. M. Agile (Data) Science: A (Draft) Manifesto. *CoRR abs/2104.12545* (2021).

[164] Günther, S., and Hagen, M. Assessing Query Suggestions for Search Session Simulation. *Sim4IR: The SIGIR 2021 Workshop on Simulation for Information Retrieval Evaluation* (2021). http://ceur-ws.org/Vol-2911/paper6.pdf.

[165] Guo, F., Liu, C., and Wang, Y. M. Efficient Multiple-Click Models in Web Search. In *WSDM* (2009), ACM, pp. 124–131.

[166] Gysel, C. V., and de Rijke, M. Pytrec_eval: An Extremely Fast Python Interface to Trec_eval. In *SIGIR* (2018), K. Collins-Thompson, Q. Mei, B. D. Davison, Y. Liu, and E. Yilmaz, Eds., ACM, pp. 873–876. https://doi.org/10.1145/3209978.3210065.

[167] Gysel, C. V., Kanoulas, E., and de Rijke, M. Lexical Query Modeling in Session Search. In *ICTIR* (2016), ACM, pp. 69–72.

[168] Haak, L. L., Fenner, M., Paglione, L., Pentz, E., and Ratner, H. ORCID: A System to Uniquely Identify Researchers. *Learned Publishing 25*, 4 (2012), 259–264.

[169] Hagen, M., Potthast, M., Büchner, M., and Stein, B. Twitter Sentiment Detection via Ensemble Classification Using Averaged Confidence Scores. In *ECIR* (2015), vol. 9022 of *Lecture Notes in Computer Science*, pp. 741–754.

[170] Hagen, M., Verberne, S., Macdonald, C., Seifert, C., Balog, K., Nørvåg, K., and Setty, V., Eds. *Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part I*, vol. 13185 of *Lecture Notes in Computer Science*. Springer, 2022. https://doi.org/10.1007/978-3-030-99736-6.

[171] Hanbury, A., Kazai, G., Rauber, A., and Fuhr, N., Eds. *Advances in Information Retrieval - 37th European Conference on IR Research, ECIR 2015, Vienna, Austria, March 29 - April 2, 2015. Proceedings*, vol. 9022 of *Lecture Notes in Computer Science*. 2015. https://doi.org/10.1007/978-3-319-16354-3.

[172] Harman, D. K., Ed. *Proceedings of the First Text REtrieval Conference, TREC 1992, Gaithersburg, Maryland, USA, November 4-6, 1992*, vol. 500–207 of *NIST Special Publication*. National Institute of Standards and Technology (NIST), 1992.

[173] Hasibi, F., Balog, K., and Bratsberg, S. E. On the Reproducibility of the TAGME Entity Linking System. In *ECIR* (2016), vol. 9626 of *Lecture Notes in Computer Science*, Springer, pp. 436–449.

[174] Hawking, D., von Billerbeck, B., Thomas, P., and Craswell, N. *Simulating Information Retrieval Test Collections*. Morgan & Claypool, 2020.

[175] He, J., Zhai, C., and Li, X. Evaluation of Methods for Relative Comparison of Retrieval Systems Based on Clickthroughs. In *CIKM* (2009), ACM, pp. 2029–2032.

[176] He, Y., Tang, J., Ouyang, H., Kang, C., Yin, D., and Chang, Y. Learning to Rewrite Queries. In *CIKM* (2016), ACM, pp. 1443–1452.

[177] Herdagdelen, A., Ciaramita, M., Mahler, D., Holmqvist, M., Hall, K. B., Riezler, S., and Alfonseca, E. Generalized Syntactic and Semantic Models of Query Reformulation. In *SIGIR* (2010), ACM, pp. 283–290.

[178] Heroux, M. A., Barba, L., Parashar, M., Stodden, V., and Taufer, M. Toward a Compatible Reproducibility Taxonomy for Computational and Computing Sciences. https://www.osti.gov/biblio/1481626.

[179] Hersh, W. R., Turpin, A., Price, S., Chan, B., Kraemer, D., Sacherek, L., and Olson, D. Do Batch and User Evaluation Give the Same Results? In *SIGIR* (2000), ACM, pp. 17–24.

[180] Hersh, W. R., Turpin, A., Price, S., Kraemer, D., Chan, B., Sacherek, L., and Olson, D. Do Batch and User Evaluations Give the Same Results? An Analysis from the TREC-8 Interactive Track. In *TREC*

(1999), vol. 500–246 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[181] Hersh, W. R., Turpin, A., Sacherek, L., Olson, D., Price, S., Chan, B., and Kraemer, D. Further Analysis of Whether Batch and User Evaluations Give the Same Results with a Question-Answering Task. In *TREC* (2000), vol. 500–249 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[182] Heumüller, R., Nielebock, S., Krüger, J., and Ortmeier, F. Publish or Perish, but Do Not Forget Your Software Artifacts. *Empirical Software Engineering 25*, 6 (2020), 4585–4616.

[183] Hiemstra, D., Moens, M.-F., Mothe, J., Perego, R., Potthast, M., and Sebastiani, F., Eds. *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part I*, vol. 12656 of *Lecture Notes in Computer Science*. Springer, 2021. https://doi.org/10.1007/978-3-030-72113-8.

[184] Hofman, J. M., Goldstein, D. G., Sen, S., and Poursabzi-Sangdeh, F. Expanding the Scope of Reproducibility Research through Data Analysis Replications. In *WWW (Companion Volume)* (2020), ACM / IW3C2, pp. 567–571.

[185] Hofmann, K. Fast and Reliable Online Learning to Rank for Information Retrieval. *SIGIR Forum 47*, 2 (2013), 140.

[186] Hofmann, K., Li, L., and Radlinski, F. Online Evaluation for Information Retrieval. *Found. Trends Inf. Retr. 10*, 1 (2016), 1–117.

[187] Hofmann, K., Whiteson, S., and de Rijke, M. Fidelity, Soundness, and Efficiency of Interleaved Comparison Methods. *ACM Trans. Inf. Syst. 31*, 4 (2013), 17:1–17:43.

[188] Hofstätter, S., Althammer, S., Sertkan, M., and Hanbury, A. Establishing Strong Baselines for TripClick Health Retrieval. In *ECIR (2)* (2022), vol. 13186 of *Lecture Notes in Computer Science*, Springer, pp. 144–152.

[189] Holland, S., Hosny, A., Newman, S., Joseph, J., and Chmielinski, K. The Dataset Nutrition Label: A Framework to Drive Higher Data Quality Standards. *CoRR abs/1805.03677* (2018).

[190] Hopfgartner, F., Hanbury, A., Müller, H., Eggel, I., Balog, K., Brodt, T., Cormack, G. V., Lin, J., Kalpathy-Cramer, J., Kando, N., Kato, M. P., Krithara, A., Gollub, T., Potthast, M., Viegas, E., and Mercer, S. Evaluation-as-a-Service for the Computational Sciences: Overview and Outlook. *ACM J. Data Inf. Qual. 10*, 4 (2018), 15:1–15:32.

[191] Hopfgartner, F., Kille, B., Lommatzsch, A., Plumbaum, T., Brodt, T., and Heintz, T. Benchmarking News Recommendations in a Living Lab. In *CLEF* (2014), vol. 8685 of *Lecture Notes in Computer Science*, Springer, pp. 250–267.

[192] Huang, J., Oosterhuis, H., Cetinkaya, B., Rood, T., and de Rijke, M. State Encoders in Reinforcement Learning for Recommendation: A Reproducibility Study. In *SIGIR* (2022), ACM, pp. 2738–2748.

[193] Huang, J., Oosterhuis, H., de Rijke, M., and van Hoof, H. Keeping Dataset Biases out of the Simulation: A Debiased Simulator for Reinforcement Learning Based Recommender Systems. In *RecSys* (2020), ACM, pp. 190–199.

[194] Hutson, M. Artificial Intelligence Faces Reproducibility Crisis. *Science (New York, N.Y.) 359*, 6377 (2018), 725–726.

[195] Huurnink, B., Hofmann, K., de Rijke, M., and Bron, M. Validating Query Simulators: An Experiment Using Commercial Searches and Purchases. In *CLEF* (2010), vol. 6360 of *Lecture Notes in Computer Science*, Springer, pp. 40–51.

[196] Ingram, W. A., and Fox, E. A. Preparing Code and Data for Computational Reproducibility. In *JCDL* (2020), ACM, pp. 565–566.

[197] Ingwersen, P. Cognitive Perspectives of Information Retrieval Interaction: Elements of a Cognitive IR Theory. *J. Documentation 52*, 1 (1996), 3–50.

[198] Ioannidis, J. P. A. Why Most Published Research Findings Are False. *PLOS Medicine 2*, 8 (Aug. 2005). https://doi.org/10.1371/journal.pmed.0020124.

[199] Ivie, P., and Thain, D. Reproducibility in Scientific Computing. *ACM Comput. Surv. 51*, 3 (2018), 63:1–63:36.

[200] Jagerman, R., Balog, K., and de Rijke, M. OpenSearch: Lessons Learned from an Online Evaluation Campaign. *ACM J. Data Inf. Qual. 10*, 3 (2018), 13:1–13:15.

[201] Jansen, B. J., Booth, D. L., and Spink, A. Patterns of Query Reformulation during Web Searching. *J. Assoc. Inf. Sci. Technol. 60*, 7 (2009), 1358–1371.

[202] Järvelin, K., and Kekäläinen, J. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Trans. Inf. Syst. 20*, 4 (2002), 422–446.

[203] Järvelin, K., Price, S. L., Delcambre, L. M. L., and Nielsen, M. L. Discounted Cumulated Gain Based Evaluation of Multiple-Query IR Sessions. In *ECIR* (2008), vol. 4956 of *Lecture Notes in Computer Science*, Springer, pp. 4–15.

[204] Jiang, J., and Allan, J. Correlation between System and User Metrics in a Session. In *CHIIR* (2016), ACM, pp. 285–288.

[205] Jimenez, I., Arpaci-Dusseau, A. C., Arpaci-Dusseau, R. H., Lofstead, J. F., Maltzahn, C., Mohror, K., and Ricci, R. PopperCI: Automated Reproducibility Validation. In *INFOCOM Workshops* (2017), IEEE, pp. 450–455.

[206] Joachims, T., Granka, L. A., Pan, B., Hembrooke, H., and Gay, G. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *SIGIR* (2005), ACM, pp. 154–161.

[207] Joachims, T., Swaminathan, A., and Schnabel, T. Unbiased Learning-to-Rank with Biased Feedback. In *WSDM* (2017), ACM, pp. 781–789.

[208] Jones, K. S. A Statistical Interpretation of Term Specificity and Its Application in Retrieval. *J. Documentation 60*, 5 (2004), 493–502.

[209] Jones, R., Rey, B., Madani, O., and Greiner, W. Generating Query Substitutions. In *WWW* (2006), ACM, pp. 387–396.

[210] Jones, T., Turpin, A., Mizzaro, S., Scholer, F., and Sanderson, M. Size and Source Matter: Understanding Inconsistencies in Test Collection-Based Evaluation. In *CIKM* (2014), ACM, pp. 1843–1846.

[211] Jordan, C., Watters, C. R., and Gao, Q. Using Controlled Query Generation to Evaluate Blind Relevance Feedback Algorithms. In *JCDL* (2006), ACM, pp. 286–295.

[212] Jose, J. M., Yilmaz, E., Magalhães, J., Castells, P., Ferro, N., Silva, M. J., and Martins, F., Eds. *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part I*, vol. 12035 of *Lecture Notes in Computer Science*. Springer, 2020. https://doi.org/10.1007/978-3-030-45439-5.

[213] Kamphuis, C., de Vries, A. P., Boytsov, L., and Lin, J. Which BM25 Do You Mean? A Large-Scale Reproducibility Study of Scoring Variants. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 28–34.

[214] Kamps, J., Koolen, M., and Trotman, A. Comparative Analysis of Clicks and Judgments for IR Evaluation. In *WSCD@WSDM* (2009), ACM, pp. 80–87.

[215] Kando, N., Ed. *Proceedings of the First NTCIR Workshop on Research in Japanese Text Retrieval and Term Recognition, NTCIR-1, Tokyo, Japan, August 30 - September 1, 1999*. National Center for Science Information Systems (NACSIS), 1999.

[216] Kapoor, S., and Narayanan, A. Leakage and the Reproducibility Crisis in ML-based Science. https://arxiv.org/abs/2207.07048, 2022.

[217] Keller, J., and Munz, L. P. M. TEKMA at CLEF-2021: BM-25 Based Rankings for Scientific Publication Retrieval and Data Set Recommendation. In *CLEF (Working Notes)* (2021), vol. 2936 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 1700–1711.

[218] Kelly, D. Methods for Evaluating Interactive Information Retrieval Systems with Users. *Found. Trends Inf. Retr. 3*, 1-2 (2009), 1–224.

[219] Kelly, D., and Azzopardi, L. How Many Results per Page?: A Study of SERP Size, Search Behavior and User Experience. In *SIGIR* (2015), ACM, pp. 183–192.

[220] Kelly, D., Dumais, S. T., and Pedersen, J. O. Evaluation Challenges and Directions for Information-Seeking Support Systems. *Computer 42*, 3 (2009), 60–66.

[221] Kendall, M. G. *Rank Correlation Methods*. Griffin, Oxford, England, 1948.

[222] Keskustalo, H., Järvelin, K., Pirkola, A., Sharma, T., and Lykke, M. Test Collection-Based IR Evaluation Needs Extension toward Sessions - A Case of Extremely Short Queries. In *AIRS* (2009), vol. 5839 of *Lecture Notes in Computer Science*, Springer, pp. 63–74.

[223] Khandel, P., Markov, I., Yates, A., and Varbanescu, A. L. ParClick: A Scalable Algorithm for EM-based Click Models. In *WWW* (2022), ACM, pp. 392–400.

[224] Kharitonov, E., Macdonald, C., Serdyukov, P., and Ounis, I. Optimised Scheduling of Online Experiments. In *SIGIR* (2015), ACM, pp. 453–462.

[225] Kieffer, S. ECOVAL: Ecological Validity of Cues and Representative Design in User Experience Evaluations. *AIS Trans. Hum. Comput. Interact. 9*, 2 (2017), 4.

[226] Kiesel, J., Meyer, L., Kneist, F., Stein, B., and Potthast, M. An Empirical Comparison of Web Page Segmentation Algorithms. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 62–74.

[227] Kiseleva, J., Williams, K., Awadallah, A. H., Crook, A. C., Zitouni, I., and Anastasakos, T. Predicting User Satisfaction with Intelligent Assistants. In *SIGIR* (2016), ACM, pp. 45–54.

[228] Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J. B., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., Willing, C., and Team, J. D. Jupyter Notebooks - a Publishing Format for Reproducible Computational Workflows. In *ELPUB* (2016), IOS Press, pp. 87–90.

[229] Kohavi, R., Deng, A., Frasca, B., Longbotham, R., Walker, T., and Xu, Y. Trustworthy Online Controlled Experiments: Five Puzzling Outcomes Explained. In *KDD* (2012), ACM, pp. 786–794.

[230] Kohavi, R., Deng, A., Longbotham, R., and Xu, Y. Seven Rules of Thumb for Web Site Experimenters. In *KDD* (2014), ACM, pp. 1857–1866.

[231] Kohavi, R., Longbotham, R., Sommerfield, D., and Henne, R. M. Controlled Experiments on the Web: Survey and Practical Guide. *Data Mining and Knowledge Discovery 18*, 1 (2009), 140–181.

[232] Kohavi, R., Tang, D., and Xu, Y. *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing*. Cambridge University Press, Cambridge, 2020.

[233] Kowald, D., Schedl, M., and Lex, E. The Unfairness of Popularity Bias in Music Recommendation: A Reproducibility Study. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 35–42.

[234] Kriegel, H.-P., Schubert, E., and Zimek, A. The (Black) Art of Runtime Evaluation: Are We Comparing Algorithms or Implementations? *Knowledge and Information Systems 52*, 2 (2017), 341–378.

[235] Kruger, J., and Dunning, D. Unskilled and Unaware of It: How Difficulties in Recognizing One's Own Incompetence Lead to Inflated Self-Assessments. *Journal of personality and social psychology 77 6* (1999), 1121–34.

[236] Kuhlthau, C. C. *Seeking Meaning: A Process Approach to Library and Information Services*, vol. 2. Libraries Unlimited Westport, CT, 2004.

[237] Kusa, W., Hanbury, A., and Knoth, P. Automation of Citation Screening for Systematic Literature Reviews Using Neural Networks: A Replicability Study. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 584–598.

[238] Labhishetty, S., and Zhai, C. An Exploration of Tester-Based Evaluation of User Simulators for Comparing Interactive Retrieval Systems. In *SIGIR* (2021), ACM, pp. 1598–1602.

[239] Labhishetty, S., and Zhai, C. RATE: A Reliability-Aware Tester-Based Evaluation Framework of User Simulators. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 336–350.

[240] Lee, C. J., Sugimoto, C. R., Zhang, G., and Cronin, B. Bias in Peer Review. *J. Assoc. Inf. Sci. Technol. 64*, 1 (2013), 2–17.

[241] Leipzig, J., Nüst, D., Hoyt, C. T., Ram, K., and Greenberg, J. The Role of Metadata in Reproducible Computational Research. *Patterns 2*, 9 (2021), 100322.

[242] Li, H., Lu, H., Huang, S., Ma, W., Zhang, M., Liu, Y., and Ma, S. Privacy-Aware Remote Information Retrieval User Experiments Logging Tool. In *SIGIR* (2021), ACM, pp. 2615–2619.

[243] Li, H., Zhuang, S., Mourad, A., Ma, X., Lin, J., and Zuccon, G. Improving Query Representations for Dense Retrieval with Pseudo Relevance Feedback: A Reproducibility Study. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 599–612.

[244] Li, L., Kim, J., and Zitouni, I. Toward Predicting the Outcome of an A/B Experiment for Search Relevance. In *WSDM* (2015), ACM, pp. 37–46.

[245] Li, S., Abbasi-Yadkori, Y., Kveton, B., Muthukrishnan, S., Vinay, V., and Wen, Z. Offline Evaluation of Ranking Policies with Click Models. In *KDD* (2018), ACM, pp. 1685–1694.

[246] Liberman, M. Fred Jelinek. *Comput. Linguistics 36*, 4 (2010), 595–599.

[247] Lin, J. The Neural Hype and Comparisons against Weak Baselines. *SIGIR Forum 52*, 2 (2018), 40–51.

[248] Lin, J. A Proposed Conceptual Framework for a Representational Approach to Information Retrieval. *SIGIR Forum 55*, 2 (2021), 4:1–4:29.

[249] Lin, J., Crane, M., Trotman, A., Callan, J., Chattopadhyaya, I., Foley, J., Ingersoll, G., MacDonald, C., and Vigna, S. Toward Reproducible Baselines: The Open-Source IR Reproducibility Challenge. In *ECIR* (2016), vol. 9626 of *Lecture Notes in Computer Science*, Springer, pp. 408–420.

[250] Lin, J., Ma, X., Lin, S.-C., Yang, J.-H., Pradeep, R., and Nogueira, R. Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations. In *SIGIR* (2021), ACM, pp. 2356–2362.

[251] Lin, J., Mackenzie, J. M., Kamphuis, C., Macdonald, C., Mallia, A., Siedlaczek, M., Trotman, A., and de Vries, A. P. Supporting Interoperability between Open-Source Search Engines with the Common Index File Format. In *SIGIR* (2020), ACM, pp. 2149–2152.

[252] Lin, J., Nogueira, R., and Yates, A. *Pretrained Transformers for Text Ranking: BERT and Beyond*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2021.

[253] Lin, J., and Yang, P. The Impact of Score Ties on Repeatability in Document Ranking. In *SIGIR* (2019), ACM, pp. 1125–1128.

[254] Lin, J., and Zhang, Q. Reproducibility Is a Process, Not an Achievement: The Replicability of IR Reproducibility Experiments. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 43–49.

[255] Lipton, Z. C., and Steinhardt, J. Troubling Trends in Machine Learning Scholarship. *ACM Queue 17*, 1 (2019), 80.

[256] Liu, B., Craswell, N., Lu, X., Kurland, O., and Culpepper, J. S. A Comparative Analysis of Human and Automatic Query Variants. In *ICTIR* (2019), ACM, pp. 47–50.

[257] Liu, J., Sarkar, S., and Shah, C. Identifying and Predicting the States of Complex Search Tasks. In *CHIIR* (2020), ACM, pp. 193–202.

[258] Liu, T.-Y. Learning to Rank for Information Retrieval. *Found. Trends Inf. Retr. 3*, 3 (2009), 225–331.

[259] Liu, Y., Xie, X., Wang, C., Nie, J.-Y., Zhang, M., and Ma, S. Time-Aware Click Model. *ACM Trans. Inf. Syst. 35*, 3 (2017), 16:1–16:24.

[260] Lucic, A., Bleeker, M. J. R., de Rijke, M., Sinha, K., Jullien, S., and Stojnic, R. Towards Reproducible Machine Learning Research in Information Retrieval. In *SIGIR* (2022), ACM, pp. 3459–3461.

[261] Lucic, A., Bleeker, M. J. R., Jullien, S., Bhargav, S., and de Rijke, M. Reproducibility as a Mechanism for Teaching Fairness, Accountability, Confidentiality, and Transparency in Artificial Intelligence. In *AAAI* (2022), AAAI Press, pp. 12792–12800.

[262] Ludewig, M., and Jannach, D. Evaluation of Session-Based Recommendation Algorithms. *User Model. User Adapt. Interact. 28*, 4-5 (2018), 331–390.

[263] Ma, X., Sun, K., Pradeep, R., Li, M., and Lin, J. Another Look at DPR: Reproduction of Training and Replication of Retrieval. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 613–626.

[264] MacAvaney, S., Macdonald, C., and Ounis, I. Reproducing Personalised Session Search over the AOL Query Log. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 627–640.

[265] MacAvaney, S., Yates, A., Feldman, S., Downey, D., Cohan, A., and Goharian, N. Simplified Data Wrangling with `ir_datasets`. In *SIGIR* (2021), ACM, pp. 2429–2436.

[266] Macdonald, C., McCreadie, R., Santos, R. L. T., and Ounis, I. From Puppy to Maturity: Experiences in Developing Terrier. In *OSIR@SIGIR* (2012), University of Otago, Dunedin, New Zealand, pp. 60–63.

[267] Macdonald, C., and Tonellotto, N. Declarative Experimentation in Information Retrieval Using PyTerrier. In *ICTIR* (2020), ACM, pp. 161–168.

[268] Macefield, R. Usability Studies and the Hawthorne Effect. *Journal of usability studies 2*, 3 (2007), 145–154.

[269] Mackenzie, J., and Moffat, A. Modality Effects When Simulating User Querying Tasks. In *ICTIR* (2021), ACM, pp. 197–201.

[270] Mackenzie, J. M., Mallia, A., Petri, M., Culpepper, J. S., and Suel, T. Compressing Inverted Indexes with Recursive Graph Bisection: A Reproducibility Study. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 339–352.

[271] Mackie, S., McCreadie, R., Macdonald, C., and Ounis, I. Experiments in Newswire Summarisation. In *ECIR* (2016), vol. 9626 of *Lecture Notes in Computer Science*, Springer, pp. 421–435.

[272] Majumder, P., Mitra, M., Pal, D., Bandyopadhyay, A., Maiti, S., Mitra, S., Sen, A., and Pal, S. Text Collections for FIRE. In *SIGIR* (2008), ACM, pp. 699–700.

[273] Makridakis, S., Spiliotis, E., and Assimakopoulos, V. Statistical and Machine Learning Forecasting Methods: Concerns and Ways Forward. *PLOS ONE 13*, 3 (Mar. 2018), 1–26. https://doi.org/10.1371/journal.pone.0194889.

[274] Malkevich, S., Markov, I., Michailova, E., and de Rijke, M. Evaluating and Analyzing Click Simulation in Web Search. In *ICTIR* (2017), ACM, pp. 281–284.

[275] Mallia, A., Siedlaczek, M., Mackenzie, J. M., and Suel, T. PISA: Performant Indexes and Search for Academia. In *OSIRRC@SIGIR* (2019), vol. 2409 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 50–56.

[276] Mallia, A., Siedlaczek, M., and Suel, T. An Experimental Study of Index Compression and DAAT Query Processing Methods. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 353–368.

[277] Manotumruksa, J., Rafailidis, D., Macdonald, C., and Ounis, I. On Cross-Domain Transfer in Venue Recommendation. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 443–456.

[278] Mao, J., Chu, Z., Liu, Y., Zhang, M., and Ma, S. Investigating the Reliability of Click Models. In *ICTIR* (2019), ACM, pp. 125–128.

[279] Marchionini, G. *Information Seeking in Electronic Environments*. No. 9. Cambridge University Press, 1997.

[280] Markov, I., Borisov, A., and de Rijke, M. Online Expectation-Maximization for Click Models. In *CIKM* (2017), ACM, pp. 2195–2198.

[281] Marrero, M. APONE: Academic Platform for ONline Experiments. In *DESIRES* (2018), vol. 2167 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 47–53.

[282] Marrero, M., and Hauff, C. A/B Testing with APONE. In *SIGIR* (2018), ACM, pp. 1269–1272.

[283] Maurera, F. B. P., Dacrema, M. F., and Cremonesi, P. An Evaluation Study of Generative Adversarial Networks for Collaborative Filtering. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 671–685.

[284] Maxwell, D. *Modelling Search and Stopping in Interactive Information Retrieval*. PhD thesis, University of Glasgow, UK, 2019.

[285] Maxwell, D., and Azzopardi, L. Agents, Simulated Users and Humans: An Analysis of Performance and Behaviour. In *CIKM* (2016), ACM, pp. 731–740.

[286] Maxwell, D., and Azzopardi, L. Simulating Interactive Information Retrieval: SimIIR: A Framework for the Simulation of Interaction. In *SIGIR* (2016), ACM, pp. 1141–1144.

[287] Maxwell, D., and Azzopardi, L. Information Scent, Searching and Stopping - Modelling SERP Level Stopping Behaviour. In *ECIR* (2018), vol. 10772 of *Lecture Notes in Computer Science*, Springer, pp. 210–222.

[288] Maxwell, D., Azzopardi, L., Järvelin, K., and Keskustalo, H. Searching and Stopping: An Analysis of Stopping Rules and Strategies. In *CIKM* (2015), ACM, pp. 313–322.

[289] Maxwell, D., and Hauff, C. LogUI: Contemporary Logging Infrastructure for Web-Based Experiments. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 525–530.

[290] Mayr, P. Sowiport User Search Sessions Data Set (SUSS). GESIS - Leibniz-Institute for the Social Sciences. Data File Version 1.0.0, https://doi.org/10.7802/1380, 2016.

[291] McPhillips, T. M., Song, T., Kolisnik, T., Aulenbach, S., Belhajjame, K., Bocinsky, K., Cao, Y., Chirigati, F., Dey, S. C., Freire, J., Huntzinger, D. N., Jones, C., Koop, D., Missier, P., Schildhauer, M., Schwalm, C. R., Wei, Y., Cheney, J., Bieda, M., and Ludäscher, B. YesWorkflow: A User-Oriented, Language-Independent Tool for Recovering Workflow Information from Scripts. *CoRR abs/1502.02403* (2015).

[292] Miksa, T., and Rauber, A. Using Ontologies for Verification and Validation of Workflow-Based Experiments. *Journal of Web Semantics 43* (2017), 25–45.

[293] Mitchell, M., and Jolley, J. *Research Design Explained*. Wadsworth Cengage Learning, 2010.

[294] Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., and Gebru, T. Model Cards for Model Reporting. In *FAT* (2019), ACM, pp. 220–229.

[295] Mitra, B., and Craswell, N. An Introduction to Neural Information Retrieval. *Found. Trends Inf. Retr. 13*, 1 (2018), 1–126.

[296] Moffat, A., Bailey, P., Scholer, F., and Thomas, P. Incorporating User Expectations and Behavior into the Measurement of Search Effectiveness. *ACM Trans. Inf. Syst. 35*, 3 (2017), 24:1–24:38.

[297] Moffat, A., Scholer, F., Thomas, P., and Bailey, P. Pooled Evaluation over Query Variations: Users Are as Diverse as Systems. In *CIKM* (2015), ACM, pp. 1759–1762.

[298] Moffat, A., and Zobel, J. Rank-Biased Precision for Measurement of Retrieval Effectiveness. *ACM Trans. Inf. Syst. 27*, 1 (2008), 2:1–2:27.

[299] Mölder, F., Jablonski, KP., Letcher, B., Hall, MB., Tomkins-Tinch, CH., Sochat, V., Forster, J., Lee, S., Twardziok, SO., Kanitz, A., Wilm, A., Holtgrewe, M., Rahmann, S., Nahnsen, S.,

and Köster, J. Sustainable Data Analysis with Snakemake. *F1000Research 10*, 33 (2021).

[300] Moreo, A., and Sebastiani, F. Re-Assessing the "Classify and Count" Quantification Method. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 75–91.

[301] Mühleisen, H., Samar, T., Lin, J., and de Vries, A. P. Old Dogs Are Great at New Tricks: Column Stores for Ir Prototyping. In *SIGIR* (2014), ACM, pp. 863–866.

[302] Mukherjee, R., Shetty, S., Chattopadhyay, S., Maji, S., Datta, S., and Goyal, P. Reproducibility, Replicability and beyond: Assessing Production Readiness of Aspect Based Sentiment Analysis in the Wild. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 92–106.

[303] Müllner, P., Kowald, D., and Lex, E. Robustness of Meta Matrix Factorization against Strict Privacy Constraints. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 107–119.

[304] Munafò, M. R., Nosek, B. A., Bishop, D. V. M., Button, K. S., Chambers, C. D., Percie du Sert, N., Simonsohn, U., Wagenmakers, E.-J., Ware, J. J., and Ioannidis, J. P. A. A Manifesto for Reproducible Science. *Nature Human Behaviour 1*, 1 (Jan. 2017), 0021. https://doi.org/10.1038/s41562-016-0021.

[305] Murta, L., Braganholo, V., Chirigati, F., Koop, D., and Freire, J. noWorkflow: Capturing and Analyzing Provenance of Scripts. In *IPAW* (2014), vol. 8628 of *Lecture Notes in Computer Science*, Springer, pp. 71–83.

[306] Nagarajan, P., Warnell, G., and Stone, P. The Impact of Nondeterminism on Reproducibility in Deep Reinforcement Learning.

[307] National Academies of Sciences Engineering and Medicine, Engineering, and Medicine. *Reproducibility and Replicability in Science*. The National Academies Press, Washington, DC, 2019. https://nap.nationalacademies.org/catalog/25303/reproducibility-and-replicability-in-science.

[308] Neophytou, N., Mitra, B., and Stinson, C. Revisiting Popularity and Demographic Biases in Recommender Evaluation and Effectiveness. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 641–654.

[309] Nosek, B. A., Ebersole, C. R., DeHaven, A. C., and Mellor, D. T. The Preregistration Revolution. *Proceedings of the National Academy of Sciences 115*, 11 (Mar. 2018), 2600–2606. https://doi.org/10.1073/pnas.1708274114.

[310] Nunzio, G. M. D., and Ferro, N. DIRECT: A System for Evaluating Information Access Components of Digital Libraries. In *ECDL* (2005), vol. 3652 of *Lecture Notes in Computer Science*, Springer, pp. 483–484.

[311] Nüst, D., Sochat, V., Marwick, B., Eglen, S. J., Head, T., Hirst, T., and Evans, B. D. Ten Simple Rules for Writing Dockerfiles for Reproducible Data Science. *Plos Computational Biology 16*, 11 (2020).

[312] Oosterhuis, H., and de Rijke, M. Optimizing Ranking Models in an Online Setting. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 382–396.

[313] Open Science Collaboration. Estimating the Reproducibility of Psychological Science. *Science (New York, N.Y.) 349*, 6251 (2015).

[314] Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., and Johnson, D. Terrier Information Retrieval Platform. In *ECIR* (2005), vol. 3408 of *Lecture Notes in Computer Science*, Springer, pp. 517–519.

[315] Ozertem, U., Jones, R., and Dumoulin, B. Evaluating New Search Engine Configurations with Pre-Existing Judgments and Clicks. In *WWW* (2011), ACM, pp. 397–406.

[316] Pääkkönen, T., Järvelin, K., Kekäläinen, J., Keskustalo, H., Baskaya, F., Maxwell, D., and Azzopardi, L. Exploring Behavioral Dimensions in Session Effectiveness. In *CLEF* (2015), vol. 9283 of *Lecture Notes in Computer Science*, Springer, pp. 178–189.

[317] Pääkkönen, T., Kekäläinen, J., Keskustalo, H., Azzopardi, L., Maxwell, D., and Järvelin, K. Validating Simulated Interaction for Retrieval Evaluation. *Inf. Retr. J. 20*, 4 (2017), 338–362.

[318] Palma, R., Holubowicz, P., Corcho, Ó., Gómez-Pérez, J. M., and Mazurek, C. ROHub - A Digital Library of Research Objects Supporting Scientists towards Reproducible Science. In *SemWebEval@ESWC* (2014), vol. 475 of *Communications in Computer and Information Science*, Springer, pp. 77–82.

[319] Papariello, L., Bampoulidis, A., and Lupu, M. On the Replicability of Combining Word Embeddings and Retrieval Models. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 50–57.

[320] Pasi, G., Piwowarski, B., Azzopardi, L., and Hanbury, A., Eds. *Advances in Information Retrieval - 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings*, vol. 10772 of *Lecture Notes in Computer Science*. Springer, 2018. https://doi.org/10.1007/978-3-319-76941-7.

[321] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[322] Peters, C., Ed. *Cross-Language Information Retrieval and Evaluation, Workshop of Cross-Language Evaluation Forum, CLEF 2000, Lisbon, Portugal, September 21-22, 2000, Revised Papers*, vol. 2069 of *Lecture Notes in Computer Science*. Springer, 2001.

[323] Pimentel, J. F., Murta, L., Braganholo, V., and Freire, J. A Large-Scale Study about Quality and Reproducibility of Jupyter Notebooks. In *MSR* (2019), IEEE / ACM, pp. 507–517.

[324] Pineau, J., Vincent-Lamarre, P., Sinha, K., Larivière, V., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Larochelle, H. Improving Reproducibility in Machine Learning Research(A Report from the NeurIPS 2019 Reproducibility Program). *Journal of Machine Learning Research 22* (2021), 164:1–164:20.

[325] Piwowarski, B. Experimaestro and Datamaestro: Experiment and Dataset Managers (for IR). In *SIGIR* (2020), ACM, pp. 2173–2176.

[326] Piwowarski, B., and Zaragoza, H. Predictive User Click Models Based on Click-through History. In *CIKM* (2007), ACM, pp. 175–182.

[327] Plesser, H. E. Reproducibility vs. Replicability: A Brief History of a Confused Terminology. *Frontiers Neuroinformatics 11* (2017), 76.

[328] Potthast, M., Braun, S., Buz, T., Duffhauss, F., Friedrich, F., Gülzow, J. M., Köhler, J., Lötzsch, W., Müller, F., Müller, M. E., Passmann, R., Reinke, B., Rettenmeier, L., Rometsch, T., Sommer, T., Träger, M., Wilhelm, S., Stein, B., Stamatatos, E., and Hagen, M. Who Wrote the Web? Revisiting Influential Author Identification Research Applicable to Information Retrieval. In *ECIR* (2016), vol. 9626 of *Lecture Notes in Computer Science*, Springer, pp. 393–407.

[329] Potthast, M., Gollub, T., Wiegmann, M., and Stein, B. TIRA Integrated Research Architecture. In *Information Retrieval Evaluation in a Changing World*, vol. 41 of *The Information Retrieval Series*. Springer, 2019, pp. 123–160.

[330] Potthast, M., Günther, S., Bevendorff, J., Bittner, J. P., Bondarenko, A., Fröbe, M., Kahmann, C., Niekler, A., Völske, M., Stein, B., and Hagen, M. The Information Retrieval Anthology. In *SIGIR* (2021), ACM, pp. 2550–2555.

[331] Pradeep, R., Chen, H., Gu, L., Tamber, M. S., and Lin, J. PyGaggle: A Gaggle of Resources for Open-Domain Question Answering. In *ECIR (3)* (2023), vol. 13982 of *Lecture Notes in Computer Science*, Springer, pp. 148–162.

[332] Pradeep, R., Liu, Y., Zhang, X., Li, Y., Yates, A., and Lin, J. Squeezing Water from a Stone: A Bag of Tricks for Further Improving Cross-Encoder Effectiveness for Reranking. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 655–670.

[333] Prinz, F., Schlange, T., and Asadullah, K. Believe It or Not: How Much Can We Rely on Published Data on Potential Drug Targets? *Nature reviews Drug discovery 10*, 9 (2011), 712–712.

[334] Qian, X., Lin, J., and Roegiest, A. Interleaved Evaluation for Retrospective Summarization and Prospective Notification on Document Streams. In *SIGIR* (2016), ACM, pp. 175–184.

[335] Qiu, Y., and Frei, H.-P. Concept Based Query Expansion. In *SIGIR* (1993), ACM, pp. 160–169.

[336] Radlinski, F., Kurup, M., and Joachims, T. How Does Clickthrough Data Reflect Retrieval Quality? In *CIKM* (2008), ACM, pp. 43–52.

[337] Raff, E. A Step toward Quantifying Independently Reproducible Machine Learning Research. In *NeurIPS* (2019), pp. 5486–5496.

[338] Ram, K. Git Can Facilitate Greater Reproducibility and Increased Transparency in Science. *Source Code Biol. Medicine 8* (2013), 7.

[339] Rao, J., Lin, J., and Efron, M. Reproducible Experiments on Lexical and Temporal Feedback for Tweet Search. In *ECIR* (2015), vol. 9022 of *Lecture Notes in Computer Science*, pp. 755–767.

[340] Rauber, A., Asmi, A., Uytvanck, D. V., and Pröll, S. Identification of Reproducible Subsets for Data Citation, Sharing and Re-Use. *Bull. IEEE Tech. Comm. Digit. Libr. 12*, 1 (2016).

[341] Rauber, A., Miksa, T., Mayer, R., and Pröll, S. Repeatability and Re-Usability in Scientific Processes: Process Context, Data Identification and Verification. In *DAMDID/RCDL* (2015), vol. 1536 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 246–256.

[342] Rekabsaz, N., Lesota, O., Schedl, M., Brassey, J., and Eickhoff, C. TripClick: The Log Files of a Large Health Web Search Engine. In *SIGIR* (2021), ACM, pp. 2507–2513.

[343] Rendle, S., Zhang, L., and Koren, Y. On the Difficulty of Evaluating Baselines: A Study on Recommender Systems. *CoRR abs/1905.01395* (2019).

[344] Ribotta, B., Bellemin, R., Grahe, J. E., IJzerman, H., Wagge, J. R., and Mailliez, M. Bringing Replication into Classroom: Benefits for Education, Science, and Society. *PsyArXiv* (Feb. 2022). https://doi.org/10.31234/osf.io/8dhbg.

[345] Robertson, S. E., and Zaragoza, H. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr. 3*, 4 (2009), 333–389.

[346] Roegiest, A., Cormack, G. V., Clarke, C. L. A., and Grossman, M. R. TREC 2015 Total Recall Track Overview. In *TREC* (2015), vol. 500–319 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[347] Roger D. Peng. Reproducible Research in Computational Science. *Science (New York, N.Y.) 334*, 6060 (2011), 1226–1227. https://www.science.org/doi/abs/10.1126/science.1213847.

[348] Rougier, N. P., Hinsen, K., Alexandre, F., Arildsen, T., Barba, L. A., Benureau, F. C. Y., Brown, C. T., de Buyl, P., Caglayan, O., Davison, A. P., Delsuc, M.-A., Detorakis, G., Diem, A. K., Drix, D., Enel, P., Girard, B., Guest, O., Hall, M. G., Henriques, R. N., Hinaut, X., Jaron, K. S., Khamassi, M., Klein, A., Manninen, T., Marchesi, P., McGlinn, D., Metzner, C., Petchey, O. L., Plesser, H. E., Poisot, T., Ram, K., Ram, Y., Roesch, E. B., Rossant, C., Rostami, V., Shifman, A., Stachelek, J., Stimberg, M., Stollmeier, F., Vaggi, F., Viejo, G., Vitay, J., Vostinar, A. E., Yurchak, R., and Zito, T. Sustainable Computational Science: The ReScience Initiative. *PeerJ Computer Science 3* (2017), e142.

[349] Roure, D. D. The Future of Scholarly Communications. *Insights: the UKSG journal 27*, 3 (2014), 233–238. https://doi.org/10.1629/2048-7754.171.

[350] Roy, D., Mitra, M., and Ganguly, D. To Clean or Not to Clean: Document Preprocessing and Reproducibility. *ACM J. Data Inf. Qual. 10*, 4 (2018), 18:1–18:25.

[351] Roy, N., Maxwell, D., and Hauff, C. Users and Contemporary SERPs: A (Re-)Investigation. In *SIGIR* (2022), ACM, pp. 2765–2775.

[352] Ruthven, I., and Lalmas, M. A Survey on the Use of Relevance Feedback for Information Access Systems. *Knowledge Engineering Review 18*, 2 (2003), 95–145.

[353] Sakai, T., Ferro, N., Soboroff, I., Zeng, Z., Xiao, P., and Maistro, M. Overview of the NTCIR-14 CENTRE Task. In *Proceedings of the 14th NTCIR Conference on Evaluation of Information Access Technologies. Tokyo, Japan* (2019).

[354] Salton, G., Wong, A., and Yang, C.-S. A Vector Space Model for Automatic Indexing. *Commun. ACM 18*, 11 (1975), 613–620.

[355] Sanderson, M. Test Collection Based Evaluation of Information Retrieval Systems. *Found. Trends Inf. Retr. 4*, 4 (2010), 247–375.

[356] Sanderson, M., Paramita, M. L., Clough, P. D., and Kanoulas, E. Do User Preferences and Evaluation Measures Line Up? In *SIGIR* (2010), ACM, pp. 555–562.

[357] Sandve, G. K., Nekrutenko, A., Taylor, J., and Hovig, E. Ten Simple Rules for Reproducible Computational Research. *Plos Computational Biology 9*, 10 (2013).

[358] Santos, R. L. T., Marinho, L. B., Daly, E. M., Chen, L., Falk, K., Koenigstein, N., and de Moura, E. S., Eds. *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*. ACM, 2020.

[359] Scells, H., Jimmy, and Zuccon, G. *Big Brother:* A Drop-in Website Interaction Logging Service. In *SIGIR* (2021), ACM, pp. 2590–2594.

[360] Scells, H., Zhuang, S., and Zuccon, G. Reduce, Reuse, Recycle: Green Information Retrieval Research. In *SIGIR* (2022), ACM, pp. 2825–2837.

[361] Schaer, P., Breuer, T., Castro, L. J., Wolff, B., Schaible, J., and Tavakolpoursaleh, N. Overview of LiLAS 2021 - Living Labs for Academic Search. In *CLEF* (2021), vol. 12880 of *Lecture Notes in Computer Science*, Springer, pp. 394–418.

[362] Schaer, P., Breuer, T., Castro, L. J., Wolff, B., Schaible, J., and Tavakolpoursaleh, N. Overview of LiLAS 2021 - Living Labs for Academic Search (Extended Overview). In *CLEF (Working Notes)* (2021), vol. 2936 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 1668–1699.

[363] Schaible, J., Breuer, T., Tavakolpoursaleh, N., Müller, B., Wolff, B., and Schaer, P. Evaluation Infrastructures for Academic Shared Tasks. *Datenbank-Spektrum 20*, 1 (2020), 29–36.

[364] Schliski, F., Schlötterer, J., and Granitzer, M. Influence of Random Walk Parametrization on Graph Embeddings. In *ECIR (2)* (2020), vol. 12036 of *Lecture Notes in Computer Science*, Springer, pp. 58–65.

[365] Scholer, F., Shokouhi, M., Billerbeck, B., and Turpin, A. Using Clicks as Implicit Judgments: Expectations versus Observations. In *ECIR* (2008), vol. 4956 of *Lecture Notes in Computer Science*, Springer, pp. 28–39.

[366] Schuth, A., Balog, K., and Kelly, L. Overview of the Living Labs for Information Retrieval Evaluation (LL4IR) CLEF Lab 2015. In *CLEF* (2015), vol. 9283 of *Lecture Notes in Computer Science*, Springer, pp. 484–496.

[367] Schuth, A., Oosterhuis, H., Whiteson, S., and de Rijke, M. Multi-leave Gradient Descent for Fast Online Learning to Rank. In *WSDM* (2016), ACM, pp. 457–466.

[368] Sculley, D., Snoek, J., Wiltschko, A. B., and Rahimi, A. Winner's Curse? On Pace, Progress, and Empirical Rigor. In *ICLR (Workshop)* (2018), OpenReview.net.

[369] Shrestha, A., and Spezzano, F. Textual Characteristics of News Title and Body to Detect Fake News: A Reproducibility Study. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 120–133.

[370] Silvello, G., Bucco, R., Busato, G., Fornari, G., Langeli, A., Purpura, A., Rocco, G., Tezza, A., and Agosti, M. Statistical Stemmers: A Reproducibility Study. In *ECIR* (2018), vol. 10772 of *Lecture Notes in Computer Science*, Springer, pp. 385–397.

[371] Sloan, M., Yang, H., and Wang, J. A Term-Based Methodology for Query Reformulation Understanding. *Inf. Retr. J. 18*, 2 (2015), 145–165.

[372] Smith, C. L., and Kantor, P. B. User Adaptation: Good Results from Poor Systems. In *SIGIR* (2008), ACM, pp. 147–154.

[373] Smucker, M. D., and Clarke, C. L. A. Time-Based Calibration of Effectiveness Measures. In *SIGIR* (2012), ACM, pp. 95–104.

[374] Smucker, M. D., Guo, X. S., and Toulis, A. Mouse Movement during Relevance Judging: Implications for Determining User Attention. In *SIGIR* (2014), ACM, pp. 979–982.

[375] Soboroff, I., Ferro, N., and Sakai, T. Overview of the TREC 2018 CENTRE Track. In *The Twenty-Seventh Text REtrieval Conference Proceedings (TREC 2018)* (2018).

[376] Soboroff, I., Nicholas, C. K., and Cahan, P. Ranking Retrieval Systems without Relevance Judgments. In *SIGIR* (2001), ACM, pp. 66–73.

[377] Sochat, V. V., Prybol, C. J., and Kurtzer, G. M. Enhancing Reproducibility in Scientific Computing: Metrics and Registry for Singularity Containers. *PLOS ONE 12*, 11 (Nov. 2017), 1–24. https://doi.org/10.1371/journal.pone.0188511.

[378] Soufan, A., Ruthven, I., and Azzopardi, L. Untangling the Concept of Task in Information Seeking and Retrieval. In *ICTIR* (2021), ACM, pp. 73–81.

[379] Srivastava, A., Adusumilli, R., Boyce, H., Garijo, D., Ratnakar, V., Mayani, R., Yu, T., Machiraju, R., Gil, Y., and Mallick, P. Semantic Workflows for Benchmark Challenges: Enhancing Comparability, Reusability and Reproducibility. In *PSB* (2019), pp. 208–219.

[380] Stihec, J., Znidarsic, M., and Pollak, S. Simplified Hybrid Approach for Detection of Semantic Orientations in Economic Texts. In *ECIR* (2018), vol. 10772 of *Lecture Notes in Computer Science*, Springer, pp. 692–698.

[381] Stodden, V. The Legal Framework for Reproducible Scientific Research: Licensing and Copyright. *Computing in Science & Engineering 11*, 1 (2009), 35–40.

[382] Stodden, V., Seiler, J., and Ma, Z. An Empirical Analysis of Journal Policy Effectiveness for Computational Reproducibility. *Proc. Natl. Acad. Sci. USA 115*, 11 (2018), 2584–2589.

[383] Strohman, T., Metzler, D., Turtle, H., and Croft, W. B. Indri: A Language Model-Based Search Engine for Complex Queries. In *Proceedings of the International Conference on Intelligent Analysis* (2005), vol. 2, Citeseer, pp. 2–6.

[384] Tague, J., and Nelson, M. J. Simulation of User Judgments in Bibliographic Retrieval Systems. In *SIGIR* (1981), ACM, pp. 66–71.

[385] Tague, J., Nelson, M. J., and Wu, H. Problems in the Simulation of Bibliographic Retrieval Systems. In *SIGIR* (1980), Butterworths, pp. 236–255.

[386] Tan, L., Baruah, G., and Lin, J. On the Reusability of "Living Labs" Test Collections: : A Case Study of Real-Time Summarization. In *SIGIR* (2017), ACM, pp. 793–796.

[387] Tavakolpoursaleh, N., and Schaible, J. PyTerrier-based Research Data Recommendations for Scientific Articles in the Social Sciences. In *CLEF (Working Notes)* (2021), vol. 2936 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 1712–1722.

[388] Taylor, R. S. Question-Negotiation and Information Seeking in Libraries. Tech. rep., Lehigh University - Center for Information Science, 1967.

[389] Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., and Gurevych, I. BEIR: A Heterogenous Benchmark for Zero-Shot Evaluation of Information Retrieval Models. *CoRR abs/2104.08663* (2021).

[390] The Turing Way Community. The Turing Way: A Handbook for Reproducible, Ethical and Collaborative Research. Zenodo, July 2022. https://doi.org/10.5281/zenodo.7625728.

[391] Thomas, P., Moffat, A., Bailey, P., and Scholer, F. Modeling Decision Points in User Search Behavior. In *IIiX* (2014), ACM, pp. 239–242.

[392] Tosch, E., Bakshy, E., Berger, E. D., Jensen, D. D., and Moss, J. E. B. PlanAlyzer: Assessing Threats to the Validity of Online Experiments. *Proc. ACM Program. Lang. 3*, OOPSLA (2019), 182:1–182:30.

[393] Tran, A. H. M., Kruff, A., Thos, J., Krah, C., Reiners, M., Ax, F., Brech, S., Gharib, S., and Pawlas, V. Ad-Hoc Retrieval of Scientific Documents on the LIVIVO Search Portal. In *CLEF (Working Notes)* (2021), vol. 2936 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 1723–1742.

[394] Trotman, A., Clarke, C. L. A., Ounis, I., Culpepper, J. S., Cartright, M.-A., and Geva, S., Eds. *Proceedings of the SIGIR 2012 Workshop on Open Source Information Retrieval, OSIR@SIGIR 2012, Portland, Oregon, USA, 16th August 2012*. University of Otago, Dunedin, New Zealand, 2012.

[395] Trotman, A., Puurula, A., and Burgess, B. Improvements to BM25 and Language Models Examined. In *ADCS* (2014), ACM, p. 58.

[396] Turpin, A., and Hersh, W. R. Why Batch and User Evaluations Do Not Give the Same Results. In *SIGIR* (2001), ACM, pp. 225–231.

[397] Turpin, A., and Hersh, W. R. User Interface Effects in Past Batch versus User Experiments. In *SIGIR* (2002), ACM, pp. 431–432.

[398] Turpin, A., and Hersh, W. R. Do Clarity Scores for Queries Correlate with User Performance? In *ADC* (2004), vol. 27 of *CRPIT*, Australian Computer Society, pp. 85–91.

[399] Turpin, A., and Scholer, F. User Performance versus Precision Measures for Simple Search Tasks. In *SIGIR* (2006), ACM, pp. 11–18.

[400] Turpin, A., Scholer, F., Järvelin, K., Wu, M., and Culpepper, J. S. Including Summaries in System Evaluation. In *SIGIR* (2009), ACM, pp. 508–515.

[401] Ulmer, D., Hardmeier, C., and Frellsen, J. Deep-Significance - Easy and Meaningful Statistical Significance Testing in the Age of Neural Networks. *CoRR abs/2204.06815* (2022).

[402] Urbano, J., Lima, H., and Hanjalic, A. Statistical Significance Testing in Information Retrieval: An Empirical Analysis of Type I, Type II and Type III Errors. In *SIGIR* (2019), ACM, pp. 505–514.

[403] van Dalen, H. P. How the Publish-or-Perish Principle Divides a Science: The Case of Economists. *Scientometrics 126*, 2 (2021), 1675–1694.

[404] van der Walt, S., Colbert, S. C., and Varoquaux, G. The NumPy Array: A Structure for Efficient Numerical Computation. *Comput. Sci. Eng. 13*, 2 (2011), 22–30. https://doi.org/10.1109/MCSE.2011.37.

[405] Verberne, S., Sappelli, M., Järvelin, K., and Kraaij, W. User Simulations for Interactive Search: Evaluating Personalized Query Suggestion. In *ECIR* (2015), vol. 9022 of *Lecture Notes in Computer Science*, pp. 678–690.

[406] Verberne, S., Sappelli, M., and Kraaij, W. Query Term Suggestion in Academic Search. In *ECIR* (2014), vol. 8416 of *Lecture Notes in Computer Science*, Springer, pp. 560–566.

[407] Verma, M., Yilmaz, E., and Craswell, N. Study of Relevance and Effort across Devices. In *CHIIR* (2018), ACM, pp. 309–312.

[408] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, I., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy. SciPy 1.0-Fundamental Algorithms for Scientific Computing in Python. *CoRR abs/1907.10121* (2019). http://arxiv.org/abs/1907.10121.

[409] Voorhees, E. M. Variations in Relevance Judgments and the Measurement of Retrieval Effectiveness. In *SIGIR* (1998), ACM, pp. 315–323.

[410] Voorhees, E. M. Overview of the TREC-9 Question Answering Track. In *TREC* (2000), vol. 500–249 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[411] Voorhees, E. M., Alam, T., Bedrick, S., Demner-Fushman, D., Hersh, W. R., Lo, K., Roberts, K., Soboroff, I., and Wang, L. L. TREC-COVID: Constructing a Pandemic Information Retrieval Test Collection. *SIGIR Forum 54*, 1 (2020), 1:1–1:12.

[412] Voorhees, E. M., Craswell, N., and Lin, J. Too Many Relevants: Whither Cranfield Test Collections? In *SIGIR* (2022), ACM, pp. 2970–2980.

[413] Voorhees, E. M., Harman, D. K., et al. *TREC: Experiment and Evaluation in Information Retrieval*, vol. 63. Citeseer, 2005.

[414] Voorhees, E. M., Rajput, S., and Soboroff, I. Promoting Repeatability through Open Runs. In *EVIA@NTCIR* (2016), National Institute of Informatics (NII).

[415] Voorhees, E. M., Samarov, D., and Soboroff, I. Using Replicates in Information Retrieval Evaluation. *ACM Trans. Inf. Syst. 36*, 2 (2017), 12:1–12:21.

[416] Wang, S., Scells, H., Mourad, A., and Zuccon, G. Seed-Driven Document Ranking for Systematic Reviews: A Reproducibility Study. In *ECIR (1)* (2022), vol. 13185 of *Lecture Notes in Computer Science*, Springer, pp. 686–700.

[417] Wang, S., Zhuang, S., and Zuccon, G. Federated Online Learning to Rank with Evolution Strategies: A Reproducibility Study. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 134–149.

[418] Wang, X., MacAvaney, S., Macdonald, C., and Ounis, I. An Inspection of the Reproducibility and Replicability of TCT-ColBERT. In *SIGIR* (2022), ACM, pp. 2790–2800.

[419] Webber, W., Moffat, A., and Zobel, J. A Similarity Measure for Indefinite Rankings. *ACM Trans. Inf. Syst. 28*, 4 (2010), 20:1–20:38.

[420] White, R. Beliefs and Biases in Web Search. In *SIGIR* (2013), ACM, pp. 3–12.

[421] Wieling, M., Rawee, J., and van Noord, G. Reproducibility in Computational Linguistics: Are We Willing to Share? *Comput. Linguistics 44*, 4 (2018).

[422] Wilson, T. D. Information Behaviour: An Interdisciplinary Perspective. *Inf. Process. Manag. 33*, 4 (1997), 551–572.

[423] Yadan, O. Hydra - A Framework for Elegantly Configuring Complex Applications. Github, https://github.com/facebookresearch/hydra, 2019.

[424] Yan, L., and McKeown, N. Learning Networking by Reproducing Research Results. *Computer Communication Review 47*, 2 (2017), 19–26.

[425] Yang, H., Guan, D., and Zhang, S. The Query Change Model: Modeling Session Search as a Markov Decision Process. *ACM Trans. Inf. Syst. 33*, 4 (2015), 20:1–20:33.

[426] Yang, P., and Fang, H. A Reproducibility Study of Information Retrieval Models. In *ICTIR* (2016), ACM, pp. 77–86.

[427] Yang, P., Fang, H., and Lin, J. Anserini: Enabling the Use of Lucene for Information Retrieval Research. In *SIGIR* (2017), ACM, pp. 1253–1256.

[428] Yang, P., Fang, H., and Lin, J. Anserini: Reproducible Ranking Baselines Using Lucene. *ACM J. Data Inf. Qual. 10*, 4 (2018), 16:1–16:20.

[429] Yang, P., and Lin, J. Reproducing and Generalizing Semantic Term Matching in Axiomatic Information Retrieval. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 369–381.

[430] Yang, W., Lu, K., Yang, P., and Lin, J. Critically Examining the "Neural Hype": Weak Baselines and the Additivity of Effectiveness Gains from Neural Ranking Models. In *SIGIR* (2019), ACM, pp. 1129–1132.

[431] Yang, X., Ounis, I., McCreadie, R., Macdonald, C., and Fang, A. On the Reproducibility and Generalisation of the Linear Transformation of Word Embeddings. In *ECIR* (2018), vol. 10772 of *Lecture Notes in Computer Science*, Springer, pp. 263–275.

[432] Yates, A., and Unterkalmsteiner, M. Replicating Relevance-Ranked Synonym Discovery in a New Language and Domain. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 429–442.

[433] Yee, W. G., Beigbeder, M., and Buntine, W. L. SIGIR06 Workshop Report: Open Source Information Retrieval Systems (OSIR06). *SIGIR Forum 40*, 2 (2006), 61–65.

[434] Yildiz, B., Hung, H., Krijthe, J. H., Liem, C. C. S., Loog, M., Migut, G., Oliehoek, F. A., Panichella, A., Pawelczak, P., Picek, S., de Weerdt, M., and van Gemert, J. ReproducedPapers.Org: Openly Teaching and Structuring Machine Learning Reproducibility. In *RRPR* (2021), vol. 12636 of *Lecture Notes in Computer Science*, Springer, pp. 3–11.

[435] Yilmaz, E., Craswell, N., Mitra, B., and Campos, D. On the Reliability of Test Collections for Evaluating Systems of Different Types. In *SIGIR* (2020), ACM, pp. 2101–2104.

[436] Yu, R., Xie, Y., and Lin, J. H2oloo at TREC 2018: Cross-collection Relevance Transfer for the Common Core Track. In *TREC* (2018), vol. 500–331 of *NIST Special Publication*, National Institute of Standards and Technology (NIST).

[437] Yu, R., Xie, Y., and Lin, J. Simple Techniques for Cross-Collection Relevance Feedback. In *ECIR (1)* (2019), vol. 11437 of *Lecture Notes in Computer Science*, Springer, pp. 397–409.

[438] Zerhoudi, S., Günther, S., Plassmeier, K., Borst, T., Seifert, C., Hagen, M., and Granitzer, M. The SimIIR 2.0 Framework: User Types, Markov Model-Based Interaction Simulation, and Advanced Query Generation. In *CIKM* (2022), ACM, pp. 4661–4666.

[439] Zhai, C., and Lafferty, J. D. A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval. In *SIGIR* (2001), ACM, pp. 334–342.

[440] Zhang, J., Liu, Y., Mao, J., Xie, X., Zhang, M., Ma, S., and Tian, Q. Global or Local: Constructing Personalized Click Models for Web Search. In *WWW* (2022), ACM, pp. 213–223.

[441] Zhang, S., and Balog, K. Evaluating Conversational Recommender Systems via User Simulation. In *KDD* (2020), ACM, pp. 1512–1520.

[442] Zhang, X., Yates, A., and Lin, J. Comparing Score Aggregation Approaches for Document Retrieval with Pretrained Transformers. In *ECIR (2)* (2021), vol. 12657 of *Lecture Notes in Computer Science*, Springer, pp. 150–163.

[443] Zhang, Y., Liu, X., and Zhai, C. Information Retrieval Evaluation as Search Simulation: A General Formal Framework for IR Evaluation. In *ICTIR* (2017), ACM, pp. 193–200.

[444] Zobel, J. How Reliable Are the Results of Large-Scale Information Retrieval Experiments? In *SIGIR* (1998), ACM, pp. 307–314.

[445] Zobel, J. When Measurement Misleads: The Limits of Batch Assessment of Retrieval Systems. *ACM SIGIR Forum 56*, 1 (2022), 20.

[446] Zobel, J., and Moffat, A. Inverted Files for Text Search Engines. *ACM Comput. Surv. 38*, 2 (2006), 6.

URLs

[447] ACM Artifact Review and Badging Version 1.0. https://www.acm.org/publications/policies/artifact-review-badging. Wayback Machine: https://web.archive.org/web/20220618180306/https://www.acm.org/publications/policies/artifact-review-badging.

[448] ACM Artifact Review and Badging Version 1.1. https://www.acm.org/publications/policies/artifact-review-and-badging-current. Wayback Machine: https://web.archive.org/web/20221025085217/https://www.acm.org/publications/policies/artifact-review-and-badging-current.

[449] ACM Policy on Plagiarism, Misrepresentation, and Falsification. https://www.acm.org/publications/policies/plagiarism-overview. Wayback Machine: https://web.archive.org/web/20221009114820/https://www.acm.org/publications/policies/plagiarism-overview.

[450] ACM SIGIR Artifact Badging. https://sigir.org/general-information/acm-sigir-artifact-badging/. Wayback Machine: https://web.archive.org/web/20220814103836/https://sigir.org/general-information/acm-sigir-artifact-badging/.

[451] ACM SIGMOD Availability & Reproducibility. https://reproducibility.sigmod.org/. Wayback Machine: https://web.archive.org/web/20220928081308/https://reproducibility.sigmod.org/.

[452] Bitbucket. https://bitbucket.org/. Wayback Machine: https://web.archive.org/web/20221115004954/https://bitbucket.org/.

[453] Bitbucket - CENTRE 2019. https://bitbucket.org/centre_eval/c2019_irc/.

[454] Bitbucket - LL4IR implementation of TDI. https://bitbucket.org/living-labs/ll-api/src/master/ll/core/interleave.py.

[455] CodaLab. https://worksheets.codalab.org/. Wayback Machine: https://web.archive.org/web/20221006202722/https://worksheets.codalab.org/.

[456] DCAT-US Schema v1.1. https://resources.data.gov/resources/dcat-us/. Wayback Machine: https://web.archive.org/web/20221115033721/https://resources.data.gov/resources/dcat-us/.

[457] Docker Compose. https://docs.docker.com/compose/. Wayback Machine: https://web.archive.org/web/20221115213852/https://docs.docker.com/compose/.

[458] English Gigaword. https://catalog.ldc.upenn.edu/LDC2003T05. Wayback Machine: https://web.archive.org/web/20220523114731/https://catalog.ldc.upenn.edu/LDC2003T05.

[459] German Research Foundation - Good Research Practice. https://www.dfg.de/en/research_funding/principles_dfg_funding/good_scientific_practice/. Wayback Machine: https://web.archive.org/web/20220609003746/https://www.dfg.de/en/research_funding/principles_dfg_funding/good_scientific_practice/.

[460] GitHub. https://github.com/. Wayback Machine: https://web.archive.org/web/20221115000531/https://github.com/.

[461] GitHub - Anserini. https://github.com/castorini/Anserini. Wayback Machine: https://web.archive.org/web/20221028051807/https://github.com/castorini/Anserini.

[462] GitHub - Anserini Regression Tests Core17. https://github.com/castorini/anserini/blob/master/docs/regressions-core17.md. Wayback Machine: https://web.archive.org/web/20221019151532/https://github.com/castorini/anserini/blob/master/docs/regressions-core17.md.

[463] GitHub - Anserini Runbook CCRF. https://github.com/castorini/anserini/blob/master/docs/runbook-ecir2019-ccrf.md. Wayback Machine: https://web.archive.org/web/20221019151851/https://github.com/castorini/anserini/blob/master/docs/runbook-ecir2019-ccrf.md.

[464] GitHub - CLEF 2021. https://github.com/irgroup/clef2021-web-prf/. Wayback Machine: https://web.archive.org/web/20220313012116/https://github.com/irgroup/clef2021-web-prf/.

[465] GitHub - ECIR 2022. https://github.com/irgroup/ecir2022-uqv-sim. Wayback Machine: https://web.archive.org/web/20220127074251/https://github.com/irgroup/ecir2022-uqv-sim.

[466] GitHub - PyClick. https://github.com/markovi/PyClick. Wayback Machine: https://web.archive.org/web/20220809131421/https://github.com/markovi/PyClick.

[467] GitHub - SIGIR 2020. https://github.com/irgroup/sigir2020-measure-reproducibility. Wayback Machine: https://web.archive.org/web/20220803115905/https://github.com/irgroup/sigir2020-measure-reproducibility.

[468] GitHub - STELLA Project. https://github.com/stella-project/. Wayback Machine: https://web.archive.org/web/20220313100729/https://github.com/stella-project/.

[469] GitHub - ir-metadata. https://github.com/irgroup/ir_metadata/. Wayback Machine: https://web.archive.org/web/20221117151117/https://github.com/irgroup/ir_metadata/.

[470] GitHub - repro_eval. https://github.com/irgroup/repro_eval. Wayback Machine: https://web.archive.org/web/20220804095153/https://github.com/irgroup/repro_eval.

[471] GitLab. https://about.gitlab.com/. Wayback Machine: https://web.archive.org/web/20221115020251/https://about.gitlab.com/.

[472] Kaggle - Personalized Web Search Challenge. https://www.kaggle.com/competitions/yandex-personalized-web-search-challenge/overview.

[473] ML Reproducibility Challenge 2021. https://paperswithcode.com/rc2021. Wayback Machine: https://web.archive.org/web/20220810235739/https://paperswithcode.com/rc2021.

[474] Nature Reproducibility Survey by Monya Baker. https://www.nature.com/articles/533452a. Wayback Machine: https://web.archive.org/web/20221109170351/https://www.nature.com/articles/533452a; Questionnaire: http://www.nature.com/polopoly_fs/7.36741!/file/Reproduciblility%20Questionnaire.doc; Spreadsheet: http://www.nature.com/polopoly_fs/7.36742!/file/Reproducibility%20Survey%20Raw%20Data.xlsx; Figshare: https://figshare.com/articles/Nature_Reproducibility_survey/3394951/1.

[475] NVIDIA cuDNN Documentation. https://docs.nvidia.com/deeplearning/cudnn/developer-guide/index.html. Wayback Machine: https://web.archive.org/web/20220612180409/https://docs.nvidia.com/deeplearning/cudnn/developer-guide/index.html.

[476] OpenReview.net. https://openreview.net/. Wayback Machine: https://web.archive.org/web/20221104212345/https://openreview.net/.

[477] ORCID. https://orcid.org/. Wayback Machine: https://web.archive.org/web/20221115040656/https://orcid.org/.

[478] Papers with Code. https://paperswithcode.com. Wayback Machine: https://web.archive.org/web/20221028155918/https://paperswithcode.com.

[479] PVLDB Reproducibility. https://vldb.org/pvldb/reproducibility/. Wayback Machine: https://web.archive.org/web/20221009200150/https://vldb.org/pvldb/reproducibility/.

[480] ReproducedPapers.org. https://reproducedpapers.org/. Wayback Machine: https://web.archive.org/web/20220710190949/https://reproducedpapers.org.

[481] ReScience C. http://rescience.github.io/. Wayback Machine: https://web.archive.org/web/20221006043502/http://rescience.github.io/.

[482] RFC 2119 - Key words for use in RFCs to Indicate Requirement Levels. https://www.rfc-editor.org/rfc/rfc2119. Wayback Machine: https://web.archive.org/web/20221014234442/https://www.rfc-editor.org/rfc/rfc2119.

[483] SIGIR and plagiarism: an open letter by Ben Carterette. https://medium.com/@carteret.acm/sigir-and-plagiarism-e23bc2b79948. Wayback Machine: https://web.archive.org/web/20220302014442/https://medium.com/@carteret.acm/sigir-and-plagiarism-e23bc2b79948.

[484] ir-metadata.org. https://www.ir-metadata.org/. Alternative URL: https://irgroup.github.io/ir_metadata/; Wayback Machine: https://web.archive.org/web/20221117150933/https://www.ir-metadata.org/.

[485] trec_eval - Issue 20. https://github.com/usnistgov/trec_eval/issues/20. Wayback Machine: https://web.archive.org/web/20220721040650/https://github.com/usnistgov/trec_eval/issues/20.

[486] The New York Times Annotated Corpus. https://catalog.ldc.upenn.edu/LDC2008T19. Wayback Machine: https://web.archive.org/web/20221031205205/https://catalog.ldc.upenn.edu/LDC2008T19.

[487] UQV Dataset by Benham and Culpepper. https://culpepper.io/publications/robust-uqv.txt.gz. Wayback Machine: https://web.archive.org/web/20220419155035/https://culpepper.io/publications/robust-uqv.txt.gz.

[488] Yahoo - Webscope Datasets. https://webscope.sandbox.yahoo.com/.

[489] Zenodo - ir-metadata. https://zenodo.org/record/5997491.

Appendix A

ECIR Reproducibility Track

Table A.1: Overview of the reactive reproducibility studies at ECIR from 2015 to 2022. The columns are defined as follows. **Topic**: summary of the reproducibility topic and target problem. **PRIMAD & Outcome**: PRIMAD components that changed w.r.t. the original experiment; the colored component is the main focus of the study; the outcome is the authors' opinion about the success of reproducibility and is categorized into *success* (●), *partial success* (◐), and *failure* (○). **Evaluation**: method for evaluating the success of reproducibility.

Year	Ref.	Topic	PRIMAD & Outcome	Evaluation
2015	[142]	Generalization of RBP	P'RI'MA'D' ●	Revalidation on more datasets; comparing Kendall's τ of system rankings and the number of significant differences between systems
	[169]	Reproduction of classfication methods for sentiment detection in tweets	P'RI'M'A'D ●	Comparison of F1 scores; error analysis of true/false positives/negatives
	[339]	Reproduction of lexical and temporal feedback techniques for tweet search	P'RI'M'A'D' ◐	Comparison of ARP including Fisher's two-sided, paired randomization test and one-sided paired t-test; revalidation on more datasets; different training/test data splits
2016	[173]	Reproduction of entity linkings based on the TAGME system	PR'IMA'D' ◐	Comparison of ARP
	[249]	Open-Source IR Reproducibility Challenge	PRI'M'A'D ◐	Comparison of ARP (effectiveness) and query latency (efficiency)
	[271]	Reproduction of multi-document summarization methods in the newswire domain	P'RI'M'A'D' ●	Comparison of system performance including paired t-test and additional comparison to crowdsourced user judgements
	[328]	Systematic reproducibility study of author identification methods by students	P'RI'MA'D' ●	Assessments of the provided resources regarding their usefulness for reimplementations, including the clarity of the approach, availability of data, reconstructability, and the overall reproducibility
2018	[117]	Reproduction of a DL-based method for question answering tasks	P'RI'MA'D ○	Comparison of ARP

	[370]	Systematic reproducibility study of statistical and language-independent stemmers	P'RI'MA'D' ●	Comparison of ARP
	[380]	Reproduction of classifying semantic orientations in economic texts	P'RI'M'A'D ●	Comparison of lexical statistics (number of entities and sentiment direction), means of the label distributions, and accuracy
	[431]	Reproduction and generalization of the linear transformation of word embeddings	P'R'I'MA'D' ●	Comparison of ARP and statistical significance testing with McNemar's test
2019	[49]	Systematic reproducibility study of general summarization algorithms applied to legal texts in different languages	P'RI'M'A'D' ◐	Comparison of ROUGE scores and additional qualitative assessments by legal experts
	[53]	Systematic reproducibility study of recommender systems in another context (massive open online courses) and bias analysis	P'R'I'MA'D' ◐	Comparison of ARP including paired t-tests and additional analysis of popularity bias
	[270]	Reproduction and generalization of a document reordering algorithm for index compression	P'RI'MA'D' ●	Comparison of compression ratio and query efficiency; generalization on different datasets
	[276]	Systematic reproducibility study of different index compression and document-at-a-time query processing algorithms	P'RI'M'A'D ●	Comparison of compression ratio and query efficiency by systematic evaluation with a fixed retrieval method (BM25) on the same datasets
	[277]	Reproduction and extension of cross-domain recommendation approaches for venues	P'RI'M'A'D' ◐	Comparison of ARP including paired t-tests
	[312]	Reproduction of two optimization algorithms for online learning to rank	P'RI'MA'D' ◐	Comparison of ARP including paired t-tests; evaluations on three learning to rank datasets; lower- and upper-bound performance estimates by simulating different levels of noise in user click signals

	[429]	Reproduction and generalization of axiomatic retrieval methods (and integration into Anserini)	P'RI'M'A'D' ●	Comparison of ARP; revalidation on more datasets (from different domains) using different first-stage retrieval methods
	[432]	Reproduction of synonym discovery and ranking methods including the application to another domain	P'R'I'MA'D' ◐	Comparison of ARP
	[437]	Reproduction and generalization of CCRF	P'RI'M'A'D' ●	Comparison of ARP including paired t-tests; different combinations of training and test datasets
2020	[46]	Revalidation and ablation study of entity alignment in knowledge graphs based on graph convolutional network	P'RI'M'A'D' ◐	Comparison of Hits@1 scores
	[147]	Revalidation of the influence of near-duplicates removal on the reproducibility of shared task evaluations	P'RI'MA'D' ●	Comparison of ARP and relative system orderings by Kendall's τ
	[157]	Review of how research findings got integrated into Lucene	-	This paper reports anecdotes of how a block-max index feature got integrated into the Lucene software library.
	[213]	Reproducibility analysis of different BM25 implementations	P'RI'MA'D ●	Comparison of ARP (by ANOVA and Tukey's HSD) and query efficiency
	[233]	Revalidation of popularity bias of recommender systems in the music domain	P'R'I'MA'D' ●	There is no direct comparison to the original results, but the paper reconfirms the overall biased trends and findings as reported in the original work.
	[254]	Reproduction of a community benchmark (OSIRRC artifacts from 2015 [249]) and the corresponding artifacts	P'RIMA'D ◐	Comparison of ARP

	[319]	Reproduction of a method based on word embeddings that in combination with the mixture model of von Mises-Fisher is used for classification, clustering, and retrieval	P'RI'M'A'D' ○	Comparison of ARP
	[364]	Reproduction of a graph embedding method (node2vec)	P'RI'MA'D ○	Comparison of structural similarity between graph networks
2021	[5]	Revalidation of a model based on paragraph-level-interactions and BERT in the legal and patent domain	P'R'I'MA'D' ◐	Comparison of ARP and paired t-tests
	[47]	Systematic revalidation of entity alignment methods	P'RI'MA'D' ◐	Comparison of Hits@1 scores
	[123]	Reproduction of an evaluation approach based random partitions of the test collection and bootstrap ANOVA	P'RI'M'A'D' ●	Comparison of bootstrap/traditional ANOVA evaluated by the comparison of confidence intervals and agreements between statistically significant differences
	[128]	Revalidation of reliability predictions for health-related content	P'RI'MA'D' ●	Comparison of system performance (weighted accuracy); revalidation on two new datasets
	[226]	Systematic reproducibility study of five different web page segmentation methods	P'RI'MA'D' ●	Comparison of ARP
	[300]	Reproduction of a *learning to quantify* study	P'RI'MA'D' ◐	Comparison of (relative) absolute errors including two-sided paired t-tests; revalidation on three datasets
	[302]	Systematic reproducibility study of aspect-based sentiment analysis in a production-like evaluation setting	P'R'IMA'D' ○	Comparison of ARP; evaluation of *transferability* to other domains by systematically varing the in-domain training instances

	[303]	Revalidation of a privacy-preserving recommender system based on meta matrix factorization (meta-learning)	P'RIMA'D' ●	Comparison of system effectiveness; reuse of the original open-source implementation on other datasets
	[369]	Revalidation of a fake news detection method	P'RI'MA'D' ●	Comparison of ARP including significance tests; revalidation on fake news datasets from different sources covering political and gossip news
	[417]	Revalidation of federated online learning to rank	P'RIMA'D' ◐	Comparison of ARP; the original implementation is reused and evaluated on more datasets and different types of simulated user click signals
	[442]	Reproduction of passage score aggregation methods based on BERT for document retrieval	P'RI'M'A'D' ●	Comparison of ARP including paired t-tests; additional evaluation on a different dataset
2022	[51]	Systematic reproducibility study of loss functions in the context of image retrieval	P'RI'M'A'D' ◐	Comparison of ARP and further analysis by counting contributing samples
	[54]	Systematic reproducibility study of recommender systems that mitigate consumer unfairness	P'RIMA'D' ◐	Comparison of ARP and fairness measures; evaluations based on two datasets
	[148]	Revalidation of using anchor text (in webpages) as ranking feature	P'RI'M'A'D' ◐	Comparison of ARP; additional comparison of anchor text by the number of distinct terms, most frequent terms, homogenenity of the search results (Jensen-Shannon distances); revalidation on a substantially larger dataset (MS MARCO); inclusion of Transformer-based retrieval methods
	[237]	Systematic reproducibility study of two deep learning-based methods for systematic literature reviews across 23 datasets	P'RIMA'D' ○	Comparison of *work saved over sampling* (WSS@95%Recall) and Precision@95%Recall metrics

	[243]	Reproduction of dense retrieval-based pseudo-relevance feedback	P'RI'M'A'D ◐	Comparison of ARP; variation of hyperparameters and different dense retrieval methods
	[263]	Reproduction of a dense passage retrieval method for question answering	P'RI'M'A'D ●	Comparison of system effectiveness and *exact match* scores including paired t-tests
	[264]	Reproduction of session-based experiments based on the AOL logs	P'RI'MA'D' ●	Comparison of ARP including paired t-tests; the reproduced results are based on a rescraped document (webpage) collection
	[283]	Analysis of generative adversarial networks for collaborative filtering	P'RI'M'A'D ◐	Comparison of ARP
	[308]	Revalidation of popularity and demographic biases in recommender systems	P'RI'M'A'D' ●	Kruskall-Wallis significance tests between different demographic groups reconfirm the original findings about biased recommendations
	[332]	Reproduction and improvement of the general cross-encoder reranking pipeline	P'RI'M'A'D' ●	Comparison of ARP including paired t-tests
	[416]	Revalidating a method for systematic literature reviews with more recent datasets	P'RI'M'A'D' ◐	Comparison of ARP including paired t-tests

Appendix B

`ir_metadata` Schema

Platform

- `platform` → `hardware` → `cpu` → `model`
  Description: Name of the CPU model.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `platform` → `hardware` → `cpu` → `architecture`
  Description: Identifier of the CPU architecture.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `platform` → `hardware` → `cpu` → `operation mode`
  Description: Operation mode of the CPU.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `platform` → `hardware` → `cpu` → `number of cores`
  Description: Number of CPU cores.
  Type: Scalar
  Encoding: A decimal integer number; `!!int`.

- `platform` → `hardware` → `gpu` → `architecture`
  Description: Name of the GPU architecture.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `platform` → `hardware` → `gpu` → `number of cores`
  Description: Number of GPU cores.
  Type: Scalar
  Encoding: A decimal integer number; `!!int`.

- `platform` → `hardware` → `gpu` → `memory`
  Description: Amount of available memory of the GPU; string with numbers followed by GB.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- platform → hardware → ram
  Description: Amount of available RAM; string with numbers followed by GB.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.

- platform → operating system → kernel
  Description: The kernel version of the operating system.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.

- platform → operating system → distribution
  Description: The name of the operating system's distribution.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.

- platform → software → libraries
  Description: Names and versions of the software libraries and packages underlying the experiment's implementation with the following syntax <library-name>==<version>. If possible, libraries and packages of different programming languages should be in separate nodes.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.

- platform → software → retrieval toolkit
  Description: Names and versions of the retrieval toolkits underlying the experiment's implementation with the following syntax <toolkit-name>==<version>.
  Type: Sequence of scalars; !!seq.
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.
  Naming convention: indri, terrier, anserini, pyserini, pyterrier, solr, elasticsearch

Research Goal

- research goal → venue → name
  Description: Acronym (if available) or name of the venue (e.g., journal or conference) at which is the study is published. A non-exhaustive list is given by the naming conventions.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.
  Naming convention: CHIIR, CIKM, ECIR, ICTIR, IPM, IRJ, JASIST, JCDL, KDD, SIGIR, TOIS, WSDM, WWW, CLEF, NTCIR, TREC

- research goal → venue → year
  Description: Year in which the study was published (syntax: YYYY).
  Type: Scalar
  Encoding: A decimal integer number; !!int.

- research goal → publication → dblp
  Description: URL of the publication in the dblp - computer science bibliography.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

- research goal → publication → doi
  Description: DOI of the publication.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

- research goal → publication → arxiv
  Description: URL to the arXiv publication.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

- research goal → publication → url
  Description: Custom URL where is the publication is hosted.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

- research goal → publication → abstract
  Description: Abstract of the publication.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

- research goal → evaluation → reported_measures
  Description: A list of measures that were evaluated. We propose to follow trec_eval's naming convention of the measures (see naming convention).
  Type: Sequence of scalars; !!seq.
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.
  Naming convention: map, P_10, ndcg, bpref

- research goal → evaluation → baseline
  Description: The run tag of the baseline that is used in the experiments. If the actor is the original experimenter, the baseline should be adequate and state-of-the-art. If the actor is a reproducer, the baseline refers to the run that is reproduced.
  Type: Sequence of scalars; !!seq.
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.

- research goal → evaluation → significance test
  Description: Significance tests that were used as part of the experimental evaluations. If required, the corresponding correction method should be reported as well.
  Type: Sequence of mappings; !!seq [!!map, !!map, ...].

- research goal → evaluation → significance test → name
  Description: Name of the significance test.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.

Naming convention: `t-test` (Student's t-test), `wilcoxon` (Wilcoxon signed rank test), `sign` (sign test), `permutation` (permutation test), `bootstrap` (bootstrap test – shift method)

- `research goal` → `evaluation` → `significance test` → `correction method`
  Description: Name of the correction method.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.
  Naming convention: `bonferroni` (Bonferroni correction), `holm-bonferroni` (Holm–Bonferroni method), `HMP` (harmonic mean p-value), `MRT` (Duncan's new multiple range test)

Implementation

- `implementation` → `executable` → `cmd`
  Description: The software command that was used to conduct the experiments, more specifically, to make the run.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `implementation` → `source` → `lang`
  Description: All programming languages that were used for the experiments. A non-exhaustive list is given by the naming conventions. If the source code is in a git repository, the `MetadataHandler` of the `metadata` module in `repro_eval` can be used to extract the programming languages automatically.
  Type: Sequence of scalars; `!!seq`.
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.
  Naming convention: `Java`, `Python` `R`, `C`, `C++`, `Ruby`, `Go`, `Matlab`, `Shell`

- `implementation` → `source` → `repository`
  Description: The URL of the corresponding software repository.
  Type: Scalar
  Encoding: URI according to RFC2396; `!!str`.

- `implementation` → `source` → `commit`
  Description: The commit at which the repository was used for the experiments. Both long and short versions are valid.
  Type: Scalar
  Encoding: String of characters generated by `SHA-1` (RFC3174) or `SHA-256` (RFC6234).

Method

- `method` → `automatic`
  Description: Boolean value indicating if it is a automatic (`true`) or manual (`false`) run.
  Type: Scalar
  Encoding: Boolean; `!!bool`.

- `method` → `score ties`
  Description: Name or description of the method used to break score ties in the ranking.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.
  Naming convention: `(reverse) alphabetical order`, `external collection`

- `method` → `indexing` → `tokenizer`
  Description: Name of the tokenizer. If available, it can be reported by the class in the software package (see example below).
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `method` → `indexing` → `stemmer`
  Description: Name of the stemmer. If possible, the stemmer should be reported by the class name in the software package (see example below). If this is not possible, it should meet the naming conventions below.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.
  Naming convention: `Porter`, `Krovetz`, `Lovins`, `Snowball`, `n-grams`

- `method` → `indexing` → `stopwords`
  Description: Name of the stopword list. If possible, the stopword list should be reported by the resource name in the software package or by an URI (see example below). If this is not possible, it should meet the naming conventions below, e.g., by naming the corresponding `retrieval toolkit`.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.
  Naming convention: `Indri`, `Lucene`, `Smart`, `Terrier`

- `method` → `retrieval`
  Description: The retrieval approach is documented by a sequence of mappings, where each mapping represents one component of a ranking pipeline, i.e., it is also possible to report multi-stage ranking pipelines by referring to previous ranking stages.
  Type: Sequence of mappings; `!!seq [!!map, !!map, ...]`.

- `method` → `retrieval` → `name`
  Description: Name of the ranking stage component.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str.`
  Naming convention: `bm25`, `rm3`, `ax` (axiomatic reranking), `piv` (pivoted normalization method), `dir` (Dirichlet prior method), `monobert`

- `method` → `retrieval` → `method`
  Description: Class name of the retrieval method.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str.`

- `method` → `retrieval` → `params`
  Description: Parameter(s) of the retrieval method. Depending on the parameter, a single mapping is defined by the parameter name and a decimal integer or floating number.
  Type: Scalar
  Encoding: A decimal integer or floating point number; `!!int` or `!!float`.

- `method` → `retrieval` → `reranks`
  Description: Name of the component whose output will be reranked.
  Type: Scalar.
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `method` → `retrieval` → `interpolates`
  Description: Name of the components whose output will be interpolated.
  Type: Sequence of scalars; `!!seq`.
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `method` → `retrieval` → `weight`
  Description: Interpolation weight.
  Type: Scalar
  Encoding: A decimal integer or floating point number; `!!int` or `!!float`.

Actor

- `actor` → `name`
  Description: Name of the actor.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `actor` → `orcid`
  Description: ORCID of the actor.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `actor` → `team`
  Description: The actor's research team name.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `actor` → `fields`
  Description: List of the actor's research fields.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.
  Naming convention: `nlp/natural language processing`, `ir/information retrieval`, `databases`, `data analytics`, `machine/deep learning`, `statistics`, `bibliometrics`, `information systems`

- `actor` → `mail`
  Description: Mail address of the actor.
  Type: Sequence of scalars; `!!seq`.
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- actor → role
  Description: Role of the actor. Can be experimenter if is an original experiment, or reproducer if it is a reproduced experiment.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.

- actor → degree
  Description: The academic degree of the actor. Should be reported by the conventional abbreviations (see examples for the naming convention below).
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.
  Naming convention: experimenter, reproducer

- actor → github
  Description: URL with the GitHub handle of the actor.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.
  Naming convention: B.Sc., M.Sc., Ph.D.

- actor → twitter
  Description: URL with the Twitter handle of the actor.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

Data

- data → test collection
  Description: A test collection includes but is not limited to a name, source, qrels, topics, and an ir_datasets identifier.
  Type: Collection of scalars

- data → test collection → name
  Description: Name of the test collection.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); !!str.

- data → test collection → source
  Description: Official source of the collection.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

- data → test collection → qrels
  Description: Source of the qrels.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

- data → test collection → topics
  Description: Source of the topic file.
  Type: Scalar
  Encoding: URI according to RFC2396; !!str.

- `data` → `test collection` → `ir_datasets`
  Description: Identifier in `ir_datasets`.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `data` → `training data`
  Description: List of different training data sources that are used in the experiments represented as mappings, a single mapping usually has a `name` and a `source`.
  Type: Sequence of mappings; `!!seq [!!map, !!map, ...]`.

- `data` → `training data` → `name`
  Description: Name of the training data resource.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `data` → `training data` → `source`
  Description: Name of the data resource.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `data` → `other`
  Description: List of other data sources that are used in the experiments, for instance, external stopword lists, thesauri, or word embeddings. These resources are represented as mappings, a single mapping usually has a `name` and a `source`.
  Type: Sequence of mappings; `!!seq [!!map, !!map, ...]`.

- `data` → `other` → `name`
  Description: Name of the data resource.
  Type: Scalar
  Encoding: UTF-8 encoded string of characters (RFC3629); `!!str`.

- `data` → `other` → `source`
  Description: Source location of the data resource.
  Type: Scalar
  Encoding: URI according to RFC2396; `!!str`.

```yaml
platform:
  hardware:
    cpu:
      model: 'Intel Xeon Gold 6144 CPU @ 3.50GHz'
      architecture: 'x86_64'
      operation mode: '64-bit'
      number of cores: 16
    ram: '64 GB'
  operating system:
    kernel: '5.4.0-90-generic'
    distribution: 'Ubuntu 20.04.3 LTS'
  software:
    libraries:
      python:
        - 'scikit-learn==0.20.1'
        - 'numpy==1.15.4'
      java:
        - 'lucene==7.6'
    retrieval toolkit:
      - 'anserini==0.3.0'
```

Figure B.1: Platform metadata example

```yaml
research goal:
  venue:
    name: 'SIGIR'
    year: '2020'
  publication:
    dblp: 'https://dblp.org/rec/conf/sigir/author'
    arxiv: 'https://arxiv.org/abs/2010.13447'
    doi: 'https://doi.org/10.1145/3397271.3401036'
    abstract: 'In this work, we analyze ...'
  evaluation:
    reported measures:
      - 'ndcg'
      - 'map'
      - 'P_10'
    baseline:
      - 'tfidf.terrier'
      - 'qld.indri'
    significance test:
      - name: 't-test'
        correction method: 'bonferroni'
```

Figure B.2: Research goal metadata example

```yaml
implementation:
  executable:
    cmd: './bin/search arg01 arg02 input output'
  source:
    lang:
      - 'python'
      - 'c'
    repository: 'github.com/castorini/anserini'
    commit: '9548cd6'
```

Figure B.3: Implementation metadata example

```yaml
method:
    automatic: 'true'
    score ties: 'reverse alphabetical order'
    indexing:
    tokenizer: 'lucene.StandardTokenizer'
    stemmer: 'lucene.PorterStemFilter'
    stopwords: 'lucene.StandardAnalyzer'
    retrieval:
    - name: 'bm25'
        method: 'lucene.BM25Similarity'
        b: 0.4
        k1: 0.9
    - name: 'lr reranker'
        method: 'sklearn.LogisticRegression'
        reranks: 'bm25'
    - name: 'interpolation'
        weight: 0.6
        interpolates:
        - 'lr reranker'
        - 'bm25'
```

Figure B.4: Method metadata example

```yaml
actor:
  name: 'Jimmy Lin'
  orcid: '0000-0002-0661-7189'
  team: 'h2oloo'
  fields:
    - 'nlp'
    - 'ir'
    - 'databases'
    - 'large-scale distributed algorithms'
    - 'data analytics'
  mail: 'jimmylin@uwaterloo.ca'
  role: 'experimenter' # or 'reproducer'
  degree: 'Ph.D.'
  github: 'https://github.com/lintool'
  twitter: 'https://twitter.com/lintool'
```

Figure B.5: Actor metadata example

```yaml
data:
  test_collection:
    name: 'The New York Times Annotated Corpus'
    source: 'catalog.ldc.upenn.edu/LDC2008T19'
    qrels: 'trec.nist.gov/data/core/qrels.txt'
    topics: 'trec.nist.gov/data/core/core_nist.txt'
    ir_datasets: 'nyt/trec-core-2017'
  training_data:
    - name: 'TREC Robust 2004'
      folds:
        - 'disks45/nocr/trec-robust-2004/fold1'
        - 'disks45/nocr/trec-robust-2004/fold2'
  other:
    - name: 'GloVe embeddings'
      source: 'https://nlp.stanford.edu/projects/glove/'
```

Figure B.6: Data metadata example

Appendix C

Replicability of UQV Simulations

Table C.1: Retrieval effectiveness over q queries. Besides averaging the retrieval effectiveness over all available queries, we also evaluated the effectiveness over the first and the most effective queries for each topic (evaluated with Robust04, replicates Table 6.3).

	All queries				First queries				Best queries			
	q	nDCG	P@10	AP	q	nDCG	P@10	AP	q	nDCG	P@10	AP
UQV_1	150	.3704	.3307	.1263	50	.4069	.3500	.1582	50	.4595	.4480	.1858
UQV_2	52	.4254	.3519	.1812	50	.4130	.3420	.1716	50	.4150	.3420	.1734
UQV_3	68	.3748	.3279	.1371	50	.3558	.3000	.1278	50	.3815	.3300	.1430
UQV_4	123	.3848	.3415	.1514	50	.4003	.3640	.1598	50	.4593	.4240	.1981
UQV_5	500	.3719	.3078	.1387	50	.4107	.3580	.1638	50	.5118	.5320	.2331
UQV_6	136	.3915	.3463	.1509	50	.4122	.3660	.1671	50	.4661	.4480	.2024
UQV_7	50	.4732	.4340	.2039	50	.4732	.4340	.2039	50	.4732	.4340	.2039
UQV_8	156	.3610	.3115	.1328	50	.3810	.3060	.1431	50	.4445	.4420	.1868
TTS_{S1}	500	.0473	.0246	.0100	50	.1542	.0920	.0373	50	.3030	.1880	.0798
TTS_{S2}	500	.1846	.1266	.0513	50	.3197	.2640	.1113	50	.4227	.3880	.1655
$TTS_{S2'}$	500	.3107	.2420	.1046	50	.3434	.2520	.1210	50	.4478	.4320	.1765
TTS_{S3}	500	.2916	.2156	.0996	50	.1542	.0920	.0373	50	.4311	.4080	.1739
$TTS_{S3'}$	500	.3056	.2272	.1061	50	.3197	.2640	.1113	50	.4303	.4120	.1741
TTS_{S4}	500	.4139	.3540	.1606	50	.3992	.3440	.1527	50	.5688	.5900	.2795
$TTS_{S4'}$	500	.4216	.3620	.1581	50	.4200	.3500	.1477	50	.5774	.5860	.2776
$TTS_{S4''}$	500	.3417	.2862	.1264	50	.3987	.3360	.1519	50	.5580	.5880	.2706
KIS_{S1}	500	.1154	.0680	.0234	50	.2498	.1480	.0545	50	.3872	.3520	.1339
KIS_{S2}	500	.3311	.2664	.1045	50	.4208	.3440	.1639	50	.5569	.5920	.2778
$KIS_{S2'}$	500	.4305	.3864	.1664	50	.4722	.4580	.1975	50	.5729	.6140	.2800
KIS_{S3}	500	.4856	.4228	.2080	50	.2498	.1480	.0545	50	.6199	.6260	.3153
$KIS_{S3'}$	500	.5146	.4552	.2266	50	.4208	.3440	.1639	50	.6189	.6160	.3137
KIS_{S4}	500	.4482	.4044	.1818	50	.4551	.4380	.1822	50	.6018	.6420	.3107
$KIS_{S4'}$	500	.4177	.3728	.1584	50	.4090	.3500	.1414	50	.5777	.6080	.2939
$KIS_{S4''}$	500	.4101	.3622	.1565	50	.4542	.4360	.1816	50	.5978	.6360	.3039

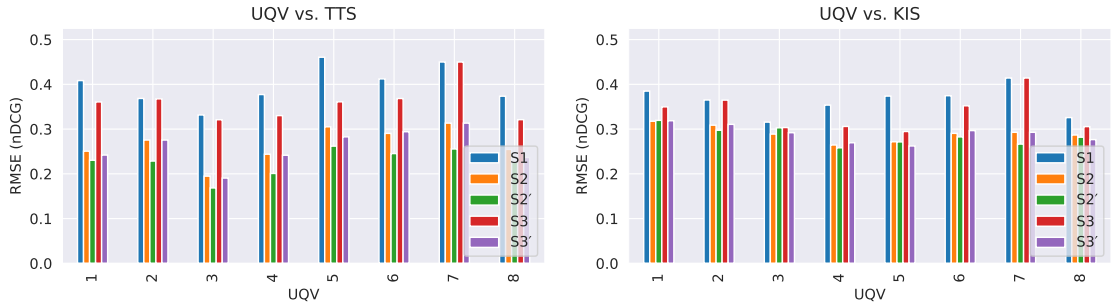

Figure C.1: RMSE between the topic scores resulting from the simulated and real UQV queries. The left plot shows the error of the $\text{TTS}_{\text{S1-S3}'}$ queries, and the right plot shows the error of the $\text{KIS}_{\text{S1-S3}'}$ queries and the UQV queries regarding queries made by eight users (evaluated with Robust04, replicates Figure 6.2).

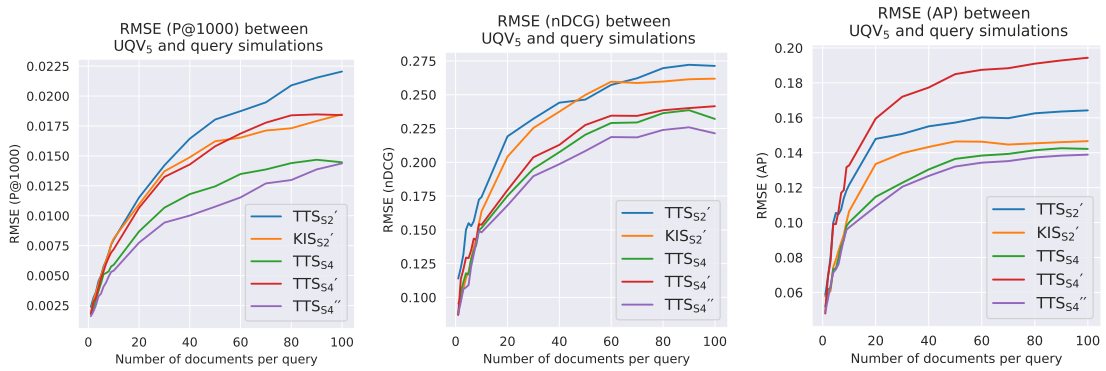

Figure C.2: RMSE instantiated with P@1000, nDCG, and AP over an increasing number of documents per query w.r.t. the fifth user in the dataset (UQV_5) (evaluated with Robust04, replicates Figure 6.3).

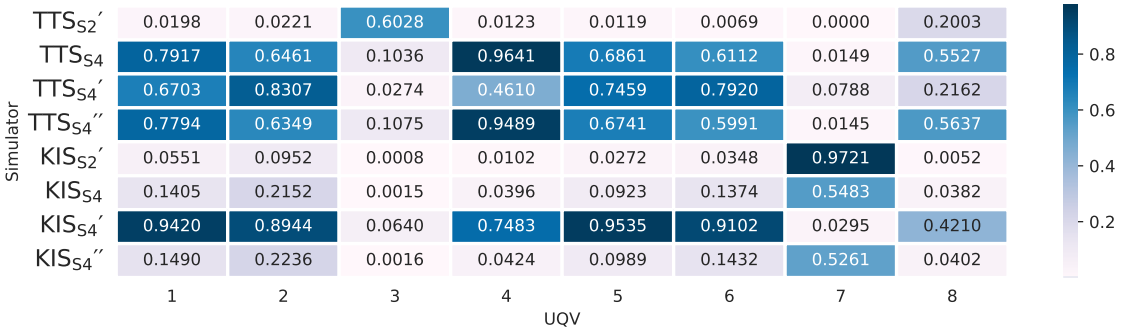

Figure C.3: p-values of paired t-tests between UQV and simulated queries. The corresponding topic scores are based on nDCG for the first query of each topic that was generated by a simulator or formulated by one of the eight users (evaluated with Robust04, replicates Figure 6.4).

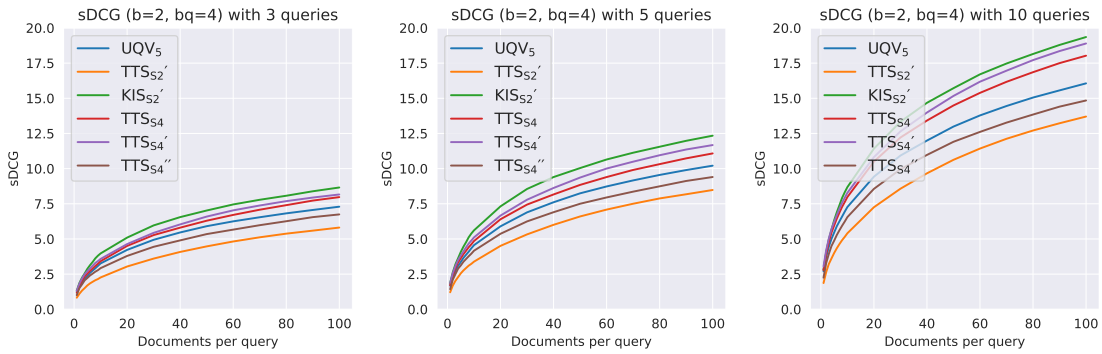

Figure C.4: Simulations with 3, 5, or 10 queries per session evaluated by sDCG (replicates Figure 6.6).

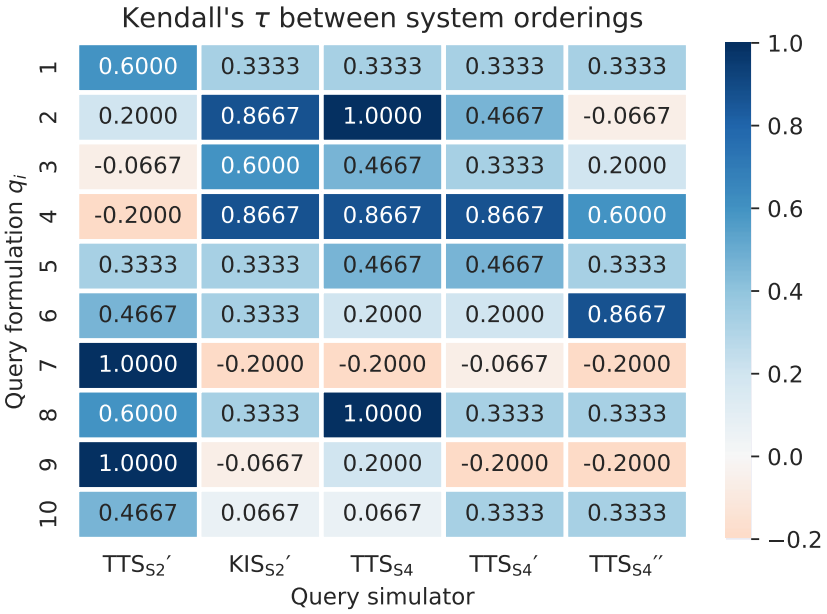

Figure C.5: Kendall's τ between the system rankings resulting from the i-th query of the simulators in reference to the system ranking of the real user (UQV_5) (evaluated with Robust04, replicates Figure 6.5).

Table C.2: MSLE between the isoquants for pre-defined nDCG levels (evaluated with Robust04, replicates Table 6.4).

nDCG	0.3	0.4	0.5
$TTS_{S2'}$	0.3804	0.5282	0.6510
$KIS_{S2'}$	0.0346	0.0661	0.0950
TTS_{S4}	0.0000	0.0417	0.0248
$TTS_{S4'}$	0.0000	0.0194	0.0308
$TTS_{S4''}$	0.1147	0.1778	0.1939

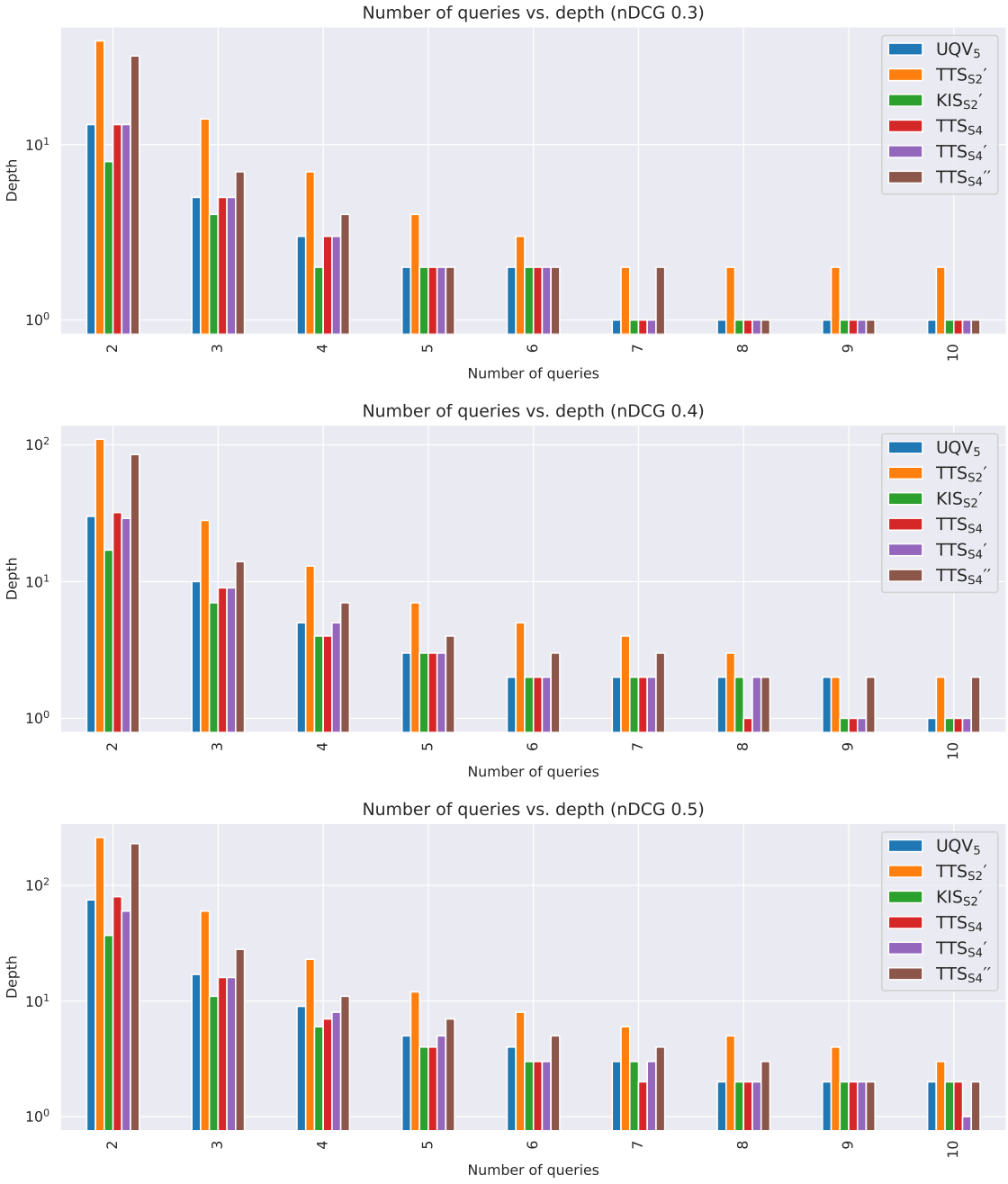

Figure C.6: Number of queries vs. browsing depth: isoquants and query simulations and UQV_5 with fixed nDCG levels of 0.3, 0.4, and 0.5 (evaluated with Robust04, replicates Figure 6.7 as bar plots). To lower the resource use and computation time, we excluded the evaluation of single queries.

Appendix D

Living Lab Evaluations

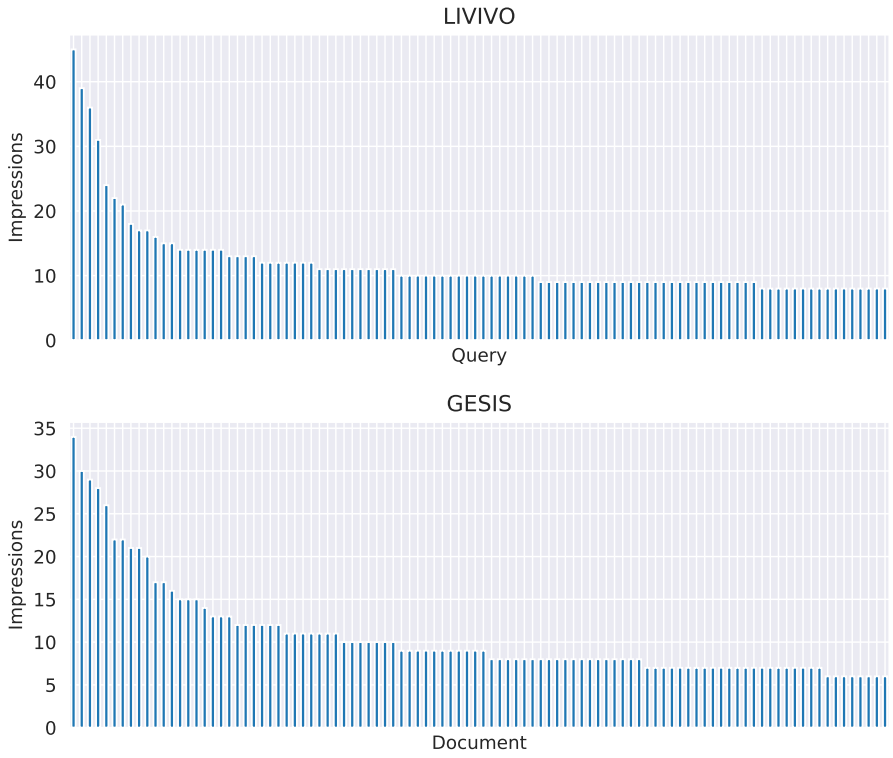

Figure D.1: Impresssions vs. queries (LIVIVO) and target documents (GESIS)

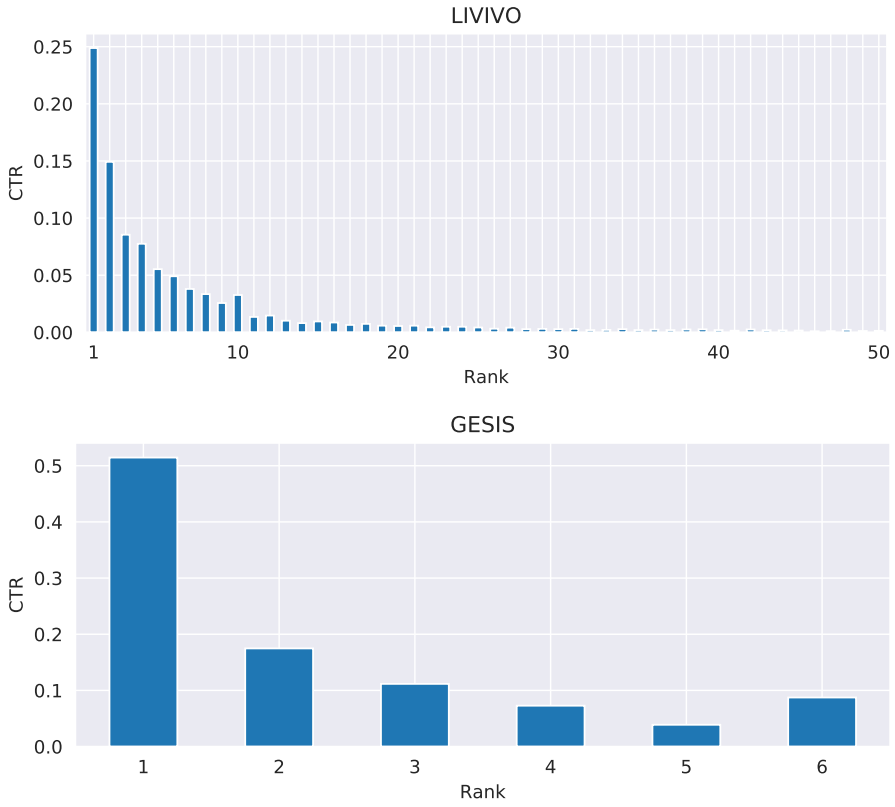

Figure D.2: CTR vs. rank

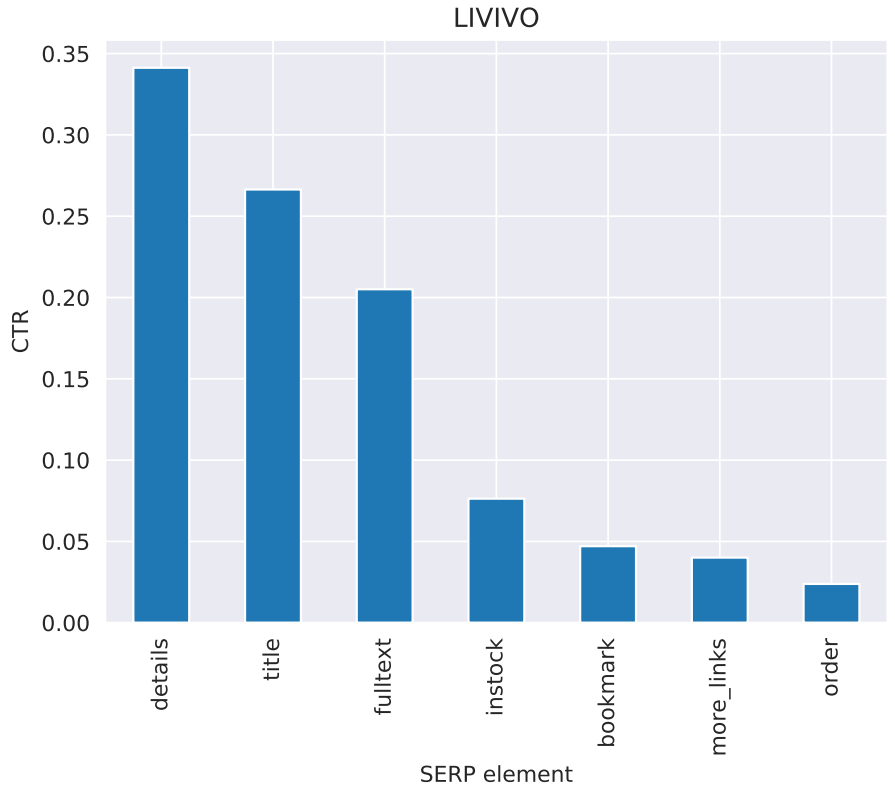

Figure D.3: CTR vs. SERP element (LIVIVO)

Acronyms

AI Artificial Intelligence.

AP Average Precision.

API Application Programming Interface.

ARP Average Retrieval Performance.

CCRF Cross-Collection Relevance Feedback.

CQG Controlled Query Generation.

CTR Click-Through Rate.

DCM Dependent Click Model.

DCTR Document-Based Click-Through Rate Model.

DL Deep Learning.

DRI Delta Relative Improvement.

EaaS Evaluation-as-a-Service.

ER Effect Ratio.

IIR Interactive Information Retrieval.

IR Information Retrieval.

IRM Interpolated Retrieval Methods.

KIS Known-Item Searcher.

KTU Kendall's τ Union.

LiLAS Living Labs for Academic Search.

LRM Lexical Retrieval Methods.

MCA Multi-Container Application.

ML Machine Learning.

MSLE Mean Squared Logarithmic Error.

nDCG Normalized Discounted Cumulative Gain.

NLP Natural Language Processing.

ORCID Open Researcher & Contributor ID.

QCM Query Change Model.

QLD Query Likelihood Model with Dirichlet Smoothing.

RBO Rank-Biased Overlap.

RBP Rank-Biased Precision.

RecSys Recommender Systems.

RI Relative Improvement.

RMSE Root Mean Square Error.

SDBN Simplified Dynamic Bayesian Network Model.

sDCG Session-Based Discounted Cumulated Gain.

SERP Search Engine Result Page.

TDI Team Draft Interleaving.

TTS TREC Topic Searcher.

UQV User Query Variants.

DOI: 10.17185/duepublico/78449
URN: urn:nbn:de:hbz:465-20230609-104129-2