THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

Query Automation for Systematic Reviews

Harrisen Scells

B.InfoTech (Hons)

0000-0001-9578-7157

A thesis submitted for the degree of Doctor of Philosophy at

The University of Queensland in 2021

School of Information Technology and Electrical Engineering

Abstract

This thesis explores and investigates query automation methods and tools to support more effective study retrieval for the creation of medical systematic reviews. A systematic review is a synthesis of very high-quality medical studies with a stringent protocol to guide creation with minimal bias and error. Modern medicine practice and healthcare policies rely heavily on systematic reviews in order to make decisions on the best available evidence.

Systematic review creation is often highly time-intensive and costly. Thus there has been much attention to attempt to automate phases in the creation of a systematic review. In this context, the definition of automation is the encoding of manual tasks into computational procedures. Existing automation research has primarily focused on the phase concerned with the critical appraisal of studies. Prior research has attempted to automate this appraisal phase by, for example, simulating human appraisal, (re-)ranking the retrieved studies, and classifying relevant studies, among others. A major drawback of these approaches is that they all rely on the initial search's retrieval effectiveness, which is contingent on the query's quality. We follow a fresh, new direction of research by investigating methods for query automation.

In investigating automatic methods for building effective queries for systematic review creation, this thesis develops three main areas of enquiry: (i) query formulation: the development of Boolean queries suitable for systematic review literature search, (ii) query refinement: the iterative process of improving the search effectiveness of a Boolean query suitable for systematic review literature search, and (iii) query exploitation: the application of methods to Boolean queries in order to improve systematic review literature search effectiveness. For each of these query automation areas, this thesis devises computational methods as well as practical tools.

This thesis makes contributions across three main areas of enquiry. They comprise of: (i) computational adaptations of human query formulation methodologies to support the automatic development of complex Boolean queries for systematic review literature search, and practical tools built upon them, (ii) a framework for the generation and identification of more effective query variations to support the automatic refinement of high-quality Boolean queries used in systematic review literature search, as well as tools built upon them, and (iii) methods to exploit intrinsic characteristics of Boolean queries in order to improve the retrieval effectiveness of systematic review literature search. We found that for each of these inquiry areas, query automation was able to improve search effectiveness, reducing the amount of time necessary to create systematic reviews.

Declaration by author

This thesis is composed of my original work, and contains no material previously published or written by another person except where due reference has been made in the text. I have clearly stated the contribution by others to jointly-authored works that I have included in my thesis.

I have clearly stated the contribution of others to my thesis as a whole, including statistical assistance, survey design, data analysis, significant technical procedures, professional editorial advice, financial support and any other original research work used or reported in my thesis. The content of my thesis is the result of work I have carried out since the commencement of my higher degree by research candidature and does not include a substantial part of work that has been submitted to qualify for the award of any other degree or diploma in any university or other tertiary institution. I have clearly stated which parts of my thesis, if any, have been submitted to qualify for another award.

I acknowledge that an electronic copy of my thesis must be lodged with the University Library and, subject to the policy and procedures of The University of Queensland, the thesis be made available for research and study in accordance with the Copyright Act 1968 unless a period of embargo has been approved by the Dean of the Graduate School.

I acknowledge that copyright of all material contained in my thesis resides with the copyright holder(s) of that material. Where appropriate I have obtained copyright permission from the copyright holder to reproduce material in this thesis and have sought permission from co-authors for any jointly authored works included in the thesis.

Publications included in this thesis

1. H. Scells, G. Zuccon, B. Koopman, A. Deacon, L. Azzopardi, and S. Geva. Integrating the framing of clinical questions via PICO into the retrieval of medical literature for systematic reviews. In Proceedings of the 26th International Conference on Information and Knowledge Management, pages 2291–2294, 2017

2. H. Scells and G. Zuccon. Generating better queries for systematic reviews. In Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 475–484, 2018

3. H. Scells and G. Zuccon. Searchrefiner: A query visualisation and understanding tool for systematic reviews. In Proceedings of the 27th International Conference on Information and Knowledge Management, pages 1939–1942, 2018

4. H. Scells, G. Zuccon, and B. Koopman. Automatic boolean query refinement for systematic review literature search. In Proceedings of the 28th World Wide Web Conference, pages 1646–1656, 2019

5. H. Scells, G. Zuccon, B. Koopman, and J. Clark. Visualising systematic review search strategies to assist information specialists. In Proceedings of the 2019 Cochrane Colloquium, 2019

6. H. Scells, G. Zuccon, B. Koopman, and J. Clark. Automatic search strategy reformulation interface for systematic reviews. In Proceedings of the 2019 Cochrane Colloquium, 2019

7. H. Scells and G. Zuccon. You can teach an old dog new tricks: Rank fusion applied to coordination level matching for ranking in systematic reviews. In Proceedings of the 42nd European Conference on Information Retrieval, pages 399–414, 2020

8. H. Scells, G. Zuccon, and B. Koopman. Sampling query variations for learning to rank to improve automatic boolean query generation in systematic reviews. In Proceedings of the 29th World Wide Web Conference, pages 3041–3048, 2020

9. H. Scells, G. Zuccon, and B. Koopman. A computational approach for objectively derived systematic review search strategies. In Proceedings of the 42nd European Conference on Information Retrieval, pages 385–398, 2020

10. H. Scells, G. Zuccon, B. Koopman, and J. Clark. Automatic boolean query formulation for systematic review literature search. In Proceedings of the 29th World Wide Web Conference, pages 1071–1081, 2020

11. H. Scells, G. Zuccon, and B. Koopman. A comparison of automatic boolean query formulation for systematic reviews. Information Retrieval Journal, pages 1–26, 2020

Other publications during candidature

1. H. Scells, G. Zuccon, B. Koopman, A. Deacon, S. Geva, and L. Azzopardi. A test collection for evaluating retrieval of studies for inclusion in systematic reviews. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1237–1240, 2017

2. D. Locke, G. Zuccon, and H. Scells. Automatic query generation from legal texts for case law retrieval. In Proceedings of the 13th Asia Information Retrieval Symposium, pages 181–193, 2017

3. H. Scells, G. Zuccon, A. Deacon, and B. Koopman. QUT ielab at CLEF eHealth 2017 technology assisted reviews track: Initial experiments with learning to rank. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017

4. H. Scells, D. Locke, and G. Zuccon. An information retrieval experiment framework for domain specific applications. In Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1281–1284, 2018

5. H. Scells, L. Azzopardi, G. Zuccon, and B. Koopman. Query variation performance prediction for systematic reviews. In Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1089–1092, 2018

6. S. Subpaiboonkit, X. Li, X. Zhao, H. Scells, and G. Zuccon. Causality discovery with domain knowledge for drug-drug interactions discovery. In Advanced Data Mining and Applications - 15th International Conference, pages 632–647, 2019

7. J. Palotti, H. Scells, and G. Zuccon. Trectools: An open-source python library for information retrieval practitioners involved in trec-like campaigns. In Proceedings of the 42nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1325–1328, 2019

8. H. Scells and G. Zuccon. Ielab at the open-source IR replicability challenge 2019. In CEUR Workshop Proceedings: Working Notes of CLEF 2019: Conference and Labs of the Evaluation Forum, pages 57–61, 2019

Contributions by others to the thesis

1. H. Li, H. Scells, and G. Zuccon. Systematic review automation tools for end-to-end query formulation. In Proceedings of the 43rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 25–30, 2020

Statement of parts of the thesis submitted to qualify for the award of another degree

No works submitted towards another degree have been included in this thesis.

Acknowledgments

Financial support

This research was supported by an Australian Government Research Training Program Scholarship. In addition, this research was also financially supported by the following bodies and awards:

1. Queensland University of Technology Excellence Top-Up Scholarship
2. The University of Queensland AOU Top Up Scholarship
3. CSIRO Top Up Scholarship
4. Google Faculty Award

Keywords

information retrieval, systematic reviews, query formulation, query refinement, automation tools

Australian and New Zealand Standard Research Classifications (ANZSRC)

ANZSRC code: 080704, Information Retrieval and Web Search, 100%

Fields of Research (FoR) Classification

FoR code: 0807, Library and Information Studies, 100%

To Amber

Contents

Chapter 1

Introduction

A systematic review is a highly specific, focused, and comprehensive literature review. Systematic reviews are used in many domains to answer challenging questions; however, this thesis is only concerned with medical systematic reviews. In medicine, systematic reviews are used extensively: most commonly for guiding clinical practice (for example, "which drug should be used to treat a patient diagnosed with a specific set of symptoms?"), but have also been used for institutional and governmental public health policy making [118]. A famous example of this is when the United Kingdom decided to ban smoking tobacco cigarettes in public places [19]). When used for guiding clinical practices, systematic reviews that synthesise high quality randomised controlled trials are often seen as the highest form of evidence to determine what action should be taken to treat patients [31]. Furthermore, there are also several instantiations of systematic reviews to support different use cases, for example: (i) rapid reviews for when evidence about a medical topic is required in a timely manner, (ii) scoping reviews for when more information is needed to begin a systematic review, and (iii) living systematic reviews for when new evidence is likely to continuously arise that can impact the results of the review and so must be periodically updated. While some of these variants of systematic reviews will be touched upon in this thesis, the main focus is on traditional systematic reviews.

Currently, creating a systematic review is an intensely methodical and time-consuming process. It typically requires the support of numerous researchers and collaborators whom all play specific roles, for example, specialised and highly trained librarians (also known as information specialists) to help search for medical literature. Furthermore, a systematic review comprises many time-consuming phases, and these phases must be performed systematically so as not to introduce bias or errors into the review.

Several studies have sought to break down the creation of systematic reviews into smaller, more defined, phases, ranging from four to fifteen [38, 52, 230, 231, 239]. This thesis classifies the creation of a systematic review into six distinct phases. Figure 1.1 provides an overview of these six high-level phases for systematic creation, and where the focus of this thesis resides in these phases. What follows is a brief explanation of each phase in Figure 1.1 in order to provide an understanding of how a systematic review is created, why this thesis chooses to focus on search strategy development, how

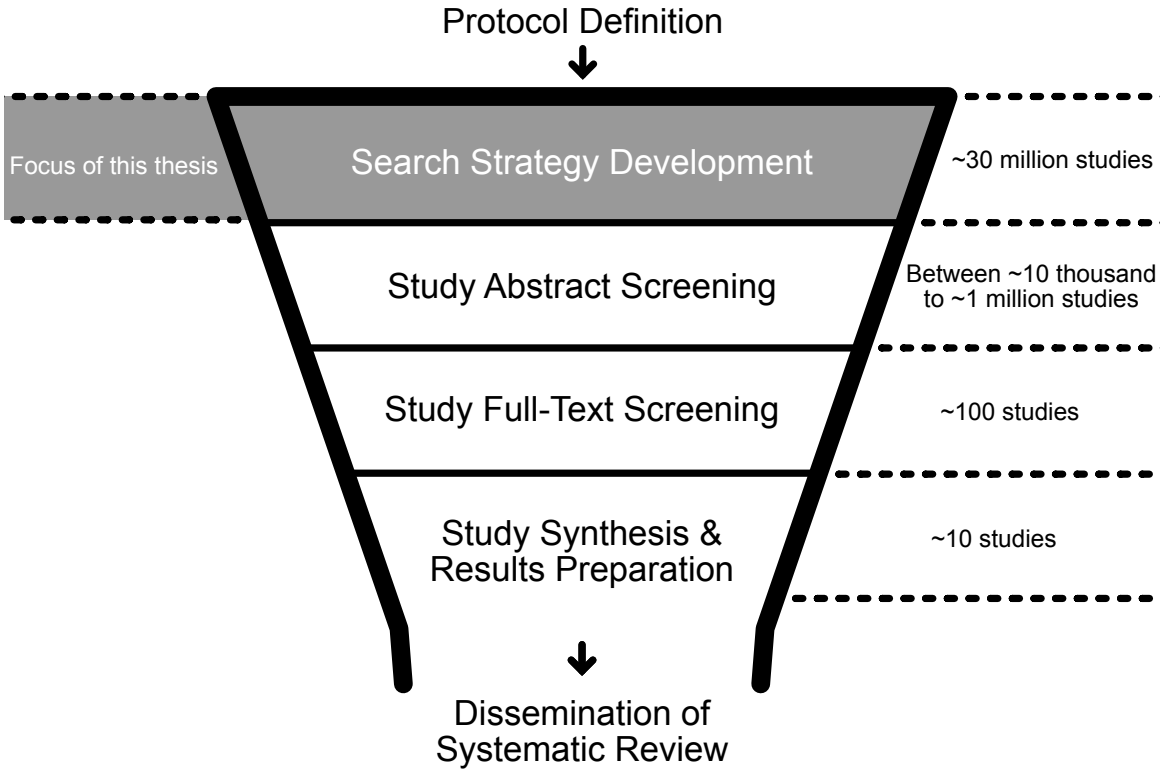


Figure 1.1: High-level phases involved in the creation of a systematic review. Each of the phases that relate to the reduction of studies is represented as a funnel. The highlighted portion of the funnel indicates the focus of this thesis. The labels on the right hand side of the funnel indicate the approximate number of studies that are the input to each step in a typical systematic review.

this thesis tackles problems that arise in the search strategy development phase that affect downstream phases through query automation, and what query automation means within the context of this thesis.

First, a *protocol* is written to guide the following phases of the systematic review. The protocol is a guiding document that sets out how the review should be completed. Next, a *search strategy* is developed, often by highly trained information specialists. The main component of a search strategy is a Boolean query that is used to retrieve medical literature for later synthesis. Exactly which studies from the literature that will be synthesised are identified next in two screening phases. During the *study abstract screening* phase, researchers critically appraise each study's abstract retrieved by the search strategy. Once a subset of potentially relevant studies has been identified, the *study full-text screening* phase begins. During this phase, the potentially relevant studies' full-text are critically appraised for inclusion in the systematic review. The results of a subset of these studies are then synthesised during the *study synthesis & results preparation phase*. Only then can the systematic review be published, in the *dissemination of systematic review* phase.

As Figure 1.1 illustrates, the steps in the systematic review creation phases that involve the highest amount of studies are search strategy development and study abstract screening. Indeed, the most costly and time-consuming aspect of systematic review creation is the study abstract screening phase. This phase contributes the most to the approximately AUD$350,000 [7, 133] cost to create a systematic review. There is often a cost associated with screening each study abstract, typically between AUD$0.30 and $2.00 [133]. Systematic reviews can also take upwards of two years to complete [133].

Furthermore, as a result of this time-consuming screening phase, systematic reviews are likely to become out of date at the time of publishing [188] (although there are other less time-intensive reasons for the delay, such as publication processes and peer review). This delay is why most research has focused on directly reducing the amount of time it takes to screen study abstracts. Here, previous work has focused on reducing the workload by automating the screening process, through methods such as active learning [51,141,243], ranking and classification [132], and text mining [9,56,109,153,210,220]. However, with the growth of online bibliographic databases such as PubMed (which has gained over 1,000,000 studies each year for the past decade; Figure 2.1), it has become clear that another approach to reducing the amount of time it takes to screen studies for systematic reviews is necessary.

In this thesis, we take a dramatically different approach to those before us by focusing our efforts on the search strategy development phase. We argue that the search strategy underpins the quality, effectiveness, and costs of a systematic review. We therefore posit that the most effective way to tackle the main problem of study abstract screening is to **reduce the total number of irrelevant studies** through computational methods that utilise the search strategy. Also, by reducing the number of irrelevant studies retrieved, we suggest that existing automated screening methods also benefit. Reducing the number of irrelevant studies is tackled through several query-centric methods, referred to as query automation. In this thesis, we research three query automation approaches: (i) query formulation – the automatic development of Boolean queries, (ii) query refinement – the automatic modification of existing Boolean queries, and (iii) query exploitation – automatic methods that exploit characteristics of Boolean queries to improve the effectiveness of a search. A survey of systematic review automation technologies by Tsafnat et al. [231] suggests that query automation has a very high potential to reduce systematic review creation costs, allowing searches to be made more comprehensive (i.e., increasing recall) or more narrow (i.e., increasing precision). In addition to computational models of query automation, we also provide implementations in the form of tools for information specialists to use to support them in developing more effective search strategies for systematic review literature search. The following section presents the query automation methods investigated in this thesis to reduce the number of irrelevant studies in the study abstract screening phase.

Summary
Systematic review creation suffers significantly from the burdens of time and money. However, one specific phase can be attributed to the lions share of the costs: the study abstract screening phase. There are opportunities to reduce these costs through automation in supporting those faced with searching and screening studies with automatic methods. We posit that query automation is the most effective and direct way to reduce the time and cost factors involved in screening to reduce the total number of irrelevant studies retrieved.

1.1 Research Questions

The primary research focuses of this thesis are the investigation into query automation methods to support the *development* of search strategies, ways to *exploit* the search strategy for screening, and tools to *assist* those who work with search strategies. As such, this thesis focuses on automating queries for systematic review literature search: specifically, the **formulation**, **refinement**, and **exploitation** of queries used to perform the systematic search. These three tasks comprise the main aspects of the **search strategy development** phase. Each of these three tasks also forms the basis for the three research questions of this thesis. The following research questions (**RQ**s) are used to guide investigation:

RQ1

Can search strategies be automatically formulated and if so, are they comparable in effectiveness to search strategies manually formulated by information specialists?

There are several facets to formulating search strategies that could be automated, for example, the translation of a query developed for one database into an equivalent query for another. However, the focus of this RQ is query formulation. Within the context of this thesis, query formulation is the fully automatic development of a search strategy without any human intervention. There has been no prior work investigating the fully automatic formulation of complex Boolean queries for systematic review literature search prior to the research in this thesis. Along with an investigation into fully automatic query formulation methods, this research question also seeks to determine if the effectiveness of such queries are comparable to those developed manually by information specialists. To this end, Chapters 3 and 4 provide a theoretical description of the two most prominent human-based query formulation methodologies. These theoretical frameworks form the basis for computational adaptations, encoding the approximate intuitions and processes information specialists take. Experimental results and in-depth analysis and comparison of the two computational adaptions follow in Chapter 5, where automatically formulated queries are compared to manually formulated queries.

RQ2

Can manually formulated search strategies be automatically refined to produce more effective search strategies?

The second research question investigates if high-quality Boolean queries, expertly manually developed for systematic review literature search can be automatically improved. There have also been relatively few studies that have investigated the automatic improvement or refinement of Boolean queries in the context of systematic review literature search outside this thesis. As such, this research question requires the investigation into new methods for this novel task and an evaluation of these methods to determine their effectiveness empirically. To address RQ2, a framework for automatically generating query variations of Boolean queries that exploits domain-specific characteristics is proposed, as well

as a framework for ranking query variations in order to predict the most effective query from a set of variations. The framework is introduced in Chapter 7 and the framework is evaluated in Chapter 8, comparing the effectiveness of automatic refinements to the effectiveness of high-quality, manually developed queries that were used to search for literature in existing systematic reviews. Further analysis of the sensitivity to training data of the framework is then presented in Chapter 9.

RQ3

Can the intrinsic characteristics of search strategies be exploited to further improve the effectiveness of systematic review literature search?

The third and final research question moves away from attempting to automate aspects of the search strategy development phase explicitly and instead aims to investigate if there are any characteristics of Boolean queries in this domain that can be exploited to improve the effectiveness of search. Specifically, this research question is about identifying and investigating methods that improve search without modifying Boolean queries in any way. Chapter 11 attempts to exploit the Boolean retrieval model by integrating a clinical question framework to better target aspects of studies that relate to the same aspects of the query. Meanwhile, Chapter 12 attempts to exploit the syntax and semantics of Boolean queries to produce a ranking of studies (i.e., existing Boolean retrieval systems do not rank studies). This research question is concerned with the indirect exploitation of existing search strategies rather than modifying or reformulating existing search strategies.

1.2 Contributions

The contributions of this thesis are categorised into: (i) theoretical Information Retrieval methods with empirical experimentation to automate the formulation, refinement, and exploitation of search strategies, and (ii) practical automation tool implementations that demonstrate support methods for an information specialist. It is expected that many of the methods presented in this thesis will be generalisable to other professional search domains, such as legal search and patent retrieval.

Firstly, this thesis will discuss the findings related to the automatic formulation of search strategies (i.e., Boolean queries), and a comparison between the methods. The findings are that fully automatic Boolean query formulation for systematic review literature search is possible, although these queries are often not as effective as those developed by information specialists. The following publications support these findings:

1. H. Scells, G. Zuccon, and B. Koopman. A computational approach for objectively derived systematic review search strategies. In Proceedings of the 42nd European Conference on Information Retrieval, pages 385–398, 2020

2. H. Scells, G. Zuccon, B. Koopman, and J. Clark. Automatic boolean query formulation for systematic review literature search. In Proceedings of the 29th World Wide Web Conference, pages 1071–1081, 2020

3. H. Scells, G. Zuccon, and B. Koopman. A comparison of automatic boolean query formulation for systematic reviews. Information Retrieval Journal, pages 1–26, 2020

Next, this thesis will discuss the findings related to the automatic refinement of search strategies (i.e., Boolean queries). The findings of these contributions are that high-quality queries developed by information specialists are not as effective as possible. More effective variations of high-quality, manually developed queries can be automatically identified; significantly reducing screening workload. The following publications support these findings:

1. H. Scells and G. Zuccon. Generating better queries for systematic reviews. In Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 475–484, 2018

2. H. Scells, G. Zuccon, and B. Koopman. Automatic boolean query refinement for systematic review literature search. In Proceedings of the 28th World Wide Web Conference, pages 1646–1656, 2019

3. H. Scells, G. Zuccon, and B. Koopman. Sampling query variations for learning to rank to improve automatic boolean query generation in systematic reviews. In Proceedings of the 29th World Wide Web Conference, pages 3041–3048, 2020

Next, this thesis will discuss the findings made which relate to the exploitation of search strategies. The findings are that there exist characteristics of Boolean queries in the context of systematic review literature search that can be exploited within this thesis and that by exploiting these characteristics, the screening workload can be significantly reduced. The following publications support these findings:

1. H. Scells, G. Zuccon, B. Koopman, A. Deacon, L. Azzopardi, and S. Geva. Integrating the framing of clinical questions via PICO into the retrieval of medical literature for systematic reviews. In Proceedings of the 26th International Conference on Information and Knowledge Management, pages 2291–2294, 2017

2. H. Scells and G. Zuccon. You can teach an old dog new tricks: Rank fusion applied to coordination level matching for ranking in systematic reviews. In Proceedings of the 42nd European Conference on Information Retrieval, pages 399–414, 2020

Finally, this thesis will present automation tools that support information specialists formulate more effective search strategies (i.e., Boolean queries) for systematic review literature search.

Tool Demonstration

Examples of query automation tools for systematic review literature search to support information specialists in developing more effective queries.

As automation tools are an important factor in creating systematic reviews in a more timely manner, two of the three parts of this thesis concludes with a tool demonstration. Several tools developed in parallel with this thesis are presented at the end of Part 1 in Chapter 6 and at the end of Part 2 in Chapter 10. The following publications support the tool demonstrations:

1. H. Scells and G. Zuccon. Searchrefiner: A query visualisation and understanding tool for systematic reviews. In Proceedings of the 27th International Conference on Information and Knowledge Management, pages 1939–1942, 2018

2. H. Scells, G. Zuccon, B. Koopman, and J. Clark. Visualising systematic review search strategies to assist information specialists. In Proceedings of the 2019 Cochrane Colloquium, 2019

3. H. Scells, G. Zuccon, B. Koopman, and J. Clark. Automatic search strategy reformulation interface for systematic reviews. In Proceedings of the 2019 Cochrane Colloquium, 2019

4. H. Li, H. Scells, and G. Zuccon. Systematic review automation tools for end-to-end query formulation. In Proceedings of the 43rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 25–30, 2020

Query automation is a currently under-utilised task within systematic review creation. Through the development of more effective queries and the exploitation of the characteristics of these queries, the number of studies to screen can be significantly decreased, while maintaining the comprehensiveness required. The main finding of this thesis is that query automation in the three forms discussed above (i.e., formulation, refinement, and exploitation) can all have considerable benefits on reducing the workload of both information specialists and researchers screening study abstracts. Automatic query formulation provides a basis for information specialists to continue to develop upon. Automatic refinement provides more effective alternative writings of queries that are more effective at retrieving fewer irrelevant studies. Query exploitation provides methods for improving the effectiveness of a search, either through more restrictive filtering, ranking studies or by exploiting the syntax or semantic of search strategies. These methods can help directly reduce the time and cost factors involved in systematic review creation, leading to more timely and cost-effective reviews and, thus, more accurate and up-to-date healthcare advice.

Chapter 2

Background and Literature Review

The focus of this thesis is query automation: that is, query formulation, refinement, and exploitation. Query automation is a severely under-investigated research area in the context of systematic reviews despite having the potential to drastically reduce the time and costs associated with constructing systematic reviews. This chapter provides a literature review of, and the background required to understand:

- What a systematic review is and the methodologies involved in the systematic review creation phases (Section 2.1). **Systematic reviews** are meta-analyses of medical literature (e.g., randomised controlled trials) used in evidence-based medicine by medical practitioners primarily for the diagnosis and treatment of symptoms their patients may have. Here, each phase in the systematic review creation pipeline is described in detail.
- Why automation of these phases is key to the future of systematic review creation and what has been done already to automate phases involved in the creation of systematic reviews (Section 2.2). This section provides a comprehensive literature review of computational models and automation tools for each of the systematic review creation phases.
- The key Information Retrieval concepts related to query automation, specifically, the evaluation of, and the searching for medical literature (Section 2.3). **Information Retrieval** is a field of Computer Science concerned with the indexing and retrieval of documents and evaluating these tasks. This section provides a background on the evaluation and retrieval techniques seen throughout the remainder of the thesis.
- The gaps this thesis aims to address through query automation (Section 2.4). This section provides a review of Information Retrieval methods for formulating, refining, and exploiting queries for search in other domains.

Those who are familiar and comfortable with **systematic reviews**, e.g., information specialists, information scientists, or researchers who conduct systematic reviews, may wish to skip these first two sections. Those familiar and comfortable with **Information Retrieval** may wish to skip the last two sections.

To frame where this thesis sits in the timeline and contexts of Information Retrieval and systematic reviews, it will begin with a brief history of both. This history also provides background on why Boolean queries are used for systematic review literature search and the historical contexts of why search for systematic review literature search is the way it is today.

Early 1950s The field of Information Retrieval has its origins with the rise of machine-organised library systems and the demise of manual processes to organise information [49]. In the United States of America, institutions are trying to build systems to replace manual and time-consuming catalogue card-based indexing methods.

1960 Arising in the early 1960s is the introduction of the Cranfield experiments [48]. These experiments would set the foundations for what would eventually become how most Information Retrieval experiments are designed, tested, and evaluated, even to this day. These early experiments used scientific abstracts as their document corpora, one of the most common use case for early Information Retrieval systems.

1964 The development of MEDLARS (Medical Literature Analysis and Retrieval System) is completed [179]. MEDLARS was developed due to the National Library of Medicine Index Mechanization Project which began in 1958. It indexes medical abstracts to be made searchable through complex Boolean queries.

1971 MEDLARS goes online, with the introduction of MEDLINE. MEDLARS becomes obsolete. Note that this pre-dates the invention of the *World Wide Web*, which came about in 1989.

1972 Cochrane emphasises the importance of randomised controlled trials in medical practice and healthcare for guiding clinical decision making in his book *Effectiveness and Efficiency* [50]. He issues a call for the uptake of systematic reviews and how they could prevent countless unnecessary medical mistakes and errors.

1992 The phrase 'evidence-based medicine' is coined [86], and subsequently changed how medical practice is thought. Medicine could now be rooted in evidence, accurate, and *systematic*. Meanwhile, the very first TREC (Text REtrieval Conference) conference for Information Retrieval was run [74]. TREC was seen as a modern incarnation of the Cranfield experiments and is still running today. MEDLINE surpasses 10 million indexed abstracts.

1993 The Cochrane Collaboration is founded in response to the call from Cochrane [37]. Today, the Cochrane organisation is one of the world leaders for high-quality systematic reviews.

1997 The National Center for Biotechnology Information introduces PubMed, a freely accessible, online retrieval system for MEDLINE.

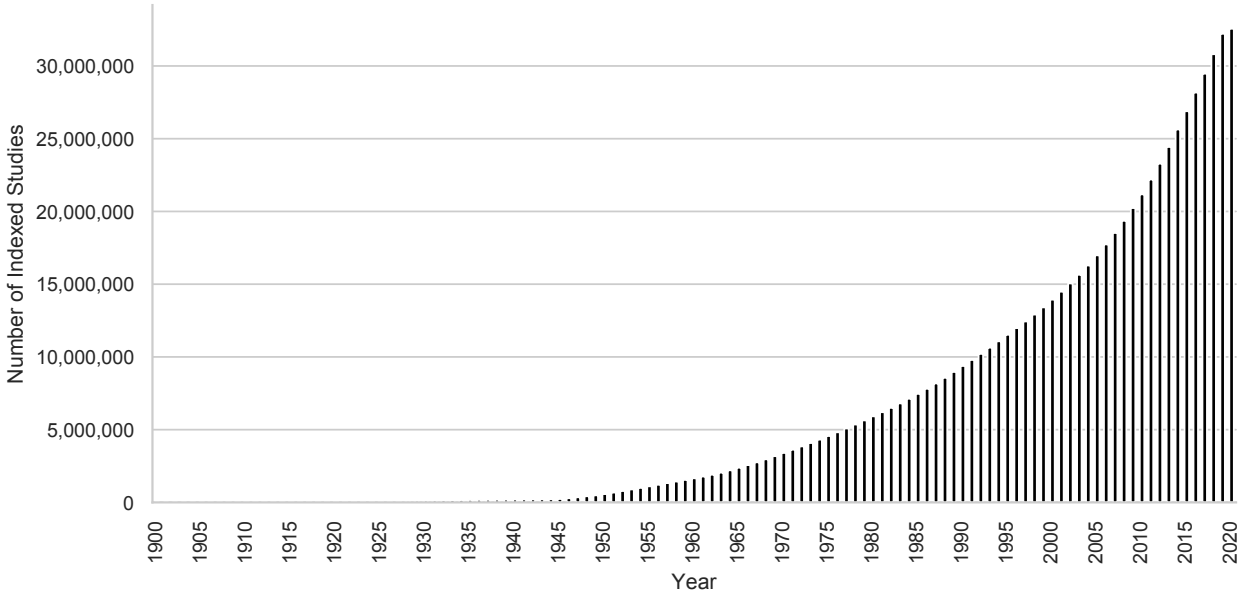


Figure 2.1: Cumulative growth of PubMed year-over-year from 1900 to early 2020.

2006 The first mentions of 'reducing workload' and 'automation' appear about systematic reviews [51] concerning using Information Retrieval methods.

2012 PubMed surpasses 20 million indexed abstracts.

2017 The Information Retrieval community widely recognises the need to address the workload problem associated with systematic reviews, and the first shared task towards tackling the problem, the CLEF Technology Assisted Reviews (TAR) track, is run [104]. Although the workload problem had been recognised and addressed independently multiple times before this point, e.g., [51, 108, 132, 153].

2018 PubMed surpassess 30 million indexed abstracts (Figure 2.1).

Only within the last two decades has the idea to guide clinical decision making with evidence taken a foothold. The combination of the uptake of systematic reviews by the medical community and the rise of searchable medical databases has seen the number of studies explode, evident in Figure 2.1. As the pool of medical literature grows, it becomes increasingly more difficult to find relevant studies. The growth of PubMed has seen a need for automatic methods to reduce the workload and costs of the systematic review process. Recently, the Information Retrieval community has increased interest in addressing the problems faced by those who create systematic reviews. These efforts will be presented later in this chapter. Before these can be discussed, however, an understanding of the phases involved in creating a systematic review is provided.

Summary

Early Information Retrieval methods are still the foundation for modern search systems for systematic review literature search. Due to the methodical nature of systematic reviews, the uptake of new systems and methods is slow. Only recently, as the amount of medical literature grows uncontrollably, has the adoption of new methods begun.

2.1 Overview of the Systematic Review Creation Process

The introduction chapter of this thesis provided a brief, high-level overview of the systematic review creation phase. Here, six high-level phases were identified that relate to different aspects of the systematic review creation pipeline. These phases are described in more detail below. The Cochrane Handbook for Systematic Reviews of Interventions presents five key characteristics of a systematic review [38]. Each phase addresses one or more characteristics of a systematic review:

1. A clearly stated set of objectives with pre-defined eligibility criteria for studies.
2. A detailed and reproducible methodology.
3. A systematic search that attempts to identify all studies that meet the eligibility criteria.
4. An assessment of the validity of the findings of the included studies.
5. A systematic presentation, and synthesis, of the characteristics and findings of the included studies.

This section aims to provide an in-depth introduction of the phases involved and the identification of research gaps. In particular, the search strategy development subsection includes a critique of current practices, and suggestions for areas of improvement, focusing on query automation.

2.1.1 Stage 1: Protocol Development

The first step in the creation of a systematic review is protocol development. The protocol addresses the first two characteristics of a systematic review: (1) *a clearly stated set of objectives with pre-defined eligibility criteria for studies*; and (2) *an explicit and reproducible methodology*. A systematic review protocol contains all the procedures and research goals, specifying the methodologies that the researchers, information specialists, and other collaborators on the review will follow. [1] Another reason for the protocol to be developed before starting any other downstream phase is to minimise any potential biases that relate to evidence identified in studies [38]. The most important aspects of a systematic review protocol are the following:

[1] Systematic review protocols are often uploaded to databases such as PROSPERO. An example of a systematic review protocol can be found at `https://www.crd.york.ac.uk/PROSPERO/display_record.php?RecordID=22124`.

Research Question A succinct definition of the exact medical problem the systematic review aims to answer. In most scenarios, this is defined in terms of the *population* the review covers (e.g., males aged 20-50); the *intervention* the population is administered with (e.g., weight loss drug); the criteria for *comparison or control* (e.g., controlled exercise regime); and the *outcome* to measure (e.g., weight loss). This framework for medical problem definitions is commonly referred to as the PICO framework and is used extensively within other aspects of the protocol, other phases in the systematic review creation, and other areas of medicine.

Databases Searched There exists several medical databases that a literature search can be issued. This aspect of the protocol identifies the databases to be searched, the reasons for searching them, and the reasons for not searching within other databases. The most commonly searched database is PubMed. Others include Ovid Medline, EMBASE, and CENTRAL.

Eligibility Criteria A detailed description of what a study must contain or not contain to either be included or excluded from the review. It is typically highly precise as often only a handful of studies from all the retrieved studies of a search will be relevant to the review (and therefore be included). These criteria also make use of the PICO framework to categorise the aspects of studies that contribute to the eligibility for inclusion: e.g., for intervention reviews on two different drugs, it is essential to distinguish between which drug is the intervention and which is the control.

Data Extraction Describes how reviewers will extract data from studies, and specifically, it details how the reviewers will screen studies. It lays out who in the research team will undertake the screening phase, what steps will be taken to reduce bias (e.g., double-blind screening, where two or more reviewers independently screen studies), as well as how the data will be handled, and who will oversee the data management.

Risk of Bias Assessmment Enumerates any foreseen biases that may impact the quality and outcomes of the final review. Biases can arise from under- or over-estimating the results from individual studies in the meta-analysis, incomplete data, problems with experimental design (e.g., blinding, rotation design), or biased funding sources and conflicts of interest with the research team undertaking the review. The final systematic review will grade each included study (typically low, high, or none) for each source of bias identified in this section of the protocol.

Strategy for data synthesis Describes exactly which statistical tests, assessments, and analysis will be performed on the included studies' data. It also includes a statement on how conclusions will be reached.

Metadata Includes authors and their contact details, funding sources, conflicts of interest, the language of the review, and a time frame for completion of the systematic review.

In this thesis, the protocol is exploited for some of the automation methods proposed (in particular, the research question). However, the protocol is a rich source of information for carrying out automation

tasks on the downstream activities of the review. With a completed protocol, the reviewers typically perform a pilot search to identify key 'seed' studies that are highly likely to be included in the review. These seed studies are used to assist information specialists collaborating with the reviewers, providing them with the foundation to develop the search strategy.

2.1.2 Stage 2: Search Strategy Development

The development of a search strategy is key to the creation and completion of a systematic review. It addresses the third characteristic of systematic reviews: (3) *a systematic search that attempts to identify all studies that meet the eligibility criteria*. As alluded to in Chapter 1, the main component of a search strategy that is of most interest is a complex Boolean query, developed primarily by information specialists. The search strategy should report, in addition to the Boolean query, several other vital attributes about the search [38]; specifically: the databases searched (e.g., MEDLINE), name of the host (e.g., PubMed), date search was run, years covered by search, a summary of the search strategy (e.g., a detailed description of what each aspect of the research question the Boolean query aims to search), and language restrictions (e.g., what language the search is intended to be run in).

There have been numerous studies and guides dedicated to systematic review literature search query formulation methodologies [3,38,44,85,89,90,144,150,213]. Currently, there are two prominent methodologies for query formulation for systematic review literature search. These are the conceptual method [44] and the objective method [90].

The conceptual method relies on the information specialist to take a systematic review protocol, and optionally several seed studies (i.e., known-to-be-relevant studies provided by the researchers of the systematic review a priori), to produce a Boolean query. The phase of developing such a query begins with the information specialist identifying high-level concepts to represent the systematic review's information need (e.g., the research question). Each high-level concept becomes a sub-clause in a larger Boolean query. Next, synonyms for the high-level concepts are identified and added to each of the respective sub-clauses. These synonyms are grouped using an `OR` operator. To complete the query each of the sub-clauses is grouped using an `AND` operator.

The objective method relies less on the information specialist to choose the query's keywords and instead takes a statistical approach to query formulation. Keywords from a set of seed studies are extracted and sorted using a background collection. Keywords are then classified into different aspects of the search and combined in the same manner as the conceptual method. The objective method was developed in response to criticism of the conceptual method, which relies heavily on the experience and intuition of the information specialist [90]. Objectively derived search strategies aim to be less subjective than their conceptually derived counterparts. Other aspects the objective method address, as a side-effect of the keyword extraction and sorting component, is a reduction in time and human-introduced errors such as spelling mistakes, missing synonyms, incorrect Boolean operators, missing MeSH terms, irrelevant keywords, and redundancy can be introduced due to human error [84, 187]. Despite this, appropriate seed studies must first be chosen in order for the objective

method to be effective (as will become apparent in Chapter 5). The selection of seed studies introduces new problems, such as whether they are representative of the research question.

Query Formulation Summary
When developing an initial systematic review search strategy, query formulation is difficult, time-consuming, and prone to human error and bias, even for trained experts.

Once a query has been formulated, however, the search strategy development phase is not complete. The query is then iteratively refined to improve the search quality further (e.g., the addition of keywords that are likely to retrieve more relevant studies, and the removal or replacement of over-representative keywords that retrieve too many studies). The refinement of a Boolean query occurs once the information specialist is satisfied that the query captures as many relevant studies as possible [44]. Query refinement is a challenging task, and information specialists must be confident that the refinements they make to a query do not reduce the number of relevant studies retrieved.

Like query formulation, there has been little work to refine queries for systematic review literature search automatically.

Query Refinement Definitions
Refinement can take many forms: query expansion, query reduction, and query rewriting. Query **expansion** and **reduction** deal with the addition and subtraction of keywords from queries, respectively. Query **rewriting** deals with modifying aspects of queries, e.g., changing the logical operators in a Boolean query to broaden or narrow a search.

Clark [44] provides several examples of how a query may be refined once the initial query has been formulated. He suggests drawing upon the expertise of colleagues to identify key studies and studies by specific authors that the query should be retrieving. Terms from key studies can then be added to the search strategy (query expansion). Note that this advice does not include query reduction or query rewriting. The goal of query refinement in search strategy development methodology research typically only focuses on broadening the query's scope. This broadening only increases the amount of work required by the screeners but reduces the likelihood that a study is missed. During the query refinement process, the query is weakly evaluated using 'seed studies' to gauge the effectiveness of the search. Through experience, the information specialist can determine if the query is satisfactory for the reviewers to begin screening. In some cases, to assist in this evaluation process, the search strategy is peer reviewed [38]. This peer-review process may also capture the previously described errors. However, determining the effectiveness of the query once it has been refined, is still a subjective process, and tied to the expertise of the review board.

Query Refinement Summary
Refining the Boolean query component of a search strategy is subject to the same human errors and bias as in the formulation process. Furthermore, the query is typically only weakly evaluated, limited by the information specialist's expertise, if performed at all.

The Boolean query component of search strategies has several characteristics that are ripe for exploitation to improve the effectiveness of a search. The exploitation of query characteristics can lead to the retrieval of fewer studies, reducing the amount of time needed to screen abstracts or the inducing of a ranking of studies.

The majority of queries formulated for systematic review literature search use the PICO framework for high-level concept organisation (i.e., using the conceptual query formulation method). Although PICO is used to develop and organise the Boolean query, this characteristic is not exploited to constrain the retrieval of studies (i.e., although the query is developed using PICO categories, keywords in the query search the entire abstract). A randomised control trial study by Schardt et al. [206] compared PICO use against no standardised framework to guide the formulation of search strategies. The study found that PICO uses tended to retrieve results for screening with greater precision than those search strategies formulated without the use of PICO: this further motivates investigating the use of PICO as operators in queries to constraint retrieval in Chapter 11.

In modern Information Retrieval systems, a ranking over the retrieved items is induced by exploiting the relationship between a query and the documents it retrieves. In systematic review literature search, all of the studies retrieved by the query are screened for inclusion eligibility. Therefore, a ranking of studies is typically only ever considered in combination with a cut-off technique. A cut-off enables only the top-k ranked studies to be screened. However, current advice from organisations such as Cochrane does not suggest to use such methods [38].

Query Exploitation Summary
Current practice in systematic review literature search ignores intrinsic characteristics (such as PICO) about the queries used to search, restricting the effectiveness of the query.

The review of current search strategy development practices has produced three key observations. These are the gaps that are addressed in the three parts of this thesis.

Observation 1: Query formulation is difficult, requiring the expertise of trained information specialists. Despite this, subjective bias and human-introduced errors still arise in queries.

Observation 2: Query refinement is critical to the success of an effective search strategy. Despite this, the evaluation methodologies used to gauge the effectiveness of queries are lacking, and the refinement process is entirely subjective.

Observation 3: Query exploitation is often completely overlooked, despite the types of queries used for systematic review literature search having characteristics that lend themselves to methods of exploitation to increase search effectiveness.

2.1.3 Stage 3: Study Abstract Screening

Once studies have been retrieved from one or more databases, the systematic review researchers can begin the study abstract screening phase. This process also addresses the third characteristic of systematic reviews: (3) *a systematic search that attempts to identify all studies that meet the eligibility criteria*. Before any reviewer looking at any of the retrieved studies, several pre-processing steps are applied to the entire set of studies. For example, if multiple databases are searched, the results of all databases are merged, and duplicate instances of the same study are removed. The retrieved studies are then often loaded into tools to assist with the study abstract screening phase [98, 155, 158, 226, 242]. These tools usually present the researcher one study at a time, sometimes with highlighted keywords and phrases (i.e., user-specified or automatically annotated PICOs). It is important to note that most of these tools do not apply any form of automation, and instead act as a convenient way for researchers to assess studies. It is common for each study retrieved to be screened at least twice by two independent researchers. Those studies that meet all of the inclusion criteria set out in the study protocol and identified by all researchers are sent to the next phase where the full-text of studies are obtained. A study that meets only some of the inclusion criteria is not eligible for full-text screening and will be explicitly excluded [38]. When there are disagreements over whether a study should be included after all studies have been independently screened, a discussion between reviewers resolves the conflict, or other researchers are contacted to resolve it.

2.1.4 Stage 4: Study Full-Text Screening

After abstracts of studies have been identified as eligible for inclusion, the full-text of those studies are obtained and a second screening phase performed in the same manner as the study abstract screening phase, but on the full-text. This phase addresses the fourth characteristic of systematic reviews: (4) *an assessment on the validity of the findings of the included studies*. It is in this phase that studies are assessed for inclusion in the results of the systematic review. To decide this, a risk of bias process is performed on each study as described in the systematic review protocol. During this process, it is explicitly stated whether the study presents low enough risk of bias to be synthesised in the results, or the risk of bias is too high, and it is excluded.

2.1.5 Stage 5: Study Synthesis and Results Preparation

Once the final set of full-text studies are identified, the synthesis of studies and results preparation may begin. This phase addresses the fifth characteristic of systematic reviews: (5) *A systematic presentation, and synthesis, of the characteristics and findings of the included studies*. The results of studies are combined in the way proposed by the systematic review protocol. The way studies are synthesised is not relevant to this thesis.

2.1.6 Stage 6: Dissemination of Systematic Review

The systematic review is then peer-reviewed and published in the target journal venue as specified in the review protocol. It is common for updates to be applied to published systematic reviews, for example, if more studies are identified or undertaken post-publishing. As systematic reviews are so heavily relied upon as strong evidence sources, updating the review with new data is essential. One primary reason for this is that additional data may change the outcome of a review. Recently, there has been a push within the systematic review community for *living* systematic reviews. These documents are updated continuously and can be highly useful in unexpected events such as global pandemics where new information may arise daily [128].

2.2 Automation of Systematic Review Creation

With an overview of each of the required phases of systematic review creation provided above, this section aims to summarise the computational methods and software tools used for systematic review automation. The focus of this thesis is on query automation for systematic review literature search. This section aims to paint a picture of why this thesis should focus on query automation by highlighting the lack of research about query automation compared to research that targets other systematic review creation aspects. A summary of the computational methods and software tools that address automation tasks in the different systematic review creation phases is presented in Table 2.1. Each following subsection discusses computational methods and software tools that address automation tasks for each phase.

Computational methods may not have an associated system that end-users (i.e., information specialists or researchers) can use. Automation software (possibly derived from a computational method) are systems those end-users can use. This distinction is made in this thesis because tools that researchers and information specialists can use are seen as more valuable than any computational method they cannot use. The use of software tools for automating phases of the systematic review creation process has demonstrable effects, as evidenced by the creation of a traditional systematic review in only two weeks [45] (note that it takes, on average 1-2 *years* for the creation of traditional systematic reviews [188], when no or only some automation is used). For some of the methods and tools that live on a fine line between the distinction of computational method and software tool, they have been categorised based upon where they are most influential. For example, a popular tool grounded in a standard computational method is classified as a software tool, but an underused tool based on a groundbreaking computational method is classified as a computational method.

All of the phases described in the previous section except for dissemination of systematic review have at least one associated computational model or automation tool to address an automation task. This likely because reviewing and publishing currently require humans (e.g., peer review).

Phase	Automation Task	Computational Methods	Software Tools
Protocol Definition	Templating	-	RevMan [1]
Search Strategy Development	Query Formulation	**Part 1**	2dSearch [181] polyglot [47] AutoDoc [122] AutoFormulate [122]
	Query Refinement	**Part 2**	QueryVis [193] QueryLens [122]
	Query Exploitation	**Part 3** PICO [24, 68]	-
Study Abstract Screening	Retrieval Models	[4,5,8,39,53,70,71,96,109, 119, 132, 139, 147, 149, 196, 214, 215, 248, 255]	-
	Text Mining	Classification [2, 111, 131, 221, 227] Term Identification [9] Simulation [153]	RobotSearch [131] Covidence [98]
	Active Learning	[51, 56, 59, 60, 69, 120, 141, 243]	rayyan [155] abstracktr [242]
	De-duplication	[100]	EndNote [25] Deduplicator [171]
Study Full-Text Screening	Snowballing	-	ParsCit [166]
	Text Mining	[178]	SARA [45]
Study Synthesis & Results Preparation	Data Extraction	[97, 223]	-
	Risk of Bias Assessment	-	RobotReviewer [129]
	Data Synthesis	[224]	Meta-Analyst [241] RevMan-HAL [229] RevMan Replicant [45]

Table 2.1: Overview of the computational methods and software tools that have been proposed in previous research to automate each aspect of systematic review creation.

2.2.1 Automating Stage 1: Protocol Definition

Automatic construction of systematic review protocols is not an actively researched area of automation. As such, only the task of **templating** the protocol has been investigated. Although Tsfnat et al. [231] note that there are areas that could warrant investigation, such as the automatic formulation of a research question and the identification of previous or related systematic reviews (to decide whether to go ahead with the protocol definition phase). Templating is an automation task that seeks to automatically structure and verify all of the components of a protocol. For Cochrane systematic reviews, the RevMan [1] software package is recommended to be used to define the protocol [38], as it

verifies all of the components of the protocol are complete. Exactly what parts of a protocol, e.g., the research question, inclusion criteria, risk of bias, etc., is an ongoing area of research [94]. Ensuring a protocol's completeness is critical to a systematic review as the protocol's primary goal is to reduce biases. The potential for automation in templating the protocol is low, however, and mainly benefits from methodological improvements to systematic review protocol definition policies of organisations and publishers.

Although not mandatory for the protocol, another ripe area for investigation into protocol definition is the **identification of seed studies**. Although it could be considered part of search strategy development, it precedes the main tasks of search strategy development and should be performed by the review's researchers (as opposed to the information specialists). Seed studies are used extensively in search strategy development to identify terminology to include in the search and gauge the effectiveness of the search. Seed study identification is currently not an active area of research, and there have been no previous works attempting to tackle this task. However, it is currently highly tedious to perform (as a small-scale literature search is undertaken to obtain these studies) and is subject to selection bias (resulting from shallow searching and random error in non-systematic searching).

Protocol Definition Automation Summary
There are very few tasks associated with protocol definition that may be automated. The amount of benefit to gain from automating these tasks is negligible compared to more time-consuming and costly systematic review creation phases.

2.2.2 Automating Stage 2: Search Strategy Development

The main tasks of search strategy development are **query formulation** and **query refinement**. The distinction between query formulation and refinement at first may appear to be somewhat vague, as both are tasks of the search strategy development phase (i.e., Figure 1.1). Within this thesis, query formulation is defined as the bootstrapping phase of search strategy development, where an initial query is constructed from the ground up. On the other hand, the query refinement phase begins with a query and results in an enhanced version of the input query (e.g., increasing the effectiveness of the query by broadening or narrowing the scope of the query). Both query formulation and query refinement are an under-looked area of systematic review automation. Indeed these areas of research are a major focus of this thesis due to a lack of research addressing them (although several automatic Boolean query formulation and refinement methods have been proposed in the past for other domains, discussed later in more detail within Section 2.4). This thesis contributes several automation tools that target search strategy development. These tools are signified in this section with a $\star$ symbol.

There are a variety of ways for humans to introduce errors into query formulation and refinement. While computational methods to formulate and refine Boolean queries for systematic review literature search are uninvestigated outside of this thesis, several tools have arisen that attempt to address errors introduced through human error and ease the burden of query formulation and refinement. To

overcome errors, the 2dSearch tool [181] provides a way for expert searchers to formulate queries in a structured and visual manner. This tool attempts to counteract errors related to the structure of Boolean queries. The AutoFormulate⋆ [122] tool goes one step further and automatically formulates Boolean queries given a set of seed studies. To help information specialists better understand their query, reducing errors such as spelling mistakes, missing or incorrect keywords, and redundancy, QueryVis⋆ [193] visualises Boolean queries. It visualises queries as a graph and annotates it with the number of (relevant) studies retrieved by each query clause. Another tool that attempts to reduce errors in search strategies is AutoDoc⋆ [122], which attempts to identify spelling mistakes and report errors automatically. QueryLens⋆ [122] is a tool designed to automatically refine Boolean queries using seed studies to guide the exploration of more effective variations of the original query. Finally, the polyglot tool [47] can be used by information specialists to automatically translate Boolean queries from the syntax of one medical literature database to any other — reducing errors that may result from the manual translation into other query languages.

Most of the research into the automation of search strategy development is focused on the exploration of software tools to support information specialists. The potential impact of these types of tools on the search strategy development phase and the downstream tasks is very high: more effective search strategies that retrieve fewer studies while retrieving all or most relevant studies result in fewer studies for the researchers to screen. However, research into computational methods for improving search strategy development does not exist outside of this thesis. Arguably, methods to improve queries' effectiveness have the highest impact on all other methods in other phases of systematic review creation. A poorly formulated query can impact both the time it takes to complete the systematic review and the review's overall quality (i.e., if the query fails to retrieve the majority of relevant studies).

Query exploitation is an under-looked area of potential for improving study abstract screening. Demner-Fushman and Lin [68], and Boudin et al. [24] have used PICO information to aid the retrieval of studies for systematic review screening. The former study used PICO, along with other information, to re-rank studies. The latter, instead, devised two approaches for using PICO within the retrieval process. The first approach involved creating separate language models for each of the PICO elements, using text that has been automatically annotated for each PICO element. Then queries and documents were matched using a mixture model that combined the separate language models, where specific importance weights were given to matches for specific PICO elements (e.g., weighting higher matches for the population element). Empirical results only showed limited improvements and no principled way to set the weights of each language model. A second approach was proposed that exploited both PICO information and structured information of the research studies. Empirical results showed improvements in terms of precision at ten and average precision; however, the impact of parameter tuning was unclear. To the best of our knowledge, there is no other research in the literature that directly exploits the Boolean query to support the study abstract screening phase outside of this thesis. An investigation into new techniques that directly exploit Boolean queries for this phase is presented in Part 3.

Search Strategy Development Summary
While several tools have arisen to support information specialists develop effective queries, they are relatively simplistic. Currently, no computational methods exist that provide theoretical frameworks to support the development of more effective queries. Furthermore, systematic reviews typically require Boolean queries to be used for the retrieval of literature.

2.2.3 Automating Stage 3: Study Abstract Screening

Once a search strategy has been fully developed through the formulation of a query and the subsequent refinement, the search strategy is executed on a medical database to retrieve studies for screening. As the study abstract screening phase is so time-consuming and costly, there has been much effort to reduce researchers' work. Indeed, there have been various computational models and automation tools to address the needs of researchers: through retrieval models, text mining, active learning, and de-duplication. The primary way to assist researchers to complete the screening phase more efficiently is through screening prioritisation. The process of screening prioritisation can be approximately split into two Information Retrieval methods: retrieval models (that seek to augment the retrieval process without interaction from a human, e.g., ranking functions), and active learning (that seek to continuously re-rank studies based on human feedback). Text mining may also help with screening prioritisation, although these methods are typically used for classification tasks (as opposed to ranking) [153,210,220]. De-duplication is another automation task that is not focused on screening prioritisation as it is only reducing the number of studies to screen.

Karimi et al. [109] showed that combining Boolean retrieval with BOW ranking has the potential for significant time savings (although it does require additional effort on the researchers part in formulating new BOW queries). [2] Here, the Boolean query is used for an initial retrieval of studies, and new BOW queries are then exploited to induce a ranking of those documents. The second query is used to speed up the screening process by ranking more relevant studies first (allowing the following phases of systematic review creation to begin in parallel). This process of deriving a ranking for systematic review literature search can be generalised as the task of *screening prioritisation*. All approaches to support study abstract screening that do not directly use the search strategy are broadly classified in this thesis as '**retrieval models**'. There has been a surge of research due to the CLEF Technology Assisted Reviews (TAR) track [104, 105, 107], where the majority of participants apply standard Information Retrieval methods. The CLEF TAR task provides two main streams: screening prioritisation and stopping prediction. Screening prioritisation has been defined above. *Stopping prediction*, on the other hand, is the identification of a rank position (given a ranking of studies) where screening may end early; with the aim that there is a small or no loss in recall, but a large decrease in the total number of studies to screen. Participants applied relevance feedback [8, 71, 96, 139, 147, 149, 215, 255], automatic supervised [53, 70, 119, 196, 214], and automatic unsupervised methods (which do not rely on any relevance feedback or human intervention) [4, 5, 39, 248]. Meanwhile, the stopping prediction task

[2] The definition of a BOW query is provided in the key IR concepts Section 2.3

has seen little participation and naïve techniques such as static score-based cut-offs [103], as well as techniques based on continuous relevance feedback [69]. These approaches do not use the Boolean queries directly; they instead resort to other representations, such as the title of the review (a sentence), which is contrived and unrealistic in the context of systematic review literature search. The information need is not captured as well as in the search strategy developed by an information specialist. However, the reason for using the title is that contemporary ranking methods rely on such a representation. It is notable that the ranking system of PubMed in their online search system [77] does not support Boolean queries, and instead attempts to translate the Boolean query into this contrived query representation before ranking.

Screening Prioritisation Summary
Existing methods to support screening prioritisation for systematic review literature search lean primarily on BOW queries, or other query representations. These representations limit the effectiveness of searches as the comprehensiveness of Boolean queries are not fully exploited.

Rather than relying upon the search system alone to produce a ranking, another method for screening prioritisation, called **active learning**, attempts to iteratively re-rank studies using feedback from humans. A small portion of participants to the CLEF TAR tasks [104, 107] used active learning to achieve significantly higher performance than the retrieval model approaches [59, 60]. One of the first to apply active learning for screening prioritisation in systematic review literature search was Cohen et al. [51], who used a voting perceptron-based classification model to continuously classify studies that contain evidence about a topic or not. Wallace et al. [243] is also notable for the construction of an active learning model that instead used support vector machines to discriminate between studies that were relevant or irrelevant with particular attention placed on an undersampling technique to tackle the problem of highly imbalanced ratios of positive and negative examples. More recently, Miwa et al. [141] proposed an active learning method that integrates certainty into a support vector machine classifier. This method attempts to mitigate 'hastily constructed' models biased towards a small subset of positive studies by weighting highly likely positive studies. One approach to active learning that does not rely directly upon the Boolean query used is the method of seed-driven document ranking (SDR) proposed by Lee and Sun [120]. This method uses a variant of query-by-example, where statistics about terms in studies (in this case clinical concepts) are used to induce a ranking (e.g., through the similarity of the items to be ranked and the example item). Although this method can outperform the participants from the CLEF TAR task, one limitation with SDR is that the effectiveness of the ranking is largely dependent on the chosen seed study. Furthermore, there is currently no way to predict how effective a ranking will be given a seed study. Methods that instead directly exploit Boolean queries for more effective searches and screening prioritisation will be presented in Part 3.

Perhaps, due in part to the success of active learning research in the literature, active learning tools have since arisen to support humans. One of the most prominent tools for this purpose is Rayyan [155]. Similarly to other computational methods proposed in the literature, Rayyan also uses a support vector

machine to classify relevant or irrelevant studies. However, it is unclear if it attempts to account for imbalances in the positive and negative class distributions as others have done beforehand. For some research groups conducting systematic reviews, active learning tools are a practical and genuinely useful component to the construction of systematic reviews. As Norman et al. [148] notes, active learning can significantly impact reducing the screening workload by massively cutting the number of studies needed to screen through stopping the screening process early. However, the Cochrane Handbook for Systematic Reviews of Interventions does not recommend the 'automatic elimination of [studies]' in the process of constructing a systematic review [38].

Active Learning Summary
Active learning has been shown to dominate shared Information Retrieval tasks on screening prioritisation, and provide practical benefits for researchers. However, the effectiveness of any active learning system is still limited by the query used to retrieve studies.

The scope of **text mining** for supporting study abstract screening goes beyond screening prioritisation and cut-off prediction. For example, Kim et al. [111] developed a method for the automatic classification of sentences into PICO classes. This method can provide benefits for researchers screening studies as it allows them to quickly identify relevant portions of a study without reading the entire abstract; however, it can not be used to prioritise one study over another. Work by Stansfield et al. [221] presented a model to cluster studies into discrete topics of interest. This model can have various benefits, ranging from informing how researchers should focus systematic reviews, to assisting information specialists in becoming confident their search strategy covers enough of the literature. Other approaches to text mining include automatic classification of randomised controlled trials [2, 131, 227], the automatic identification of expansion terms for search strategies [9], or simulating a second screener [82, 153].

Text Mining Summary
For systematic review creation, text mining can be useful; however, methods either address tasks that require relatively less time and effort than other more time consuming or costly tasks; or the amount of time saved from using these methods is marginal.

The last automation task to assist with study abstract screening is **de-duplication**. This task is usually the step before the researchers of the review starting to screen studies. De-duplication is the removal of duplicate studies from the set of studies retrieved by a search strategy. It is also common for multiple searches to be issued to different databases of studies; therefore, this step also removes the same studies retrieved by searching within different databases. Jiang et al. [100] proposed a rule-based algorithm to de-duplicate studies for bibliographic databases like PubMed. It uses document fields and substring matching to de-duplicate studies, however, no empirical evaluation was performed to quantify the effectiveness of the method. Two tools dominate the task of de-duplication: EndNote [25] and the SRA Deduplicator [171]. EndNote de-duplicates using simple exact matching on the authors, year, and title. The SRA Deduplicator uses a hand-coded, heuristic rule-based system similar to that of

Jiang et al. [100]. Rathbone et al. [171] compared the SRA Deduplicator with EndNote and found the Deduplicator found almost 50% more duplicates than EndNote, on a single search comprising approximately 4,500 studies.

De-duplication Summary
There already exist several tools and methods that effectively de-duplicate studies. De-duplication is an area of research where more sophisticated techniques may not have the potential to significantly reduce duplicate studies further.

2.2.4 Automating Stage 4: Study Full-Text Screening

While most automation tasks are focused on study abstract screening, several tasks are also focused on study full-text screening. In this phase of systematic review creation, there are far fewer studies to screen. Therefore, the automation tasks that address this phase are less focused on traditional Information Retrieval techniques.

One of the main tasks of the study full-text screening phase is identifying any literature that was not discovered through the previous study abstract screening phase (i.e., the search strategy did not retrieve it). This study identification task is typically achieved through a process called **snowballing**. Here, new studies may be identified by 'chasing references': Following the trail of citations from studies eligible for inclusion to new, potentially eligible studies. This process is usually performed manually by the researchers of the review, as there is little to be done in the way of automation. Indeed there are very few published tools that address this task (likely as a result of in-house tools or manual efforts), and, to the best of our knowledge, there are no computational methods that address this task (likely due to the simplicity of the task). One of the few tools that performs the snowballing task is called ParsCit [166]. This tool uses deep learning (word embeddings and an LSTM) to parse reference strings from studies. By extension, this also enables researchers to identify all of the references for several studies quickly, rather than expending manual effort to process them.

The other tasks involved in study full-text screening are broadly categorised as **text mining**. There has also been limited research for this task, likely for the same reason snowballing has limited research: the gap in effort required between manual and (semi-)automated methods is small. One task for study full-text screening is assessing the data within studies. Rodriguez-Esteban and Iossifov [178] propose an approach to mining figures from the full-text of studies to enable the faster assessment of the data within them. They use an SVM classifier to classify different types of figures, and then annotate those figures to enable retrieval over them. This method would support the researchers of the systematic review by allowing them to understand which studies have figures that contribute to answering the research question of the review. Perhaps the most crucial task in full-text screening is the acquisition of the full-text studies themselves. The SARA tool [45] automatically obtains the full-text versions of abstracts potentially eligible for inclusion by requesting access to various sources. This tool saves time from the researchers of the systematic review by allowing them to focus on other tasks that can be completed in parallel while the full-text of studies is obtained automatically.

2.2.5 Automating Stage 5: Study Synthesis and Results Preparation

Once the full-text of potentially eligible studies have been obtained and screened for inclusion in the systematic review, the next phase may begin. Within the study synthesis and results preparation phase, the full-text studies are used to produce a systematic review, answer the research question, and produce medical recommendations. This phase is concerned with the extraction of data from the full-text of studies, assessing the risk of bias of that data, and synthesising that data with other studies.

Manually extracting data from studies is highly intensive and critical to the success of the review. The types of data extracted, for example, are the results of clinical trials, the number of participants in those trials, or what interventions were tested. A study on automation tools to create a systematic review by Clark et al. [46] found that, after using automation tools for many phases of systematic review creation, the data extraction phase took approximately the same amount of time as screening. The **data extraction** automation task attempts to automate this tedious phase of systematic review creation. Summerscales et al. [223] developed a conditional random field classifier for named entity recognition of treatments, groups, and outcomes in medical studies. This classification of entities in studies assists the researchers of the systematic review by enabling them to quickly identify the parts of studies that are most relevant to the part of the review being written. Relatedly, Hsu et al. [97] proposed a model for automated statistical analysis extraction from studies using a part of speech tagger in combination with a concept annotator. This model would help the researchers of a systematic review as it would provide faster access to the relevant portion of studies required for results synthesis.

Following the extraction of data, studies are assessed for risk of bias. Bias can arise in many forms from *selection bias* (i.e., how participants in randomised controlled trial studies are assigned an intervention or control), to *conflicts of interest* (e.g., researchers affiliated to a company whose primary interest is to sell drugs that treat the condition under review). **Risk of bias assessment** is an automation task that seeks to automatically assess these biases in the full-text studies eligible for inclusion in the systematic review. Perhaps the most well-known tool to enable the automatic risk of bias assessment is RobotReviewer [129]. This tool uses a novel multi-task SVM model to simultaneously label sentences as different classes of bias (e.g., the previously mentioned selection bias or conflicts of interest). RobotReviewer supports six different risks of bias assessments: random sequence allocation, allocation concealment, blinding of participants and personnel, blinding of outcome assessment, incomplete reporting of outcomes, and selective reporting.

Once data has been extracted from the full-text of studies, and the studies have been assessed for bias, the data synthesis process can begin. This process has the potential for automation through the **data synthesis** task. A computational model proposed by Summerscales et al. [224] produces statistical summaries of results from studies using standard name entity recognition techniques with hand-coded heuristics and templates. Other more advanced approaches have been developed that are somewhat related to systematic review result synthesis, such as extracting comparisons in biomedical texts [78] and the synthesis of clinical answers for precision medicine [68]. Such statistical synthesis would assist reviewers in manually combining and analysing results from studies, potentially improving the quality and effectiveness of the review (through the identification and computation of statistics

that may not be found manually). To this end, several tools have been proposed that assist further in the synthesis of results, although to a much lesser extent than the previously discussed computational model. The Meta-Analyst tool [241] produces a meta-analysis of studies: this is a statistical report detailing the effects that individual studies are statistically underpowered. This tool can also be used to identify differences between studies that can be ignored to boost the statistical power of the systematic review. Meta-Analyst computationalises several previously tedious, manual statistical processes that are often quite difficult to compute. Less critical, yet still tedious to write sections of the review, such as the abstract, results, and discussion can also be automated through the use of the RevManHAL tool [229]. The RevMan Replicant tool [45] can also be used to automate the writing of the results section. Both of these tools use an XML-structured file format, commonly used to develop systematic reviews. They use the file to write the results using pre-defined templates automatically. The automatic generation of this text content reduces the amount of time the researchers of a systematic review spend writing content, and the content is typically structured in the same way regardless if a tool is used.

Study Synthesis and Results Preparation Summary
several automation techniques are used in practice to automate the study synthesis and results preparation step in the systematic review creation process.

2.2.6 Systematic Review Automation Conclusion

This section has provided a comprehensive review of the computational methods and software tools that seek to automate many of the systematic review creation process steps. This review has identified that query automation is a severely under-looked area of research in this domain. Furthermore, three clear areas of query automation have arisen in which the remainder of this thesis will focus on specifically. These are query formulation, query refinement, and query exploitation. This thesis will take an Information Retrieval approach to address these areas of research. Therefore, the next two sections explore the key concepts required to understand how new methods will be implemented and evaluated, and the literature gaps that new methods will fill.

2.3 Key Information Retrieval Concepts

This section aims to provide the background for how Information Retrieval methods can be evaluated, both from an information specialist perspective, an Information Retrieval researcher perspective, and some background on how Information Retrieval systems search for and rank documents. While Information Retrieval is a broad domain, this section presents the core aspects and tasks involved in this thesis. The topics in this section lay the foundations for the later chapters.

2.3.1 Evaluating Information Retrieval Systems

Evaluation provides the basis for understanding whether a query or search system is effective or not. The evaluation of a query for systematic review literature can be seen from two perspectives. The first is the information specialist, who likely has some knowledge about evaluation, but cannot rigorously do so, likely due to the search system. For example, a practice taken by information specialists is to evaluate how good a query is given the total number of documents reported in PubMed, but this is a relatively weak measurement. This measurement is weak because it does not take into account the relevance of studies. The other perspective is the Information Retrieval community, compelled by the Cranfield experiments and the resulting TREC workshops. The Cranfield experiments developed an evaluation framework for Information Retrieval systems that relies on (i) documents, (ii) queries, and (iii) relevance assessments for document-query pairs. Under this framework, an Information Retrieval system may be evaluated using several evaluation measures that use statistics about which documents were retrieved for which queries. This style of evaluation is commonly referred to as *system-based evaluation*. Information specialists perform this style of evaluation for their searches, although in a more ad-hoc fashion. Information specialists are sometimes given a set of seed studies (i.e., studies that are known to be relevant a priori to search strategy development) used to identify search terms to include in the query and to provide them with information about what kinds of studies are likely to be relevant to the systematic review. Although it is not explicitly stated that information specialists could and should do so in any literature about developing search strategies, e.g., the Cochrane Handbook [38], one may combine the IDs for the seed documents in the target database (e.g., PubMed) with the rest of the Boolean query, as seen in Figure 2.2. [3] This combination effectively limits the search to only the seed studies. From here, the information specialist can measure what portion of the seed studies have been retrieved by the search. Once they have measured this, they can remove the seed studies from the search to obtain how many studies the query retrieves overall. Using this information, the information specialist can, in this ad-hoc manner, measure the *recall* (sensitivity) and *precision* (specificity) of their search, at least for the seed studies. These are two widely used set-based measures for measuring the effectiveness of a search. In this thesis, recall and precision play a large role and are among the main evaluation measures used. The following two sections provide a formal definition of the evaluation measures that will be used in the chapters to come and the data that will be used to make these measurements.

[3] Note that unlike most modern web search systems, the IDs of documents are visible to the users, allowing them to be included in the query.

(($term_1$ OR $term_2$ OR ... OR $term_n$) AND ($pmid_1$ OR $pmid_2$ OR ... OR $pmid_n$))

Figure 2.2: Example of how a query may be constructed to enable its evaluation, in an ad-hoc fashion, by combining the PMIDs of seed studies with the rest of the query.

Information Retrieval Evaluation Measures for Systematic Reviews

The evaluation measures used in this thesis include the aforementioned recall and precision, other set-based measures, and some rank-based measures. Set-based measures deal with all of the documents retrieved by a system, for a query, and assume that they are unordered. On the other hand, Rank-based measures are concerned with the order of the documents retrieved by a system for a query and may consider a rank cut-off. A formal definition for, and the reasoning behind why each evaluation measure is chosen for this work is presented next.

Recall computes the proportion of relevant documents retrieved to the total relevant documents in the relevance assessments:

$$\text{recall} = \frac{|\text{relevant retrieved}|}{|\text{relevant}|}$$

Where $|...|$ indicates the number of documents in these sets. There may be very few studies relevant to a systematic review [205], and the loss of a relevant study may have a significant impact on the results of a systematic review [32]. Recall is an important measure to compute for systematic review literature search as it shows how many relevant documents are being retrieved.

Precision computes the proportion of relevant documents retrieved to the total number of documents retrieved:

$$\text{precision} = \frac{|\text{relevant retrieved}|}{|\text{retrieved}|}$$

Retrieving many studies will cause precision to be very small, regardless of how many relevant studies are retrieved. Precision is an important measure to compute for systematic review literature search as it represents an approximation of how much work must be done through screening to obtain the corresponding recall.

The balance between recall and precision must be considered for systematic review literature search. It may be easy to obtain a high recall, but to also ensure high precision is what information specialists are trained to do. This balancing act between recall and precision is also encapsulated within the F_β measure. This measure is the harmonic mean between recall and precision. In other words, the score for F_β is higher when both recall and precision are high:

$$F_\beta = (1+\beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) \cdot \text{recall}}$$

F_β is parameterised to be biased towards recall or precision (favouring one over the other). When $\beta = 1$, recall has the same weight as precision. When $\beta = 0.5$, precision has twice as much weight as recall. When $\beta = 3$, recall has twice as much weight as precision.

The next set-based evaluation measures that will be described have arisen out of Information Retrieval approaches for automating systematic review screening. These measures may not be familiar to the Information Retrieval community but are used often for evaluating Information Retrieval

approaches in the context of systematic reviews. The first measure is Work Saved over Sampling (WSS, Equation 2.2). Originally proposed by Cohen et al. [51] for use in a study classification scenario, WSS seeks to measure 'the percentage of papers that meet the original search criteria that the reviewers do not have to read':

$$WSS = \frac{TN+FN}{TN} - (1 + \frac{TP}{TP+FN}) \tag{2.1}$$

TN, FN, TP, and FN are the true negatives, false negatives, true positives, and false negatives made by the classifier. Of course, it does not make sense to use this measure directly to evaluate a retrieval scenario. Therefore, it was adapted in work by Scells et al. [204] for this purpose:

$$WSS = \frac{N + |\text{retrieved}|}{N} - (1 + \frac{|\text{relevant retrieved}|}{|\text{retrieved}|}) \tag{2.2}$$

Where N is the size of the collection. WSS is an important measure in evaluating search for systematic review literature search as it is a measure of how much work could be saved by the reviewers.

Number Needed to Read (NNR) is a measure that estimates how many non-relevant studies on average a reviewer must screen before coming across a relevant study [16]:

$$NNR = \frac{1}{\text{precision}} \tag{2.3}$$

NNR was proposed as an analogy to Number Needed to Treat (NNT) [117], which was introduced to 'summarise the effect of a treatment in terms of the number of patients a clinician needs to treat to prevent one adverse effect' [54].

Finally, the last two set-based evaluation measures were specifically designed as a result of the the CLEF Technology Assisted Review (TAR) tasks [104]. The first is Reliability [57], a loss measure (i.e., where smaller values are better) that has two components: $loss_r$ (Equation 2.4) and $loss_e$ (Equation 2.5),

$$loss_r = 1 - (\text{recall})^2 \tag{2.4}$$

$$loss_e = (n/(R+100) * 100/N)^2 \tag{2.5}$$

where n is the number of documents retrieved, N is the size of the collection, and R is the total number of relevant documents:

$$\text{Reliability} = loss_r + loss_e \tag{2.6}$$

Reliability penalises any losses in both recall ($loss_r$) and in effort ($loss_e$) during the screening phase into a single measure.

The final set-based evaluation measure seen in this thesis is Total Cost [104] (TC). This measure simply multiplies the number of studies needed to screen by the cost associated with screening a single abstract of a study:

$$TC = |\text{retrieved}| \cdot \text{screening cost} \tag{2.7}$$

Total Cost attaches an estimate of a monetary or temporal value associated with the screening phase. This measure is important as it allows for a more direct assessment of the cost associated with screening studies.

All of these measures assume, however, that the relevance assessments are complete. It is easy to imagine that small changes to a query can greatly impact the set of studies that are retrieved. There are many aspects to a query that can be changed, those of which are described in detail further in Chapter 7, that can have a large impact on the set of studies that are retrieved. Imagine the scenario where the authors of a review have finished screening all of the studies for a search. This scenario means that they now have *complete relevance assessments* for the studies retrieved. Afterwards, the same reviewers may wish to update their review: they may use the same query or modify it. In either situation, there will now be studies for which no assessments have been made; either due to more recently published studies now being retrieved, or modifications to the query that caused it to retrieve a different set of studies. In both cases, it would be useful for the reviewers to estimate the number of new relevant studies they may find. In Information Retrieval terminology, these studies which do not have any relevance assessments are referred to as *unjudged*. Several measures account for unjudged documents, namely Rank Bias Precision (RBP) [143], however, these are typically used for rank-based evaluation where a user is modelled by a level of persistence for continuing to examine documents. However, the type of user that screens studies for systematic review literature search is extremely persistent and will examine every retrieved study. To this end, the intuition of unjudged documents in RBP was extended by Scells et al. [201] in the form of two heuristics that can be applied to bound the effect of unjudged documents given an evaluation measure. The first is an *optimistic residual* heuristic that assumes all unjudged studies are relevant, as done by RBP [143]:

$$P(\text{relevant}|d) = 1$$

This heuristic states that unjudged studies are always considered to be relevant. The second is a *maximum likelihood residual* heuristic that provides a stricter bound than the optimistic residual heuristic. As the name states, the probability of a study to be relevant is calculated as the maximum likelihood of relevance given the set of judged studies:

$$P(\text{relevant}|d) = \frac{|\text{relevant}|}{|\text{relevant}| + |\text{non-relevant}|}$$

These heuristics can be combined with an evaluation measure to bound the effectiveness of unjudged studies. Why one may do so is because these heuristics simply change the distribution of relevant studies. Intuitively, the maximum likelihood residual heuristic provides a balance between the existing pessimistic view of unjudged studies (i.e., all unjudged studies are non-relevant), and the optimistic residual heuristic.

Evaluation Measures Summary
In practice, information specialists typically only evaluate their search using recall, and their evaluation is limited to a handful of studies. Limited efforts have been made to evaluate the effectiveness of search strategy development methodologies. The Information Retrieval community has developed several more sophisticated measures that can reveal more about the effectiveness of a search.

The next set of evaluation measures, as seen in this thesis, are all rank-based measures. These measures are concerned with how studies are ordered. These measures assume, however, that the studies are ordered in the first place. In traditional systematic review literature search, all of the studies that are retrieved are assessed. One reason to order studies for systematic review literature search is to begin the systematic review's downstream phases in parallel. Only when compromises are made in the screening phase does it make sense for an ordering of studies to be invoked. The primary compromise in this context is when screening is stopped early. The approach to determine when to stop screening is typically referred to as stopping prediction or cut-off prediction. Both stopping screening and parallelising downstream activities are not commonplace for systematic review creation, although they are beginning to see some adoption [148].

In order to evaluate how good an ordering of studies is, an assumption about ordering is made. That is, studies are ordered by their likelihood of relevance to the query. That is the assumption made for the rank-based evaluation measures used in this thesis. However, there are alternative assumptions for the rank-based evaluation of documents, e.g., assuming that studies are ordered to maximise diversity. These alternative assumptions about orderings are not relevant to the evaluation context in this thesis and therefore, are not discussed. Furthermore, each rank-based evaluation measures attempt to model different user behaviours.

The C/W/L framework [142] will be used to explain how the rank-based evaluation measures discussed below model user behaviour. This framework enables the dissection of evaluation measures into three mathematically related functions that can be used to describe user behaviour. Each function defines a distribution describing how different aspects of user behaviour are modelled. Note that while set-based evaluation measures can be modelled using C/W/L, it is generally not useful as will become apparent shortly. It is easiest to begin with the definition of W.[4] Put simply, W is the amount of attention a user will give to a study at rank position i, such that $\sum_i^\infty W(i) = 1$. Seen as a probability distribution, it is then possible to define C using W: C is the continuation probability at a given rank position i:

$$C(i) = \frac{W(i+1)}{W(i)}$$

The continuation probability at a rank position i is how likely the user will continue to the next study.

Both C and W are closely tied, so much so that one can be derived from the other, as shown by Moffat et al. [142]:

$$W(i) = W(1) \cdot \prod_{j=1}^{i-1} C(j)$$

The final function of this framework is L, which defines the probability that a study at rank i is the last one to be observed:

$$L(i) = \frac{W(i) - W(i+1)}{W(1)}$$

[4]Note that while each function is typically written as, e.g., W_M to signify the corresponding evaluation measure M, it has been omitted where it is not needed for clarity.

For set-based evaluation measures, the user model can be defined in the same way for all measures using L, given the number of studies retrieved, k:

$$L_{\text{set-based measure}}(i) = \begin{cases} 1 & \text{when } i = k \\ 0 & \text{otherwise} \end{cases}$$

The first rank-based measure used in this thesis is Average Precision (AP – written as MAP, Mean Average Precision, when talking about the average AP over multiple queries). It is defined as the sum of the precision for each relevant study in the rank list divided by how many relevant studies in total:

$$AP = \frac{\sum_{i=1}^{k}(\text{rel}(i)/k)}{|\text{relevant}|}$$

Where $\text{rel}(i)$ is the relevance of the study at rank position i (e.g., 1 if relevant, 0 if non-relevant). AP is a very frequently used measure in Information Retrieval tasks. The measure itself does not model a reviewer in the screening phase well but is used to compare other works in this thesis that used AP. Robertson [174] suggests a user model for AP when the user has complete knowledge of where the relevant studies are in the unseen portion of studies after a rank position i. AP can be expressed in C/W/L as [142]:

$$C_{AP}(i) = \begin{cases} \frac{\sum_{j=i+1}^{k}(\text{rel}(j)/j)}{\sum_{j=i}^{k}(\text{rel}(j)/j)} & \text{if } \sum_{j=i+1}^{k}(\text{rel}(j)/j) > 0 \\ 0 & \text{otherwise} \end{cases}$$

This indicates that the user's behaviour is to continue examining studies until all of the relevant studies have been found. Note that this is impossible if a real user were to screen studies for systematic review literature search because the user does not know how many relevant documents there are, but is used to make comparisons to other work in screening prioritisation that uses the measure.

Reciprocal Rank (RR) measures the position of the first relevant document:

$$\text{RR} = \frac{1}{d} \tag{2.8}$$

Where d is the depth in the ranking that the first relevant document appears. This measure is trivial to define as a continuation probability:

$$C_{RR}(i) = \begin{cases} 1 & \text{if } rel(i) = 0 \\ 0 & \text{if } rel(i) = 1 \end{cases}$$

Given this definition of a user model, the user will examine every study until they identify the first relevant study. This measure models a reviewer in the abstract screening phase slightly better than AP as it does not require complete knowledge of where all relevant studies appear in a ranking.

Another related measure is R-precision (Rprec): the precision measured at the rank position equal to the total number of relevant studies:

$$Rprec = \sum_{i}^{|\text{relevant}|} \frac{rel(i)}{|\text{relevant}|}$$

The user model for Rprec can be defined in terms of the following continuation probability:

$$C_{Rprec}(i) = \begin{cases} 1 & \text{if } i < |\text{relevant}| \\ 0 & \text{otherwise} \end{cases}$$

Similarly to AP, Rprec does not accurately model real user behaviour in the context of systematic review literature search, as it relies on the user knowing exactly how many studies are relevant. The final related rank-based evaluation measure is Last Relevant (Last Rel). This is an evaluation measure that was born out of professional search scenarios [104]. It measures the rank position of the last relevant document seen. The user model for this measure is akin to that of AP. Again, it assumes the user knows how many relevant studies there are in the unseen portion of the ranking. It is also included for comparison to other works in this thesis. These four evaluation measures are related to each other as they explicitly rely on the relevance of a study to define their continuation probability. These measures are typically classed as '*adaptive*' user models (sometimes also referred to as 'cascade' user models).

The other rank-based measure discussed in this thesis is normalised discounted cumulative gain (nDCG) [99] — classed as a *static* user model as the continuation probability of this class of rank-based evaluation measures is dependent only on the rank position of studies. nDCG is a normalised form of another measure within a group of gain-based evaluation measures. The idea behind this group of measures is that a user receives gain from examining relevant studies. Therefore, there is a maximum amount of gain available from examining the studies in a ranking. This is referred to as the Cumulative Gain (CG) of a ranking. It is the sum of the degree of relevance for all relevant studies in a ranking. The degree of relevance means the graded score of relevance for a study: typically, relevance assessments are binary (1 for relevant, 0 for non-relevant). In some cases, as will be seen in the coming chapters, it is useful to have multiple grades for higher degrees of relevance (e.g., a study that is screened for inclusion but after reading the full text was deemed not suitable for inclusion in the final review could be deemed less relevant than a study that was included in the final review) – although not necessarily required to compute nDCG.

One of the assumptions made earlier about rankings is that the higher ranked a study is, the more relevant it should be. Therefore, the lower rank a study appears, the less attention should be placed on it. Järvelin and Kekäläinen [99] propose to model this behaviour by discounting the gain accumulated from each study by dividing the relevance label for the study by the log of the rank position of the study:

$$DCG = \sum_{i}^{k} \frac{rel(i)}{\log_2(i+1)}$$

However, the problem with DGC is that when comparing across many searches, the total number of retrieved studies, k changes. To counteract this issue, an *ideal* ordering of studies, ordered by relevance (i.e., using the relevance assessments of studies to induce a ranking) is used to normalised DCG:

$$nDCG = \frac{DCG}{IDCG}$$

Where *IDCG* is the ideal ordering of studies. Moffat et al. [142] define the user model of DCG using the following continuation probability:

$$C_{DCG}(i) = \frac{\log_2(i+1)}{\log_2(i+2)}$$

Much like the original intention when DCG was first proposed, this continuation probability models user behaviour by smoothly progressively reducing the likelihood that the next document will be viewed. Note that this definition is only valid if no cut-off is specified; i.e., all studies are still screened. A cut-off can be applied at given rank positions for nDCG, but it is not used in this thesis.

Evaluation measures provide a way to assess the effectiveness of an Information Retrieval system given a search scenario. They typically have foundations in specific user behaviour and attempt to approximate this behaviour through a user model. Most evaluation measures are targeted directly at measuring how effective a search system is through the studies that are retrieved, but other times they focus on other aspects of search like cost and effort. User behaviour is unpredictable and deeply misunderstood. However, there have been great efforts to study user behaviour and better model it, for example, using information foraging theory [161] or economic models [15]. One common belief about users is that they use Information Retrieval systems to satisfy an *information need* [246]. An information need can be instantiated as the requirement to satisfy a task that depends on identifying relevant information. In the systematic review literature search, the information need is the requirement to identify relevant studies that potentially satisfy the research question posed by the review protocol. When a system is evaluated using different measures and targeting different information needs, this is referred to as *batch evaluation*. One major part of Information Retrieval evaluation that enables batch evaluation of systems in the manner that has been described is the use of *test collections*.

Test Collections for Evaluating Systematic Review Literature Search

A test collection is a corpus of documents, topics, and relevance assessments. While documents and relevance assessments should be familiar terminology from the previous section, the idea of a topic is new. A topic can be seen as the encapsulation of an information need through a structured description. The parallel that could be made within the context of systematic review literature search is the protocol of the review. In fact, in this thesis, systematic review protocols are typically used as topics in test collections. In addition to a description of the information need, each topic contains relevance assessments for a portion of the documents in the collection. It is typically too expensive to collect relevance assessments for all documents in a collection due to the size of the collection and number of topics. In typical test collections, many searches or systems contribute to a pool of documents that will be assessed for a topic [219]. In the test collections used in this thesis, documents that are retrieved by a search strategy are used as the assessments.

There are only a handful of test collections for evaluating systematic review literature search, and each of them is used for different experimental use-cases within this thesis. A description of each collection follows. Other test collections have also arisen; however, they were developed for addressing problems in other phases in the systematic review creation pipeline, e.g., results preparation. They

have been included in the list for completeness. The test collections used in this thesis are denoted in the list below with a ⋆ symbol.

CLEF TAR 2017 [104] ⋆ 50 Cochrane DTA reviews as topics. Each topic contains study title, protocol, relevance assessments (binary relevance — study was included or excluded from the review), Boolean query, and all of the studies retrieved by the Boolean query in the PubMed database.

CLEF TAR 2018 [107] ⋆ Extends upon the topics from the CLEF TAR 2017 collection. Thirty new topics (all still DTA reviews), eight topics removed from the 2017 collection (as they were deemed 'not reliable for training or testing purposes'), 72 in total. Each topic contains the same attributes as in 2017.

CLEF TAR 2019 [105] Extends upon the topics from the CLEF TAR 2018 collection. Twenty intervention review topics are also added. Each topic contains the same attributes as in the previous two years.

Scells et al. [205] ⋆ 125 Cochrane systematic reviews of intervention as topics. Each topic contains the Boolean query used to search for the original systematic review, a translated Elasticsearch query, PICO annotations for each query, and graded relevance assessments: excluded and not retrieved (*l1*), included and not retrieved (*l2*), excluded and retrieved (*l3*), and included and retrieved (*l4*).

Alharbi and Stevenson [6] 25 Cochrane systematic review of interventions as topics. This collection also includes the updates to each systematic review after it has been published. It is focused on updating the review after it has been published (a phase following the dissemination of systematic review in the systematic review creation pipeline). This collection is not appropriate for the query automation methods in this thesis.

Norman et al. [146] 63 Cochrane DTA reviews as topics. This collection is focused on the study synthesis and results preparation phase, containing 5,848 test results for 1,354 diagnostic tests from 1,738 diagnostic studies. This collection is not appropriate for query the automation methods in this thesis.

Summary of Test Collections used for Systematic Review Literature Retrieval
Information Retrieval test collections for medical systematic review literature search are growing in number and size of topics. However, a more diverse range of test collections is needed to compare systematic review automation systems better.

2.3.2 Retrieval Techniques used in this Thesis

Searching for literature using Boolean queries is the primary means to obtain studies to screen for inclusion in systematic reviews. This section provides a background on how Information Retrieval systems perform the task of retrieval given a query, and how documents may be ranked to enable users to find the information they are looking for more efficiently. Within the context of systematic reviews, this translates to how a Boolean query retrieves abstracts of studies, and how it could rank studies, given the query. Note the use of *could* — modern Information Retrieval systems are highly effective at ranking documents using the query provided by a user; yet existing Information Retrieval systems for systematic review literature search still do not rank studies given the Boolean query provided. What follows is a background, firstly on the *Boolean retrieval model* — how documents are retrieved in this thesis; and secondly, on *ranking functions* — how Information Retrieval systems rank documents, with a focus on functions that operate within the Boolean retrieval model.

The Boolean Retrieval Model

Searching for study abstracts for systematic review literature search traditionally involves the use of highly complex Boolean queries. These queries are the main component of a search strategy. To enable users to search for studies in early Information Retrieval systems, the Boolean retrieval model was developed [216]. The Boolean retrieval model enables the retrieval of documents through set-based operations that satisfy equivalent Boolean logic expressions. For example, as visualised in Figure 2.3, the union (conjunction) of documents is achieved though the `AND` operator, the intersection (disjunction) of documents is achieved though the `OR` operator, and the compliment (negation) of documents is achieved through the `NOT` operator. These operations form the basis for the Boolean retrieval model.

Retrieval of documents using the Boolean retrieval model in practice requires knowledge about which documents contain each of the terms in the query. Early practitioners of Information Retrieval realised that books usually have this functionality in the form of an *index*: a list of significant words or phrases the author believes the reader may be interested in, with adjacent page numbers that refer to where in the book each word or phrase appears. Using this concept, Information Retrieval systems were built that used an index of all of the terms from all of the documents, where terms point to a list of documents they appear. What has been described here is referred to as an *inverted index*; and is used as the foundation for most modern Information Retrieval systems, regardless of the retrieval model used [26]. Many retrieval models have arisen that better exploit the inverted index to improve retrieval effectiveness than the Boolean retrieval model. However, newer retrieval models compromise over the Boolean retrieval model to address limitations about it, namely the ability to effectively rank documents by foregoing the ability to represent a complex information need.

For the vast majority of Information Retrieval systems, Boolean queries add a layer of complexity and tedium that stands in the way of efficiently and effectively finding information. Web search today is a ubiquitous tool that enables anyone with an internet connection to find information about almost anything. The reason for this is the low barrier to entry: users of these systems may type anything they

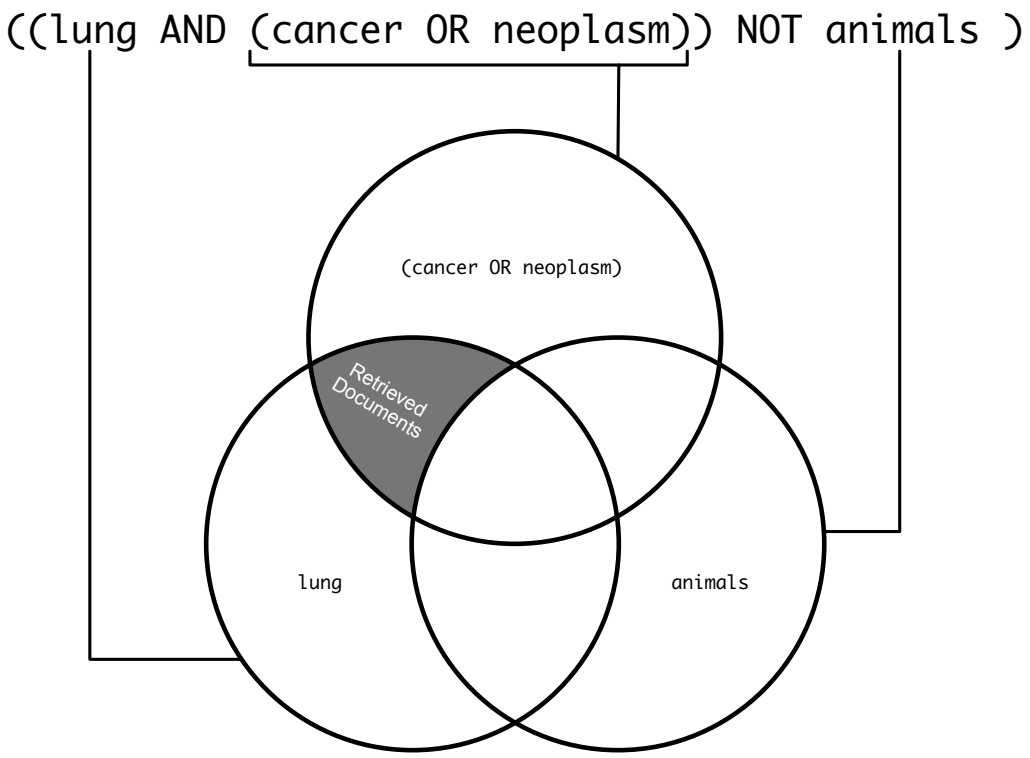


Figure 2.3: A high-level overview of how the Boolean retrieval model works as a translation of a textual Boolean query into a Venn-diagram, showing how the sets of documents each of the Boolean clauses matches overlap to determine which set of documents to retrieve and return to the user. Note that this example is an oversimplification of a typical Boolean query used for systematic review search.

like into the search box. This flexibility is opposed to the rigid and systematic way queries must be constructed for systematic review literature search.

Despite these limitations, Boolean queries are the de facto standard for systematic review literature search. They permit total control over the search system, allowing experts to effectively express complex information needs while also controlling which documents are retrieved by the system. This control also affords researchers other benefits, for example, allowing searches to be reproducible. Reproducibility is important for several reasons, including re-running a search to determine if new studies have been published (which may trigger an update to a systematic review), or if the outcome of the review is in doubt (where a third-party can essentially replicate the entire systematic review creation process).

Boolean Retrieval and Systematic Review Literature Search
Boolean retrieval permits information specialists complete control over the target search system. Queries developed for systematic review literature search are highly complex, however contemporary systems, which are much simpler to use, cannot capture this complexity. New methods to improve systematic review literature search must include or extend Boolean queries to see any practical adoption and use by information specialists.

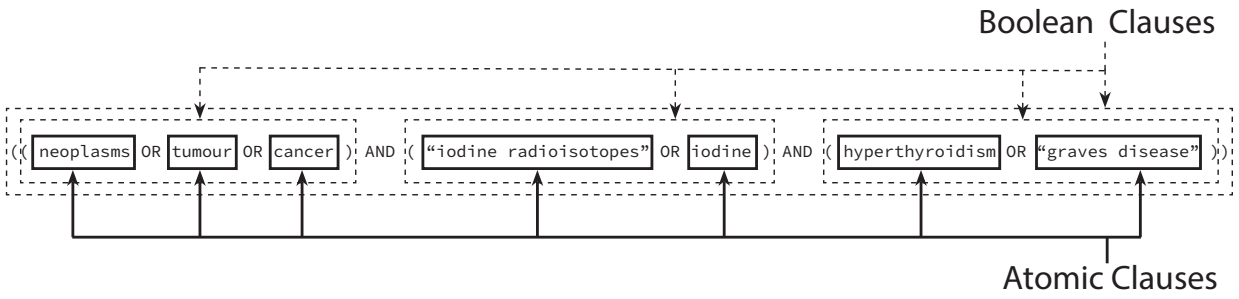


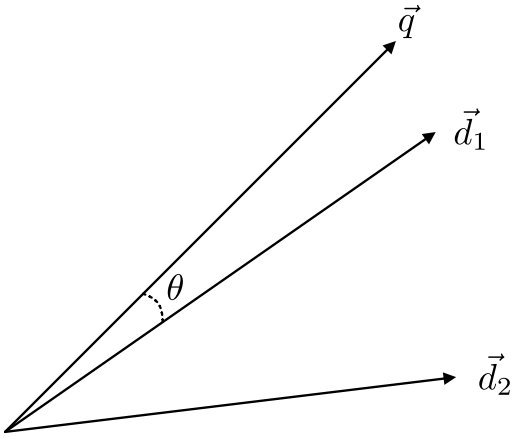
Figure 2.4: Types of clauses in a Boolean query. Dashed lines surround Boolean clauses, solid lines surround atomic clauses.

Ranked Retrieval

Several approaches to ranking documents retrieved by Boolean queries were proposed in the 1980s and 1990s outside of the context of systematic review creation. Most of these approaches rely on users explicitly weighting terms in the query [183], probabilistic retrieval using fuzzy set theory [28, 169] and term dependencies [65]. A drawback of these methods is their heavy reliance on the users to impose a ranking over retrieved documents (e.g., the requirement that users must specify individual term weightings). Users are often unable to provide such weights, or weighting terms creates an additional hindrance in using the retrieval system.

A ranking function for Boolean queries which relies solely on the Boolean query structure, without further user intervention, is Coordination Level Matching (CLM) [126]. The intuition behind CLM is that nested sub-clauses of a Boolean query could be considered separate, but related queries, and therefore documents that appear in multiple clauses should be ranked higher. For example, a common way information specialists to formulate Boolean queries for systematic review literature search is to break a search down into three of four categories based on the Population, Intervention, Controls, Outcomes (PICO) framework [44]. Query terms from each category become a clause in the Boolean query, grouped by a single `AND` operator [44]. Formally, using CLM, the score of a document d is the number of Boolean clauses of the query Q that are satisfied. A clause can be considered a single atomic keyword or a group of several keywords or other nested groupings by a single Boolean operator (Boolean clause). Figure 2.4 visualises the differences between atomic clauses and Boolean clauses. One of the most considerable downsides to using CLM is that it results in many ties when ranking documents. Ties are certainly not desirable, especially for searches that retrieve many thousands of documents. We build upon this in Chapter 12.

One of the earliest (and most well known) systems of retrieval that did not use Boolean query syntax is the vector space model [186] (VSM), originating in the 1970s. The approach behind this ranking function is visualised in Figure 2.6. Both documents and queries are represented in a high-dimensional vector space. Query and document vectors ($\vec{q}$ and $\vec{d}_i$) are of length V: the number of unique terms in all indexed documents. Elements of the vector are represented as a binary number, indicating a term's presence or absence within a document or query. Documents are ranked in terms of how close their vector is to a query, given an appropriate similarity function. One common similarity function is cosine



$q =$ lung cancer

$d_1 =$ Lung cancer has been transformed from a rare disease into a global problem and public health issue. The etiologic factors of lung cancer become more complex along with industrialization, urbanization, ...

$d_2 =$ Venous thromboembolism (VTE) includes pulmonary embolism (PE) and deep vein thrombosis. PE is the third most common cause of cardiovascular death worldwide after stroke and heart attack. ...

$\vec{q} =< t_{1,q}, t_{2,q}, ..., t_{n,q} >$

$\vec{d_i} =< t_{1,d_i}, t_{2,d_i}, ..., t_{n,d_i} >$

Figure 2.6: The vector space model and how the relevance of a document is computed as the cosine similarity between the document and query. In this figure, the angle of distance between $\vec{d}_1$ and $\vec{q}$ is smaller than between $\vec{d}_2$ and $\vec{q}$. In fact, d_2 could be considered irrelevant to the query as the distance is so large.

similarity. Therefore to score a document given a query, the similarity can be computed as follows:

$$\text{score}(d_i, q) = \cos\theta(\vec{d_i}, \vec{q}) = \frac{\sum_{j=1}^{V} d_{i_j} \cdot q_j}{\sqrt{\sum_{j=1}^{V} {d_{i_j}}^2 \cdot \sum_{j=1}^{V} {q_j}^2}}$$

A ranking of documents can then be induced by ordering them using this score. One criticism that could be made of VSM is that it ranks documents by minimising the distance between a query and a document, rather than ranking documents by how relevant they are to the query.

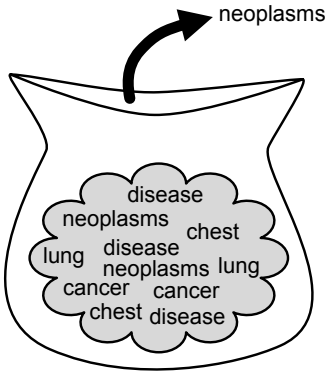


Figure 2.5: Bag of Words

The ease with which queries formulated using natural language has over queries formulated using Boolean clauses, has been empirically found to be highly beneficial to user experience [75]. It led to further research into more effective ways to rank natural language queries. One of the most prominent, and today possibly most widespread, classes of ranking functions are *bag of words* (BOW) models. BOW rankers treat documents and queries as sets of unordered terms. [5] This idea is illustrated in Figure 2.5. These BOW models still typically use Boolean retrieval to match terms to documents: morphologically, a BOW query is equivalent to a Boolean query were each term is separated by a `OR` operator. To rank documents given a query under the BOW model, each term in the query is considered independent, and a document is scored with a function, s, one term at a time. The final score for a document is typically the sum of the scores for each term t in the query:

$$\text{score}(d_i, q) = \sum_{t \in q} s(d_i, t)$$

[5] Originally, these models used single terms (1-grams), but have since evolved to include phrases (n-grams) to model term dependencies, e.g., by including phrases to more effectively score documents.

Therefore a ranking can be produced by ordering documents by their score. The most fundamental term scoring function could be considered to be *term frequency* (TF). TF simply counts the number of times a term appears in a document:

$$TF(d_i,t) = |t \in d_i|$$

The assumption of scoring documents using TF is that a document is more relevant to a query the more terms in the query occur in the document. The most considerable downside to TF is that documents are only ranked by the importance of a term in a document. However, some terms could be considered more important to the query than others. *Inverse document frequency* attempts to model this by weighting a term by a function of the frequency it appears in the collection:

$$IDF(t) = \log \frac{N}{n_t}$$

N is the total number of documents in the collection, and n_t is the number of documents the term appears. IDF weights terms higher the more 'rare' or infrequent they are in the collection. Both TF and IDF combine to form one of the most common BOW ranking models. Indeed most BOW models use statistics about term frequency and collection term frequency.

These ideas from BOW models eventually grew into the idea of *probabilistic* ranking functions, based on the probability ranking principle [175, 260] (PRP). One implementation of the PRP is the binary independence model [254] (BIM). The BIM seeks to rank documents in decreasing order of the probability of relevance to a query. It does this by considering documents and queries as binary vectors of term occurrence — the same representation as to the basic VSM.

One of the most widespread ranking algorithms used in modern search systems is Okapi BM25 [176]. BM25 extends the BIM with two term saturation and document length normalisation components:

$$BM25(d_i,q) = \log \frac{N - n_t + 0.5}{n_t + 0.5} \cdot \frac{TF(d_i,q) \cdot (k+1)}{TF(d_i,q) \cdot k \cdot (1 - b + b \cdot \frac{|D|}{|D_{avg}|})} \tag{2.9}$$

The k and b parameters in Equation 2.9 control term saturation and document length normalisation, respectively. Term saturation controls how much the term frequency component should contribute to a score: the higher k the more contribution given. Document length normalisation controls how much the length of documents should contribute to the score: a higher value favours longer documents. Typical values for these parameters are $k = 1.2$ and $b = 0.5$. For most search scenarios where natural language queries are used, BM25 performs well.

Instead of deriving a ranking function for items, another approach is to *learn* a ranking function using features about items intended to be ranked.

Learning to Rank

Learning to Rank (LTR) is a machine learning approach to ranking items given a set of features about those items. A ranking can be induced over the set of items by learning an ordering of items in several ways. In this section, the type of LTR discussed is offline (the model is trained on example

data and tested on unseen data) and feature-based (item d, topic q pairs are represented as a vector $\vec{x} = \Phi(d,q)$, where Φ is a feature extractor). Traditionally, the items being ranked are documents, and the topic is a query. There also exist other forms of LTR in Information Retrieval, such as Online LTR (OLTR) [95, 152, 154, 256, 259]. These will not be discussed as they are not used in this thesis.

The features for training LTR models are typically extracted from search systems and are usually statistics about the item and topic; for example, when ranking documents for a query, one feature could be the term frequency of the query terms in the document. Within this thesis, LTR is used extensively, primarily for ranking queries given an information need as opposed to ranking documents as in traditional LTR. While no new LTR methods are proposed in this thesis, the following section aims to provide a high-level overview of the different approaches to different LTR methods such that the experiments in this thesis are understandable and validates the choices for the use of LTR in this thesis.

Liu [124] defines LTR models as a discriminative training machine learning method: i.e., a LTR model can be defined in terms of the input space, the output space, the hypothesis space, and a loss function. To this end, LTR models generally fall within three classes: **pointwise** (e.g., least squares polynomial using the probability ranking principle [81], logistic inference [83], staged logical regression [55], PrRank [63], gaussian ordinal regression [41–43], regression-based subset ranking [62], McRank [123]), **pairwise** (e.g., RankSVM [102], RankBoost [80], RankNet [29], AdaRank [252], LambdaRank [30], LambdaMART [249]), and **listwise** (e.g., ListNet [33], Coordinate Ascent [136] SoftRank [225], ListMLE [251], RankCosine [167]). A description of the input, output, and hypothesis spaces, as well as the loss function for each class of LTR models is provided as follows.

pointwise

input space A feature vector for an item.

output space The degree of relevance for each item.

hypothesis space Functions that predict the relevance degree of an item given a feature vector. Items are ranked by the score assigned by the prediction function.

loss function The corresponding loss function for the scoring algorithm is used (e.g., regression loss, classification loss, ordinal regression loss).

pairwise

input space A pair of items represented by feature vectors.

output space The pairwise preference between two items.

hypothesis space Bi-variate functions h that predict the relative ordering between two items given the feature vectors of both items. For example, given a set of three items $\{d_1, d_2, d_3\}$,

and the application of a hypothesis function,

$$h(d_1, d_2) = \{+1, -1\}$$
$$h(d_1, d_3) = \{-1, +1\}$$
$$h(d_2, d_3) = \{-1, +1\}$$

The ordering of the items can be induced through the pairwise preferences: thus, the final ordering should be d_3, d_1, d_2.

loss function The measurement of the inconsistency between the pairwise preference predicted by the hypothesis function and the ground truth label. The loss function of pairwise LTR models aims to minimise the number of inversions in the ranking.

listwise

input space A set of items represented by feature vectors.

output space A permutation of the input items.

hypothesis space Multi-variate functions that predict a new permutation of items. In practice, this is typically achieved by assigning a score to each item and re-ranking the items in descending order of score.

loss function Two types of loss functions exist for listwise LTR methods that depend on the technique. The first is the direct optimisation of an evaluation measure, e.g., NDCG. The second depends on the ranking method (and thus classified as not dependent on an evaluation measure).

One of the main benefits of LTR is that a well-trained model with discriminative features can rank items to a very high degree of effectiveness. The main downside to LTR, however, is that features may be computationally expensive to compute. In production systems, LTR is often used as a 'second-stage' ranker in a cascade of rankers where items are first ranked using an inexpensive but relatively effective method such as BM25, and then the top-k documents are re-ranked using LTR [245]. Another downside to offline LTR models is that they have difficulty adapting to changes in search behaviour. Online LTR models attempt to address this by continually updating their hypothesis function in response to changes in user behaviour. Finally, all LTR methods rely on labelled data for training. Often, this is either difficult to obtain, costly, or both. Another popular way of combining features about items without the use of labels is called rank fusion.

Rank Fusion

Rank fusion is a popular family of techniques to combine different search systems or ranking functions into a single ranked list. Wu [250] provides a general mathematical framework to base rank fusion methods upon: given a collection of items D, and a group of retrieval systems, i.e., a ranking function, search engine, or any other system that retrieves and ranks items. All retrieval systems execute a query

q on D, and each provide a ranked list of items $r_i =< d_{i_1}, d_{i_2}, ..., d_{i_m} >$, providing a final set of rankings R. A rank fusion method then attempts to combine all ranked lists into a single list. There are two ways to combine these ranked lists into a single list: **score-based** methods and **rank-based** methods. A survey of the most popular score-based and rank-based methods are provided below as several of these methods are used within this thesis.

The aim of score-based rank fusion methods, as noted by Wu [250], is to provide a global score for a document $g(R,d)$ among all of the ranked lists it appears in, where d is the document to be scored, and R is the set of rankings of items. Items are then re-ranked using the global score to produce the final ranking. The two most widely used score-base rank fusion techniques are CombSUM and CombMNZ [79]. The scoring function of CombSUM is as follows:

$$g(R,d) = \sum_{r_i \in R} s(d, r_i)$$

The global score of an item is computed by summing the item's score as it appears in each ranked list. If an item is not retrieved in one of the lists, then a default score (typically 0) is applied. The second method, CombMNZ, is an extension of CombSUM:

$$g(R,d) = |d \in R| \cdot \sum_{r_i \in R} s(d, r_i)$$

The global score of an item is computed again by summing the item's score as it appears in each ranked list, however this time if the item is not retrieved a default score is not applied. Instead, the summed score is multiplied by the number of ranked lists the item appears. Vogt et al. [237] suggest one of the intuitions behind the success of these methods is the *chorus effect*: that 'several retrieval approaches suggest that an item is relevant to a query'. A document is likely to be relevant if several retrieval systems retrieve it and rank it highly. That being said, it is also likely that some retrieval systems are more effective than others. For this reason, other fusion methods have been proposed that assign weights to each ranked list; for example, another form of CombSUM that linearly combines the weight of a ranker with the score of an item from that ranker:

$$g(R,d) = \sum_{r_i \in R} w_i \cdot s(d, r_i)$$

Of course, one may learn the weights to assign to each ranker. One of the simplest and empirically effective methods of weight assignment is performance-related weighting [14]. Here, a ranker's weight is assigned the performance of that ranker over a range of representative training queries, e.g., AP or NDCG. One final aspect of score-based rank fusion methods is that the scores of items are typically normalised within each ranking. Normalisation ensures that the rank fusion is not biased towards some rankers that assign unbounded scores (e.g., BM25, where scores are based on statistics and can be larger than 1) and others that may assign bounded scores (e.g., the vector space model, where scores lie in the range [0,1] due to the use of cosine similarity). Typically, min-max normalisation is used to ensure the scores of all items between rankers are within the same scale.

The aim of rank-based rank fusion methods is to re-rank items according to each item's rank positions in a ranking (thus ignoring the score). Although these methods ignore the scores of items in

each ranked list, a global score for an item is still computed to re-rank the items. There are several methods based on election voting algorithms, for example, Borda count [14]:

$$g(R,d) = \sum_{r_i \in R} \frac{|r_i| - \text{rank}(r_i,d) + 1}{|r_i|}$$

The score of an item for a ranked list is the number of items in the list ranked lower than it. $\text{rank}(r_i,d)$ is the rank position of the item in a ranking. A rank-based method should be used if the quality of scores assigned to documents is low, or if documents are not assigned scores. In this thesis, only score-based rank fusion methods are used, as they typically achieve a better ranking.

Rank fusion is used in this thesis in a novel way for ranking the documents retrieved by a Boolean query. Different clauses in a Boolean query can be considered independent queries, and thus documents retrieved by different atomic clauses can be fused in several different ways.

2.4 Query Automation Methods

With an understanding of the phases of systematic review creation, the current methods and tools used to assist in the automation of these phases (as well as research gaps), and the key Information Retrieval concepts that enable the evaluation and development of new methods and tools, this section provides a review of the existing query automation methods that exist for other domains. This review will further motivate and demonstrate the need to focus on query automation. Query automation can be categorised into three main classes: query formulation, query refinement, and query exploitation. The first, query formulation, is concerned with constructing queries, possibly given none, or some, input data (e.g., document relevance judgements — seed studies). In this thesis, the sole focus on query formulation is the automatic construction of queries. The second, query refinement is concerned with the automatic narrowing, broadening, and rewriting of a query for the explicit purpose of improving precision or recall. Finally, the third class of query automation is query exploitation, which is concerned with the enhancement or measurement of queries to improve search effectiveness. The distinction between these classes of query automation will be made more evident within the following three subsections.

Through a review of the literature on each of the query automation categories, the gaps in the research will be identified, focusing on gaps in the systematic review domain. This section will also identify where this thesis fits within other research about systematic review literature search and search for literature in other domains.

2.4.1 Query Formulation

Formulation is the development of a query from scratch using some data (or even none at all). Encoding information specialists' knowledge and intuitions into computational models would allow for the automatic construction of Boolean queries suitable for systematic review literature search. Indeed, this is the path taken for Part 1, where two query formulation methodologies are computationalisied

by encoding the steps information specialists would take to develop queries. There is relatively little research in this area of query automation.

Although, automatic formulation of queries is not uncommon in specialised domains, for example, the generation of SQL queries from natural language statements [10, 159, 164], Boolean query generation for legal search [112], and deriving clinical queries from patient narratives [113]. The Boolean query generation for legal search method from Kim et al. [112] is one of the only methods that tackles query formulation for Boolean queries. In this method, a Binary decision tree classifies (pseudo-)relevant documents using terms from those documents. Each terminating leaf node in the decision tree represents a positive or negative response to a set of documents. Boolean queries are derived by following the path from the root node in the tree to each positive terminating leaf node. The method then uses an LTR model to identify the most effective automatically formulated query. This method has not been demonstrated to be effective in the systematic review literature search domain.

Query Formulation Gaps
There have been almost no works investigating the automatic formulation of Boolean queries for systematic review literature search. There is a clear gap that can be filled with new methods that computationalise existing manual methods of query formulation, that are already used to develop Boolean queries for systematic review literature search. This thesis contributes automatic query formulation methods based on these manual methods and rigorously compares the two.

2.4.2 Query Refinement

Refinement is the narrowing (reduction), broadening (expansion), and rewriting of a query to improve the retrieval effectiveness of a query. What follows is a survey of automatic query expansion, query reduction, and query rewriting methods; both in domains outside systematic review literature search and for it (where applicable). The lack of methods specifically addressing systematic review literature search is what Part 2 seeks to address.

Query Reduction

Several studies have arisen that investigated query reduction, both for web search [18, 115] and professional search [125]. Kumaran and Carvalho [115], and Balasubramanian et al. [18] explored automatic query reduction methods for ad-hoc and web queries. In doing so, they faced issues in terms of the feasibility of exploring the full space of query reductions. To address this, they devised several heuristics to limit the search space. They also explored methods that can automatically select reduced queries, given the original queries, to increase search effectiveness. The three most common methods for BOW query reduction are proportional inverse document frequency (IDF-r) [113], Kullback-Leibler divergence for informativeness (KLI) [228], and parsimonious language models (PLM) [93]. Each of these methods expects a set of terms T and produces a subset of ordered pairs $T' = \{\langle t, s\rangle | t \in T\}$

where s is the score assigned to a given term t. The reduction is achieved by first ordering each pair in T' by the score and then selecting the top-k terms by applying a score cut-off. Locke et al. [125] investigated these three methods and appropriate cut-off values for legal search and found that these methods were inferior to expertly developed Boolean queries. At the time of writing, there do not exist any similar query reduction methods for Boolean queries.

Query Expansion

Voorhees [238] provided one of the first effective query expansion methods for Web search, and the idea of query expansion for Web search has since become a popular area of research [21, 22, 40] with the rise in the ubiquity of such systems. Somewhat closer to the focus of this thesis, query expansion methods in Web search have also been applied to assist consumers in finding more accurate health information [101] and in the clinical space to assist clinicians in finding evidence to support their medical practice [17]. Also related is a study by Verberne et al. [236], who investigated term scoring methods for query expansion in several professional search domains.

Carpineto and Romano [36] provides a survey of automatic query expansion methods. In it, they offer a taxonomy of query expansion methods. This taxonomy has been reproduced here in Figure 2.7. Perhaps the three query expansion methods used most extensively are stemming, thesauruses, and model-based (embeddings). Stemming is the process of deriving a 'root' stem from a word though morphological normalisation. One of the most common stemming algorithms is Porter's algorithm [165]. Thesaurus-based automatic query expansion methods use large lexicon databases to identify synonyms though lexical semantic relationships. Perhaps the most prolific lexicon database is WordNet [138]. Thesaurus-based approaches have also seen use in domain-specific applications, e.g., SNOMED-CT in the clinical domain [262]. More contemporary automatic query expansion methods rely on model-based approaches, specifically, word embeddings such as word2vec [137]. This family of embedding methods learn a vector representation from natural language, e.g., word2vec uses word co-occurrences. The use of algebraic operations on vectors allow one to express semantic relationships between words. Embedding approaches have had much success, and have also been applied to more specific domains, outside of Web search, e.g., the medical domain [262].

While query expansion has seen numerous methods applied to Web search, there have been relatively few studies that have explored query expansion in more specialised domains. Furthermore, those studies that focus on more domain-specific areas of search are not typically concerned with Boolean queries. Few methods have considered query expansion for Boolean queries, and almost none for systematic review literature search.

Query Rewriting

Query rewriting is different from query reduction and query expansion as it could be considered the application of both methods at once. For example, it is commonly known that the same information needs can be expressed with many queries — it is uncommon for there to be only a single query that can

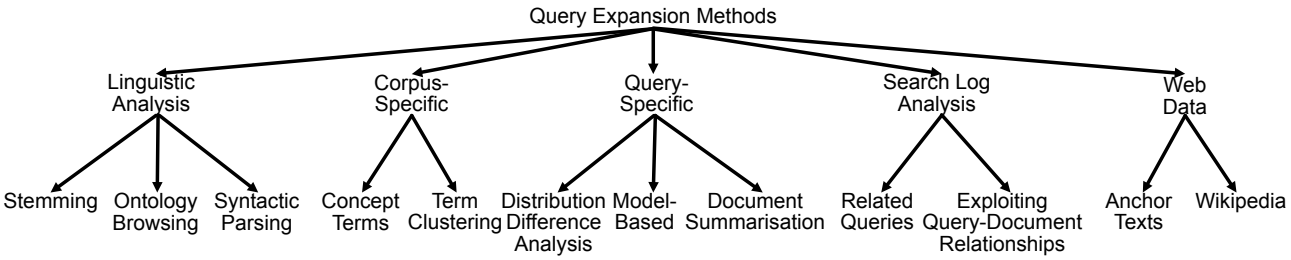


Figure 2.7: Taxonomy of query expansion methods as identified by Carineto and Romano [36].

express an information need for most search tasks. Therefore, query rewriting is the task of identifying a variation of a query that improves retrieval effectiveness. There are very few studies that have investigated query rewriting. For example, work by Sheldon et al. [208] where variations of a query are ranked using an LTR model that can directly optimise evaluation measures. Another approach has been to apply neural translation language models, commonly used to translate words and phrases from one language to another, to the translation of terms in a query to terms in a document [261].

Query Refinement Gaps
There are only a handful of studies investigating automatic methods of query refinement for Boolean queries in systematic review literature search. There is a clear gap in the research for new methods that can automatically apply the three query refinement methods discussed above to Boolean queries to improve the systematic review literature search effectiveness. This thesis contributes a novel method of query refinement and studies how different refinement methods can affect the retrieval of studies for systematic reviews.

2.4.3 Query Exploitation

The exploitation of queries is significantly broader than query formulation and refinement. In this thesis, query exploitation covers any aspect of searching that uses or enhances a query somehow. The three most common methods of query exploitation are *relevance feedback*, *query extensions*, and *query performance prediction* (QPP). Each query exploitation method is described in detail below in order to frame the exploitation methods in Part 3. While the methods discussed below are only broadly related to the methods presented in this thesis, they provide good coverage of the range of activities involving query exploitation.

Relevance Feedback

The first query exploitation method to be discussed is relevance feedback. Relevance feedback is the exploitation of a query representation in order to shift the intention of the query towards more relevant documents. Note that although this could be achieved with query expansion/reduction/rewriting (and therefore this could be considered query refinement), these methods only seek or broaden or narrow a search. The difference between broadening/narrowing and changing query intent distinguishes

relevance feedback as a query exploitation method from query refinement (as it enhances the query in some way as opposed to broadening/narrowing). The most common and prolific use for relevance feedback is term (re)-weighting. The Rocchio algorithm is one of the most well-known relevance feedback methods for term weighting [177]. Given a vector representation of a query and set of documents (e.g., the representation given by the vector space model), the query vector is updated such that it maximises the difference between the average vector representing the relevant documents and the average vector representing the non-relevant documents:

$$\vec{q}' = \underbrace{\alpha \cdot \vec{q}}_{\text{Original query.}} + \underbrace{(\beta \cdot \frac{1}{|\text{relevant}|} \cdot \sum_{\vec{d}_i \in \text{relevant}} \vec{d}_i)}_{\text{Positive relevance feedback.}} - \underbrace{(\gamma \cdot \frac{1}{|\text{non-relevant}|} \cdot \sum_{\vec{d}_i \in \text{non-relevant}} \vec{d}_i)}_{\text{Negative relevance feedback.}}$$

The three parameters α, β, and γ are used to scale the vectors to place more weight on the original query, positive documents, or negative documents. If no relevant documents are known about a priori, pseudo-relevant documents may be used instead (pseudo-relevance feedback). Under this scheme, the top-k retrieved documents are considered relevant and are used for positive relevance feedback. The bottom-k retrieved documents are then used as negative relevance feedback. The Rocchio algorithm works exceedingly well for queries that can easily be represented as a word vector. Unfortunately, however, the syntax of Boolean queries do not permit such representation to exist. That has not stopped others from attempting to apply the same intuitions of relevance feedback to Boolean queries. For example, Dillon et al. [72] proposed a method to add negative feedback to Boolean queries by adding `NOT` clauses for terms in a negative prevalence range. They defined prevalence of a term t across the set of retrieved documents as:

$$\text{prevalence}(t) = \text{prevalence}^{+}(t) - \text{prevalence}^{-}(t)$$

Where positive prevalence (prevalence^{+}) and negative prevalence (prevalence^{-}) are defined as the following ratios:

$$\text{prevalence}^{+}(t) = \frac{|t \in \text{relevant}|}{|\text{relevant}|}$$

$$\text{prevalence}^{-}(t) = \frac{|t \in \text{non-relevant}|}{|\text{non-relevant}|}$$

As controlled by some threshold value, terms with high negative prevalence are then `NOT` -ed into the original query as a new clause. Surprisingly, the authors found that this process failed to improve precision, although the experiments were performed on a small collection. One major downside to this algorithm is that the addition of negative clauses is solely dependent on the negative prevalence factor (which can lead to excessively long queries). The other prominent method of relevance feedback for Boolean queries is the disjunctive normal form (DNF) method [182]. This method seeks to improve the one proposed by Dillon et al. in two ways. The first is an equivalent term scoring method to prevalence called relevance weight:

$$\text{relevance weight}(t) = \left(\frac{|t \in \text{relevant}|}{|\text{relevant}|} - \frac{n_t}{N} \right) \cdot \ln \left(\frac{N}{n_t} \right)$$

Where the negative prevalence is replaced by the ratio of the term occurring in the entire collection. The score is also smoothed by the inverse document frequency of the term t. The second proposed improvement is the concept of a retrieval parameter T that specifies the expected number of documents the query will retrieve. This parameter controls the exclusion of terms using an estimation for how many documents the conjunction of a clause to the original query will retrieve (removing the need to explicitly specify a prevalence threshold as in the method proposed by Dillon et al.). Salton et al. [185] empirically evaluated the two relevance feedback methods across four different collections and found that the DNF method achieved 10-80% higher recall than the method by Dillon et al. and that the DNF method achieved 10-80% higher precision than the original queries. In contrast, the method by Dillon et al. was usually 10-50% worse.

These relevant feedback methods seek to modify the query in some way in order to produce a more refined search. Unfortunately, there are several downsides with the methods discussed above. Firstly, the Boolean query representation does not permit term re-weighting using vector-based approaches such as Rocchio. Secondly, methods that attempt to apply the intuitions of relevance feedback to Boolean queries do so by greatly increasing the queries' size, as they add many clauses into the query. This size is undesirable within the context of systematic review literature search, where the interpretability of a query is an important factor in the reproducibility and future updating of a systematic review. It is yet unclear how to apply relevance feedback to Boolean queries while preserving the carefully designed structure seen in a systematic review literature search. This area is not explored in this thesis and is left for future work.

Query Extensions

Query extensions seek to exploit the syntax of queries in some way that improves the effectiveness of the search. While the literature for query extensions on BOW queries is limited, there has been a vast amount of effort to extend the Boolean retrieval model in some way; namely to induce a ranking of documents.

Perhaps the most well-known extended Boolean model (EBM) is coordination level matching (CLM) [126], which has been discussed earlier in Section 2.3.2. Rankings produced by CLM typically perform poorly. This poor performance is because the amount of information about the query being exploited to produce a document ranking is low. CLM has been noted to be more effective when weighting occurrences of documents by, for example, IDF or TF-IDF [64]. Which weighting scheme to use for CLM is then unclear, and some documents may be ranked higher than others using different weighting schemes. Moreover, when computing scores, CLM does not account for the different Boolean operators present in the query, i.e., scores are summed in the same manner irrespective of the operator used, e.g., `AND`, `OR`.

Another well-known EBM is the fuzzy set model [168]. Under this retrieval model, the syntax of Boolean queries is exploited through fuzzy set theory to induce a ranking over the documents retrieved. Fuzzy set theory extends classical set theory primarily through the definition of a membership function. The membership function assigns values to members (documents) of a set where a value of 1 indicates

full membership, 0 indicates no membership, and any number in between indicates partial membership. A ranking can be derived by ordering documents in a fuzzy set by their decreasing membership grade order. However, what has been described above is how a fuzzy set of documents is retrieved for a single term. Much like how the Boolean retrieval model uses logical operators to manipulate sets of documents, equivalent operations on fuzzy sets can also be defined. Bordogna and Pasi [23] demonstrate how to achieve this. One limitation of the fuzzy model is that many ties in the ranking will occur if the membership of most documents retrieved is 1. Furthermore, although this model allows for greater flexibility in the representation of user needs than the Boolean model, the complexity involved may have prevented widespread adoption compared to the Boolean model's simplicity and understandability.

One EBM model that seeks to discriminate relevant documents better while maintaining the understandability of Boolean retrieval is p-norm [184]. This model attempts to apply the intuitions from the VSM, namely that documents should be ranked according to their relative 'distance' from a query. At a high level, the p-norm model suggests that Boolean operators that partially match terms in documents should be accounted for when scored. [6] Under the p-norm model, document scores lie in the range [0,1], where 0 signifies lack of the term in the document and 1 signifies the term's absolute presence in the document.A p parameter specifies the 'strictness' applied to the Boolean operator ($1 \leq p \leq \infty$). When $p = 1$, the model acts in the same way as the definition of the VSM. When $p \to \infty$, the model acts in the same way as the conventional Boolean retrieval model (and the fuzzy-set model). As individual terms may be weighted alongside the strictness of the Boolean operators, it allows users to control the importance of specific clauses, while retaining the structural control over the query. However, one criticism that could be made of this model is that there is an additional cognitive burden associated with the assignment of term weights and p-norms for each clause. Furthermore, this model requires additional query and costly document processing; although efforts have been made to address this efficiency limitation [163]. Finally, although this model is better at discriminating relevance than the Boolean retrieval model, CLM, and the fuzzy set model, there are still more ties in rankings than other BOW approaches, e.g., TF-IDF, BM25, VSM.

It should be noted that other EBM (and EBM-related) approaches exist [27, 156, 232, 244], however, none are as prominent or influential as the three discussed above, and all suffer from additional cognitive effort or low discriminatory power.

Query Performance Prediction

The final category of query exploitation methods is query performance predictors. Unlike relevance feedback and query extensions, query performance predictors (or QPPs) are not concerned with directly improving search effectiveness. Instead, a QPP seeks to infer the effectiveness of the query. This inference can then be used in combination with other methods (e.g., relevance feedback) to gauge the effect these methods have on a query in the absence of relevance judgements. A significant amount of

[6] For example, given two query terms t_1 and t_2 that are connected by a Boolean operator, a document that contains both terms should be scored higher than a document that contains only one of the terms.

attention has been paid to the development of QPP methods. QPP methods can be considered as either pre-retrieval or post-retrieval.

Pre-retrieval predictors use statistics about queries and the collection in order to make a prediction. These statistics include *Query Length* [145] (the number of characters in a query), *Term Length* [92] (the number of non-stopword terms in a query), *Inverse Document Frequency* [66, 91, 257], *Inverse Collection Term Frequency* [116] (the ratio of unique terms in the collection to the term frequency of a term in a document), *Collection Query Similarity* [257] (the collection frequency of each term in the query multiplied by the IDF of the term), *Query Scope* [162] (the ratio of the number of documents that contain at least one of the query terms to the number of documents in the collection, and *Simplified Clarity Score* [92]) (the maximum likelihood of a query language model of each term in a query).

Post-retrieval predictors use the results, such as the retrieval status value and rank of documents to predict the effectiveness of a query. For instance, measures such as *Weighted Expansion Gain* [110] (the quality of retrieved pseudo relevant documents by measuring the likelihood that they will have topic drift), *Weighted Information Gain* [258] (the weighted entropy of the top k ranked documents), *Normalised Query Commitment* [211] (the amount of potential drift in the list of top k documents by measuring the standard deviation of their retrieval scores.), and *Clarity Score* [66] (how much the query language model diverges from the collection language model).

The QPPs presented above all have a major limitation when considering the context of this thesis. That is, all of these QPPs are focused on predicting the performance of BOW queries. As of writing, the only publication that has suggested QPPs for a Boolean query context is that of Kim et al. [112]. In it, they suggest adaptions of existing QPPs for BOW queries. However, the predictive power of these adaptations is unknown, and they likely suffer from the same limitations of existing QPPs. For example, QPPs are often designed such that they aim to correlate with retrieval effectiveness. With this in mind, Hauff et al. [88] found that most QPPs are indeed not directly correlated with retrieval performance and that the predictive power of a QPP is often dependent on the collection. An effective QPP is said to have a strong linear correlation between the retrieval model and the QPP score. Hauff et al. [87] also found that this correlation is often insufficient and does not accurately reflect the performance of a query for a retrieval system, or that the correlation is high only for specific tasks; however, no solutions to this problem have been proposed.

The ability to effectively predict a query's performance is vital for developing automatic methods that automatically formulate and refine queries. Scells et al. [190] investigated existing QPPs for the task of identifying more effective BOW query variations for systematic review literature search. It was found that no QPPs were able to consistently predict more effective queries for this task. Unfortunately, existing query performance predictors do not apply to the query formulation or refinement tasks within the context of this thesis (Boolean queries). Therefore, there is a clear gap for predicting how effective a Boolean query is at the task of systematic review literature search. This thesis contributes a framework for the task of ranking Boolean queries in terms of their effectiveness to refine Boolean queries automatically.

Query Exploitation Gaps
Query formulation, refinement, and exploitation are widely researched and well-understood in many domains, especially Web search. However, research into such topics for systematic review literature search is almost non-existent, providing fertile grounds for an investigation into domain-specific methods. This thesis contributes two query exploitation methods that seek to improve the effectiveness of Boolean queries while preserving the structure and clarity and without imposing any additional cognitive strain on information specialists.

2.4.4 Summary of Query Automation for Systematic Reviews

This section has identified several gaps in the knowledge about Boolean query automation, both in the domain of systematic reviews and other domains. The main gaps in the literature for the three areas of query automation this thesis seeks to fill are:

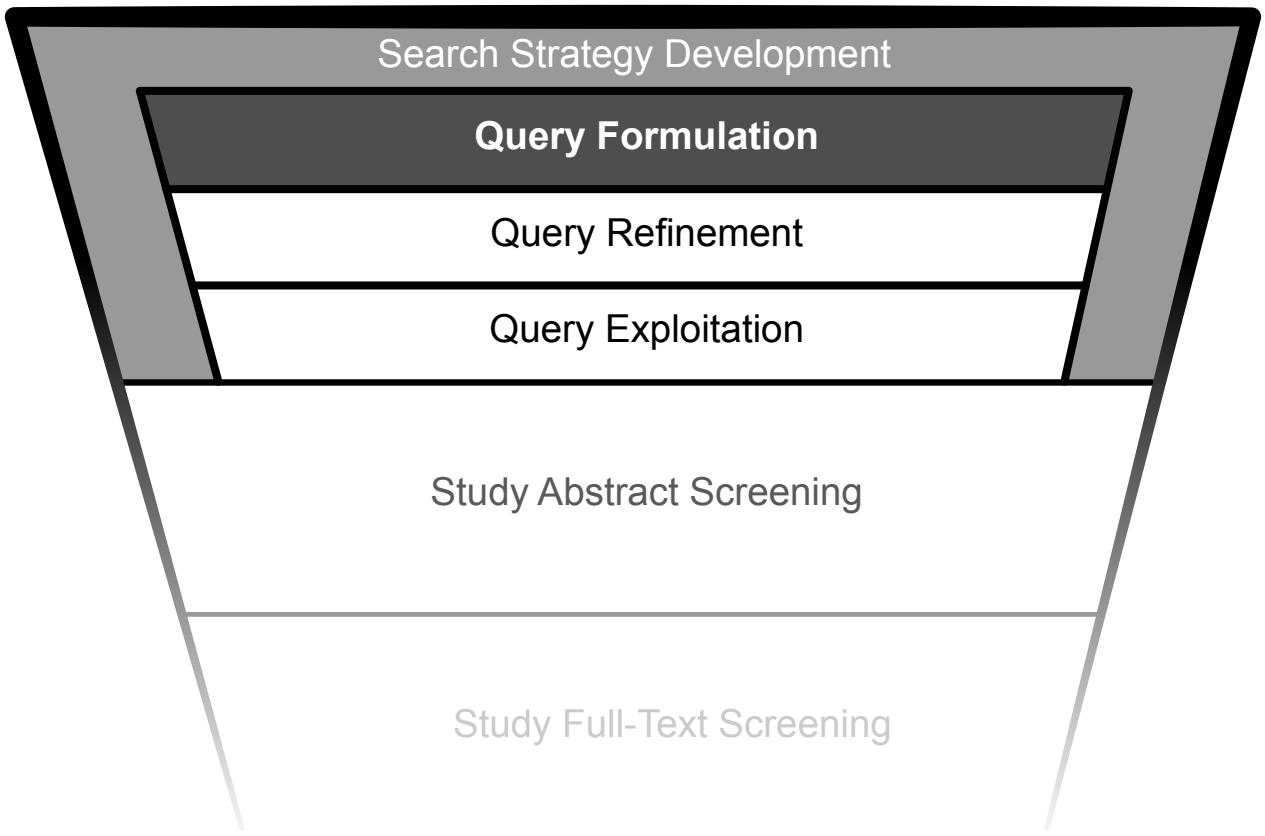
Query Formulation Lack of automatic Boolean query formulation for systematic review literature search. Manual query formulation is time-consuming, error-prone, and requires the expertise of an information specialist. Automatic query formulation is expected to reduce query formulation errors and provide a less biased starting point for further refinement. This thesis addresses this gap by computationalising two existing, manual approaches to Boolean query formulation, and comprehensively comparing them.

Query Refinement Lack of automatic Boolean query refinement for systematic reviews. There is a great amount of subjectivity in how Boolean queries should be improved for systematic review literature search. Automatic query refinement is expected to greatly improve systematic review literature search effectiveness, where queries typically retrieve far more studies than necessary. This thesis addresses this gap by developing a computational framework for the automatic selection of more effective queries, produced by generating variations of existing high-quality Boolean queries used in systematic review literature search.

Query Exploitation Lack of appropriate techniques for exploiting Boolean queries used in systematic review literature search. While there are several exploitation methods for Boolean queries, they all drastically change the syntax or have additional cognitive overhead. More desirable methods that do not impose such constraints are expected to provide immediate benefits to those that develop Boolean queries for systematic review search (as it does not require new knowledge of how to exploit query syntax). This thesis addresses this gap by developing two domain-specific query exploitation techniques that significantly improve systematic review literature search effectiveness.

Part 1

Query Formulation



Is it possible for a computer to effectively formulate Boolean queries for systematic review literature search? How do these queries compare in effectiveness to those developed by information specialists? These questions guide the research contained within Part 1 of this thesis. Specifically, the chapters in this part are guided by the investigation of the first research question of the thesis:

RQ1

Can search strategies be automatically formulated and if so, are they comparable in effectiveness to search strategies manually formulated by information specialists?

Formulating a query for a systematic review is a challenging task and is commonly undertaken by highly trained information specialists. It involves constructing a complex Boolean query in order to find all relevant studies.

Information specialists use domain knowledge, query formulation guidelines (e.g., [38, 44]), experience, and intuition in order to formulate queries [44]. This part of the thesis investigates the two most prominent Boolean query formulation methods for systematic review literature search and computational adaptations of these methods that result in automated Boolean query formulation. Chapters 3 and 4 present the methodological approaches of the conceptual and objective methods, and how they have been computationally adapted to fully automatic query formulation processes, respectively. The computational adaptations of the conceptual and objective methods are then compared with each other to determine which is more effective at automatic query formulation in Chapter 5. Finally, Chapter 6 demonstrates novel tools that have been developed as part of this thesis to support query formulation, guided by usefulness of tools in this domain:

Tool Demonstration: Query Formulation

Examples of query automation tools for systematic review literature search to support information specialists in formulating more effective queries.

One of the tools demonstrated is a product of the computational adaptations presented in the first two chapters of this part. The demonstration of tools is presented as a case study from the perspective of an information specialist.

Chapter 3

Conceptual Query Formulation

This chapter introduces the *conceptual* query formulation methodology used by information specialists and a computational adaptation of this methodology to support the automatic formulation of queries. The conceptual method is the most commonly used approach to develop Boolean queries for systematic review literature search [44]. Under this approach, several high-level concepts are identified, either from seed studies or through initial searches, that represent the research question of the review. Once the information specialists have identified the high-level concepts they will use to develop the search; they use both their expertise and a number of tools to assist them in identifying synonyms and related keywords to their high-level concepts. They add to and refine the query in an iterative process until they feel they are satisfied. This refinement process (visualised in Figure 3.1) is achieved using a number of seed studies to gauge the effectiveness of the search. Commonly, only a handful of seed studies are used in the conceptual query formulation process [44].

3.1 The Conceptual Methodology

Typically, information specialists begin the conceptual query formulation process by analysing a brief statement about the topic of the systematic review, its protocol, and a possible handful of citations to clinical studies (seed studies) that the review's researchers have identified as being relevant to the review that will be conducted [38, 44].

They then identify core high-level *concepts* that emerge from the provided information. In collaboration with the researchers, information specialists then refine these concepts that form the basic structure of a query. Next, a 'proto-query' is developed, and a small subset of the citations retrieved by the proto-query are screened to determine approximately how many studies may be relevant when screened. Finally, the 'proto-query' is enlarged to include synonyms of the previously identified concepts. At this stage, field restrictions (e.g., 'match only on title') may also be applied to each query keyword. The query is then 'translated' to query languages appropriate for use in various medical databases to form a search strategy, e.g., into Ovid or PubMed. Different query languages cater to different advanced operators; e.g., Ovid allows for proximity operators, while PubMed does

not. However, most commonly supported operators in this context are Boolean operators such as AND, OR, NOT, field restrictions (title, abstract, both), and MeSH keywords[1].

Formulating queries, according to the method described above, can take several weeks, if not months [109, 172]. This process adds to the *significant costs* (both in time and money) involved in creating systematic reviews. It takes, in fact, up to two years and a quarter of a million dollars to complete a systematic review [133]. Query formulation in this context is *demanding*, involving lengthy interactions with the search system and the underlying collection to elicit relevant terms that characterise relevant citations. Formulated queries are often *far from optimal*; i.e., they retrieve more citations than necessary (and in fact, they commonly retrieve orders of magnitudes more false positives — irrelevant citations — than true positives — relevant citations), while they may still not guarantee total recall (i.e., it is unclear what the number of false negatives may be). For example, previous studies have investigated search strategies from a representative set of published high-quality systematic reviews and showed that better queries than those originally in the reviews were possible; these were queries that reduced the number of false positives, while not reducing recall (or providing bounded recall losses) [192, 197]. Note that even small increases in precision can have a significant impact on both the total cost of a review and the time required to produce a review [51, 209].

The following section describes a computational framework for automatically formulating Boolean queries for systematic review search, that approximates the conceptual process of information specialists. Within our framework, we present several methodologies for approximating phases of the conceptual query formulation process.

3.2 Automating the Conceptual Method

The conceptual method is the most commonly used approach to develop Boolean queries for systematic review literature search. Under this approach, several high-level concepts are identified, either from seed studies or through initial searches, representing the research question of the review. Often, these concepts are categorised using the PICO question scheme. PICO stands for **P**opulation, **I**nterventions, **C**ontrols, and **O**utcomes. It is a way of framing medical questions, in terms of the information needed to answer them. It is common for the title and research question of a systematic review to be framed using PICO. Once the information specialist has identified the high-level concepts they will use to develop the search; they use both their expertise and possibly several tools to assist them in identifying synonyms and related keywords to their high-level concepts. They add to and refine the query in an iterative process until they feel they are satisfied. This refinement process is achieved using several seed studies to gauge the effectiveness of the search. Commonly, only a handful of seed studies are used in the conceptual query formulation process [44].

The automatic formulation framework comprises four steps (Figure 3.1): query logic composition (1), entity extraction (2), entity expansion (3), keyword mapping (4). These steps approximate the process an information specialist undertakes when formulating queries. In *query logic composition*

[1]The **Me**dical **S**ubject **H**eadings (MeSH) ontology is a hierarchically organised index of biomedical concepts.

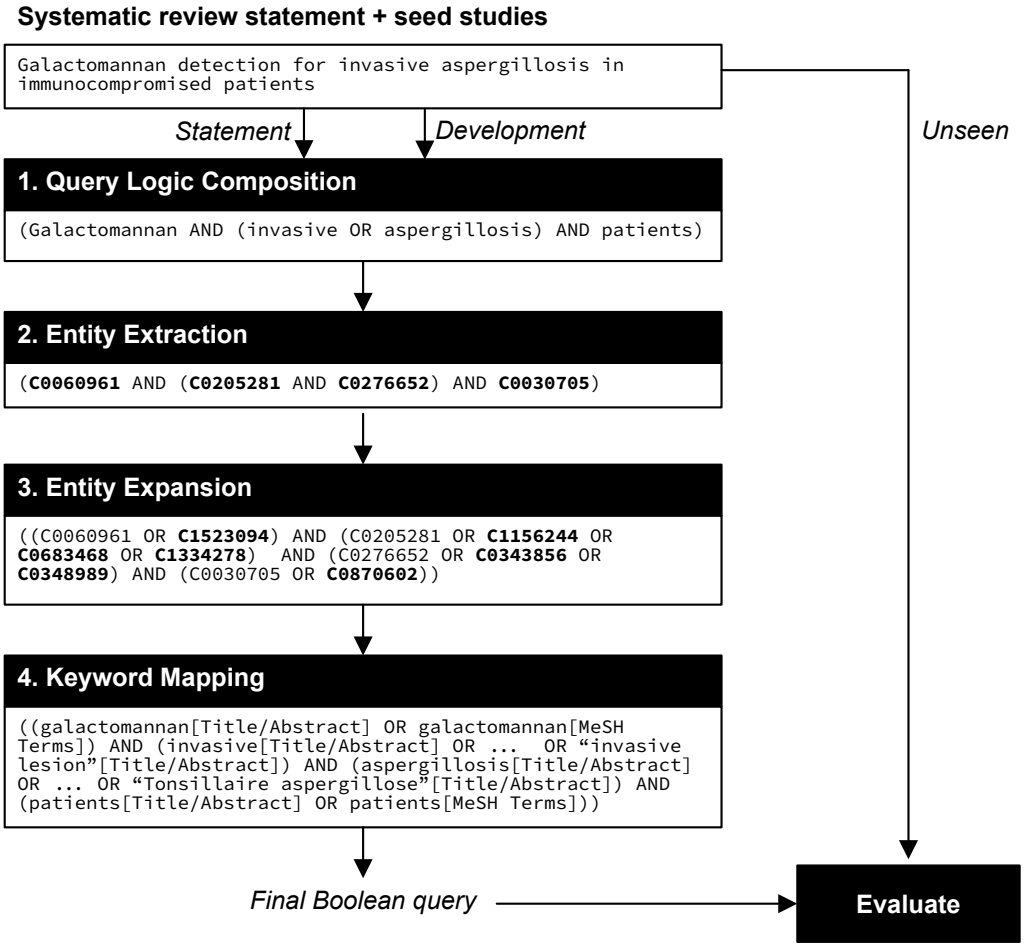


Figure 3.1: Overview of the computational adaptation of the conceptual approach. Three high-level concepts are identified from the systematic review's topic statement: *galactomannan detection*, *invasive aspergillosis*, and *patients*. These are then logically composed into a Boolean syntax and entities extracted (defined as entries in the UMLS). Entities are then (optionally) expanded. Entities are then mapped to keywords and evaluated to determine the retrieval effectiveness.

(step 1), a logical hierarchy of high-level concepts in a query is extracted from a short description of the systematic review (the brief topic statement). In *entity extraction* (step 2), the high-level concepts in the query are extracted and represented with entities from a reference entity repository (i.e., a medical terminology or thesaurus, e.g., UMLS[2]). In *entity expansion* (step 3), a query is broadened by adding related entities, within suitable locations in the query's logical structure. In *keyword mapping* (step 4), entities in the query are mapped to one or more keyword expressions[3]: keywords replace entities in the query's logical structure. The third step is optional, and valid queries can be obtained by applying step 4 immediately after step 2: this may result in narrower queries being formulated.

The conceptual method's computational adaptation is seeded using a single sentence describing the high-level research statement the systematic review aims to address and several seed studies. Seed

[2]The **U**nified **M**edical **L**anguage **S**ystem (UMLS) is an integration of a number of key medical and biomedical terminologies, including MeSH.

[3]Each keyword may be formed by an n-gram or phrase.

studies are split into Development ($\frac{2}{3}$) and Unseen ($\frac{1}{3}$). The Development set is used to perform a term reduction step in query logic composition, removing non-contributing terms from the query. The titles of systematic reviews are used as the input statement as they are typically written in PICO format. This computational approach differs from the manual approach; primarily, it does not perfectly reflect the process an information specialist would use to add keywords to a query as there is no universally agreed-upon approach for this. The input seed studies are identified from the relevance assessments for topics. The original conceptual adaptation proposed a pipeline of processing steps to transform a statement into a Boolean query. This pipeline is illustrated in Figure 3.1.

The proposed automatic Boolean query formulation framework is derived by approximating the processes that an information specialist commonly undertakes when formulating a query. Each of the steps of the framework (Figure 3.1) are described in detail as follows.

Query Logic Composition Once provided with a high-level overview of the topic of a review, the information specialist usually proceeds by deriving the key *high-level concepts* of the review, which in turn inform the logical structure of the final query. An example of a high-level concept from the example query in Figure 3.1 is "Immunocompromised Patients". Ultimately, high-level concepts are used to identify which *keywords* to use in the query. In this step, the information specialist also decides the main logical structure of the query in terms of Boolean operators. Typically, the specialist groups the highest-level concepts with `AND` operators, as all these concepts need to be in relevant citations and related lower-level concepts with `OR` operators (as these keywords form alternative expressions for referring to the high-level concept) [44].

Terms from the seed sentence are composed into a logical structure for a Boolean query. This step approximates the information specialist in identifying the initial high-level concepts and categorising them into the different Boolean clauses that will eventually contain the synonyms for these terms. These word boundaries are then used to specify where terms should be added into Boolean clauses of a query.

In this thesis, we use an embedding approach to cluster similar terms into the same Boolean clauses. We propose two methods to extract terms and phrases from the input statement: the first split the statement into uni-grams, and the second split the statement into phrases using the RAKE algorithm [180]. Keywords are first mapped to UMLS concepts. An embedding for each UMLS concept is then obtained using the model proposed by van der Vegt et al. [234] Keywords are clustered by measuring the cosine similarity of their embedding between other keyword embeddings. A minimum similarity threshold is used to determine if a keyword belongs in an existing cluster or if a new cluster should be created. In our empirical testing, a value of 0.3 was found to provide the best separation of concepts. A keyword is added into the cluster which contains the most similar keyword. If the keyword does not meet the minimum similarity threshold, it is added to a new cluster containing itself.

Seed studies can be used to tune the effectiveness of queries at different stages in the computational, conceptual pipeline. We integrate seed studies into the logical composition step. We

use a portion of the relevance assessments to tune the query by reducing keywords that do not contribute to the search while maximising coverage. This is done by first constructing a set of binary keyword vectors K for each seed study corresponding to each extracted keyword; $\vec{s_k} \in K$. Once the keywords have been clustered as in the query logic composition step, the result is a set containing each set of clustered keyword vectors $C = \{K_1, K_2, \ldots, K_n\}$. The maximum coverage for a new Boolean clause $K_i \in C$ is the logical disjunction of all term vectors corresponding to that clause: $coverage(K_i) = \vec{s_{k_1}} \vee \vec{s_{k_2}} \vee \cdots \vee \vec{s_{k_n}}$. Each keyword in the clause is then tested to determine if it reduces the coverage, or in other words, the removal of the keyword causes a change the in coverage vector. If no change is detected, the keyword is discarded. This process is formalised as follows: For each $K_i \in C$ let

$$K_i' = \{\vec{s_{k_i}} \in K_i \mid coverage(K_i - \vec{s_{k_i}}) \neq coverage(K_i)\}$$

Now let $C' = K_1' \cap K_2' \cap \cdots \cap K_n'$. Each set of keywords $K_i' \in C'$ then becomes a clause in the Boolean query, where each keyword in a K_i' set is joined by an `OR` operator, and an `AND` operator joins each set of keywords in C'.

Entity Extraction Following the logical composition of the review's topic statement into a proto-query, entities are extracted from the query's high-level concepts. To represent entities, we use the Unified Medical Language System (UMLS). [4] We use UMLS, rather than MeSH (which is commonly used as part of systematic reviews queries) as UMLS comprises a more extensive scope of topics than MeSH, and provides many alternative ways to refer to an entity. Note that MeSH is also a subset of the UMLS. The mapping between high-level concepts and UMLS entities (also known as CUIs) is performed through any of the available medical entity extraction algorithms, e.g., MetaMap [11] or QuickUMLS [218], that use natural language processing and term matching between the keywords to express the high-level entity and the terms used to describe entities in UMLS.

Entity Expansion Once the entities that form the query have been extracted, an optional expansion step can be applied to broaden the query with related entities. This step is optional — if not applied, the resulting query may, however, be narrow and thus not suited for recall-oriented searches; but may be sufficient for precision-oriented searches. To identify expansions, entities (UMLS CUIs) are embedded into a high-dimensional vector space (using word embeddings techniques [234]). The cosine similarity between an embedding for each entity in the proto-query and any entity in the embedding space is computed. For each proto-query entity, the k entities with the highest cosine similarity are selected and included in the query, respecting the query's logical structure, and the entities contained in the proto-query. Note that other methods can perform entity expansion: we could consider methods of medical entity similarity surveyed by relevant literature [67, 114, 160] — we leave this for future work.

[4] Although other suitable terminologies may be used (e.g., MeSH, SNOMED CT, etc.).

Keyword Mapping After the mapping and optional expansion of UMLS concepts, the concepts must then be mapped into appropriate keywords (a single UMLS concept may have several aliases, i.e., textual representations – alternate spellings, word orderings, etc. – due to the origin of the concept in different ontologies). The Keyword Mapping step performs this action. There are several work techniques to perform this task, including using the preferred term in UMLS, using all of the aliases for a concept, or using only the most frequently used term (a method proposed by Jimmy et al. [101]). We found experimentally that the preferred term in UMLS of an entity provides us with the highest gains in effectiveness.

3.3 Summary

This approach to automatic query formulation mimics the practice of information specialists formulating Boolean queries for systematic review literature search using the conceptual query formulation methodology. Moreover, these queries require minimal data: one sentence and a handful of seed studies. The motivation behind this research is to reduce the bias and subjectivity aspects of conceptual query formulation. It is envisioned that this research will not replace the information specialist performing query formulation but instead assist them in formulating more effective queries in a shorter period of time. An interactive system, using our query formulation approach, could help the searcher to further expand and refine queries as they see fit, help in training purposes to allow trainees to 'get a feel' for how to search for literature without starting from scratch, or for researchers conducting rapid reviews where time is a major factor, and the accuracy of the review (with respect to finding every possible relevant study for the research question) is not as important as it is in traditional systematic review settings. Overall, more effective query formulation has the potential to reduce the time spent screening and appraising literature, leading to more timely and cost-effective systematic reviews, in turn leading to up-to-date, evidence-based medicine and therefore more accurate diagnosis and decisions by clinical professionals.

Chapter 4

Objective Query Formulation

This chapter introduces the objective query formulation methodology used by information specialists, and a computational adaptation of this methodology to support the automatic formulation of queries.

The *objective* [90, 213] method for deriving systematic review search strategies aims to avoid the unrigorous, subjective aspects of the conceptual approach. In this method, a small set of 'gold standard' studies are first identified — these serve to (semi-)automatically identify keywords to add to the query and validate its effectiveness. The gold standard set is akin to the seed studies considered in the conceptual approach, but generally much more extensive (conceptual: a handful; objective: in the order of 10s-100s). Despite the name, this method is still ad-hoc and involves manual trial-and-error for choosing a subset of the identified keywords to add to the query, and where to place keywords in the query. In addition, this method has only been evaluated on a handful of use-case systematic reviews; thus, its effectiveness and generalisability beyond these cases are as yet unknown.

In this chapter, we propose a computational adaptation of the objective methodology proposed by Hausner et al. [90] for objectively deriving medical systematic review search strategies. Our approach does not require manual human involvement, nor trial-and-error procedures, and, is capable of generating a query automatically, given a set of relevant studies as input. The primary goal of this research is to develop a more transparent and less subjective method to search strategy development by computationally adapting and extending the current objective approach. Our method for automatically and objectively deriving search strategies is an adaptation of the method initially proposed by Hausner et al. [90]. A high-level process overview of the objective method is shown in Figure 4.1. This figure highlights manual aspects of the original method that we computationalise, and extensions to the original method our method makes, seeking to reduce bias further. The following two sections first describe the original method and the specific computational adaptations and extensions we make.

4.1 The Objective Methodology

The objective method [90] starts by identifying studies either by scanning the references of similar, already published systematic reviews or by issuing broad queries to medical databases (e.g., PubMed,

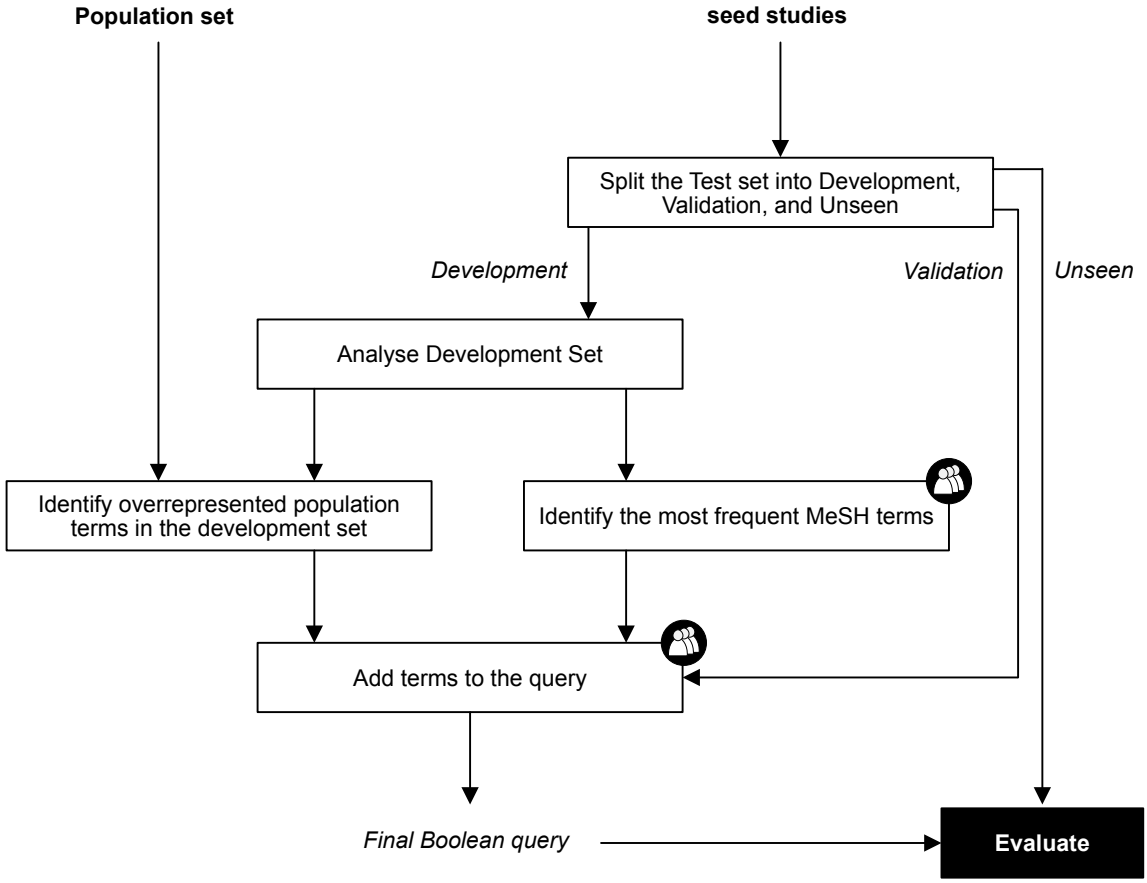


Figure 4.1: The process used for deriving queries using the objective method. The dashed line signifies an extension of the objective method not in the originally proposed method. The symbol refers to the processes in the objective method, which currently require manual intervention. Automating these manual processes is the main focus of this work.

MEDLINE) and screening a subset for identifying gold-standard references. These references form the 'test' set, which is divided into a development set ($\frac{2}{3}$) and a validation set ($\frac{1}{3}$). The title and abstract of each of the references in the test set are then analysed by ranking each term by the frequency it appears in each of the references (i.e., document frequency – DF). Next, these terms are filtered to include the top 20% of terms according to DF. Simultaneously, a population set (i.e., a background collection) of 10,000 studies is identified by issuing an empty search to PubMed and restricting results to the last 12 months (the default ranking of PubMed is by descending publication date). According to DF, the previously filtered terms are filtered yet again to include the bottom 2% of terms. Finally, the 20 most frequent MeSH terms are identified from studies in the development set. A Boolean query is then developed by dividing terms into three categories (which form three clauses, linked with the Boolean `AND` operator): (1) terms relating to health conditions, (2) terms relating to a treatment, and (3) terms relating to the types of study design to be included. Through a time-consuming process of trial-and-error, filtered terms and MeSH terms are then added to one of the three clauses of the query depending on the category of the term. Terms inside a category are combined using an `OR` operator, and the three categories are then combined with an `AND` operator. The effectiveness of the query is

then compared to the validation set. While the objective method seeks to remove the human in the loop to reduce bias in query formulation, several manual steps require a human, for example, adding the identified terms into the query. In the next section, we propose a framework that automates the entire objective method pipeline to formulate Boolean queries without human involvement.

4.2 Automating the Objective Method

We propose a number of computational modifications to this method that seeks to remove human subjectivity from the process further. We also improve the process by which the evaluation of the resulting queries is undertaken in several ways. In our modified methodology, we begin with the same test set; however we split into ($\frac{2}{4}$) development, ($\frac{1}{4}$) validation, and ($\frac{1}{4}$) unseen. The unseen set is used to approximate how the query will perform on studies which are not assessed (i.e., a study which is relevant, but which may never be retrieved by the query, and therefore never screened for potential relevance). It also allows us to develop the query on the development set, tune parameters values on the validation set, and study their effectiveness on the unseen set. The first difference is that in addition to terms (uni-grams), we also consider phrases (n-grams). From here out, a 'keyword' represents either a term of a phrase. We then follow the same method of filtering keywords using the development set and the population set and identifying MeSH terms to use. To automatically assign a category for a keyword, the semantic type of a keyword is used. The semantic type is obtained automatically by mapping keywords to UMLS concepts via MetaMap [11] (version 2018 with options set to their default values and using UMLS2018AA). If a keyword does not map to a concept in MetaMap, it is discarded. Once a semantic type is obtained for a keyword, it is automatically categorised in a two-fold process: (i) if the semantic type is present in Table 4.1a, then the keyword is mapped accordingly [233], (ii) if the semantic type is not present, then the semantic group of the semantic type is used to broadly categorise the keyword according to Table 4.1b. Note that some keywords may be discarded in step (ii) due to the semantic group they belong to, denoted by 'None' in the table. Following this process, the identified MeSH terms are then added to one or more of the three categories according to the top-level MeSH parent in Table 4.1c. Once all of the identified keywords are categorised, the computational assembly of the Boolean query takes place.

We also take a computational approach to the assembly of the Boolean query. A naïve approach could involve trying all combinations of keywords in a category with all combinations of all other keywords in all other categories (similar to the manual trial-and-error employed in the original method). However, the complexity of this approach presents itself as infeasible: $O(n!^3)$ (where n is the number of keywords for a given category, assuming all categories have the same number of keywords, in the worst case). Instead, we compute the maximum number of studies in the development set retrievable using the filtered keywords and MeSH terms by first representing the set of relevant studies retrieved for each category as a binary array (e.g., health conditions $\vec{c}$=[1,1,1,1,1,1,1,1], treatments $\vec{t}$=[0,1,1,1,1,1,1,1] and study type $\vec{s}$=[1,1,0,1,1,1,1,1]), where 1 indicates that the relevant study referred by that position in the array is retrieved. Then we perform conjunction (bitwise `AND`) on the three binary arrays to

Uzuner et al. [233] Relationship		Hausner et al. [90] Category
Test	→	Treatment
Treatment	→	Treatment
Diagnosis	→	Condition

(a) How a relationship as identified by Uzuner et al. [233] maps to a category.

Semantic Group		Hausner et al. [90] Category
ACTI	→	Treatment
ANAT	→	Condition
CHEM	→	Treatment
CONC	→	None
DEVI	→	Treatment
DISO	→	Condition
GENE	→	Condition
GEOG	→	Study Type
LIVB	→	Condition
OBJC	→	Treatment
OCCU	→	Condition
ORGA	→	Study Type
PHEN	→	Condition
PHYS	→	Condition
PROC	→	Treatment

(b) How a semantic group maps to a category.

MeSH top-level heading		Hausner et al. [90] Category
Anatomy	→	Condition
Organisms	→	Condition
Diseases	→	Condition
Chemicals and Drugs	→	Treatment
Analytical, Diagnostic and Therapeutic Techniques, and Equipment	→	Treatment
Psychiatry and Psychology	→	Condition
Phenomena and Processes	→	Condition
Disciplines and Occupations	→	Condition
Anthropology, Education, Sociology, and Social Phenomena	→	None
Technology, Industry, and Agriculture	→	None
Humanities	→	None
Information Science	→	Study Type
Named Groups	→	Condition
Health Care	→	None
Publication Characteristics	→	Study Type
Geographicals	→	Study Type

(c) How a top-level MeSH heading maps to a category.

Table 4.1: Processes for mapping keywords to a Hausner et al. [90] category in a query.

obtain a new binary array (i.e., $\vec{c} \wedge \vec{t} \wedge \vec{s} = \vec{q}$ =[0,1,0,1,1,1,1,1]) which represents the set of relevant studies in the development set that can be retrieved by the query that includes all keywords (i.e. the maximal query). Note that when there are no keywords present for a category[1] then the category is removed from the conjunction which forms $\vec{q}$. The logical conjunction of the three vectors has the same effect as executing the query, thus greatly increasing the number of comparisons made (i.e., it reduces computation time). Further note that it is not guaranteed that the set of categories which contain keywords from the development set, when combined using a Boolean `AND` operator, will retrieve all the relevant studies in the validation set — this is true regardless of using our technique for speeding up query assembly or trying all combinations. Next, iteratively, each keyword from each category is temporarily removed, and a new binary vector ($\vec{v}_i$) is computed, containing the set of relevant studies in the development set retrieved without that keyword. If $\vec{q} \wedge \vec{v}_i = \vec{q}$, that is, if the removal of the keyword does not affect the number of relevant studies in the development set retrieved by the rest

[1] e.g., there are no keywords that can be categorised into Study Type (**s**), but there are keywords categorised into Conditions (**c**) and Treatments (**t**).

of the keywords, then the removal of that keyword from the category is made permanent. In other words, that keyword contributes nothing overall to the query (or its contribution is redundant as its contributing studies are also retrieved by other query keywords) and is removed from its respective clause. Note that this technique could also be used in an interactive system to highlight to a user those keywords that do not contribute to the set of retrieved documents, or for evaluating existing search strategies. The iteration proceeds by considering one candidate keyword for removal at a time; keywords are ordered descending by the sum of the components of their document vectors, i.e., their total document frequency, thus the order of keywords removed is deterministic. The complexity of this approach is $O(3n)$: each keyword is required to be only tested once for inclusion in the final query, rather than all possible combinations. The resulting query is guaranteed to retrieve the maximum number of relevant studies possible in the development set (based on the keywords identified in the previous process).

We further propose to tune the keyword cut-off thresholds parameters for the filtering steps. Rather than fixing the thresholds at 20% for development and 2% for the population (as done by Hausner et al. [90]), we perform a grid search (independently for each query) over combinations of thresholds to find the parameters best suited (optimising for F_1, F_3, recall) for a particular query. We also apply the same strategy to identify the number of MeSH terms to add to a query. The development set is used to identify keywords; then, we evaluate queries using the validation set to select the best combination of parameters. The query can then be evaluated fairly on the unseen set.

4.3 Summary

This chapter presented a computational approach to objectively deriving search strategies for systematic reviews. This approach adapts and extends the proposal of Hausner et al. [90], to further reduce human subjectivity in an otherwise objective methodology. The manual objective method included human intervention; our computational adaptations approximate the steps a human would take.

The computational method presented here is envisioned to be integrated into tools for assisting researchers conducting systematic reviews (for example, query suggestion [112]). The aim is not to replace humans constructing search strategies — at the very least, several gold-standard studies are still needed to seed this approach, (as is typically the case in this context [120]). Query formulation is currently a highly subjective and error-prone process, and reducing subjectivity and mistakes in search strategy construction can only lead to less biased, reproducible, and timely systematic reviews.

Chapter 5

Comparison of Automatic Query Formulation

The previous two chapters introduced two popular methodologies for Boolean query formulation for systematic reviews and methods for computationally adapting these methodologies to support the automatic formulation of queries. This chapter presents a comprehensive set of experimental results and analysis comparing these automatic query formulation methods to each other, and to manually formulated queries. This chapter is guided by the first overarching research question this thesis aims to address (**RQ1**), and illustrated by Figure 5.1.

RQ1

Can search strategies be automatically formulated and if so, are they comparable in effectiveness to search strategies manually formulated by information specialists?

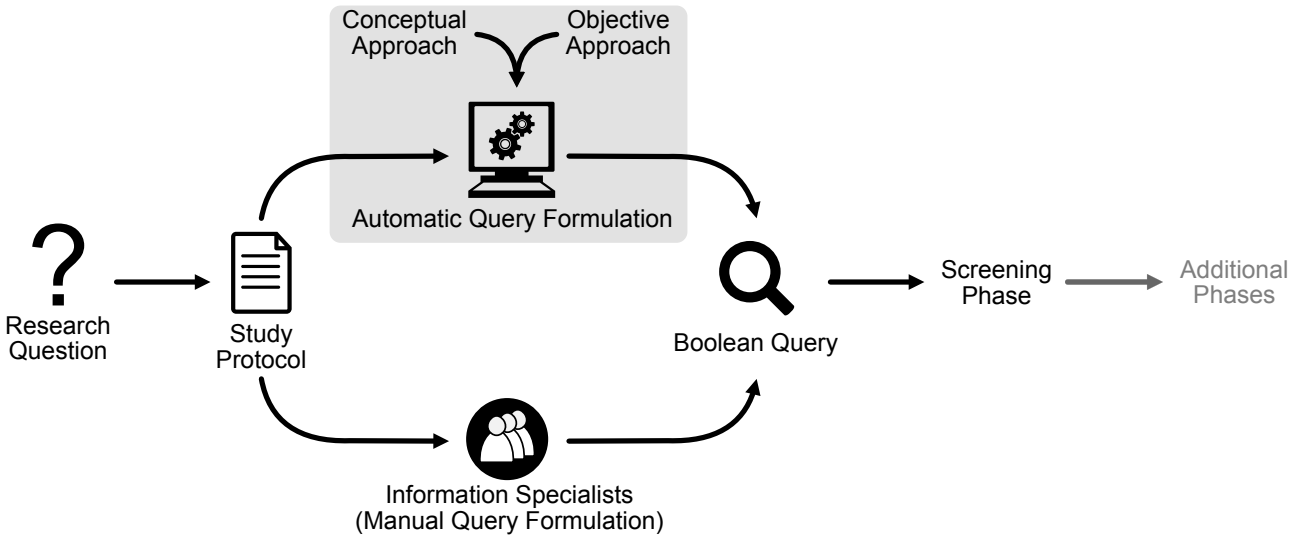


Figure 5.1: The initial phases of systematic review creation that this study focuses on. The highlighted area indicates the aspect of the creation phase that we focus on: query formulation. In particular, we propose to automate a currently manual task, indicated below the highlighted area.

This chapter seeks to compare the differences between queries derived from the two automatic query formulation methods and queries derived from each of the automatic query formulation methods and the original, manually formulated queries. The two computational approaches will be compared to each other to determine which one is more appropriate for systematic review literature search. We also assess the effectiveness of the two computational approaches as starting points for further manual refinement. Finally, an analysis of seed studies and their impact on query effectiveness will be undertaken. Seed studies are commonly used in the development of queries for systematic review literature search. Specifically, they are highly relevant studies that are identified before starting the review. However, there currently does not exist a test collection with the original seed studies used for manual query formulation in this domain. For this reason, we instead study how sensitive the two automatic query formulation approaches are to different initial seed studies.

5.1 Research Questions

An investigation into the comparison of automatic query formulation methods between themselves and manually formulated queries and addressing the limitations of prior research underpin the research questions of this chapter. The first two research questions guide the investigation into comparing automatic query formulation methods between themselves and manually formulated queries.

RQ1.1

How does automatic query formulation compare to manual query formulation in terms of search effectiveness?

Specifically, **RQ1.1** aims to identify which automatic query formulation is most effective when compared to a manually formulated query. This comparison is achieved though a batch-style evaluation of automatically formulated and manual queries.

RQ1.2

What factors of automatically formulated queries contribute the most to effectiveness?

Meanwhile, **RQ1.2** aims to identify the factors of automatically formulated queries that contribute to their effectiveness, e.g., choice of terms vs. phrases or the number of seed studies. Note that the focus is on the comparison between the two automatic query formulation methods, not the manually formulated queries. The next two research questions guide the investigation into limitations identified in the previous two chapters.

RQ1.3

Which automatic query formulation method provides the most effective starting point for manual refinement?

Specifically, **RQ1.3** aims to identify which automatic query formulation method provides the most effective query once manually refined, and how these manually refined queries compare to the same originally manually formulated queries. This is achieved through a small-scale case study.

RQ1.4

How sensitive to variation in the initial seed studies are the automatic query formulation approaches?

Finally, **RQ1.4** aims to identify how sensitive each automatic query formulation method is to the initial set of seed studies. Different portions of relevance judgements are sampled for seed studies as input, and statistical variances are studied. In answering these research questions, this chapter makes the following contributions:

- A comprehensive comparison of two automatic query formulation methods to manually formulated queries, and between each other.
- A case study investigating the suggestion that the automatic query formulation methods are good starting points for further manual refinement.
- A comprehensive investigation into the effect seed studies have on the automatic query formulation methods, particularly how sensitive the effectiveness of queries are to a given set of seed studies.

5.2 Experimental Setup

Experiments are conducted on the CLEF TAR 2018 set of diagnostic test accuracy systematic reviews [107]. The CLEF TAR 2018 collection was designed as a shared Information Retrieval task to develop methods to support the screening phase of systematic review creation. We adapt this collection for the use of automatic query formulation. The CLEF TAR task has run for three years; however the 2017 collection is a subset of the 2018 collection, and the 2019 collection contains systematic reviews of various types (including the 2017 collection). In this work, only the 2018 collection is used to control for the type of systematic review (i.e., diagnostic systematic reviews are much more challenging to search literature for than intervention reviews [121]). This test collection contains titles, relevance assessments, and queries for 75 systematic reviews. The PubMed entrez API [189] is used for retrieval and statistics (e.g., document frequency).

Queries are formulated using the computational adaptations of the methodologies described in the previous two chapters. As there are multiple ways for the conceptual and objective methods to be run (e.g., terms versus phrases), we make this distinction clear as an **instantiation** of one of the methods. We define a set of queries formulated with different samples of seed studies for the same topic as an **iteration**. However, this creates a new problem, which is that the query formulation methods may be sensitive to the seed studies used. In total, we perform 30 iterations per query formulation instantiation,

per topic. This method was found to provide us with a sufficiently powered statistical test. Note that the random split for a given iteration is the same across all query formulation instantiations and approaches. Next, to compare the originally formulated queries for each topic to the queries we automatically formulate, we evaluate the original queries on the unseen portion of each iteration (giving 30 runs also for the original queries). In our results, we compare the average performance of a given evaluation measure across all iterations and all topics, for each instantiation of a query formulation approach.

In this work, we use a custom Elasticsearch index containing the UMLS terminology for mapping terms to and from UMLS entities. The matching of terms to entities is handled by the default ranking function of Elasticsearch 7.5.2 (BM25). For mapping term to a single entity, we always choose the top-most ranked entity. This method of entity mapping has been shown to be empirically comparable to MetaMap [140].

5.3 Results

5.3.1 Comparison to Original Queries

This section aims to address RQ1.1:

RQ1.1

How does automatic query formulation compare to manual query formulation in terms of search effectiveness?

We address this question by comparing the automatic query formulation methods to the original, manually formulated queries. This comparison is made in Table 5.1. The results in this table are the average performance across all topics and all iterations of seed study split.

We find that none of the automatic query formulation methods can outperform the original, manually formulated queries. Indeed, often the queries formulated automatically are significantly worse than the original queries. The highest performing automatic method in terms of recall and WSS (evaluation measures critical to the construction of an effective systematic review) is the Term/Recall/MeSH instantiation of the objective method. The highest performing automatic method in terms of precision is the Term/F_3 instantiation of the objective method.

In almost all measures, the objective instantiations outperform the conceptual instantiations. Often, there are significant differences between the effectiveness of the objective and conceptual methods.

The term-based instantiations are almost always more effective between the conceptual and the objective methods than the phrase instantiation counterparts. This difference suggests that more effective queries for systematic review literature search contain single terms instead of a mixture of terms and phrases.

The impact of MeSH terms on queries (at least within the objective instantiations) is clear also: the gains achieved in recall are typically higher in objective instantiations that add MeSH terms to those queries that do not. This also suggests that the use of MeSH terms is critical for maximising recall.

		Precision	$F_{0.5}$	F_1	F_3	Recall	WSS
	Original	0.0217	0.0267	0.0407	0.1439	0.9338	0.9181
Cptl.	Phrase	**0.0023***	**0.0027***	**0.0038***	0.0107*	0.5129*	0.4878*
Cptl.	Term	0.0021*	0.0026*	0.0037*	**0.0114***	**0.6286***	**0.5990***
Objective	Phrase/F_3	0.0031*	0.0037*	0.0055*	0.0213*	0.3572*†	0.3571*†
Objective	Phrase/F_3/MeSH	0.0029*	0.0036*	0.0055*	0.0235*	0.5657*	0.5653*
Objective	Phrase/Recall/	0.0006*	0.0007*	0.0012*	0.0053*	0.5532*	0.5513*
Objective	Phrase/Recall/MeSH	0.0005*	0.0006*	0.0010*	0.0048*	0.7935*†	0.7899*†
Objective	Term/F_3	**0.0053***‡	**0.0065***‡	**0.0099***‡	**0.0365***‡	0.4432*‡	0.4430*‡
Objective	Term/F_3/MeSH	0.0050*‡	0.0061*‡	0.0092*‡	0.0356*‡	0.5482*	0.5478*‡
Objective	Term/Recall	0.0004*‡	0.0004*‡	0.0007*‡	0.0032*‡	0.8126*‡	0.8058*‡
Objective	Term/Recall/MeSH	0.0002*‡	0.0003*‡	0.0005*‡	0.0022*‡	**0.8780**‡	**0.8692**‡

Table 5.1: Results of each automatic query formulation method averaged across each topic, averaged across each of the 30 iterations of query formulation. Statistical significance ($p < 0.05$) between the original queries is indicated by *, between conceptual (Cptl.) phase and objective phrase formulation is indicated by †, between conceptual (Cptl.) term and objective term formulation is indicated by ‡.

5.3.2 Factors Contributing to Effectiveness

This section aims to address RQ1.2:

RQ1.2

What factors of automatically formulated queries contribute the most to effectiveness?

To answer this, we study the correlations between factors that may contribute to the effectiveness of queries and the actual effectiveness of queries.

Factor 1: Choice of Keywords

The first factor that is investigated is the choice of keywords in queries. Specifically, how phrases and MeSH terms contribute to effectiveness, and whether the effectiveness of automatically formulated queries is due to the choice of keywords. Figure 5.2 illustrates how phrases and MeSH terms affect the originally manually formulated queries. Interestingly, while phrases caused a decrease in retrieval effectiveness for most automatically formulated queries (as illustrated in Table 5.1, term vs. phrase instantiations), the addition of phases is correlated with retrieval effectiveness for the original queries. This correlation also holds when MeSH terms are added to the automatically formulated queries (also presented in Table 5.1, MeSH vs. no MeSH instantiations), indeed in the original queries, there is a moderately strong correlation between the number of MeSH terms and effectiveness. These findings suggest that more effective queries contain both phrases and MeSH terms and that both phrases and MeSH terms are conducive of effectiveness. However, as the results in Table 5.1 illustrate, the identification and combination of these keywords are the most critical factors in terms of effectiveness. Furthermore, while more phrases can have a positive impact on the effectiveness of a query, the choice of phrases is more important.

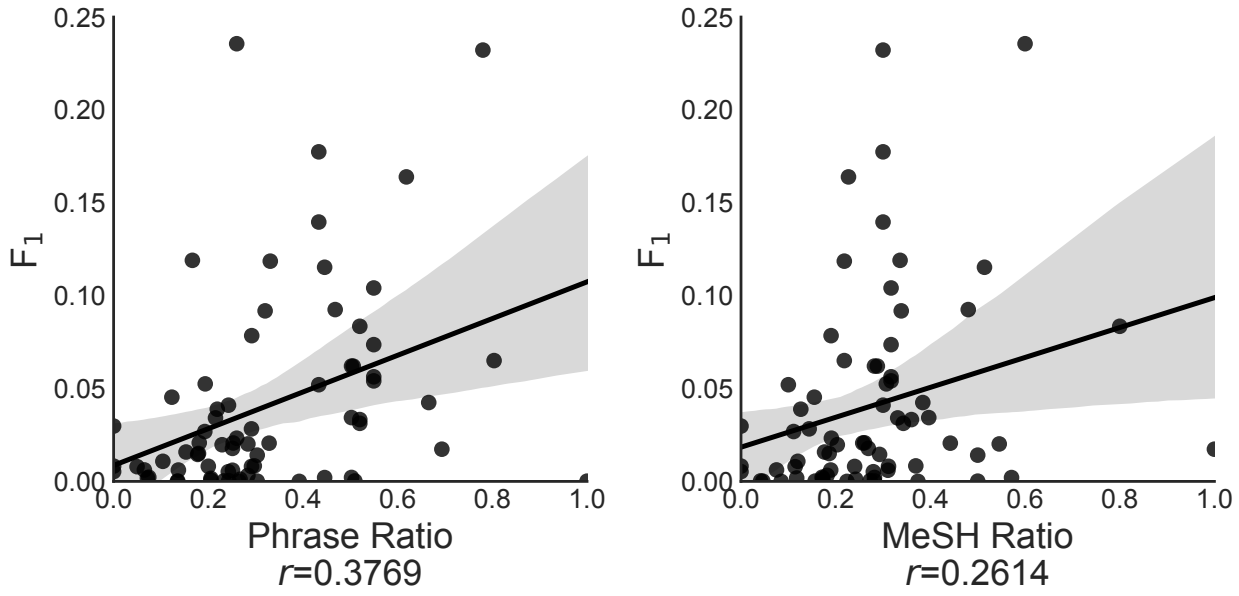


Figure 5.2: Correlations between ratio of phrases ($\frac{|\text{phrases}|}{|\text{keywords}|}$) and ratio of MeSH terms ($\frac{|\text{MeSH terms}|}{|\text{keywords}|}$) in the original, manually formulated queries, and the effectiveness of those queries. Pearson's correlation coefficient *r* is indicated beneath each plot.

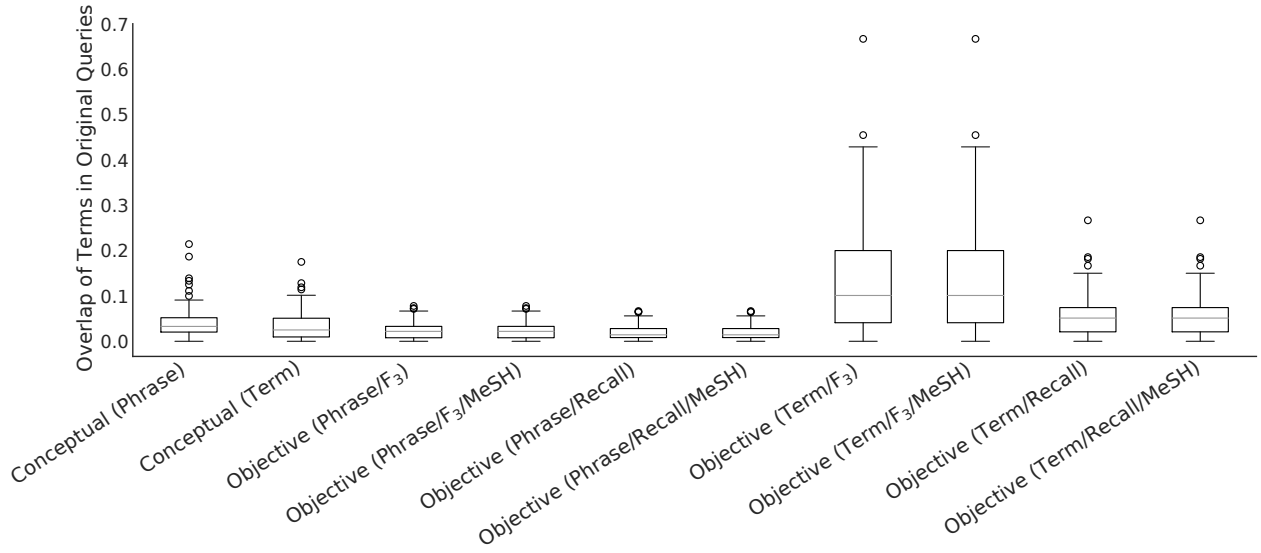


Figure 5.3: Overlap between of keywords (e.g., terms, phrases, MeSH terms) between the originally, manually formulated queries and each instantiation of the automatic query formulation methods. The overlap is normalised by the total number of terms in an automatically formulated query. Terms from all iterations of each instantiation of an automatic query formulation method are combined to compute the overlap.

Figure 5.3 furthers the point that while the identification of correct terms is important, how those terms are combined in a Boolean expression is equally important. This figure illustrates the normalised overlap of terms between the original, manually formulated queries and each instantiation of an automatic query formulation method. Interestingly, most automatically formulated queries have less than 10% of terms in common with the original queries. The intuition that a high keyword overlap between automatically formulated queries and original queries results in high retrieval effectiveness does not hold when considering the objective Term/F_3 and Term/Recall instantiations. The objective Term/Recall/MeSH instantiation achieved the highest recall and WSS of all automatic formulation instantiations (including conceptual methods), while the objective Term/F_3 achieved the highest precision and F-measures. However, the objective Term/Recall instantiations have a lower term overlap than objective Term/F_3. This result suggests that to obtain high recall, the choice of keywords is less important than the way keywords are combined in a Boolean expression, as although the objective Term/F_3 instantiations have a higher overlap in terms than the objective Term/Recall instantiations (suggesting that they are more similar to the original queries), they obtain almost double the recall.

Factor 2: Number of Seed Studies

The second factor that we investigate is the number of seed studies used in the automatic query formulation methods. This is because each topic has different numbers of seed studies for input. Note that this is reflective of reality as there is no set number of seed studies used in query formulation: an information specialist may be provided many, one or even no seed studies to develop a query. We perform this analysis with the intuition that the more seed studies that can be used, the more successful a query formulation method should be at producing an effective query. The correlations between the number of seed studies used for query formulation and query effectiveness is presented in Figure 5.4.

First, looking at the conceptual methods: The number of seed studies is weakly negatively correlated with precision but more strongly correlated with recall, for both the phrase and term instantiations. The conceptual term instantiation is indeed strongly correlated with recall.

Next, investigating the objective instantiations: The inverse is true for the phrase-based instantiations: more seed studies are more strongly correlated with precision and less correlated with recall. Indeed, more seed studies do not necessarily correlate with an increase in recall for many of the objective approaches. However, the term-based instantiations of the objective method which optimise for recall do see a moderate correlation in both precision *and* recall; indicating that for at least these instantiations, more seed studies do indeed correlate with more effective queries.

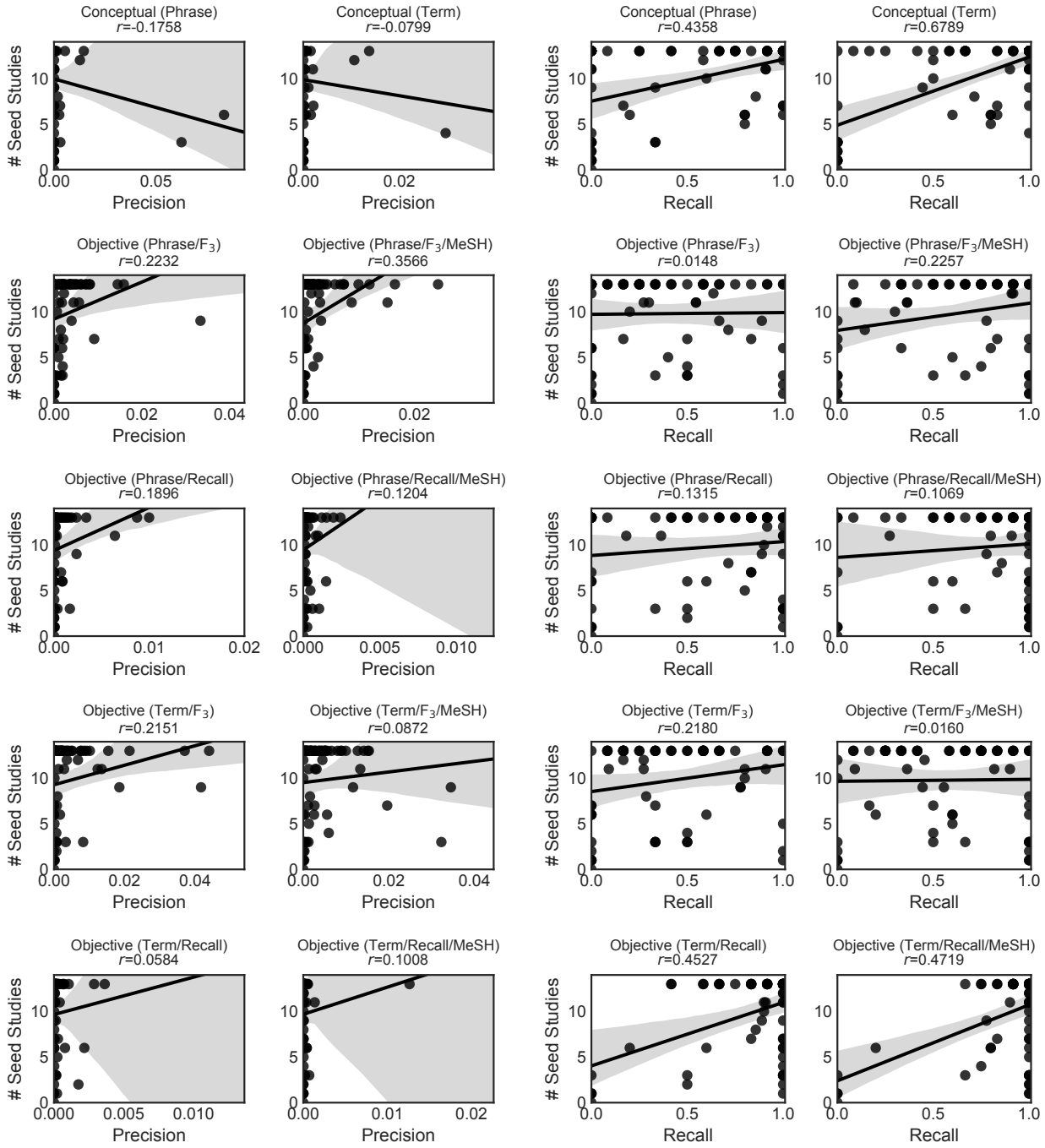


Figure 5.4: Correlation between number of seed studies and effectiveness of queries. Each point in a plot refers to an iteration of the automatic query formulation approach given in the title (thus the x-axis is averaged across topics). Plots on the left correspond to precision, plots on the right refer to recall. Pearson's *r* is indicated between the two variables beneath the title of each plot.

	Precision	$F_{0.5}$	F_1	F_3	Recall	WSS
Original	0.0263	0.0324	0.0494	0.1686	0.8869	0.8232
Conceptual (Formulated)	0.0025	0.0031	0.0049	0.0220	0.6458	0.6177
Conceptual (Refined)	0.0020	0.0025	0.0040	0.0188	0.9166	0.9159
Objective (Formulated)	0.0001	0.0002	0.0003	0.0017	0.9687	0.9607
Objective (Refined)	0.0009	0.0011	0.0018	0.0090	0.9375	0.9368

Table 5.2: Results of case study using manual query refinement after automatic query formulation.

5.3.3 Effectiveness after Manual Refinement

This section aims to address RQ1.3:

RQ1.3

Which automatic query formulation method provides the most effective starting point for manual refinement?

We address this question through a case study where we manually refine a small subset of the automatically formulated queries by removing terms from the query. We take a small subset of queries (approximately 10% of topics, 8 in total) from the highest performing iterations in terms of recall and manually apply query reduction using the validation set to validate the effectiveness of the queries. The results of this case study are presented in Table 5.2. We first compare the results of the automatically formulated queries and the same queries but manually refined. The queries automatically formulated using the conceptual approach performs the worst (mirroring the results in Table 5.1). However, once refined, the recall of these queries outperforms the same original queries, with a marginal drop in precision.

On the other hand, manually refining the objective queries resulted in a small drop in recall and a small increase in precision. This result suggests that queries automatically formulated using the conceptual approach can be much more effective once refined. The objective approach formulates queries with a very high recall that is difficult to maintain while increasing precision when refining. This finding leads to an overarching result about query formulation in this domain: it may be easier to refine a query to increase recall when precision is high than it is to refine a query to maintain recall and increase precision when recall is high.

5.3.4 Sensitivity to Seed Studies

This section aims to address RQ1.4:

RQ1.4

How sensitive to variation in the initial seed studies are the automatic query formulation approaches?

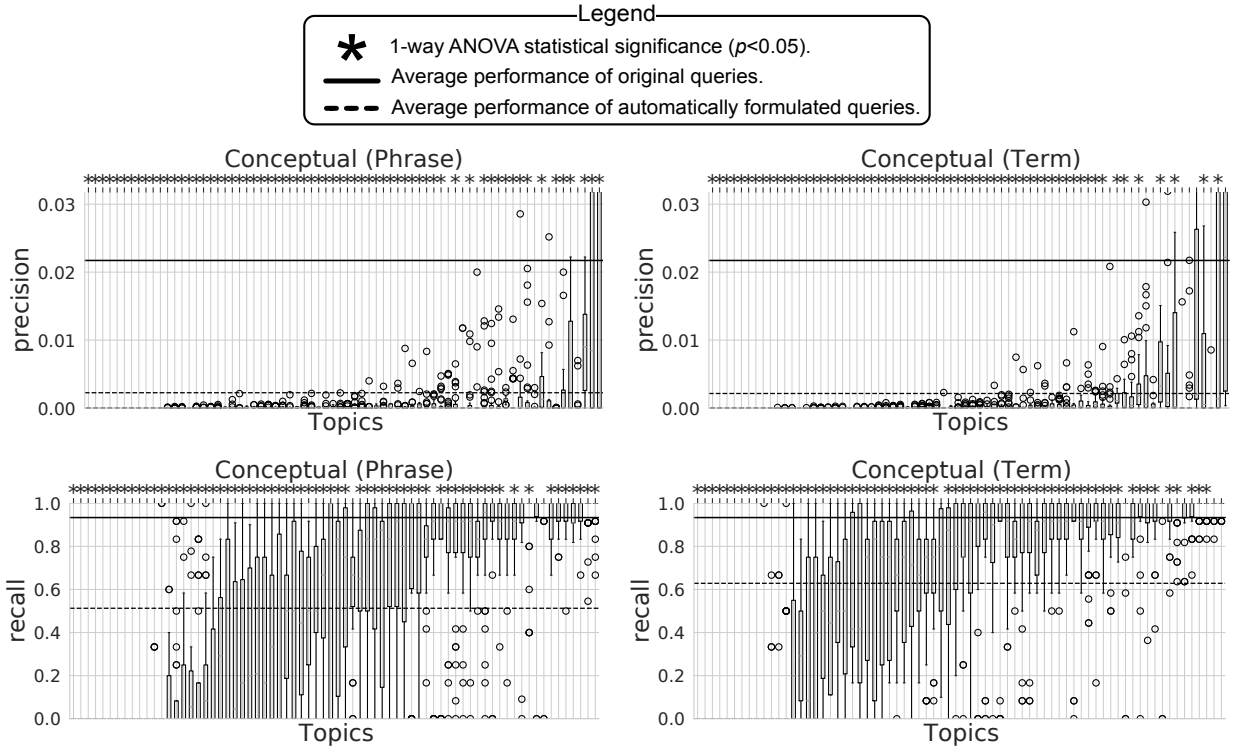


Figure 5.5: Sensitivity of the two instantiations of the automatic conceptual query formulation method, measured using precision and recall. Left: phrase-based instantiation, right: term-based instantiation. Approximately one-third of topics obtain high recall with relatively low sensitivity (far right topics in bottom plots), while another third is highly sensitive to seed studies (middle topics in bottom plots). The last third of topics did not retrieve any relevant studies (far left topics in bottom plots). Meanwhile, for both instantiations, many topics obtained very low precision, except for a handful from the conceptual Term method; although these topics mostly vary in effectiveness.

We address this question by analysing the per-topic performance of each instantiation of both of the fully automatic Boolean query formulation methods. This is shown in Figures 5.5, 5.6, and 5.7. Each figure illustrates the per-topic breakdown of the effectiveness of each instantiation of an automatic Boolean query formulation method (given in the title) in terms of precision or recall (the y-axis). Boxes in each plot are ordered by the average performance of each topic to show better the differences between instantiations of the query formulation methods. Note that this means that plots cannot be compared to each other using the x-axis.

First, examining the topic breakdown plots for the conceptual instantiations in Figure 5.5, we note that while some topics are able to achieve reasonably high performance, a number of topics result in an overall poor average performance. The recall plots specifically tell an interesting story: several high-performing topics show low variability *and* closely match the performance of the original queries. Meanwhile, the majority of topics display very high variability in effectiveness. The precision of topics in the conceptual instantiations is overall poor, and for most topics, there are little variations in the low performance. However, approximately one-quarter of topics from both instantiations are sensitive to seed studies, causing variation in retrieval effectiveness, suggesting that the choice of seed studies does indeed impact effectiveness for these topics. Approximately one-quarter of topics in both instantiations retrieve no relevant studies across all iterations. For the automatic conceptual method, the choice of seed studies can have a considerable impact on the overall effectiveness of individual queries, especially recall.

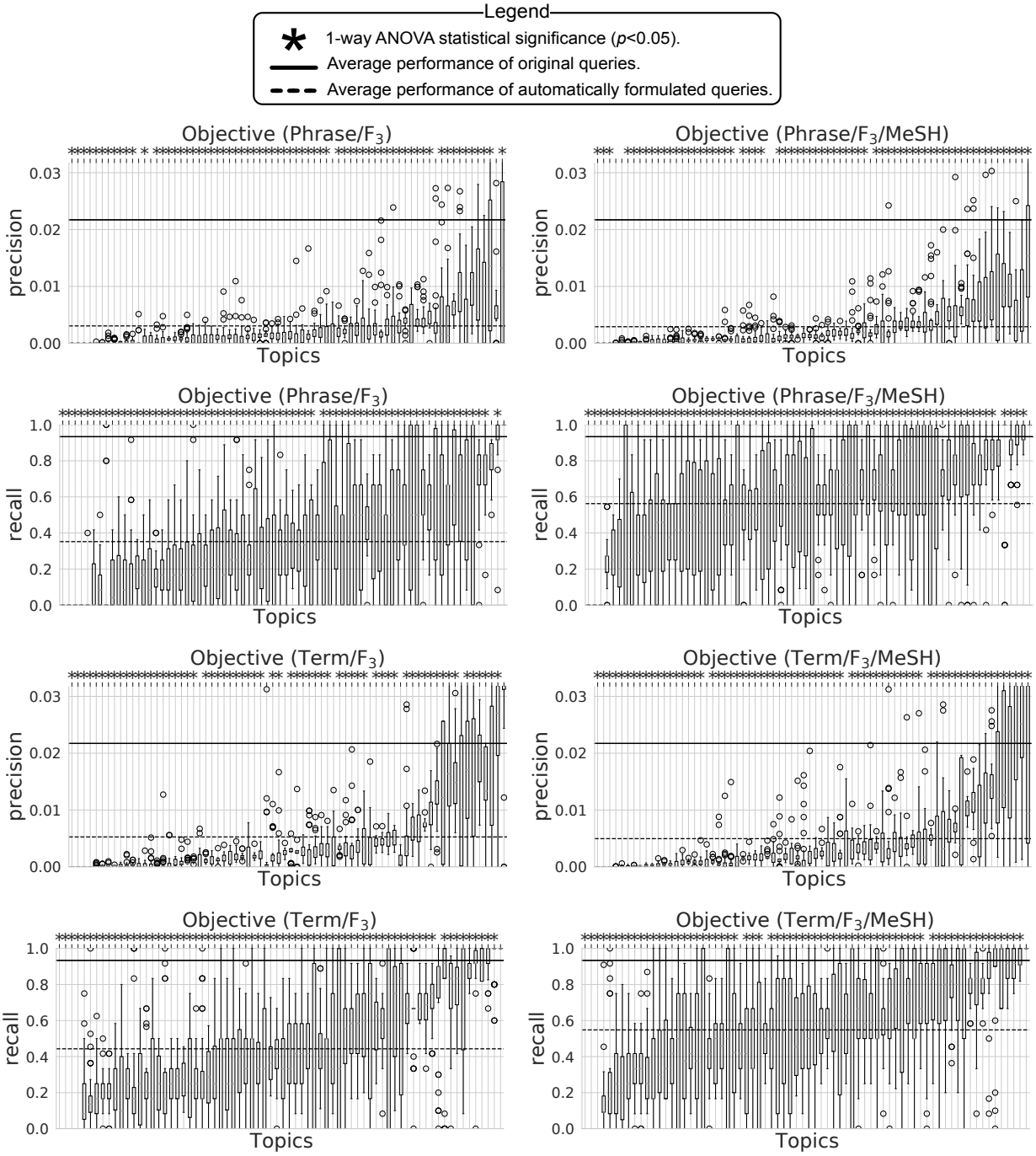


Figure 5.6: Objective F_3-optimised query formulation. Left: without MeSH terms, right: with MeSH terms. Overall, these instantiations have the highest variability in terms of both recall and precision. There is little difference between the phrase-based instantiations and the term-based instantiations

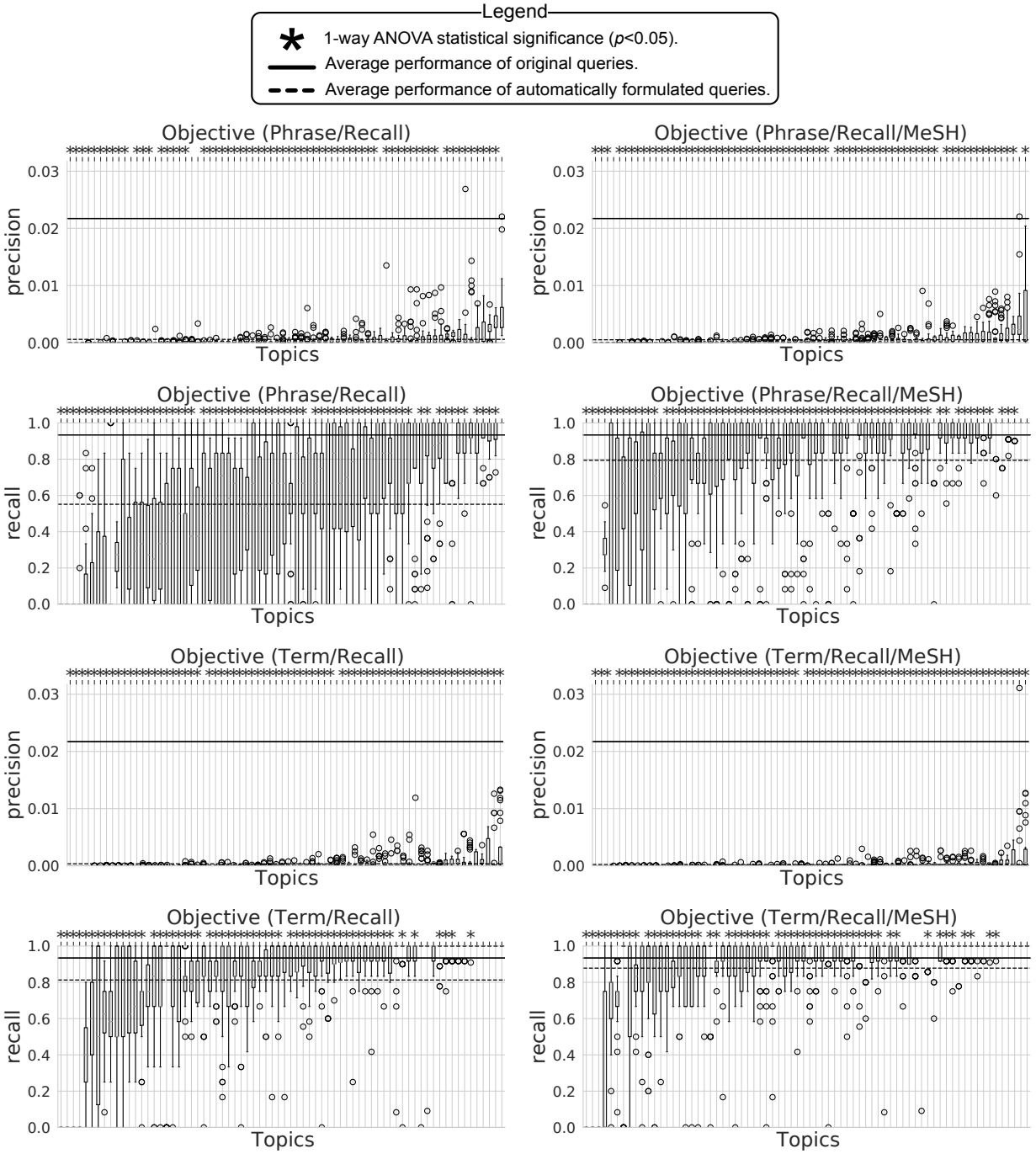


Figure 5.7: Objective recall-optimised query formulation. Left: without MeSH terms, right: with MeSH terms. The term-based instantiations have the lowest variability in recall overall, while also achieving the highest recall overall. Meanwhile, the phase-based instantiations are similar in variability to the F_3-optimised instantiations.

Next, we report the variability for the objective instantiations which optimise for F_3 in Figure 5.6. Overall, there is a high amount of variability in the effectiveness of queries for both precision and recall, for all instantiations. Between the phrase and the term instantiations, there is little difference in variability due to the seed studies. Indeed for most topics, there is a high statistical difference between the variability of the original queries and the automatically formulated queries. Comparing these instantiations to the previous conceptual instantiations on a per-topic basis, these instantiations are more likely to produce queries that retrieve more relevant studies: fewer topics overall retrieve zero studies. However, when averaged across each iteration, the conceptual instantiations achieve a higher recall as there is less variability. This result may be because these instantiations attempt to put some weight towards precision during the tuning process: overall, these instantiations obtain the highest precision out of all other instantiations studied in this chapter.

Finally, we report the variability for the objective instantiations which optimise for recall in Figure 5.7. The most immediate result is the relatively low variability in recall for the objective Term/Recall/MeSH instantiation. This instantiation is the most tolerant to sensitivity in seed studies among all methods. Comparing across all other instantiations, the queries automatically formulated for this topic have the lowest overall variability. The average recall across all topics and iterations for this particular instantiation is very close to the original queries. However, this must be balanced with the significant difference in precision between these queries and the original ones. Comparing these queries to the objective queries tuned for F_3, there is a large difference in the variability between the phrase-based instantiations and the term-based instantiations. This is unlike the instantiations tuned for F_3, where there is little difference in the variability of queries between phrase-based and term-based instantiations. For all instantiations in Figure 5.7, the precision of almost all topics is low, as expected when queries are tuned for recall. Unlike the conceptual instantiations or the objective instantiations tuned for F_3, the variability in precision among these topics is low for almost all topics.

5.4 Discussion

5.4.1 Comparison to Original Queries

The first research question,

RQ1.1

How does automatic query formulation compare to manual query formulation in terms of search effectiveness?

guided the investigation into comparing Boolean queries derived from two automatic query formulation methods to original, manually formulated Boolean queries. We found that the **automatic query formulation methods** investigated in this work are **only somewhat effective compared to the original, manually formulated queries**. This finding demonstrates the utility of information specialists in applying their expertise to query formulation. That being said, we suspect that the intellectual

burden involved in query formulation can be massively reduced through the use of automatic query formulation. It is also worth noting that the original, manually formulated queries have undergone the scrutiny of colleagues and peer review. These steps are likely to greatly improve the quality and effectiveness of queries making the comparison performed in this study more stringent (as we do not have access to the queries before these quality control steps).

5.4.2 Factors Contributing to Effectiveness

The second research question,

RQ1.2

What factors of automatically formulated queries contribute the most to effectiveness?

guided the investigation into two specific factors that were seen to likely contribute the most to the effectiveness of queries. We performed an extensive analysis to determine if these factors of the automatically formulated queries lead to effective queries. Firstly, we found that in order to have a high recall, it is not necessary to have the same terms as the original queries. We found that a high term overlap with the original query instead leads to high precision. Next, we identified that queries with MeSH terms are more conducive of higher recall and that the more MeSH terms in a query, the more effective that query. For all instantiations of the conceptual and objective methods, we also found that phrases reduce recall while increasing precision. Higher numbers of keywords in the original queries strongly correlated with higher effectiveness, suggesting that the use of many phrases, MeSH terms, and terms could be beneficial. The key finding for us, however, was that the **choice of keywords is more important than the number of keywords**. We also found that generally, **the more seed studies used for automatic query formulation, the more effective queries were**, both in precision and recall. However, there is still work to be done to reduce the variability of query formulation. As the conceptual and objective methods are deterministic, the only variable introduced to each method is the set of seed studies used to start the query formulation process. This variable indicates that one of the most important factors contributing to the effectiveness of queries is the selection of seed studies.

5.4.3 Effectiveness after Manual Refinement

The third research question,

RQ1.3

Which automatic query formulation method provides the most effective starting point for manual refinement?

guided a case study that involved the manual refinement of automatically formulated queries. We found that when some manual effort is expended on refining the automatically formulated queries, i.e., through query reduction, the **queries can become as effective as the original (manually formulated)**

queries. Specifically, we found that the automatic conceptual approach should be chosen when recall is the preferred measure to optimise a search for, and the automatic objective approach should be chosen when precision is the preferred measure to optimise a search. The automatically formulated, manually refined queries obtain a higher recall than the manual, original queries. However, the precision of the manually refined queries is still lower than the original queries. Furthermore, it was found that the conceptual queries obtained a much higher recall once manually refined, while approximately maintaining their precision.

Note that the refinement was performed by us and not an experienced information specialist. It is likely that if an experienced information specialist were to refine the automatically formulated queries, then both precision and recall could be increased, in line with the effectiveness of the original queries.

5.4.4 Sensitivity to Seed Studies

The fourth research question,

RQ1.4

How sensitive to variation in the initial seed studies are the automatic query formulation approaches?

guided the investigation into the sensitivity of the automatic query formulation methods in terms of retrieval effectiveness. Almost all of the automatic Boolean query formulation methods investigated in this chapter were **highly sensitive to the initial seed studies**. While we cannot know for certain if manually formulated queries using the conceptual or objective approaches are as sensitive to seed studies, as this would require humans to develop searches, the effectiveness of manually formulated queries is almost always higher than automatic approaches. While the conceptual query formulation methods offer a more consistent base for manual query refinement than the objective F_3 instantiations, the most consistent and least sensitive to seed studies was the objective (Term/Recall/MeSH) instantiation. However, as the results of the manual refinement show, these queries are more difficult to refine (to increase precision while maintaining recall) than the conceptual (Term) instantiation which was easier to refine (to increase recall while maintaining precision). A large-scale user study must be undertaken to truly determine the most effective base for manual query refinement. We leave this for future research.

5.5 Summary

This chapter presented the evaluation of the conceptual and objective automatic Boolean query formulation methods. Instantiations of these methods were compared with each other between formulation methods and within instantiations of a method. Automatic instantiations of the objective and conceptual methods were also compared to queries formulated manually for the same topics. An analysis to determine which factors produced more effective queries was undertaken and how sensitive

the automatic query formulation instantiations are to seed studies. We also performed a small case study to determine which instantiation of the highest performing formulation method provides the best starting point for manual query refinement and how the sensitivity to seed studies may affect this. Our main findings are that while the automatic Boolean query formulation instantiations of the objective and conceptual methods on their own cannot beat the performance of the original queries, with some manual refinements (in this case query reduction), they can be more effective. The conceptual, computational adaptions should be used for this purpose as they achieved the highest precision once refined, and a comparable recall to the original queries. If no manual query refinements are desired, the objective adaptations are a more suitable choice; however, the trade-off between precision and recall depends on the evaluation measure optimised. We also found that both automatic methods are sensitive to seed studies and that the instantiations of these methods that are term-based are generally less sensitive to variation and more effective. Instantiations that use MeSH terms generally have a higher recall with a trade-off in precision, and instantiations that use phrases generally have higher precision with a trade-off in recall.

The results of this chapter impact both new techniques for automatic Boolean query formulation for systematic review literature search and manual approaches. Our empirical findings confirm the intuitions that queries should prefer terms to increase recall and carefully chosen phrases to increase precision. The choice of seed studies can significantly impact the resulting query, and these should be chosen carefully to ensure maximum coverage of relevant studies. These findings resolve **RQ1** of this thesis:

RQ1

Can search strategies be automatically formulated and if so, are they comparable in effectiveness to search strategies manually formulated by information specialists?

This chapter demonstrated that search strategies could indeed be automatically formulated. However, when compared to strategies manually formulated by information specialists, there were not found to be as effective. Despite this, it was also found that the queries became comparable to those formulated originally through minimal manual query refinement.

The end goal of this line of research is to integrate it into tools for information specialists to use to reduce the cognitive burden of query formulation, to provide a less subjective basis for query formulation, and to ultimately improve the systematic review creation process by reducing the total number of studies to screen for inclusion in the systematic review. One such tool is demonstrated in the following chapter and concludes this part of the thesis.

Chapter 6

Tools to Support Query Formulation

This chapter describes tools that leverage the previous research into automation query formulation methods. These tools address specific problems in the pipeline of search strategy development relating to query formulation. Although none of these tools has been empirically evaluated to determine their utility (e.g., in a user study setting), some have been used to develop queries for real systematic reviews [45]. One of the research aims of this thesis is to demonstrate practical query automation tools to assist information specialists.

Tool Demonstration: Query Formulation

Examples of query automation tools for systematic review literature search to support information specialists in formulating more effective queries.

What follows is a description of three query automation tools developed, including the motivation behind each.

6.1 Description of Tools

The following tools are described in this chapter:

AutoFormulate Fully automatic formulation of Boolean queries suitable for systematic review literature search. Implementation based on Chapter 4.

KeywordSuggest Suggestion of semantically related keywords for use as search terms in a Boolean query. Reduces cognitive load and bias.

AutoDoc Framework to support the documentation of search strategies for publishing alongside a systematic review. Aids in the reproducibility and overall quality of a systematic review.

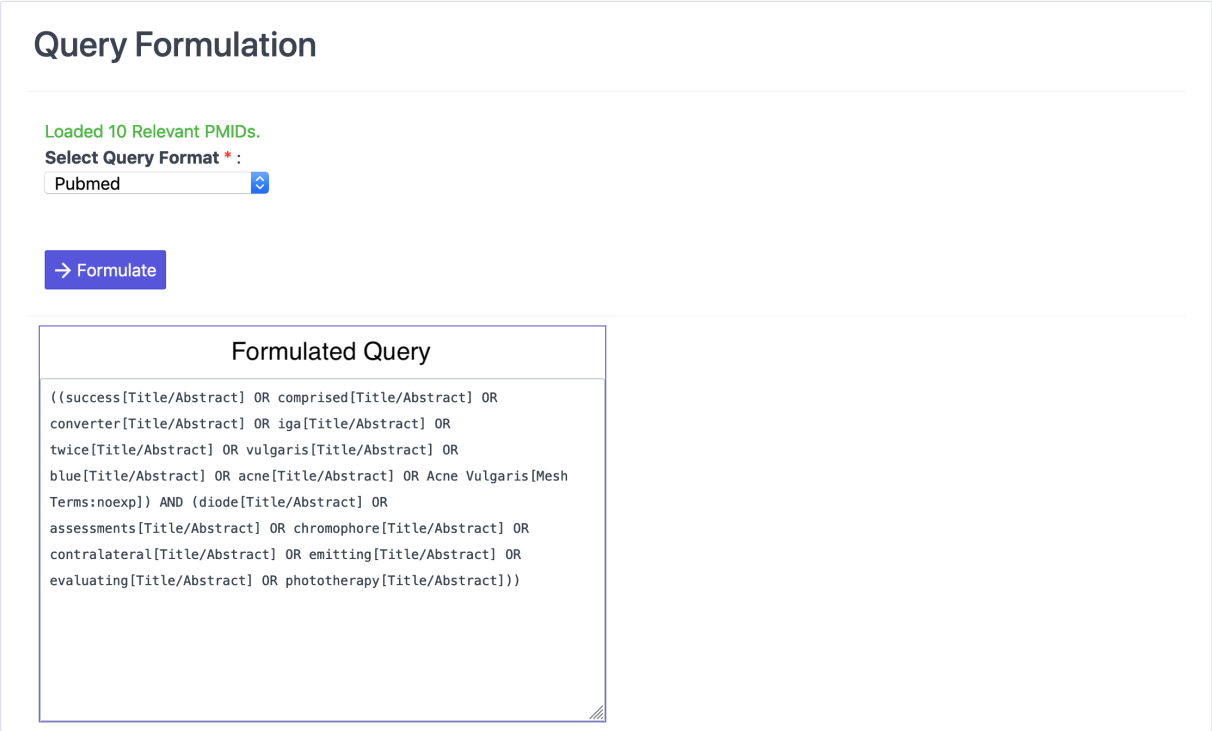


Figure 6.1: Formulating a PubMed query using the AutoFormulate tool.

6.1.1 AutoFormulate

AutoFormulate (Figure 6.1) is an interface for supporting the automatic formulation of search strategies using the objective method. It was developed to reduce the information specialist workloads, unnecessary errors, and bias in formulating search strategies. It is based on the computational adaptation of the objective query formulation approach, described in Chapter 4. It does not require manual human involvement, trial-and-error procedures, and is capable of automatically generating a query. AutoFormulate provides a direct interface to the computational adaptation of the objective method. To use this tool, information specialists first input a set of seed studies. These seed studies are divided randomly into two sets for the purpose of query development and validation. The development set is used to identify suitable terms for the query. The validation set is used to tune which combination of terms results in the most effective best query (queries can be tuned for different evaluation measures, e.g., precision, recall, F_β, etc.). By default, this automation tool uses F_3 as the evaluation measure to optimise.

The motivation behind the development of this tool was to provide a way for information specialists to interact with the computational models presented earlier in this part. For areas of medicine where an information specialist may not be familiar with, this tool provides a good starting point for further development and refinement.

6.1.2 KeywordSuggest

KeywordSuggest (Figure 6.2) is a tool that provides synonym suggestions given an input keyword. Identifying synonyms for high-level concepts is a challenging and time-consuming task for information

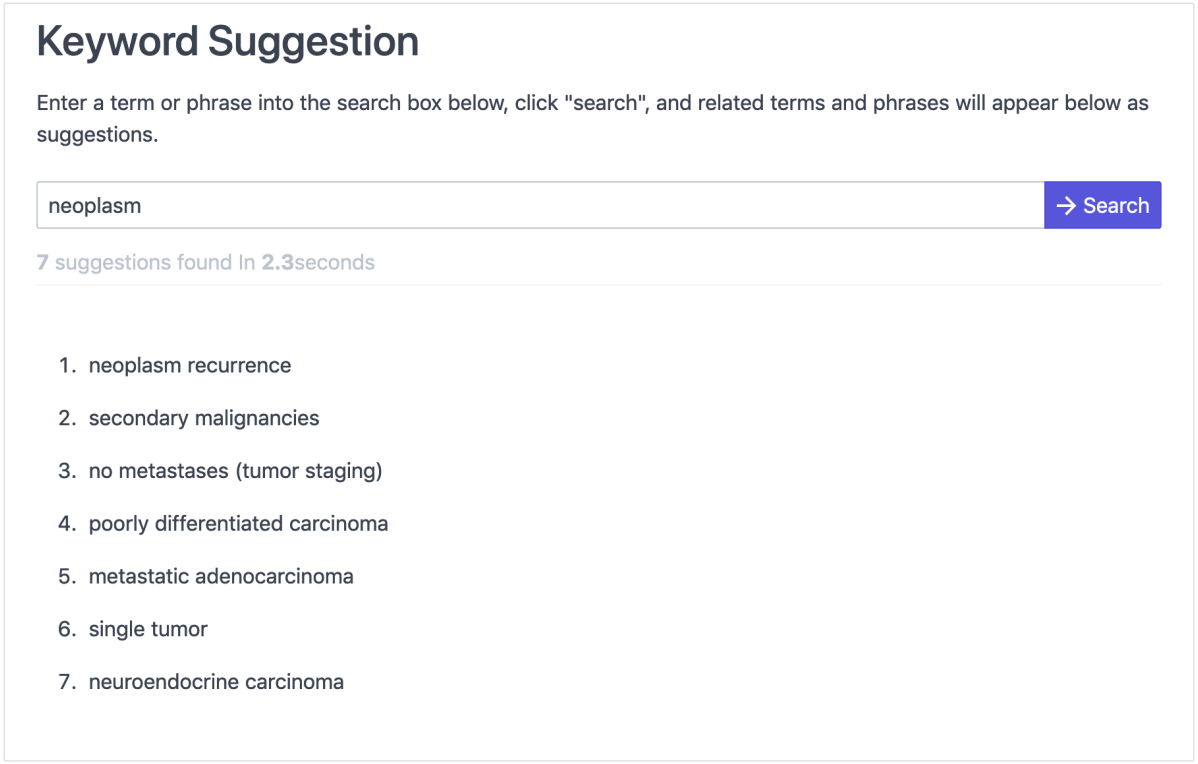


Figure 6.2: Suggesting keywords using the KeywordSuggest tool. Given an input term, suggestions from multiple sources are presented for consideration.

specialists, often requiring domain expertise and small-scale literature searches. KeywordSuggest utilises clinical concept embeddings [234] and PubMed term frequency statistics to produce a set of semantically related terms and phrases to a given input term or phrase. Keywords are ranked and merged into a single list by normalising the scores from each source. This tool supports information specialists by identifying terms that they can add to their query. It works similarly to how an author would use a thesaurus to find a more diverse vocabulary to describe something, where information specialists are interested in choosing terms that comprehensively cover a high-level concept. This tool also integrates into another tool that will be seen in the coming chapters.

The motivation behind this tool was to provide a way for information specialists to identify synonyms for a keyword they may not be familiar with. There are very few tools that allow information specialists to perform this kind of action.

6.1.3 AutoDoc

AutoDoc (Figure 6.3) is the first tool of its kind for supporting the documentation of search strategies. Search strategies are included in systematic review for reproducibility purposes. According to the Cochrane Handbook, the report of the search strategy used should consist of seven elements [38]: (1) databases searched (e.g., PubMed), (2) name of the host (e.g., Ovid), (3) date search was run, (4) years covered by search, (5) complete search strategy, (6) summary of the search strategy, and (7) language restrictions. When reporting search strategies, authors rarely include all seven elements [253]. Several studies [61, 84, 134] find that most reviewers inadequately report their search strategies and Sampson

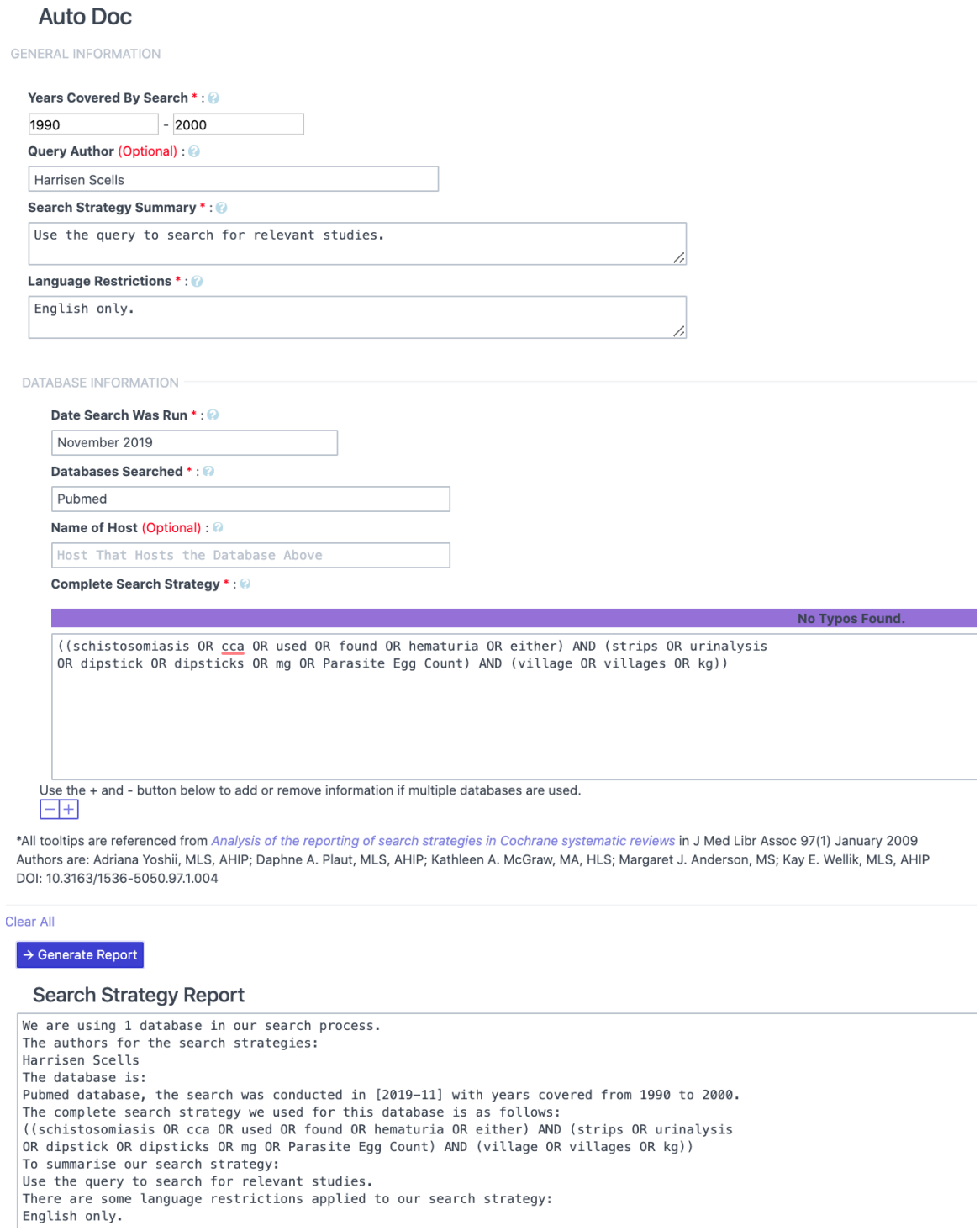


Figure 6.3: Documenting the query using AutoDoc. This tool validates both queries and each of the seven elements that should be reported. Once validated, a report can be generated and can be sent to the authors of the systematic review.

and McGowan [187] find that even well-reported search strategies commonly contain errors such as spelling mistakes and redundant search terms. The Cochrane Handbook has seven recommended elements for reporting search strategies. 65 systematic reviews were analysed by McGraw et al. [134], who found that none included all 7 of the recommended elements. Meanwhile, in a study that contains a larger dataset, Golder et al. [84] reports that only 13 of the included 277 search strategies were found to be reproducible, and another 63 studies did not provide sufficient information on their search strategy. Both McGraw et al. [134] and Rethlefsen et al. [173] agree that when librarians or other 'information professionals' co-author systematic reviews it is linked with higher quality reported search strategies. However, none of these studies offers a framework for designing reproducible search strategies. AutoDoc validates both search strategies and the seven elements that should be reported. Firstly, the tool checks for spelling mistakes and logical errors in the query. Next, it validates the forms that information specialists must fill that cover each of the seven elements that should be reported. After the query and forms have been validated, a report is generated. The generated search strategy report can finally be copied into the relevant section of a systematic review.

The motivation behind the development of this tool was to provide a convenient system for information specialists to document their searches. This tool promotes the best practices for search strategy documentation, and its use may help curb the inconsistencies in reporting that affect the reproducibility of searches for systematic review literature search.

6.2 Case Study

This section describes a small case study for typical use cases of the previously described tools to demonstrate that these query automation tools address gaps in the search strategy development phase. This chapter aims to show how tools support information specialists in developing more effective search strategies.

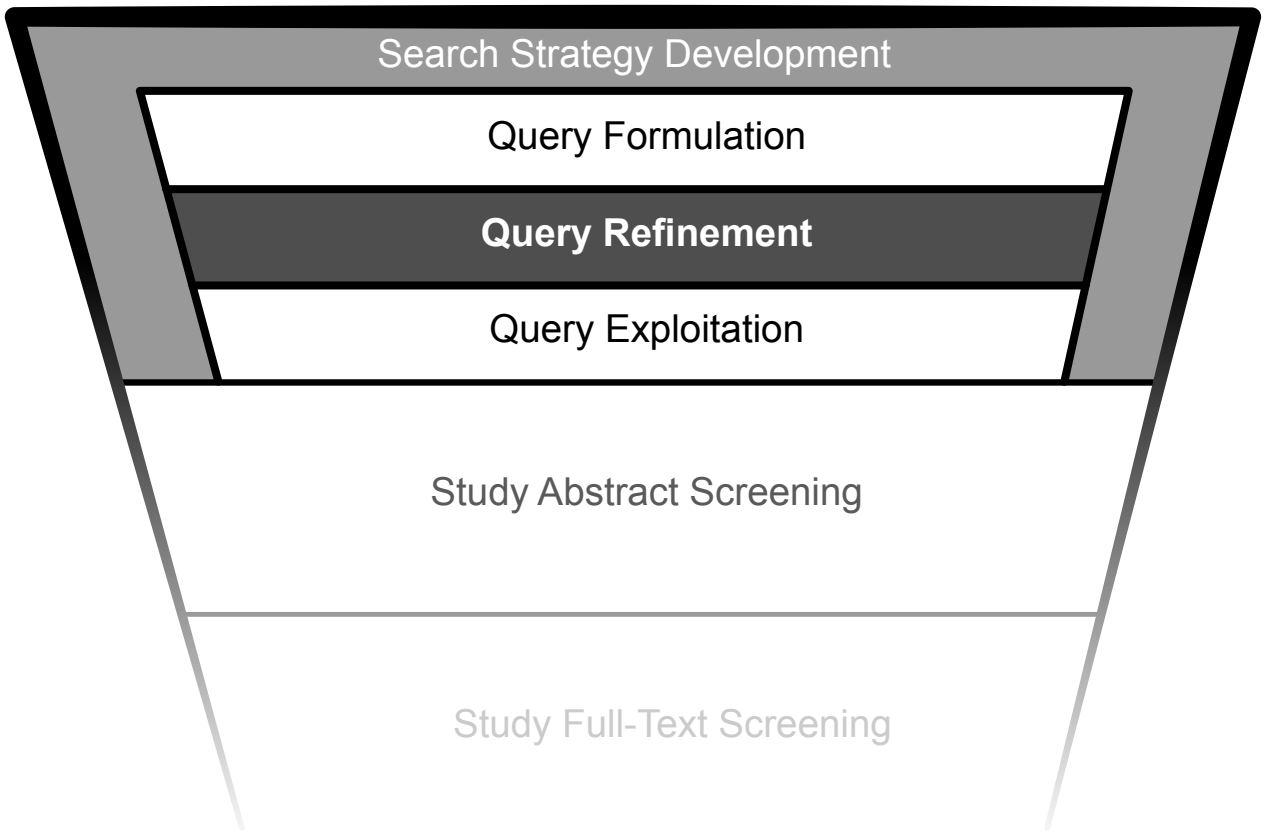
The AutoFormulate tool can be seen as the first step an information specialist may take towards search strategy development. They may use it to develop an initial search strategy for later refinement or use it to get another perspective for alternative ways to formulate a query. As the tool is based in an existing methodology for query formulation (as described earlier in this part), the information specialist can be confident that the presented query is reasonably effective, if not comparable to a manually developed query from scratch.

The KeywordSuggest tool may be used by an information specialist following the conceptual method of query formulation. Here, they must identify synonyms for high-level concepts in their search. This tool eases the identification of such keywords by suggesting semantically related keywords for their search. As keywords are suggested from multiple medical sources, the information specialist can be confident that the suggestions are appropriate.

The AutoDoc tool is highly useful for information specialists documenting their final search strategy. There are numerous inconsistencies in the reporting of search strategies, despite there existing a standard for reporting. This tool addresses the many problems that can arise when a search strategy is ready to be included in a systematic review.

Part 2

Query Refinement



Are the queries developed by information specialists as effective as they can be? What aspects of queries should information specialists change in order to improve their effectiveness? These questions guide the investigation into query refinement techniques that make up Part 2 of this thesis. Query refinement is a crucial task in the search strategy development phase of systematic review creation. It has arguably the most considerable impact on the downstream phases of systematic review creation. An unrefined query is likely to retrieve an excessive number of studies, drastically increasing the amount of time needed in the following study abstract screening phase. Part 2 begins with the background of a proposed theoretical framework to generate and select more effective Boolean queries. This framework is described in detail in Chapter 7. Chapter 7 also introduces several more direct sub-research questions that will be investigated in order to answer the overarching research question of this part of the thesis:

RQ2

Can manually formulated search strategies be automatically refined to produce more effective search strategies?

Chapter 8 presents an empirical evaluation of the previously introduced theoretical framework, specifically the application of the framework to achieve query refinement. Chapter 9 presents a detailed investigation into the sampling strategies to improve further the effectiveness of the selection component of the framework. The part concludes with a final chapter on the demonstration of several tools used for query refinement (Chapter 10):

Tool Demonstration: Query Refinement

Examples of query automation tools for systematic review literature search to support information specialists in refining existng queries into more effective queries.

One of the tools in Chapter 10 is an implementation of the framework that this part of the thesis presents. In addition to a description of these tools, a case study is used to present how tools might be used in the pipeline of search strategy development.

Part 1 focused on the formulation of Boolean queries from scratch. It was found that the automatic formulation of queries could not match the effectiveness of manually formulated queries. However, one finding as a result of Part 1, is that manual refinement could improve the effectiveness of these queries. Part 2 goes one step further to investigate if manually formulated queries are indeed as effective as they can be. The following chapter presents a framework for generating alternative Boolean queries to study whether more effective queries are indeed possible and if more effective queries can be automatically selected.

Chapter 7

Generating Queries with Query Transformation Chain

This chapter describes a framework for the automatic generation of Boolean query variations and the automatic selection of more effective variations than the original query. This framework aims to automatically refine a Boolean query for systematic review literature search into a more effective query. As such, this chapter (and by extension, Part 2) is concerned with addressing **RQ2** of this thesis:

RQ2

Can manually formulated search strategies be automatically refined to produce more effective search strategies?

Retrieval of literature from a medical database for systematic reviews is performed using complex Boolean queries. Boolean queries (and retrieval) are the accepted standard for searching citations when compiling a systematic review [38]. An example of one such query is visible in Figure 7.1. These queries contain several Boolean operators such as `OR`, `AND`, and `NOT`, field restrictions, nesting, and the explosion of terms in an ontology (MeSH). The query in Figure 7.1 is composed of 19 clauses: each represented as a line in the query and may contain one or more operators.

Query formulation is an important step in the definition of the search strategy of a systematic review. However, a poorly formulated query may retrieve only a subset of the relevant citations to the review study or, conversely, may retrieve an extremely large number of citations, while there may only be few relevant citations. In particular, the retrieval of a large number of citations is often a problem for the compilation of systematic reviews because all of the retrieved citations need to be screened for inclusion in the systematic review (akin to performing relevance assessment). This study abstract screening phase, commonly performed by two reviewers, is expensive and time-consuming, often requiring several person-months to complete [108], thus adding to the costs (both monetary and in terms of time) required for the compilation of a systematic review. Previous work has reported that it could take experienced reviewers between 30 seconds and several minutes [243] to screen a

```
1. Diabetic Ketoacidosis/
2. Diabetic Coma/
3. ((hyperglyc?emic or diabet*).tw adj emergenc*.tw.)
4. (diabet*.tw. and (keto* or acidos* or coma).tw.)
5. DKA.tw.
6. or/1-5
7. Insulin Lispro/
8. Insulin Aspart/
9. Insulin, Short-Acting/
10. (glulisine or apidra).tw.
11. (humulin or novolin).tw.
12. (lispro or aspart).tw.
13. (novolog or novorapid).tw.
14. (insulin* adj3 analogue*).tw.
15. acting insulin*.tw.
16. or/7-15
17. 6 and 16
18. (humans/ not exp animals/)
19. 17 and 18
```

Figure 7.1: A typical Boolean query found in a systematic review. Note that the line numbers form part of query: the query is nested by referring to the line numbers, for instance `or/1-5` on line 6 means that lines 1 through 5 should be combined with a Boolean `OR` operator. The fields to search on (`.tw.`), the Medical subject headings (`/`) and their explosion (subsumption [262] – `exp`) are also encoded in the query.

single study (title, abstract and metadata). Some notable examples highlight the effect this has on the timeliness of reviews; for example in Shemilt et al.'s *scoping* review, 1.8 million studies were screened, of which only about 4,000 were found to be potentially eligible [210].

To ensure the queries used by systematic reviews are of high quality (i.e., meet standards for effectiveness and bias before screening citations for inclusion in the review.), reviews often receive the support of information specialists to assist in the query formulation process, and queries are often submitted to an expert panel for review (along with the protocol of the systematic review). However, when formulating queries, little is known about their retrieval performance when applied to answer the questions posed in systematic reviews [3]. Past research has found that only 30% of citations are retrieved using the Boolean queries defined in the protocol of the systematic review [85] (51% of citations were discovered by pursuing references of references, and 24% by personal knowledge or contacts[1]) — a recall problem. On the other hand, past research has also shown that queries may retrieve an overly large set of citations compared to those relevant, as it was for Shemilt et al.'s study where only 0.22% of the retrieved citations were relevant [210] — a precision problem.

In this chapter, we question whether the Boolean queries used in systematic reviews are the most effective possible (e.g., highest recall/precision), or whether more effective queries are possible and how these can be obtained. Given this research direction, four research questions are proposed to guide the investigation.

[1]Percentages add up to more than 100% because the same citations appeared in multiple sources.

RQ2.1

Is it possible to automatically formulate Boolean queries that are more effective than those used initially within search strategies of systematic reviews?

To answer RQ2.1, we devise a set of transformations that can be applied to clauses of Boolean queries for generating alternative valid queries to be issued for retrieval. Transformations are applied at the level of clauses and consist of changing the operators used in the original queries.

RQ2.2

Can alternative, more effective Boolean queries, generated from the original systematic review queries, be automatically selected?

To answer RQ2.2, we cast the problem of formulating Boolean queries that are more effective than the original query, into a machine learning problem: ranking the Boolean queries generated from the original query so that the queries that are predicted to be better than the original are ranked at the top of the suggested alternative queries. This problem is tackled by training a learning to rank model and a k-nearest neighbour approach to query selection.

We define an effective query refinement as one where fewer citations are retrieved, but the number of relevant citations stays the same (or with very minor, tolerable reductions). In Information Retrieval terms, this equates to increasing precision, while maintaining or even improving recall. In the case where recall is improved, note that in the case of systematic reviews, several studies that are used to synthesise results are often found *outside* the search process, either by snowballing references (i.e., finding relevant studies by examining the references of retrieved citations) or having prior knowledge of the existence of relevant studies. Therefore, while the first two research questions relate to the effectiveness of queries, the following research question focuses on the investigation into what aspects result in more effective queries:

RQ2.3

Which types of transformations are the most effective at refining Boolean queries in systematic review literature search?

To answer **RQ2.3**, we analyse the automatically generated and selected queries to determine what kind of transformations are made and whether these transformations positively impact the effectiveness of such a query.

RQ2.4

What impact do unjudged documents have on the effectiveness of refined Boolean queries in systematic review literature search?

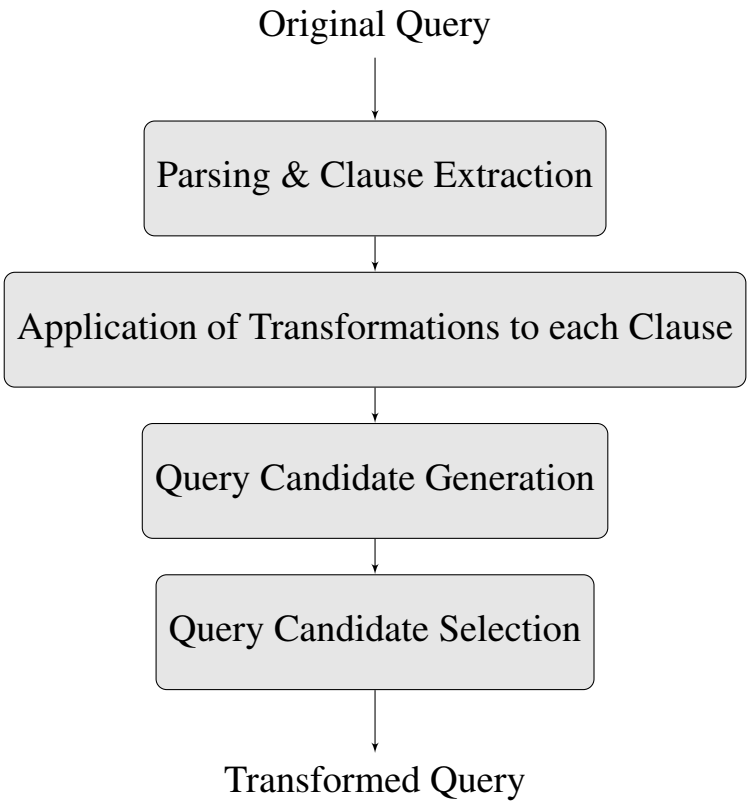


Figure 7.2: Pipeline of the method used for generating transformations of original systematic review queries.

To answer **RQ2.4**, we evaluate by considering the unjudged documents that may be retrieved by the automatically refined queries. We do so by considering two methods for computing which portion of the unjudged citations (i.e. the residual) should be used for computing relevance.

Next, we describe our general method for generating query transformations from the original Boolean query. We then detail the set of considered transformations (Section 7.1), the approach used for query candidate generation (Section 7.3) and those for query candidate selection (Section 7.5), including the features used to represent query candidates for ranking.

Figure 7.2 provides a schematic view of our method to generate alternative Boolean queries. The first step consists of parsing the original Boolean query to extract each clause from the query (remember, a clause is a line of the larger Boolean query, see Figure 7.1). Note that a clause may contain a reference to one or more previous clauses, as it is the case for line 19 in Figure 7.1, which combines the results of the clauses at lines 17 and 18 using the operator `AND`.

Once clauses are extracted, one or more transformations are applied to each clause, and the resulting transformed clauses recorded, along with the original clause. For example, a transformation that changes `AND` into `OR` may be applied to the clause in line 19 of Figure 7.1. Note that multiple transformations may be applied. For example, the clause in line 14 could be transformed by replacing or modifying the `ADJ` operator, and by removing the field restriction (`.tw`) – these will count as two transformations being applied.

Original and transformed clauses are assembled to form a new candidate query (query candidate generation). For each clause, only one among all clause variations (original and transformed versions

of the clause) is selected to form a query; no clause is dropped from or added to a candidate query. This assembly results in each candidate query having the same number of clauses as the original query. The output of this process is a set of candidate queries, which includes the original query.

The next step in the pipeline of Figure 7.2 is the selection of a candidate transformed query (or the original query) to replace the original query. This step is achieved via a candidate selection function. Two candidate selection functions are described later: a learning to rank approach and a k-nearest neighbour approach. These approaches can suggest more than one transformed query; however, each approach is restricted to select only the top-ranked query (i.e. top-$k = 1$) to suggest to the user as better queries.

7.1 Description of Transformations

To rewrite a query, we define a set of transformations ($\mathfrak{T}$) that are possible for all queries. A transformation is a query rewriting strategy that belongs to one of three classes of transformations: *Query Reduction* (i.e., removing elements from the query), *Query Replacement* (i.e. replacing an element of a query with something else), and *Query Expansion* (i.e. adding a new element to a query). The desired effect of these transformations is to increase or reduce the number of citations retrieved, as each type of transformation affects the result set size in different ways. We apply transformations at the query clause level. We do not consider typical query reduction and expansion strategies (e.g., [217, 262]). However, some of the considered transformations are effectively expansions or reductions, e.g., the explosion of MeSH terms.

Next, we describe the family of transformations that we consider in this work (more transformations are possible, but their development and investigation are left for future work). A *family* groups similar transformations together: individual transformations involve specific terms or operators, but the transformations could be represented by the same set of features.

1. **Logical Operator Replacement** The logical operator replacement transformation syntactically replaces the Boolean operators in a query. For example, `(A AND B)`→`(A OR B)`. This transformation only considers `AND` and `OR` operators. This transformation is visualised in Figure 7.3, where the operator depth represents the depth in the logic tree of the Boolean operator that is replaced (depth of root is zero). The transformation `AND` →`OR` will have the likely impact of increasing recall, often to the expense of precision. The transformation `OR` →`AND` is likely to have the opposite effect. We leave the exploration of the `NOT` operator to future work.

2. **Field Restrictions** The field restrictions transformation syntactically modifies the fields an atomic clause is restricted to in Boolean search. For example,

```
(cancer[Title] AND lungs[Title])
                ↓
(cancer[Title/Abstract] AND lungs[Title])
```

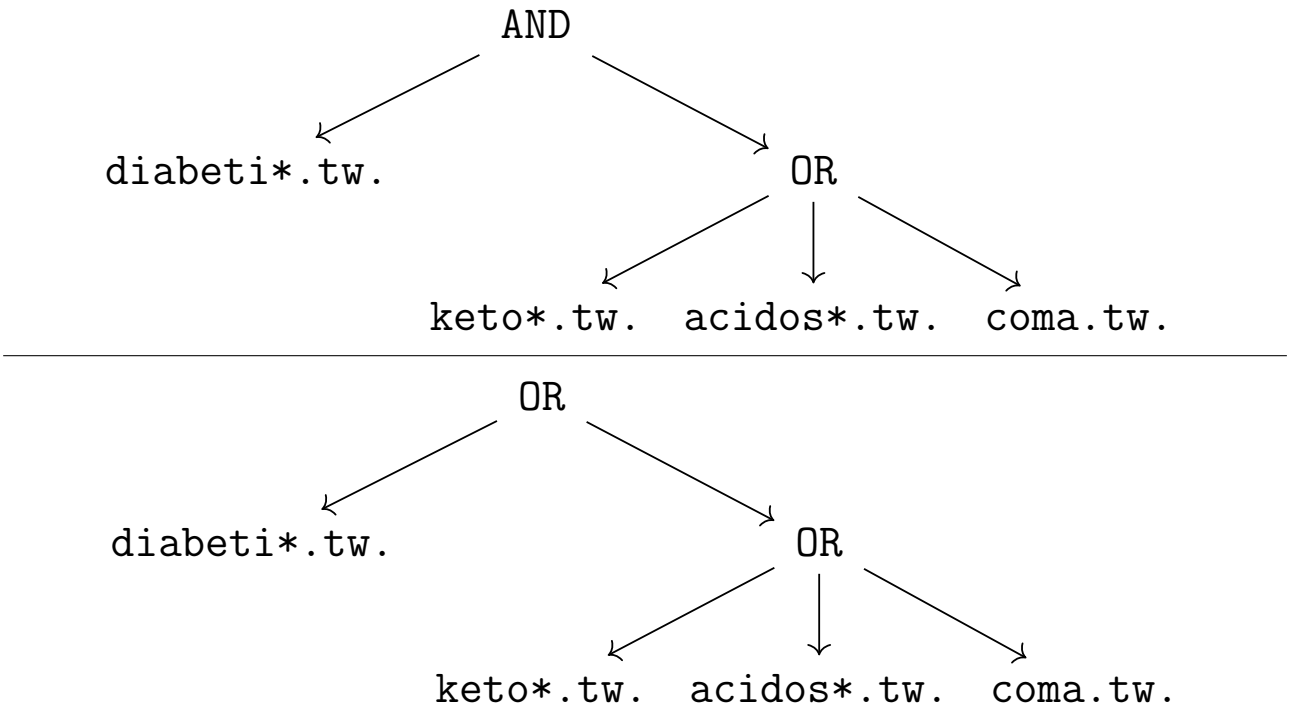


Figure 7.3: A Logical Operator Replacement transformation applied to a clause. This diagram represents the clause `(diabet*.tw. and (keto* or acidos* or coma).tw.)` in tree form. Above: the original clause; below: the transformed clause. The replacement type is `AND` →`OR` and the operator depth is 0.

This transformation produces the following modifications to fields: `title` to `abstract`, `title` to `title & abstract`, `abstract` to `title`, `abstract` to `title & abstract`, `title & abstract` to `title`, and `title & abstract` to `abstract`.

3. **MeSH Explosion** MeSH is a medical ontology organised in a tree structure. Explosion is the subsumption of all nodes underneath a particular term node in the ontology. MeSH explosion can be considered a semantic transformation in that it implicitly adds terms to a query. This transformation toggles if a MeSH term should be exploded or not. Systematic review query languages all support some way for explosion to be toggled, normally in the retrieval model, thus not explicitly modifying the semantics of a query.

4. **MeSH Parents** The MeSH Parents transformation acts in the opposite direction of the explosion transformation: moving up one level in the tree (if possible), thus selecting a broader MeSH term. For example, if the MeSH term is `Skull Neoplasms`, then this transformation selects the parent term `Bone Neoplasms`.

5. **Clause Removal** The clause removal transformation semantically modifies a query by removing a non-atomic clause from a query. This transformation will remove atomic clauses from a non-atomic clause until no more atomic clauses are left (at which point the non-atomic clause that groups them is removed).

6. **Concept Embeddings Expansion** The concept embeddings expansion (CEE) transformation

Transformation	Classification	Applicability
Logical Operator Replacement	Syntactic	Non-Atomic
Field Restrictions	Syntactic	Atomic
MeSH Explosion	Semantic	Atomic
MeSH Parents (new)	Semantic	Atomic
Clause Removal (new)	Semantic	Atomic
cui2vec Expansion (new)	Semantic	Atomic

Table 7.1: Classification (semantic or syntactic) and Applicability (transformation can be applied to atomic or non-atomic clauses) of transformations used in this work.

is a semantic query expansion transformation that uses a pre-trained set of clinical concept embeddings [20] to find related keywords.[2] We first map keywords to concepts using QuickUMLS [217], restricting matches to preferred candidates. Next, we find the top-k similar concepts (which are then used as expansions E) using the clinical concept embeddings via cosine similarity; scores are then normalised using the softmax function. A mapping derived by Jimmy et al. [101] is used to map concepts to their most common string. In our experiments, we expand to a maximum of five candidates. To integrate the new expansions into the Boolean query, we replace the original keyword with a logical `OR` operator that groups the expansions. The attributes (e.g., the fields that the keyword should search on) of the expansions are inherited from the original keyword. The original keyword is included in the expansion terms.

7.2 Application of Transformations

Transformations can be applied to clauses to create variations of a clause. These variations are generated on the basis that a transformation can be applied to a clause. Each transformation (e.g., the Logical Operator Replacement) may produce 0 or more candidate clauses. Formally, the set of transformations that can be applied to a clause, denoted as $\mathcal{T}'_c$, is defined as:

$$\mathcal{T}'_c = \{\forall \tau \in \mathcal{T} | a(\tau, c) = 1\}$$

The applicability function a is a Boolean function that determines if an individual transformation τ can be applied to the clause c:

$$a(\tau) = \begin{cases} 1, & \text{if } \tau \text{ is applicable to clause } c \\ 0, & \text{otherwise} \end{cases}$$

The applicability of τ to a clause is determined by aspects of the clause, i.e. a transformation cannot be made if the clause does not contain criteria for the transformation to be applied. For example, if the clause does not contain any `ADJ` operators, then the Adjacency Replacement transformation cannot be applied. Clause variations $c_{\tau_1}, c_{\tau_2}, c_{\tau_3} \ldots c_{\tau_m}$, denoted C', are the application of the transformations in $\mathcal{T}'_c$ to the original clause c.

[2] As the name suggests, the embeddings are for clinical *concepts* (rather than terms), identified as **Concept Unique Identifiers** (CUI) from the UMLS Metathesaurus. A single CUI may represent different names or *strings* in different source vocabularies of the UMLS Metathesaurus.

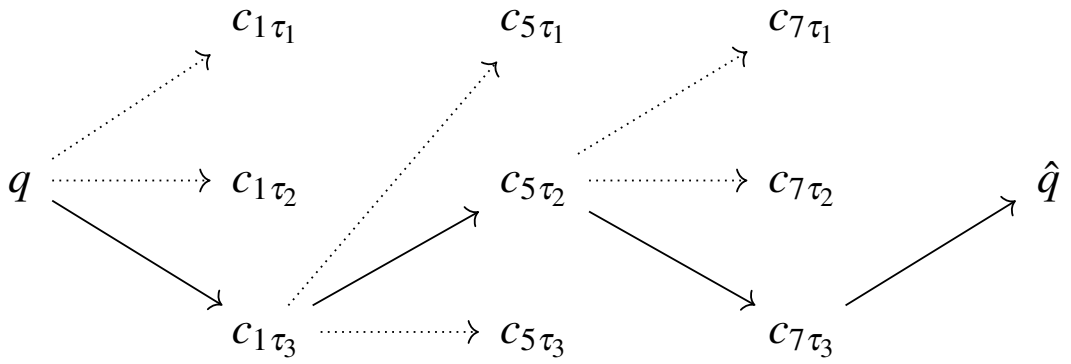


Figure 7.4: Query transformation chain with three transformations (τ_1, τ_2, and τ_3) being applied to three clauses (c_i) of a query (q) to produce the final rewritten query ($\hat{q}$). Lines between transformations represent possible transformations. Solid lines indicate the path followed (possible transformations for paths that have not been taken are not shown).

7.3 Query Candidate Generation

Candidate query generation is the process of assembling clauses to form a Boolean query. For each original clause in the Boolean query, the original clause or one of its variations is selected to form a candidate query. Figure 7.4 exemplifies the transformation of an initial Boolean query q into the transformed candidate query $\hat{q}$ through the selection of transformation τ_3 applied to clause c_1, transformation τ_2 applied to clause c_5, and the application of transformation τ_3 again, but applied to clause c_7.

The result of the candidate query generation process for an original Boolean query q is a set of queries $\hat{\mathbb{Q}}_q$ that includes the query q and all possible query transformations. The size of the set of candidate queries depends on the number of clauses and the number of transformations that can be generated for each clause. Assuming that a query q contains n clauses and for each clause, a total of maximum m variations (including the original) can be created, then the total number of candidate queries that can be generated for the original query q is $|\hat{\mathbb{Q}}_q| \leq m^n$. Note that this is an upper limit on the number of variations; often, different clauses may allow for a different number of variations. While numerous transformations could be applied to some clauses, other clauses may only allow a restricted set of transformations.[3] Because of the high number of possible candidate queries generated, we shall employ a greedy approach to reduce the search space of query candidates in our empirical evaluation. This greedy approach is also used for query candidate selection and is described next among other query candidate selection approaches.

7.4 Query Candidate Features

The ranking of query candidates requires discriminative features in order to select the most effective query. We use 39 features in total, described in Table 7.2. Two notable features which we consider to be novel are Δ, which computes the difference between measurements indicated in Table 7.2 with †, and ω_i which represents the ith transformation applied to a query in the query chain process. For

[3]The exact number of query candidates can be computed as $\prod_{i=1}^{n} m_i$, where m_i is the exact number of variations that can be produced for a specific clause i.

performance and effectiveness reasons, we omit general-purpose QPPs (Query Performance Predictors) as in similar work [112]. This is because: (i) most QPP measurements rely on costly retrieval statistics which becomes prohibitive with the number of queries that are generated; and (ii) previous work has shown that QPPs are often not correlated with retrieval effectiveness [88, 190] and thus may hamper the quality of ranking.

7.5 Query Candidate Selection

Query candidate selection is the process by which the next candidate to transition to in the query chain is chosen. The formalisation of candidate selection is described as follows. Following the generation of candidate queries $\hat{\mathbb{Q}}_q$ from the initial query q, the next candidate q^* is selected as the next transition in the query chain via the maximisation of function $f(\hat{q})$ where:

$$q^* = \underset{\hat{q} \in \hat{\mathbb{Q}}_q}{\text{argmax}} f(\hat{q}) \tag{7.1}$$

The candidate generation and selection process is repeated until the stopping criteria is met. Here, we define the stopping criteria with two objectives: (i) the query chain reaches a depth of five, and (ii) the number of retrieved citations for a candidate query is zero.

When generating queries for training purposes, we only considered the stopping criteria when the depth of a chain reached the maximum length of five. We did this so that we would have sufficient negative training examples. In practice, this resulted in a severely imbalanced training set with the majority of examples being negative and only a small portion positive. In training models to maximise the candidate selection objective function, we discovered empirically that removing the majority of negative examples (i.e., rebalancing the positive and negative examples via negative sub-sampling) increased the effectiveness of ranking.

The maximisation function $f(\hat{q})$ can be learnt as a machine learning problem. In this work, we learn $f(\hat{q})$ as both a learning to rank task (similarly as in [192]) and a nearest neighbour task (new to this work). For both tasks, we train a different model for each evaluation measure that we seek to optimise. For the learning to rank task, we use DART: an ensemble of boosted regression trees which incorporates dropouts [170]. In our empirical testing, we found this method to be the best for this task compared to other state-of-the-art techniques such as LambdaMART. For the nearest neighbour task, we create a model that records the most effective query in each topic (given an evaluation measure), and the distance to and reduction in effectiveness for all other queries in that topic. For an unseen set of candidate queries, each target query in the set is ranked by minimising the distance and maximising the score of the queries in the model mentioned above.

To determine an upper bound on the effectiveness of the query transformations, we devise an oracle approach that selects query candidates based on ground truth information (relevance assessments). In particular, we set $f(\hat{q}) = E(\hat{q})$, where $E(\hat{q})$ represents the score assigned to the set of citations retrieved by candidate query $\hat{q}$ by evaluation measure E, e.g. E may be precision or recall. Thus, the oracle candidate selection function selects the query $q^* \in \hat{\mathbb{Q}}_q$ that retrieves the set of citations that maximises evaluation measure E. Ties between queries are chosen randomly.

Feature	Description	Applicability
Depth	How deep in the query the transformation was made.	Both
Clause Type	The type of clause the transformation was applied to (atomic/non-atomic).	Both
Children Count	Number of sub-clauses a non-atomic clause has.	Non-Atomic
Transformation Type	The type of transformation (Table 7.1) made.	Both
Logical Operator Replacement Type	The type of operator this type of transformation modified.	Non-Atomic
MeSH Depth	The depth that the MeSH term was exploded from in the MeSH ontology.	Atomic
MeSH Parent	Whether a MeSH parent transformation is made.	Atomic
Restriction Type	Which type of field restriction was made when this transformation was applied.	Atomic
Clause Removal	Whether a clause removal transformation has been made.	Atomic
Concept Embedding	Whether a cui2vec concept embedding expansion has been made.	Atomic
Concept Embedding Expansions	The number of expansions that were made when this transformation was applied.	Atomic
MeSH Exploded	If the MeSH heading in an atomic clause is exploded or not.	Atomic
Truncated	If an atomic clause has a character replacement expression (e.g., wildcard).	Atomic
Retrieval Size†	Total number of citations retrieved by the query.	Both
# Boolean Clauses†	Total number of Boolean clauses in the query.	Non-Atomic
# Boolean Keywords†	Total number of atomic clauses (keywords) in the query.	Non-Atomic
# Truncated†	Total number of atomic clauses that have a truncated term.	Non-Atomic
# Fields†	Total number of fields in the query.	Both
# MeSH Keywords†	Total number of non-atomic clauses that contain a MeSH term.	Non-Atomic
# MeSH Exploded†	Total number of non-atomic clauses that have an exploded MeSH term.	Non-Atomic
# MeSH Non-Exploded†	Total number of non-atomic clauses that do not have an exploded MeSH term.	Non-Atomic
Average MeSH Depth†	The average depth MeSH terms in the query appear.	Non-Atomic
Maximum MeSH Depth†	The maximum depth MeSH terms in the query appear.	Non-Atomic
Δ	The difference of a feature between q and $\hat{q}$, where applicable (†).	Both
ω_i	n additional features that correspond to the sequence of transformations made.	Both

Table 7.2: Features used in the candidate selection task. Applicability denotes to which types of clauses a feature applies to: *Atomic* refers to clauses which consist of a single keyword, *Non-Atomic* refers to clauses which group several atomic clauses with a logical operator, and *Both* indicates that the feature can be applied to either type of clause.

7.6 Sampling Query Candidates

In generating training data for automatic query candidate selection methods, we encountered an exponential growth in queries generated. This growth is computationally undesirable: for example if on average 100 candidates are generated for a query, and the total length of a chain is five then a total of 10^9 queries would be generated (n^d, where n is the number of candidates and d is the depth). Additionally, if it takes 30 seconds to generate a set of queries and features on average, then for that query, it would take over 950 years to complete. Thus, we sample queries to cut the total number of queries included for training. Note that the $O(n^d)$ complexity does not affect testing as we are not exploring the space of transformations for a query. Instead, we are ranking candidates, selecting and generating transformations for the top-1. Our sampling approach is as follows: for a seed query q, we generate a set of candidate queries $\hat{\mathbb{Q}}_q$; from this set, we randomly sample a minimum of 20 candidate queries, and if there are more than 20 queries generated, we sample a further 10%. We then cut further by removing any identical queries that have already been processed. Finally, of the queries generated from the previous step, we randomly sample 10% of these new queries and repeat the process. A retrospective study on the effect sampling has on the effectiveness of candidate selection functions is included in Chapter 9.

7.7 Evaluation

We evaluate a standard ad-hoc search task. Relevance assessments are obtained from the studies that are reported in each systematic review as *included* (relevant), and *excluded* (non-relevant). Queries obtained through query refinement may retrieve citations that were not assessed by the reviews (i.e., unjudged); thus, relevance assessments are incomplete. At the first stage, we assume all citations which were not judged as non-relevant. Remember, however, that unlike in traditional ad-hoc information retrieval evaluation, no pooling using a large variety of systems was executed: only one query and one system contributed to the relevance assessments — other, non-retrieved but relevant citations may very well exist. Thus, later in the analysis of the results, we use two heuristics to bound the effect of unjudged documents on the effectiveness of the studied methods. To this aim, we use and further extend the intuition of residual analysis described for the rank biased precision measure [143]. These heuristics are defined at the end of Subsection 2.3.1 in Chapter 2 (Background and Literature Review).

7.8 Summary

This chapter presented the Query Transformation Chain (QTC) framework. This framework is capable of generating and selecting variations of a given Boolean query for query refinement. However, this chapter did not evaluate the effectiveness of such a framework. The following chapter seeks to do just this by describing how instantiations of the framework can be made and analysing the modifications made to queries to determine if the QTC framework can refine existing high-quality, expertly developed queries into more effective ones.

Chapter 8

Evaluation of Query Transformation Chain

This chapter presents an empirical evaluation of the query transformation chain framework. The previous chapter was an introduction and description of the framework. Here, implementations of the framework are evaluated in order to address **RQ2**:

RQ2

Can manually formulated search strategies be automatically refined to produce more effective search strategies?

This chapter begins with a description of the experimental settings chosen to perform the experiments, with each experiment addressing one of the four sub-research questions presented in Chapter 7.

8.1 Experimental Setup

Experiments were performed on a test collection of 125 systematic reviews [205]. The CLEF TAR collections [104] were not used because these collections contain a smaller number of topics and therefore, less training data. To perform the experiments, we used QueryLab to construct pipelines for each stage [191]. Queries are executed directly on Pubmed using the Entrez API [189].

8.1.1 Query Candidate Generation

In order to train a model to select candidate queries, training data is required. We achieve this by exploring the space of queries using the transformation processes described in the previous chapter up to a depth of 5. However, the difference here is that rather than choosing one query as a candidate query, all queries are chosen. Rather than choosing one query to transition to in the chain, we explore all possible transitions to generate training data. Because this process is exponential, we sample using the method described in Section 7.6. In terms of training statistics, on average, 7968 queries were generated from a seed query. The highest number of queries generated for a topic was 129,282, and the lowest was 70. The total number of training data points was 717,160.

8.1.2 Query Candidate Selection

The topics are split into approximately 70% train (90 topics) and 30% test (35 topics). In the training process, 30% of the train set is used for validation. We train two methods to perform automatic candidate selection: a learning to rank model and a nearest neighbour approach. Both approaches use vectors of features to predict more effective queries. A feature vector is constructed from the features described in Section 7.4. The feature vectors are normalised column-wise using min-max normalisation, which scales each feature between 0 and 1. The queries that are refined by the candidate selection process are evaluated with six Information Retrieval measures (precision, recall, $F_{0.5}$, F_1, F_3 and work saved over sampling (WSS)). These measures are typically used to evaluate retrieval in the case of systematic reviews [104, 153, 205]. As the candidate selection function seeks to maximise an evaluation measure, a model is trained for each of the evaluation measures considered in this work for both the learning to rank and nearest neighbour candidate selectors, for a total of 12 models.

8.1.3 Learning to Rank

The DART learning to rank model is trained within the QuickRank framework [34]. We parametrise the DART model with a tree size of 100, a drop-out rate of 0.8, with uniform sampling, tree normalisation, a learning rate (shrinkage) of 0.1, and to optimise the ranking of queries using DCG@1.[1] We perform candidate selection using this model with the same ranking effectiveness metric (DCG@1). The choice of DCG@1 is motivated by identifying the top-1 ranked candidate query to transition to in a query chain. We sample the training data by removing a large portion of negative examples (two-thirds) to train a DART learning to rank model, as we discovered empirically that negative examples degrade the ranking quality.

8.1.4 Nearest Neighbour

Our nearest neighbour approach first records the feature vectors and evaluation measure values for each query in the training set. For a new unseen query, an evaluation measure value is approximated by searching for a query in the training set which minimises the distances between the two feature vectors and maximises the recorded evaluation measure score. The intuition here is to find the most effective query from the training set which is most similar to an unseen query. To find the top-1 candidate to continue the query chain, each generated candidate query is ranked by the approximated evaluation measure value described previously. We only store the deltas of distance and score from the most effective query generated from each seed query for efficiency reasons. The entire training data is used to create a nearest neighbour model, as we empirically discovered that unlike learning to rank, negative examples improved the ranking quality with the nearest neighbour approach.

[1] The exact parameters used to train the model can be found in the experiment pipeline files. These parameters were chosen based on a combination of default values and early experimental testing.

8.2 Results

8.2.1 Are Better Queries Possible?

RQ2.1

Is it possible to automatically formulate Boolean queries that are more effective than those used initially within search strategies of systematic reviews?

To answer **RQ2.1**, we explored the effectiveness of query transformations generated from the original query and compared their effectiveness. Given the exponential number of transformed queries possible to generate, we used a random sampling approach to generate candidate queries and an oracle for selecting candidates. The results are reported in Table 8.1, where * refers to statistically significant differences between the retrieval effectiveness of the queries obtained with the considered method and the original Boolean query (*Baseline*). Statistical significance was computed using a two-tailed t-test and $p < 0.01$. In answer to our first research question, we found that it *was* possible to generate Boolean queries via the transformations illustrated in this work that provided higher effectiveness than the original Boolean queries used in the (considered high-quality) systematic reviews. The oracle selectors consistently outperformed the baseline, with most improvements being statistically significant. This result highlights how significant the increase in target effectiveness can be if the right query is selected. Also, note that the oracle may not have selected the globally optimal transformed queries, as only a small subset of queries generated during the query generation process were considered.

	Precision	Recall	$F_{0.5}$	F_1	F_3	WSS
Baseline	0.0211	0.8395	0.0255	0.0373	0.1107	0.8386
Oracle$_p$	**0.3004***	0.3182	0.1596*	0.1828*	0.1905*	0.3181
Oracle$_r$	0.0000*	**0.9783***	0.0000*	0.0000*	0.0002*	0.7593*
Oracle$_{F_{0.5}}$	0.2326*	0.3970	**0.1940***	0.1977*	0.2513*	0.3969
Oracle$_{F_1}$	0.2685*	0.3632*	0.1884*	**0.2028***	0.2299*	0.3630
Oracle$_{F_3}$	0.1521	0.5453*	0.1752*	0.1533*	**0.2990***	0.5452
Oracle$_{WSS}$	0.0113*	0.9596	0.0216*	0.0140*	0.0811*	**0.9444**

Table 8.1: Effectiveness of Baseline (original query) and candidate queries chosen using the oracle candidate selector after five iterations of variation generation and selection. The measure optimised by the oracle is denoted with $_p$ (precision), $_r$ (recall), $_{F_{0.5}}$, $_{F_1}$, $_{F_3}$ (F_β measures), and $_{WSS}$ (WSS). Values in bold denote the best value obtained for that evaluation measure. * indicate statistically significant differences with the baseline (two-tailed paired t-test, $p < 0.01$).

	P	P_{mle}	P_r	R	R_{mle}	R_r	F0.5	$F0.5_{mle}$	$F0.5_r$	F1	$F1_{mle}$	$F1_r$	F3	$F3_{mle}$	$F3_r$	WSS	WSS_{mle}	WSS_r
Original	0.0211	0.0307	0.7344	0.8395	0.8635	0.9831	0.0255	0.0358	0.7353	0.0373	0.0493	0.7366	0.1107	0.1331	0.7384	0.8386	0.8626	0.9822
P_l	0.0068	0.0142	0.7100	0.6989	0.7406	0.9591	0.0084	0.0161	0.7104	0.0129	0.0214	0.7109	0.0468	0.0610	0.7117	0.6849	0.7267	0.9450
P_n	0.0207	0.0432	0.5909	0.6403*	0.6810*	0.8541	0.0248	0.0472	0.5908	0.0355	0.0591	0.5908	0.0977	0.1331	0.5912	0.6396*	0.6802*	0.8533
R_l	0.0115	0.0195	0.7948	0.8833	0.8978	0.9580	0.0139	0.0222	0.7954	0.0201	0.0292	0.7963	0.0573	0.0716	0.7976	0.7946	0.8091	0.8694*
R_n	0.0023	0.0088	**0.7985**	**0.8939**	**0.9111**	0.9865	0.0029	0.0094	**0.7986**	0.0045	0.0113	**0.7988**	0.0183*	0.0270*	0.7990	**0.8551**	**0.8723**	0.9476
$F0.5_l$	0.0113	0.0237	0.7083	0.7826	0.8046	0.9513	0.0139	0.0254	0.7081	0.0213	0.0333	0.7079	0.0793	0.1011	0.7079	0.7802	0.8022	0.9489
$F0.5_n$	0.0191	0.0225	0.8206	0.8245	0.8355	0.9690	0.0231	0.0271	0.8215	0.0336	0.0393	0.8228	0.0965	0.1118	0.8245	0.8125	0.8235	0.9570
$F1_l$	0.0206	0.0229	0.7623	0.8330	0.8503	0.9787	0.0248	0.0275	0.7631	0.0360	0.0396	0.7644	0.1007	0.1090	0.7661	0.8224	0.8397	0.9681
$F1_n$	**0.0416**	**0.0638**	0.6474	0.6726*	0.7103*	0.9188	0.0486	**0.0604**	0.6454	**0.0655**	**0.0795**	0.6430	**0.1472**	**0.1772**	0.6408	0.6719*	0.7095*	0.9181
$F3_l$	0.0344	0.0248	0.6501	0.6860	0.7104*	0.9129	**0.0355**	0.0297	0.6497	0.0436	0.0424	0.6493	0.1041	0.1147	0.6490	0.6847	0.7091*	0.9116
$F3_n$	0.0175	0.0312	0.7279	0.7850	0.8195	0.9583	0.0212	0.0339	0.7191	0.0308	0.0448	0.7155	0.0910	0.1173	0.7144	0.7838	0.8183	0.9570
WSS_l	0.0152	0.0199	0.7666	0.7590	0.7961	0.9881	0.0181	0.0237	0.7665	0.0257	0.0335	0.7663	0.0634	0.0812	**0.7661**	0.7471	0.7842	0.9762
WSS_n	0.0229	0.0277	0.7323	0.7955	0.8177	**0.9867**	0.0275	0.0331	0.7329	0.0397	0.0474	0.7337	0.1131	0.1326	0.7349	0.7916	0.8138	**0.9828**

Table 8.2: Comparison of the effectiveness of the original queries in the test set (Seed) and each of the candidate selectors. Reported are the average values across topics. Values in bold indicate the highest value in that column. Statistical significance (two-tailed t-test where $p < 0.01$) is indicated with $*$. Evaluation methods are abbreviated in the following manner: P→precision, R→recall, WSS→work saved over sampling. Residual evaluation can be identified with the $_r$ (optimistic) and $_{mle}$ (maximum likelihood) subscripts. The two candidate selection functions can be identified by the evaluation measure they optimise, and a corresponding letter: e.g., P_l corresponds to the learning to rank selector which maximised precision and $F1_n$ corresponds to the nearest neighbour selector which maximised F1.

8.2.2 Can Better Queries be Automatically Selected?

RQ2.2

Can alternative, more effective Boolean queries, generated from the original systematic review queries, be automatically selected?

To answer **RQ2.2**, we explored the effectiveness of a learning to rank approach (indicated by the subscript l) and a nearest neighbour approach (indicated by the subscript n). The results are reported in Table 8.2. Along with the results obtained when unjudged citations were considered not relevant, we also report results obtained by considering the optimistic and probabilistic (maximum likelihood estimation) residuals: these are indicated by the subscripts r and mle, respectively.

Overall, the results in Table 8.2 indicate that the automatic candidate selection methods identified refined queries that were better than the original query (Baseline), for the chosen evaluation measures. Somewhat surprisingly for this task, the nearest neighbour candidate selector often outperformed the learning to rank selector, choosing candidate queries that more often refined the query, rather than degrading or broadening it. Notably, the nearest neighbour candidate selector that was trained to maximise the F_1 measure (indicated as $F1_n$ in Table 8.2) was the most effective at selecting queries that improve over the original seed queries in terms of precision, F_1, and F_3.

Next, we explore at a topic level the differences between candidate selectors and their effects on evaluation. Figures 8.1 and 8.2 present the topic-by-topic effectiveness of the selectors with respect to the target evaluation measures. The differences between the precision candidate selectors (i.e. the learning to rank selector and the nearest neighbour selector) are noticeable. The P_l selector chose many queries which degraded the effectiveness in terms of precision. On the other hand, the P_n selector had more considerable gains for a similar number of topics, but the P_l selector shows both a reduced number of queries for which a loss was recorded and a reduced amount of loss when losses happened.

On the other hand, the R_l and R_n selectors were more similar, and many of the same topics had improved effectiveness. The results of these figures reflect the intuition that refining a query to increase precision is more difficult than improving recall. To contrast, the previous two figures, Figure 8.3 presents the effect the $F1_n$ selector had on each topic in terms of the six evaluation measures considered in this work. While many queries had improved precision and maintained or even increased recall (e.g., topic 83), approximately one-third of queries had a reduction in recall. This finding suggests that there may not have been enough training examples for this selector to choose a candidate query that is likely to cause an increase in recall.

Finally, we compare two example queries by examining the transformations that have been applied to them. The queries chosen for comparison are from topics 73 and 154. These topics were respectively the best and worst-performing ones when query transformations were obtained through the $F1_n$ candidate selector (in the case of topic 154 it was chosen because the query for 106 was too long to fit into a column). The transformed query for topic 73 obtained a large increase in precision, while maintaining recall, while the transformed query for topic 154 obtained a minor decrease in precision and a large decrease in recall.

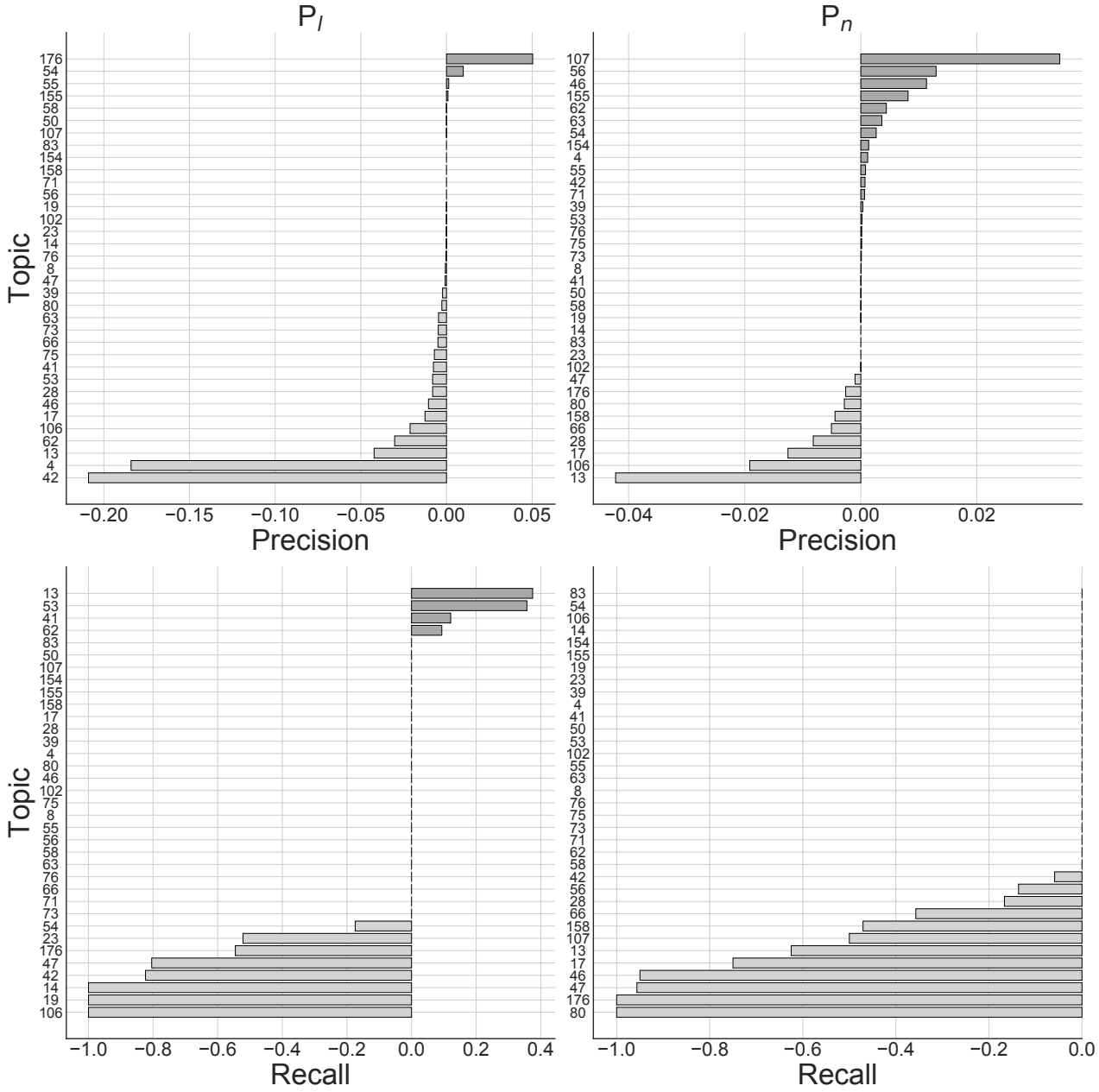


Figure 8.1: Per-topic gains in precision (top) and recall (bottom) over the Seed query for the candidate selectors which maximised precision (P_l and P_n).

The original and the transformed query for topic 73 are shown in Figure 8.5. The transformations made to the query were that lines 59 and 75 were both removed, and lines 40, 46 and 75 (which become line 66 in the transformed query) had their operators changed (from `OR` to `AND`). The original and the transformed query for topic 154 are shown in Figure 8.4. The transformations made to its query were that lines 1, 3, and 27 had their fields changed from Title and Abstract to only Title, and the MeSH term of line 7 was expanded to its parent.

Upon examination of the queries in Figures 8.5 and 8.4, it is clear that they considerably differ in length: one may question whether the length difference (in the number of clauses) is associated with the effectiveness difference — according to this hypothesis shorter queries would generally perform worse than longer queries. However, upon further analysis, we found that this is not the case. The

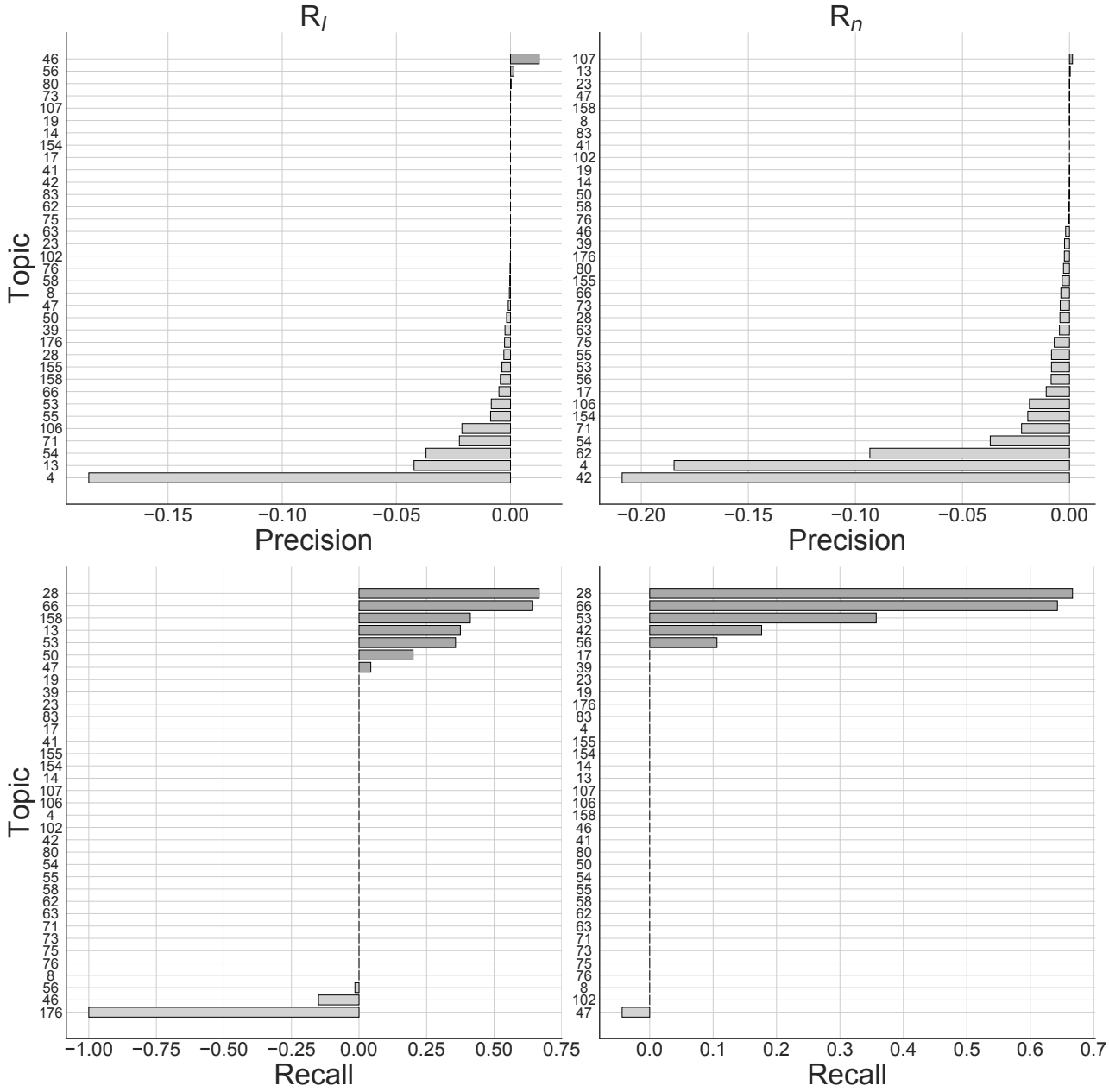


Figure 8.2: Per-topic gains in precision (top) and recall (bottom) over the Seed query for the candidate selectors which maximised recall (R_l and R_n).

Pearson's r correlation between the F_1 evaluation measure and the number of Boolean clauses in each query was only $r = 0.175$. Similar results were obtained when considering the other evaluation measures used in this study, thus suggesting that there is no direct relation between query length in terms of a number of Boolean clauses and retrieval effectiveness. This finding is similar to what we found when studying query performance predictors for this domain [190].

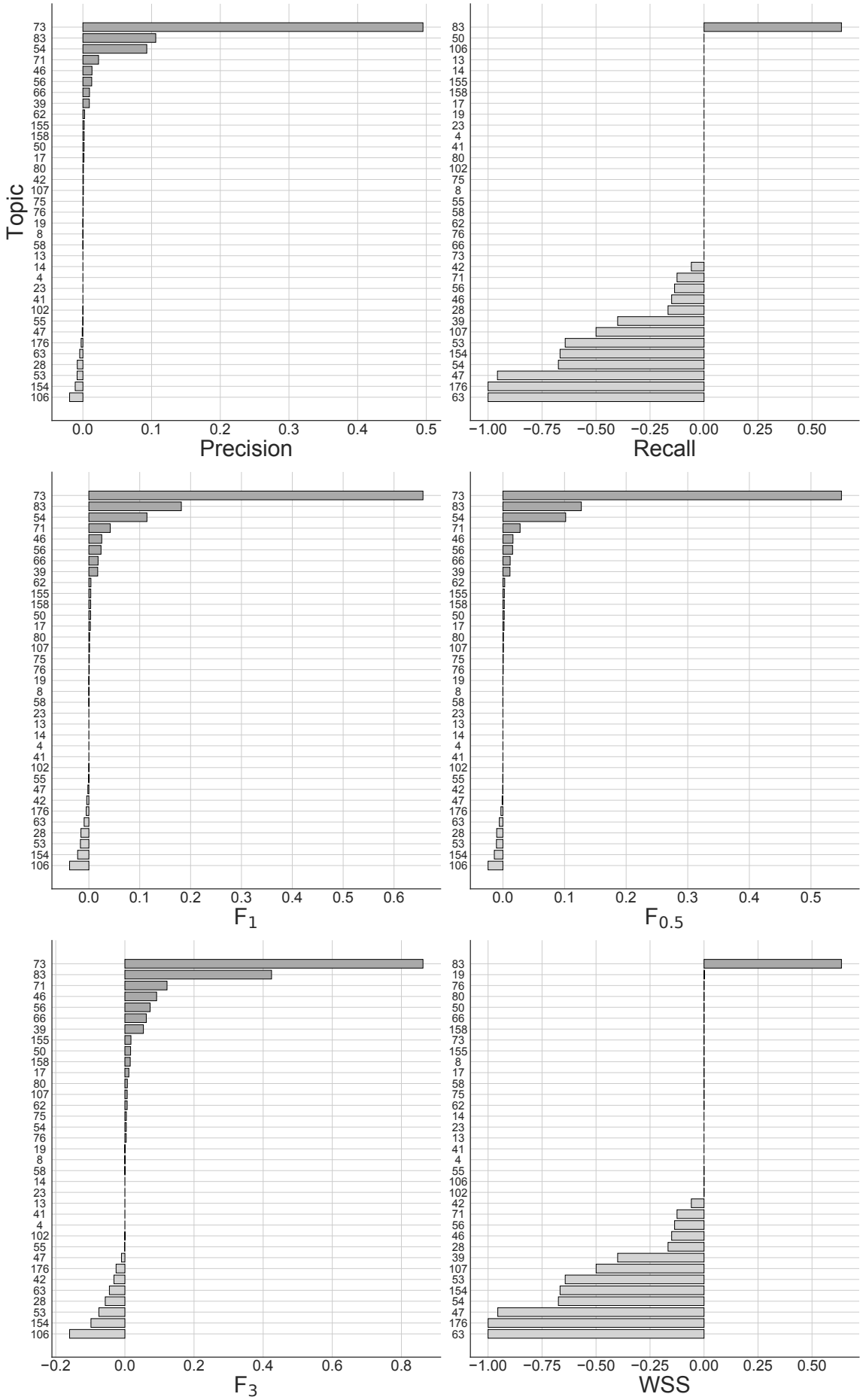


Figure 8.3: Per-topic gains over the Seed query in the evaluation measures considered in our study. Here we considered the nearest neighbour candidate selector that maximised F1 ($F1_n$.)

(a) Original query.

```
1. measles.tw.
2. rubeola.tw.
3. morbilli*.tw.
4. exp Measles/
5. exp Measles virus/
6. or/1-5
7. exp Vitamin A/
8. vitamin a.ti,ab,sh.
9. retinol.ti,ab,sh.
10. exp Dietary Supplements/
11. or/7-10
12. bitot*.tw.
13. spot*.tw.
14. 12 and 13
15. vision*.tw.
16. visual*.tw.
17. eye*.tw.
18. sight*.tw.
19. or/15-18
20. xerosis*.tw.
21. exp Vision Disorders/
22. Xerophthalmia/
23. Night Blindness/
24. xerophthalmia*.tw.
25. exp Blindness/
26. keratomalacia.tw.
27. blind*.tw.
28. 14 or 19 or 20 or 21 or 22 or
23 or 24 or 25 or 26 or 27
29. 6 and 11 and 28
```

(b) Transformed query.

```
1. measles.ti.
2. rubeola.tw.
3. morbilli*.ti.
4. exp Measles/
5. exp Measles virus/
6. or/1-5
7. Retinoids/
8. vitamin a.ti,ab,sh.
9. retinol.ti,ab,sh.
10. exp Dietary Supplements/
11. or/7-10
12. bitot*.tw.
13. spot*.tw.
14. 12 and 13
15. vision*.tw.
16. visual*.tw.
17. eye*.tw.
18. sight*.ab.
19. or/15-18
20. xerosis*.tw.
21. exp Vision Disorders/
22. Xerophthalmia/
23. Night Blindness/
24. xerophthalmia*.tw.
25. exp Blindness/
26. keratomalacia.tw.
27. blind*.ti.
28. 14 or 19 or 20 or 21 or 22 or
23 or 24 or 25 or 26 or 27
29. 6 and 11 and 28
```

Figure 8.4: Comparison between the original query (left) and a transformed query (right) for topic 154. The transformed query was chosen using the $F1_n$ selector, and was the least effectively refined query. Note that on line 28 for both queries there is a carriage return as the line was too long.

(a) Original query.

```
1. controlled clinical trial.pt.
2. placebo.ab.
3. clinical trials as topic/
4. randomly.ab.
5. trial.ti.
6. randomized.tw.
7. randomized controlled trial.pt.
8. or/1-7
9. humans/
10. animals/
11. 9 not 10
12. 8 and 11
13. abdominal.tw.
14. abdomen.tw.
15. chest.tw.
16. thoracic.tw.
17. trunk.tw.
18. or/13-17
19. exp Wounds, Penetrating/
20. 18 and 19
21. abdominal.tw.
22. abdomen.tw.
23. chest.tw.
24. thoracic.tw.
25. trunk.tw.
26. or/21-25
27. trauma*.tw.
28. injur*.tw.
29. penetrat*.tw.
30. stab*.tw.
31. or/27-30
32. 26 and 31
33. Splenic.tw.
34. spleen.tw.
35. stomach.tw.
36. gastric.tw.
37. or/33-36
38. rupture*.tw.
39. burst*.tw.
40. 38 or 39
41. 37 and 40
42. heart.tw.
43. cardiac.tw.
44. aortic.tw.
45. aorta*.tw.
46. or/42-45
47. rupture*.tw.
48. 46 and 47
49. exp Abdominal Injuries/
50. exp thoracic injuries/
51. 20 or 32 or 41 or 48 or 49 or 50
52. Blood.tw.
53. Plasma.tw.
54. 52 or 53
55. Autologous.tw.
56. 54 and 55
57. Autohaemotransfusion*.tw.
58. Auto-haemotransfusion*.tw.
59. Autohemotransfusion*.tw.
60. Auto-hemotransfusion*.tw.
61. Autotransfusion*.tw.
62. Auto-transfusion*.tw.
63. or/56-62
64. cell*.tw.
65. blood.tw.
66. 64 or 65
67. transfusion*.tw.
68. salvage.tw.
69. save*.tw.
70. or/67-69
71. 66 and 70
72. exp Blood Transfusion, Autologous/
73. exp Blood Loss, Surgical/
74. exp Blood Transfusion/
75. 63 or 71 or 72 or 73 or 74
76. 51 and 75
77. 12 and 76
```

(b) Transformed query.

```
1. controlled clinical trial.pt.
2. placebo.ab.
3. clinical trials as topic/
4. randomly.ab.
5. trial.ti.
6. randomized.tw.
7. randomized controlled trial.pt.
8. or/1-7
9. humans/
10. animals/
11. 9 not 10
12. 8 and 11
13. abdominal.tw.
14. abdomen.tw.
15. chest.tw.
16. thoracic.tw.
17. trunk.tw.
18. or/13-17
19. exp Wounds, Penetrating/
20. 18 and 19
21. abdominal.tw.
22. abdomen.tw.
23. chest.tw.
24. thoracic.tw.
25. trunk.tw.
26. or/21-25
27. trauma*.tw.
28. injur*.tw.
29. penetrat*.tw.
30. stab*.tw.
31. or/27-30
32. 26 and 31
33. Splenic.tw.
34. spleen.tw.
35. stomach.tw.
36. gastric.tw.
37. or/33-36
38. rupture*.tw.
39. burst*.tw.
40. 38 and 39
41. 37 and 40
42. heart.tw.
43. cardiac.tw.
44. aortic.tw.
45. aorta*.tw.
46. and/42-45
47. rupture*.tw.
48. 46 and 47
49. exp Abdominal Injuries/
50. exp thoracic injuries/
51. 20 or 32 or 41 or 48 or 49 or 50
52. Blood.tw.
53. Plasma.tw.
54. 52 or 53
55. Autologous.tw.
56. 54 and 55
57. Autohaemotransfusion*.tw.
58. Auto-haemotransfusion*.tw.
59. Auto-hemotransfusion*.tw.
60. Autotransfusion*.tw.
61. Auto-transfusion*.tw.
62. or/56-61
63. exp Blood Transfusion, Autologous/
64. exp Blood Loss, Surgical/
65. exp Blood Transfusion/
66. and/63-65
67. 51 and 66
68. 12 and 67
```

Figure 8.5: Comparison between the original query (left) and a transformed query (right) for topic 73. The transformed query was chosen using the $F1_n$ selector, and was the most effectively refined query.

8.2.3 What are the Best Transformations?

RQ2.3

Which types of transformations are the most effective at refining Boolean queries in systematic review literature search?

To answer **RQ2.3**, a comparison between candidate selection transformations was made. The summary of query refinement are presented in Table 8.3. The following mapping indicates which of the six transformations align with which headings in Table 8.3:

- Clauses, Keywords: *Clause Removal* & *Concept Embedding Expansion*
- `#AND` , `#OR` : *Logical Operator Replacement* (`#NOT` included for completeness)
- Fields, (Title, Abstract, MeSH, Other): *Field Restrictions*
- Exp. Mesh: *MeSH Explosion*
- Avg. MeSH, Max. MeSH: *MeSH Parents*

On average, none of the approaches selected queries which performed explicit query expansion. Indeed, a more in-depth analysis of the results identifies that none of the queries is longer than the original query. This finding suggests that, in this context, query expansion may not be an effective way to refine a query. On the other hand, query reduction was chosen very often, more so by the learning to rank selector — which tended to remove clauses from queries more often than the nearest neighbour selector. While either of the selectors did not choose the candidate queries with the *Concept Embedding Expansion* transformation applied, the *MeSH Parent* transformation was. Overall, the R_l selector chose queries with an average MeSH depth of 0.32 higher than the original query. This finding suggests that to maximise an increase in recall, this selector opted to choose a query that broadened the scope of MeSH keyword than to add new, somewhat related text-matching keywords.

In terms of directly optimising precision and recall, we further analyse the P_n, and R_l selectors, as each recorded the highest increase in precision and recall, respectively. The P_n selector showed a similar proportion of fields to $F1_n$. However, the proportion of `AND` operators was lower, and the proportion of OR operators was higher. The R_l selector had fewer Title *and* Abstract fields, and it removed atomic clauses that did not contain MeSH fields (this is implied by the fact that no transformation allowed MeSH terms to be added, and that any clause could be removed with the *Clause Removal* transformation). Also, this selector had a higher number of OR operators than AND operators. Finally, Table 8.3 shows that R_l is the only selector to increase the average MeSH depth, suggesting that this selector often applied the *MeSH Parent* transformation.

To provide further insight into how each candidate selector model affects the resulting selected queries, Table 8.4 presents the average proportion of fields and Boolean operators in queries. As the number of clauses in each of the selected queries differ from each trained candidate selector, this table

	Clauses	# AND	# OR	# NOT	Fields	Fields (Title)	Fields (Abstract)	Fields (MeSH)	Fields (Other)	Keywords	Exp. MeSH	Avg. MeSH	Max. MeSH
Seed	12.6000	4.7714	7.3714	0.4571	74.4286	27.8286	29.3143	14.3143	2.9714	41.1714	4.2286	3.1763	7.7143
P_l	-3.3429*	-1.3714*	-1.8857*	-0.0857	-23.9143*	-9.2571	-9.3143	-4.8857*	-0.4571	-12.8857*	-1.5143*	-0.2009	-0.5714
P_n	-0.1714	0.3714	-0.5429*	0.0000	-2.9429*	0.3714	-2.7429*	-0.2000	-0.3714	-0.3429	-0.5143*	-0.2425	-0.2857
R_l	-2.8286	-1.6286	-1.0857	-0.1143	-18.5429	-7.1714	-8.1429	-2.3714	-0.8571	-11.0000	-0.7143	0.3190	-0.3143
R_n	-1.0857*	-1.0286*	0.0000	-0.0571	-5.9714*	-2.5429*	-1.9429*	-1.0000*	-0.4857	-2.6286*	-0.2857	-0.0877	-0.2286
$F0.5_l$	-2.3429*	0.0000	-2.2286*	-0.1143	-13.4000*	-3.7143*	-4.9429*	-4.2571*	-0.4857	-7.2857*	-1.2571*	-0.4887	-1.4571*
$F0.5_n$	-0.3143*	-0.1429	-0.1429	-0.0286	-2.3143*	-0.2571	-1.4000*	-0.2857	-0.3714	-0.3143*	-0.4571*	-0.1620	-0.3429
$F1_l$	-4.4286*	-1.5143*	-2.6286*	-0.2857*	-21.7143*	-6.9143*	-7.5143*	-6.4857*	-0.8000	-11.8000*	-2.0571*	-1.6380*	-3.9143*
$F1_n$	-0.3143	0.5143*	-0.8286*	0.0000	-3.3429*	-0.5714	-2.0286*	-0.4000	-0.3429	-0.7714	-0.7714*	-0.2804	-0.3143
$F3_l$	-0.4857	0.2857	-0.7714*	0.0000	-6.6000	-2.6000	-2.9429	-0.6857	-0.3714	-2.6286	-0.2000	-0.2266	-0.3429
$F3_n$	0.0000	-0.0286	0.0286	0.0000	-1.0571	-0.3429	-0.3429	0.0000	-0.3714	0.0000	-0.8286*	-0.3331*	-0.2857
WSS_l	-2.2857*	-0.8571*	-1.3714*	-0.0571	-12.7429*	-5.0286*	-5.1143*	-2.2000*	-0.4000	-6.1143*	-0.4857*	-0.4084	-0.8857
WSS_n	-0.8571*	-0.2571	-0.5714*	-0.0286	-5.9714*	-2.1143*	-2.4000*	-1.0000*	-0.4571	-2.8000*	-0.6571*	-0.1420	-0.3143

Table 8.3: Summary of the query refinement process. This table shows the average difference in change of different aspects of queries obtained by each selector. A negative value indicates a reduction in the corresponding aspect, a positive value indicates an increase, and a zero value indicates that aspect was not changed. For comparison, the average values of the original *Seed* queries are reported as well. *Clauses* refers to the number of Boolean clauses in the query. *# AND*, *# OR*, and *# NOT* indicate how many of the corresponding operators were in the query. *Fields* indicates how many fields were in the query in total; this is further divided into how many of those were Title, Abstract, MeSH, or other (e.g., dates or publication type). *Exp. MeSH* indicates the number of MeSH terms that have been exploded, and *Avg. MeSH* & *Max. MeSH* indicate the average and maximum depth in the ontology of the MeSH terms. Two-tailed statistical significance where $p < 0.01$ between the Seed queries and the queries selected by each model is indicated with $*$.

	Title	Abstract	MeSH	Other	AND	OR	NOT	Exp. MeSH
Baseline	0.3327	0.3589	0.2711	0.0374	0.3542	0.6172	0.0286	0.1227
$F1_l$	0.3366*	0.3764*	0.2324*	0.0547	0.3589*	0.6298*	0.0113*	0.0704*
$F1_n$	0.3432	0.3370*	0.2901	0.0297	0.4480*	0.5223*	0.0297	0.0819*
$F3_l$	0.3730	0.3500	0.2472	0.0298	0.4287	0.5417*	0.0296	0.1196
$F3_n$	0.3376	0.3706	0.2625	0.0293	0.3447	0.6267	0.0286	0.0819*
$F0.5_l$	0.3513*	0.3675*	0.2402*	0.0410	0.4406	0.5392*	0.0202	0.1048*
$F0.5_n$	0.3590	0.3501*	0.2621	0.0289	0.3387	0.6359	0.0254	0.1044*
P_l	0.3326	0.3657	0.2656*	0.0360	0.3246*	0.6402*	0.0352	0.1024*
P_n	0.3626	0.3198*	0.2882	0.0294	0.4263	0.5441*	0.0296	0.0936*
R_l	0.3161	0.3357	0.3221	0.0262	0.3196	0.6567	0.0236	0.1153
R_n	0.3252*	0.3822*	0.2647*	0.0280	0.2729*	0.6992	0.0278	0.1223
WSS_l	0.3378*	0.3559*	0.2771*	0.0293	0.3102*	0.6641*	0.0256	0.1358*
WSS_n	0.3314*	0.3504*	0.2899*	0.0283	0.3790	0.5927*	0.0283	0.1049*

Table 8.4: Average proportion of clauses for each field, each Boolean operator, and each Exploded MeSH term, across selected queries by each query selector approach. The original queries (Seed) are included for comparison. Statistically significance (two-tailed t-test with $p < 0.01$) between the *Baseline* and listed selectors is indicated with $*$.

seeks to show the changes in the types of fields and types of Boolean operators in proportion to the size of the query (i.e., for the total number of Boolean operators and fields). For example, the $F1_n$ selector (which consistently improved the effectiveness of queries over the original seed queries for all measures) tended to increase the number of Title fields while decreasing the number of Abstract fields, as well as increase the number of `AND` operators while decreasing the number of OR operators. These refinements to queries align with the intuition that using the Title field will yield fewer results than the Abstract (or both fields combined) and that the `AND` operator is more restrictive than the OR operator (the intersection of retrieved citations as opposed to the union). Overall the results indicate that, among the investigated transformations, *syntactic* transformations play a larger role in query refinement than *semantic* transformations.

8.2.4 What is the Impact of Unjudged Studies?

RQ2.4

What impact do unjudged studies have on the effectiveness of refined Boolean queries in systematic review literature search?

To answer **RQ2.4**, the evaluation measures and the estimates obtained with the two residual heuristics are compared.

As reported in Table 8.2, these results show that the effect of unjudged citations is not significant.

When a selector chooses candidate queries that perform statistically significantly worse for an evaluation measure, the residual *mle* results are also significantly worse. Nonetheless, Table 8.2 still shows that in many cases, effectiveness increases when unjudged citations are factored into the evaluation. For example, the P_n selector does not obtain increases in precision over the seed queries, when unjudged citations are considered non-relevant. However, when using the *mle* heuristic, precision is higher, though, with the *r* heuristic, precision is also lower. From analysing the *r* heuristic, we can determine that the P_n selector retrieves many more non-relevant citations than the P_l selector. For these reasons, estimates for residual should be considered when performing high-recall tasks such as this; or in tasks that modify or change queries somehow. This consideration is especially important when relevance assessments are largely incomplete, and no system variation was considered when forming the pool for assessments: this was the case for systematic review collections available for Information Retrieval research [104, 205].

8.3 Summary

We employed two candidate selector functions, one based on learning to rank, and one based on a nearest neighbour algorithm to identify more effective queries than ones initially developed for systematic review literature search. Both candidate selectors were trained to maximise six different evaluation measures. When evaluating query refinement, we also revealed that the refinement retrieved unjudged documents; we accounted for these in the evaluation by computing the residuals of the evaluation measures. We did so by considering both an optimistic residual and a probabilistic residual (based on maximum likelihood estimation). Our results showed that the nearest neighbour model could more effectively select query refinements than the learning to rank approach. Our experiments have shown that the explicit query expansion method (*cui2vec Expansion*) proposed in this thesis is not viable for query refinement (as candidates with other transformations outrank it). However, the explicit query reduction method (*Clause Removal*), and to a lesser extent the implicit query expansion method (*MeSH Parents*) can be used to help refine queries.

The detailed analysis of how candidate selectors transformed each set of selected queries suggested that the syntactic transformations were, in fact, more suited to refinement than semantic transformations. Our experiments also showed that the effect of unjudged citations on evaluation is significant: unjudged citations should be accounted for in some way when performing the evaluation.

Given the experiments undertaken, we can answer **RQ2**,

RQ2

Can manually formulated search strategies be automatically refined to produce more effective search strategies?

It was found that more effective search strategies exist, but the query transformation chain framework was able to identify the better refinement to make automatically. The proposed method for automatic

query refinement can vastly reduce the time spent creating systematic reviews. The searching and screening processes contribute to a significant period of time and money in the systematic review creation process. Reducing the total number of studies to screen by automatically refining already highly effective queries, the time spent screening citations, and the total cost of a systematic review can be reduced. These reductions lead to faster and more up-to-date evidence-based medicine, leading to more accurate clinical decisions and better health outcomes.

However, there still exists one major limitation that was not addressed in this chapter. That is, the impact different query variations have on the effectiveness of trained models. As has been seen so far, the generation of training data is computationally expensive. Thus, the following chapter investigates whether all training data is necessary (to cut the computational costs) and if the training data can result in less effective candidate selection models. This final experimental chapter will somewhat also confirm whether the experiments in this chapter are due to random chance (were we lucky with the training data?) and if the models used in this chapter could be improved by merely changing the training data.

Chapter 9

Sampling Query Variations

One key limitation with the query transformation framework is *the approach used for sampling training queries*. The query variation generation process produces a computationally infeasible amount of queries. This happens when the original Boolean query has many clauses, or when many transformations apply to each clause, and when many iterations of the transformation process are considered: all conditions typical to (medical) systematic reviews queries. The computational complexity required for generating the training queries is exponential: $O(n^t)$, where n is the number of variations for a query and t is the number of transformations.

Researchers have noticed that the impact of sampling on effectiveness may be higher than specific aspects of the learning algorithm. For example, Wu et al. [247] identify that sampling in the context of Deep Embedding Learning has a much higher impact on model effectiveness than the loss function. In the context of Learning to Rank, Donmez et al. [73], and Aslam et al. [13] find that sampling methods have a significant impact on the effectiveness of ranking models. Donmez et al. use an active sampling approach to maximise the estimated loss differential over unlabelled data. While Aslam et al. investigates the effect sampling has on LTR in a typical ad-hoc query-document retrieval setting. This work uses sampling methods that both do and do not use prior relevance knowledge to sample, and evaluates sampling methods by how well a LTR model trained on different sampling methods can rank documents given a query. This setting is similar to the one we study in this work, although we rank queries, not documents. Similar studies have also investigated the impact of sampling in a traditional LTR context from different perspectives. For example, Kanoulas et al. [106] consider the distribution of positive and negative training examples on a large scale, and Lucchese et al. [127] investigate negative sampling to improve LTR models. However, more related to this work is the study by Mehrotra et al. [135] who develop query sampling methods to reduce labelling costs for training an Active Learning to Rank model. Key to the effectiveness of the selection of queries in that work is the informativeness and representativeness characteristics. It is unclear, however, how to apply these techniques to Boolean queries in an unsupervised setting.

One solution to that problem has been to randomly sample a predetermined number of query variations, which was performed in the previous chapter. However, the way query variations are

sampled can influence the effectiveness of the LTR model trained on these training queries. We provide evidence for this with a small study in Section 9.1, which establishes the motivation for the remainder of this chapter. In this chapter, we investigate the effect sampling has on the effectiveness of trained models and empirically establish effective sampling strategies for this problem. Furthermore, previous work has considered exploring the space of candidate queries using a breadth-first exploration method. In this chapter, we further contribute a depth-first exploration method and adapt applicable sampling strategies.

This chapter aims to address any potential limitations and downsides to the strategies for sampling query variations from the previous chapter and identify better strategies. The limitations of those chapters present three research questions to be addressed in this chapter:

RQ2.5

How does sampling affect the distribution of *sampled query effectiveness* within the sampled set?

In **RQ2.5**, the different approaches to exploring the query space will be investigated and compared (where appropriate) and how these approaches impact the automatic selection of queries.

RQ2.6

How does sampling affect the distribution of *selected query effectiveness* in the Query Transformation Chain framework?

In **RQ2.6**, the different approaches to sampling using the same approach to exploration are compared by examining what information about queries are recorded to inform how sampling should be performed and how these impact the automatic selection of queries.

RQ2.7

Are there relationships between the set of *sampled queries* and the effectiveness of the *selected queries*?

Finally, in **RQ2.7**, any relationships between sampled queries and selected queries are revealed and analysed. Specifically, this research question seeks to identify correlations between the effectiveness of sampled queries and selected queries.

9.1 Motivating Example

We motivate this work with an example whereby queries are sampled according to three sampling strategies to select query variations: (1) sample only queries which improve over the original query (*positive*) (2) sample only queries which do not improve over the original query (*negative*), (3) sample

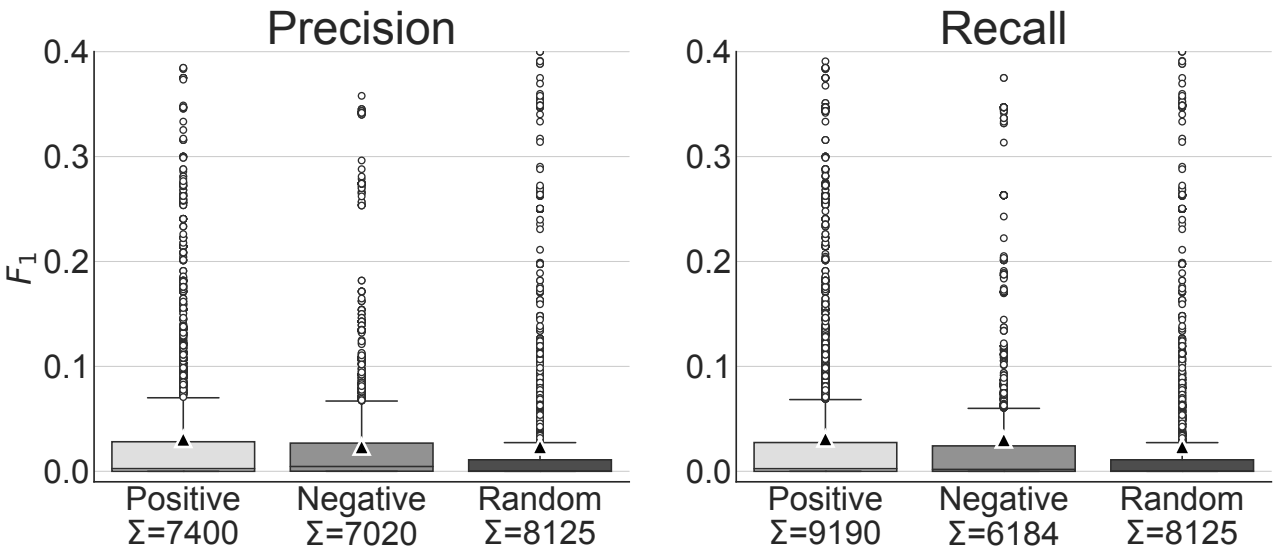


Figure 9.1: Distributions of *sampled* queries using strategies which sample positive queries, negative queries, and sampling randomly. Positive and negative sampling is performed according to precision and to recall. Mean values can be identified by a ▲. The total number of samples in each distribution (Σ) are presented beneath the respective label. Queries with F_1 above 0.4 have been omitted for clarity.

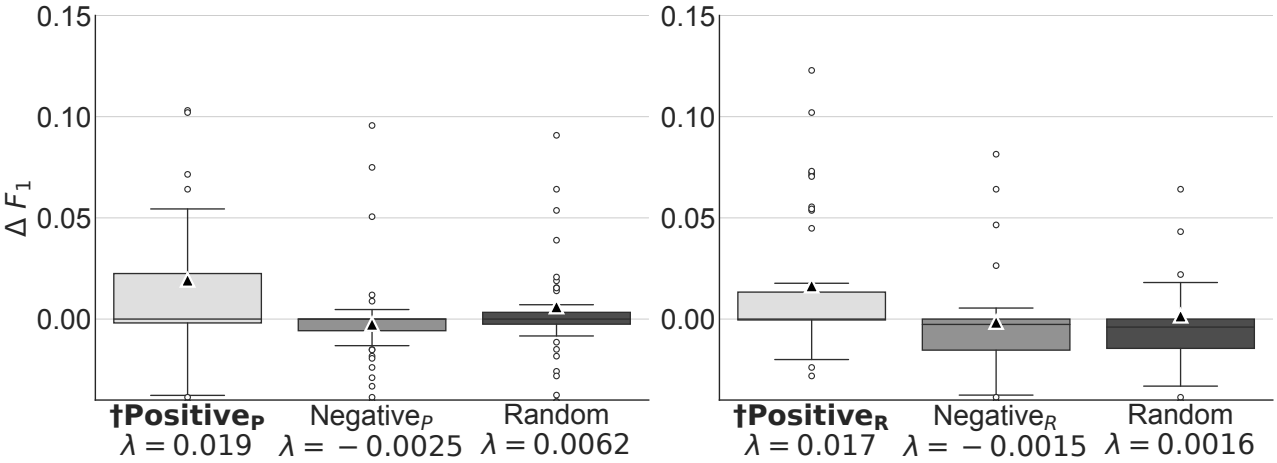


Figure 9.2: Distributions of *selected* queries, chosen from selectors trained on only positive queries (precision: Positive$_P$, recall: Positive$_R$), only negative queries (precision: Negative$_P$, recall: Negative$_R$), and randomly selected queries. Mean values (λ) in each distribution can be identified by a ▲ and are presented beneath each label. Queries with $F_1 > 0.15$ have been omitted for clarity. Statistical significance ($p < 0.05$) between the original queries and the selected queries is indicated by † on the corresponding label.

randomly. Approximately the same number of queries are sampled by each strategy (between 5000-8000). The distributions of sampled queries for precision and recall are presented in Figure 9.1 (relating to **RQ2.5**). Next, two LTR models are trained for each sampling strategy, optimising for either precision or recall. For both the precision-based and recall-based distributions of queries, the positive strategies contain queries with a higher average F_1 than the queries sampled negatively. The randomly sampled distribution of queries has the lowest average F_1 in both cases. Six models in total are created[1].

[1] (3 sampling strategies × optimise precision) + (3 sampling strategies × optimise recall)

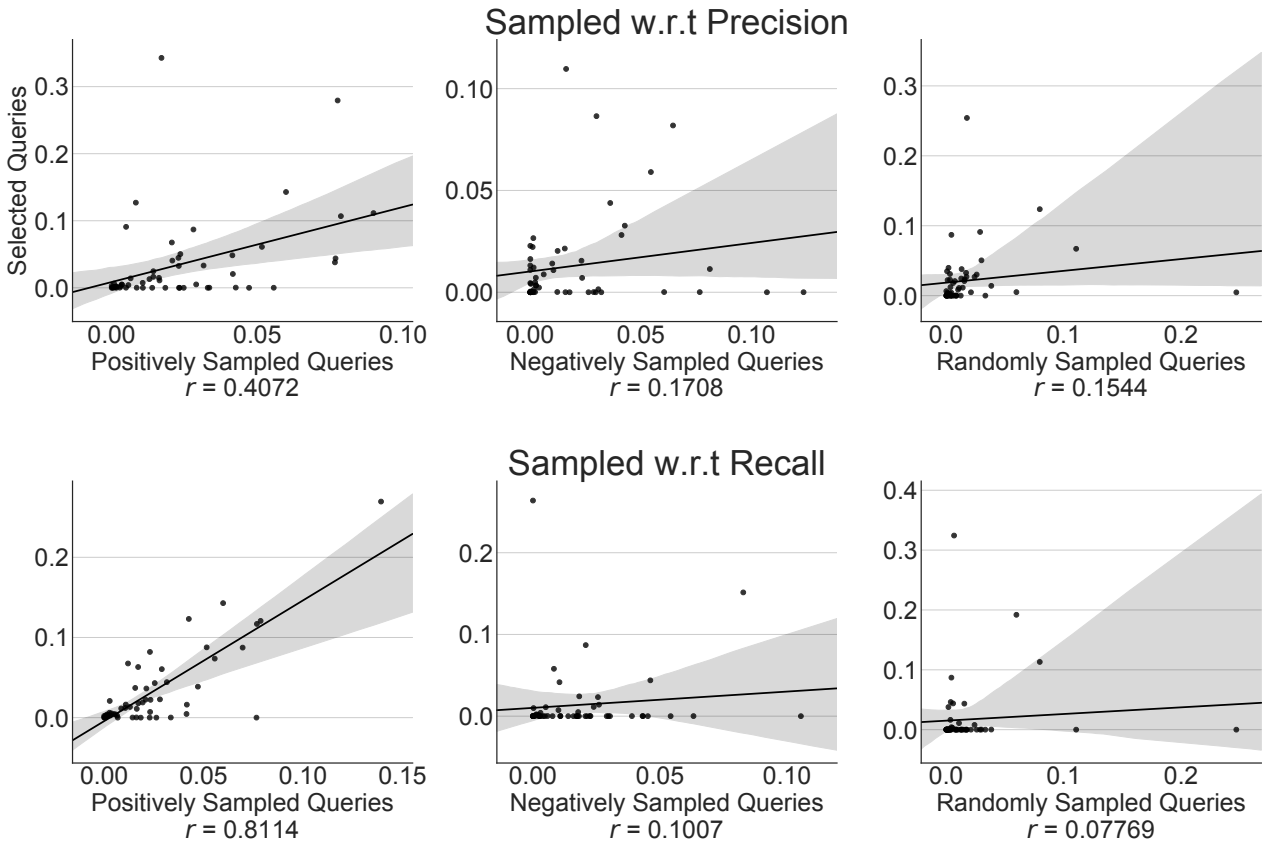


Figure 9.3: Relationship between the mean sampled query (i.e., the average value of queries from each sampled distribution; those in Figure 9.1) and selected queries (i.e., those in Figure 9.2). Horizontal axis: average F_1 of sampled queries. Vertical axis: F_1 of selected queries. The linear relationship is presented by a solid black line. Pearson's r correlation coefficient between sampled and selected queries is recorded beneath.

A summary of the results from using each LTR model to select new queries in this example automatically is shown in Figure 9.2 (relating to **RQ2.6**). The results for precision LTR showcase the impact of sampling. The model trained on only queries that delivered higher precision (Positive$_P$) produces query variations that are on average statistically significantly better than the original queries ($p < 0.05$). Likewise, the model trained on only queries that delivered higher recall (Positive$_R$) produces query variations that are also on average statistically significantly better than the original queries, but not as effective as the precision-based model. Meanwhile, both models trained on the negative queries (Negative$_P$ and Negative$_R$) select queries less effective than the original queries. The models trained on queries sampled at random (Random$_P$ and Random$_R$) select queries with effectiveness somewhere between the two other models.

The relationship between sampled queries used to train a model and queries selected by the model trained on sampled queries is presented in Figure 9.3 (relating to **RQ2.7**). Queries selected by models trained on positively sampled queries are strongly positively correlated with each other. Meanwhile, negatively and randomly sampled queries are only weakly positively correlated. Moreover, the central tendencies of the linear regression are lower between the positive and negative strategies and the negative and random strategies. This finding indicates a lower margin of error, for example, between the positive and negative model.

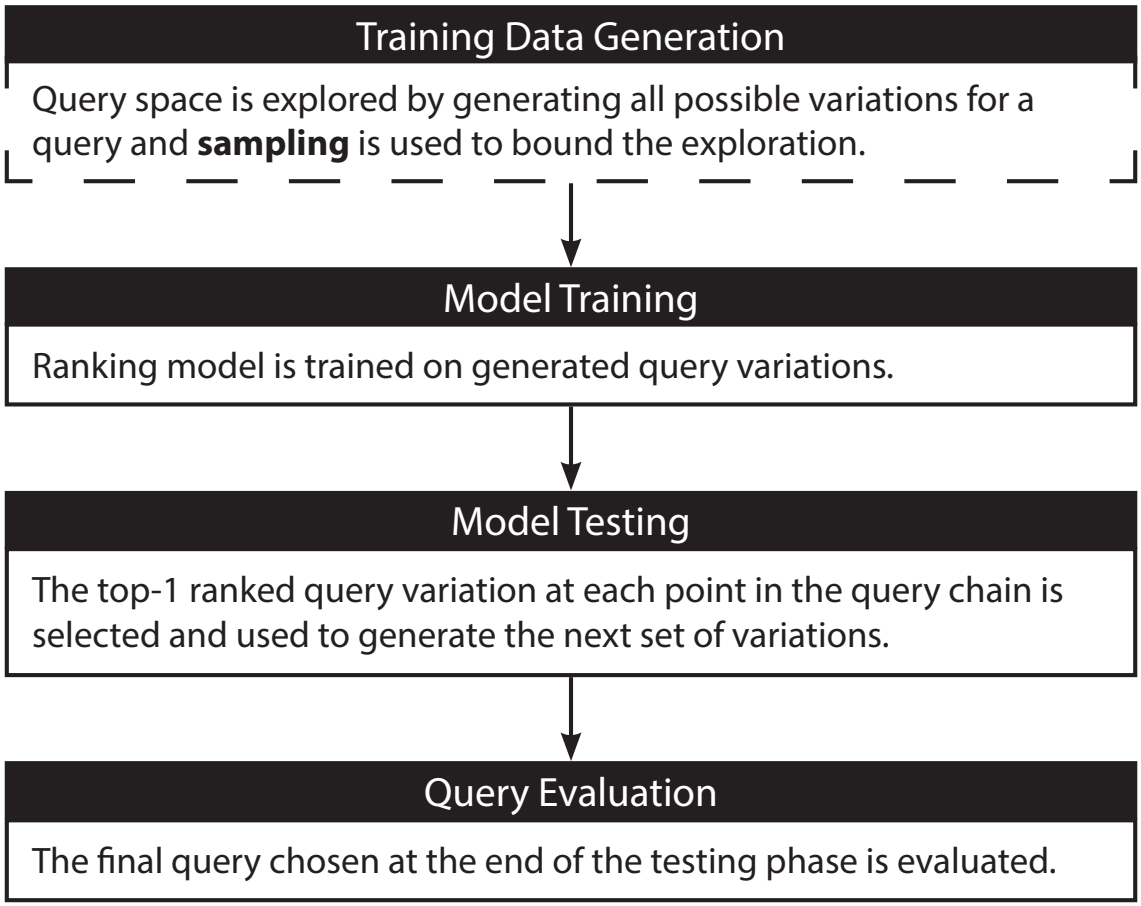


Figure 9.4: High-level overview of the experimental pipeline. Emphasis is placed on the generation phase (dotted box), where training queries are created by exploring the query space. Exploration is computationally expensive and is thus bounded by the sampling methods. In the testing phase, sampling methods are not required as only the selected query in each step in the chain is used to generate the next set of variations.

The results from this experiment highlight the importance of sampling in the context of learning models to select more effective queries for systematic review literature search. Models trained on explicit positive examples result in significantly better-selected queries. On the other hand, models trained on negative or randomly sampled queries result in selecting less effective queries. Indeed, the relationship between selected queries and queries used for training is correlated with effectiveness. This finding motivates the work and methods presented in this chapter.

9.2 Sampling

Figure 9.4 presents a high-level overview of the experimental pipeline for QTC, and where the sampling experiments fit and the impact sampling has on the rest of the pipeline.

We propose two **exploration methods** for sampling the query space to obtain training queries: *breadth-first* and *depth-first*. Each of these methods can be instantiated according to several **sampling strategies**, which leads to the sampling of different queries. In the following sections, we describe the two exploration methods and the applicable sampling strategies.

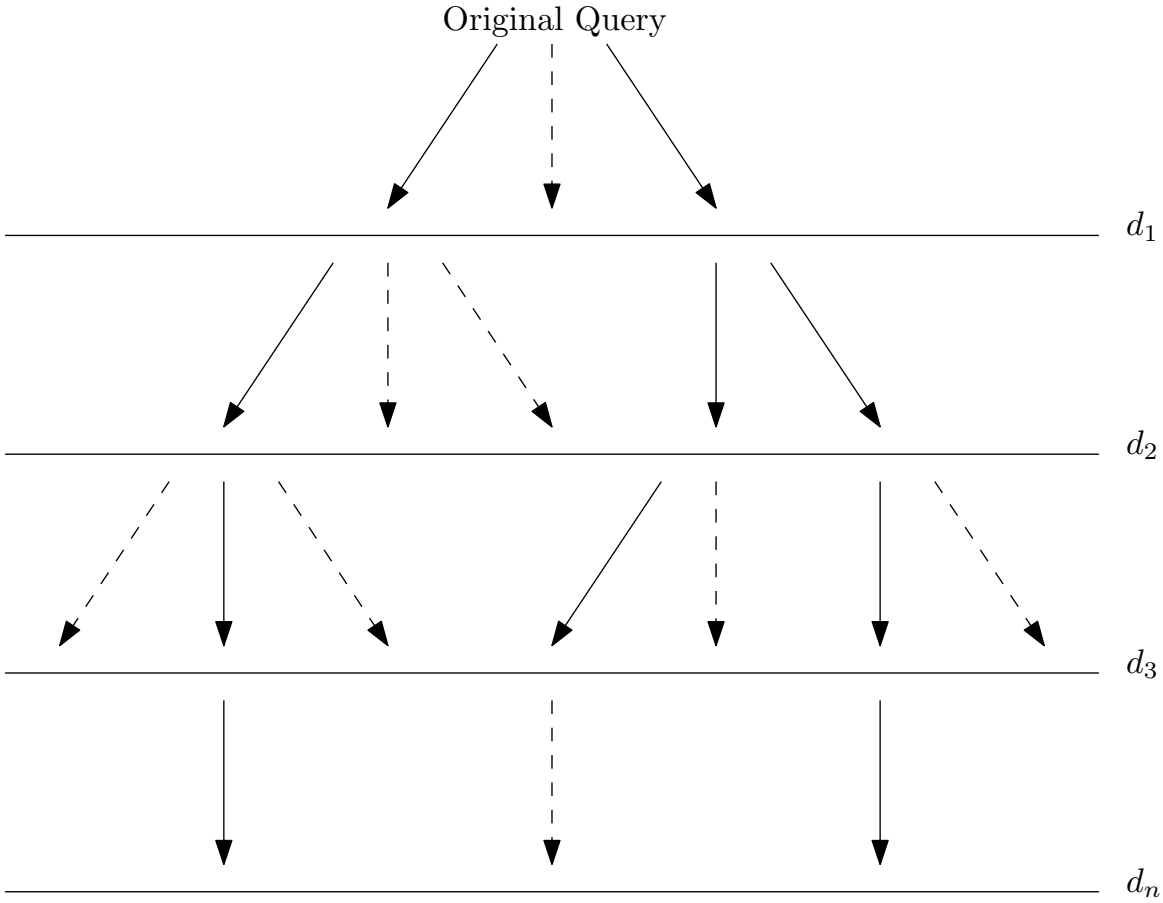


Figure 9.5: Breadth-first exploration method. Candidates are sampled when the *depth* of the chain (indicated by $d_1 ... d_n$) increases (i.e., all of the candidates have been generated from the previous queries). In this figure the queries that are sampled, and therefore continue the chain in the generation process, are indicated by a solid line. Dashed lines indicate that the query is discarded, and therefore not included in the sampled set.

9.2.1 Breadth-First

The breadth-first exploration method (visualised in Figure 9.5) generates a set of query candidates ($\hat{Q}$) from previous queries in the chain and pools them together to sample. Algorithm 1 describes this approach. The `sample` method indicates one of the following breadth-first sampling strategies. These strategies act on a *pool* of generated query variations with the result being a subset of queries (reduction). The amount of reduction is determined by a parametrised approach where two variables control the ratio of queries to sample. The first parameter, n, controls the *minimum* number of queries to sample; the second, δ, controls the percentage of queries to sample. If $|\hat{Q}| < n + (\delta \times |\hat{Q}|)$, then all queries are included in the sample. Queries are sampled into each stratum randomly.

The sampling strategies that can be instantiated by the breadth-first exploration method we consider are:

Greedy Sampling Strategies The greedy sampling strategies are adapted from the greedy candidate selector of Scells and Zuccon [192]. With these strategies, queries are sampled by choosing those which minimise the number of retrieved documents while maximising an evaluation measure. An evaluation measure has to be specified in addition to the n and δ parameters.

for d←1 **to** l **do**
  for t ∈ T **do**
    $\hat{Q}_t \leftarrow \texttt{append}(\hat{Q}_t, t(q))$;
  end
  $\hat{Q} \leftarrow \texttt{append}(\hat{Q}, \texttt{sample}(\hat{Q}_t, n, \delta)$;
  $\hat{Q}_t \leftarrow \emptyset$;
end
return $\hat{Q}$

Algorithm 1: Breadth-first exploration method. l is the chosen depth to explore and T is the set of transformations to be applied to a query q. The algorithm returns a sampled set of queries, $\hat{Q}$.

- **Greedy Naive**: Minimise the number of citations retrieved by the candidate query (N), where $N > 0$ while maximising the number of relevant citations retrieved.

- **Greedy Diversity**: Minimise the number of citations retrieved, where $N > 0$, while diversifying the queries. In this work, we use Maximal Marginal Relevance (MMR) [35] as our diversity function. To apply MMR to query re-ranking, we replace the similarity between a document and query of the original MMR formulation, with the evaluation measure score of the query ($Ev(Q)$). The similarity function $Sim()$ we use is cosine similarity. Thus, in this context MMR becomes:

$$\texttt{score} = \lambda \cdot Ev(Q) + (1-\lambda) \cdot Sim(q_i, Q) \tag{9.1}$$

Evaluation Sampling Strategies Explicit relevance judgements for sampling queries. Only the scores for a given evaluation measure are considered; thus, each strategy below must be paired with an evaluation measure (e.g., precision, recall, etc.).

- **Evaluation Stratified**: Two strata are obtained by sampling proportionately across the pooled candidate queries. Each stratum contains either the population of queries that were more (or less) effective than the original seed query given an evaluation measure.

- **Evaluation Balanced**: Sample uniformly by the score of a given evaluation measure across pooled candidate queries. Candidates are balanced into two distributions: those that are more effective than the original seed query given an evaluation measure and less effective.

- **Evaluation Positively/Negatively Biased**: Biased sampling across the pooled candidate queries. Candidates are sampled only when they are more (or less) effective than the original seed query given an evaluation measure.

- **Evaluation Diversity**: Re-rank the pooled candidate queries by applying MMR to the score of a given evaluation measure for queries, then sample uniformly across the new distribution.

Transformation Sampling Strategies The types of transformations applied to generate each variation are considered when sampling. Unlike the evaluation sampler, which aims to sample query

variations for some evaluation measure, the transformation sampler aims to sample across the types of transformations applied to queries.

- **Transformation Stratified**: Stratified (proportional) sampling across the pooled candidate queries. The number of strata is equal to the total number of types of transformations in the candidate pool (i.e., some type of transformations may not apply to any of the previous queries).
- **Transformation Balanced**: Balanced (uniform) sampling across the pooled candidate queries. Candidates are balanced into the number of types of transformations in the candidate pool.

Clustering Sampling Strategy

Similar candidates are grouped using features about the queries. k-means++ [12] with $k = 5$ is used to cluster queries. Queries are represented with the features vectors used in previous chapters to guide the query chain transformations. Queries are then sampled from each cluster in a round-robin fashion, up to the cut-off n (the number of queries to be selected).

Random Sampling Strategy The random strategy is included as a naïve approach to sampling, to determine if any previous techniques provide significant benefits over random sampling. Candidates are sampled according to a uniform likelihood.

9.2.2 Depth-First

The depth-first exploration method (visualised in Figure 9.6) uses a depth-first search to traverse the query space. Candidate queries are sampled in the same fashion as Section 9.2.1, however rather than sampling from a pool of queries, the chain of previous queries are used to determine inclusion in the sample set. In order to adapt breadth-first based sampling strategies to the depth-first method, a cost-based approach is used. Each strategy is provided with a budget of how many queries may be included in the sampled set. The benefit to using the depth-first exploration method over the breadth-first method is that queries can be sampled for a sequence of transformations, rather than the pool of variations at a certain depth. Algorithm 2 describes how the depth-first exploration method works. The following strategies can be adapted to depth-first exploration:

- **Evaluation Positively/Negatively Biased**: Sample candidates where, given an evaluation measure, the chain of queries including the most recent query is more (or less) effective than the original query.
- **Transformation Balanced**: Sample candidates where the chain of transformations applied to previous candidates is balanced in terms of which transformations have been applied.
- **Random**: Sample candidates according to a uniform likelihood.

In addition to adapted strategies, the following strategy is unique to the depth-first exploration method:

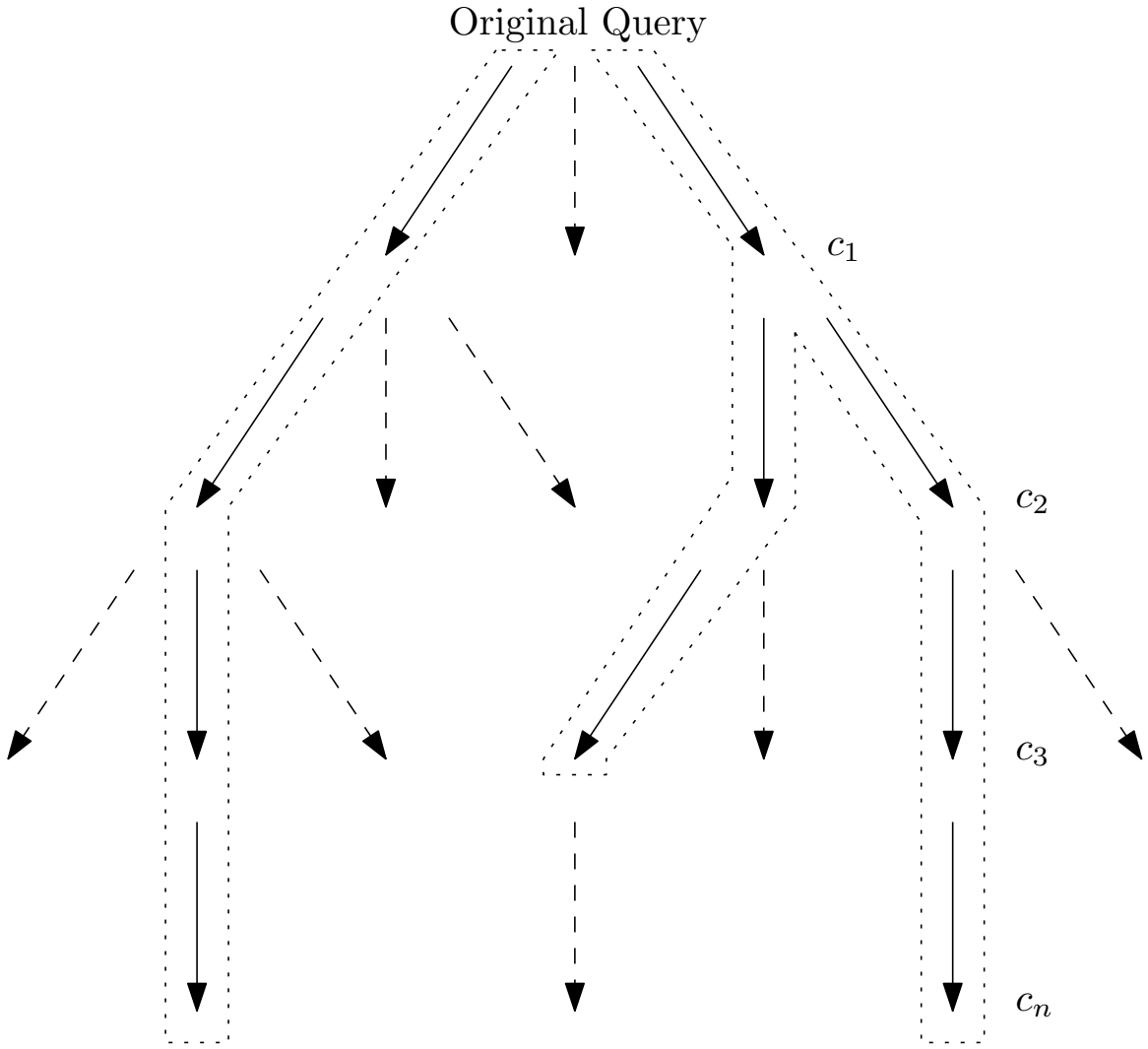


Figure 9.6: Depth-first exploration method. Candidates are sampled when the *chain* of the query (indicated by $c_1 ... c_n$) satisfies a *sampling criteria* (indicated by a dotted border around a chain of queries).

- **Transformation Biased**: Choose candidates where the chain of transformations applied to previous candidates is the same as the most recent transformation applied.

The reason only a subset of sampling strategies from the breadth-first method can be applied to the depth-first method is the way queries are pooled. e.g., the balanced and stratified evaluation methods use a pool of generated query variations at a given depth to determine a distribution and strata for the two strategies, respectively. The **Transformation Biased** depth-first strategy is an example of a strategy that could not be possible in the breadth-first method.

9.3 Experimental Setup

Experiments are performed on the same collection of systematic review queries as in Chapter 8 to evaluate the sampling strategies empirically. This collection contains 125 queries and is the most extensive collection available for this task. When generating training queries, each combination of

```
function choose(q̂) begin
    if stop(q̂) then
        return q̂;
    end
    for t ∈ T do
        if sample(t(q)) then
            Q̂_t ← append(Q̂_t, t(q));
        end
    end
    for q ∈ Q̂_t do
        Q̂ ← append(Q̂, choose(q));
    end
end
Q̂ ← choose(q);
return Q̂
```

Algorithm 2: Depth-first exploration method. T is the set of transformations to be applied to a query q. The algorithm returns a sampled set of queries $\hat{Q}$. The `stop` function determines when the budget of sampled queries is depleted. Unlike the `sample` of the breadth-first method, here, `sample` is a *policy* that determines if a query should be sampled.

exploration algorithms and sampling strategies from Section 9.2 is used to sample automatically generated queries. When the breadth-first exploration algorithm is used, n is set to 10, and δ is set to 0. When the depth-first exploration algorithm is used, a budget of 65 queries was found through empirical experimentation to produce a similar number of sampled queries for the depth-first exploration algorithm. Each sampled query is evaluated according to precision, recall, and F_1. The evaluation measure is used as the label for training LTR models (the LTR models learn to rank queries for target evaluation measures).

Some sampling strategies require parametrisation. In the case of diversity-based sampling strategies (i.e., MMR; which includes *Greedy Diversity* and *Evaluation Diversity*), the λ parameter is set to 0.3 which trades similar scoring queries for more diverse queries. This parameter was chosen to provide effective results in early experimental testing. For the breadth-first and depth-first random sampling strategies, the likelihood of choosing a candidate to sample is set to 65% (empirically found to generate similar amounts of training queries as other sampling strategies). When sampling strategies also have an evaluation measure parameter, in all cases both precision and recall are used, denoted as "Strategy$_P$" and "Strategy$_R$", respectively.

When learning a ranking model, the evaluation measures recorded for each query are used to rank queries. Thus for each sampling strategy, a model for each evaluation measure is trained and evaluated. The PubMed entrez API [189] is used as the retrieval system (allowing one to submit queries directly against PubMed). All queries are used to train the ranking model using five-fold cross-validation. To obtain validation data, the training portion of each fold is further split into 80% training and 20% validation. A LTR model is trained on each fold using the training and validation data. In total, a model is trained using each of the sets of sampled training data generated by the 25 sampling strategies

$\times$ 6 target evaluation measures $\times$ five folds, resulting in 750 LTR models. The ranking model used in candidate selection is the QuickRank [34] implementation of LambdaMART [249]. Experiments are conducted using the Querylab experimental pipeline framework [191]. Each model is set to optimise DCG@1 to place the most importance on the top-1 ranked query (i.e., the query selected to continue the chain).

When automatically selecting queries using the trained ranking models, the queries within the left-out portion of each fold are used to test each model. As there are no overlapping sets of query variations in each left-out fold, the resulting queries from this iterative transformation and selection process are evaluated using the aforementioned evaluation measures. To evaluate the effectiveness of the LTR models (which reflect the set of query variations used as training material), the models are used as candidate selectors in QTC. We use the average performance in terms of F_1 of selected queries to evaluate the performance of the LTR models; F_1 is typically used in this context.

9.4 Results

Next, we present the analysis and results of the sampling experiments; each of the following sections corresponds to a research question. The results presented here differ from those presented in the Motivating Example as they were obtained with models which learn to rank queries for precision and recall. The baseline in the results is the original queries. Gains are therefore measured with respect to these (unless otherwise stated).

9.4.1 Distribution of Sampled Queries

RQ2.5

How does sampling affect the distribution of *sampled query effectiveness* within the sampled set?

To answer **RQ2.5**, we investigate the distributions of sampled query effectiveness. Figure 9.7 presents the distributions of F_1 measure for the sampled queries and the total number of sampled queries for each sampling strategy in both exploration methods. In both the breadth-first and depth-first exploration methods, the distribution of queries sampled using Positively Biased Precision (Positive$_P$) obtain the highest average F_1. The distributions of queries sampled using the Greedy Naïve strategies (biased to precision: Greedy Naïve$_P$, and biased to recall: Greedy Naïve$_R$) obtain the second highest F_1 in the breadth-first exploration methods. Comparing the two Greedy sampling strategies with each other, the F_1 scores of queries in the distribution obtained by the Greedy Naïve sampling strategy is, on average, higher than queries in the distribution obtained using the Greedy Diversified strategy. The distribution of queries in all other breadth-first evaluation-based sampling strategies (Balanced, Diversified, Stratified, Negative) obtain similar F_1 scores to each other.

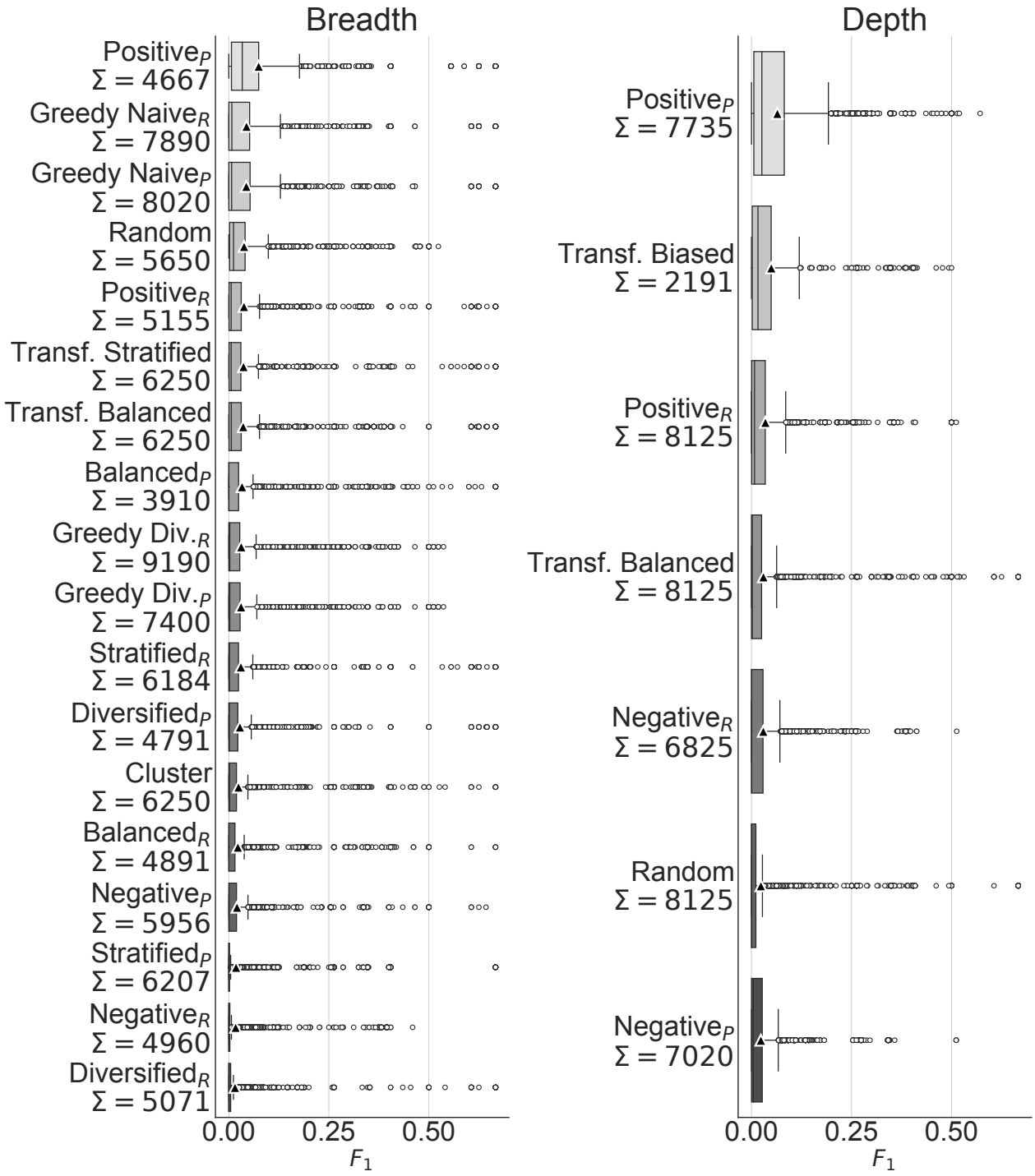


Figure 9.7: Distributions of sampled queries across the breadth-first (left) and depth-first (right) sampling strategies. Mean values in each distribution can be identified by a ▲. The total number of samples in each distribution (Σ) are presented along underneath the respective label.

Furthermore, the distribution of queries from the Transformation Stratified, Transformation Balanced, Cluster, and Random sampling strategies all obtain similar F_1 scores to the above evaluation-based strategies. In terms of the depth-based exploration method, all sampled distributions of queries except the Positively Biased Precision strategy obtain similar average F_1 scores. No correlation was found when observing the number of sampled queries in each distribution and the average F_1 score of those queries. This finding indicates that average scores of the queries in each sampling strategy are not influenced by the number of queries sampled by that strategy. In summary, the choice in sampling strategies does affect the sampled queries, the effect of which is studied next.

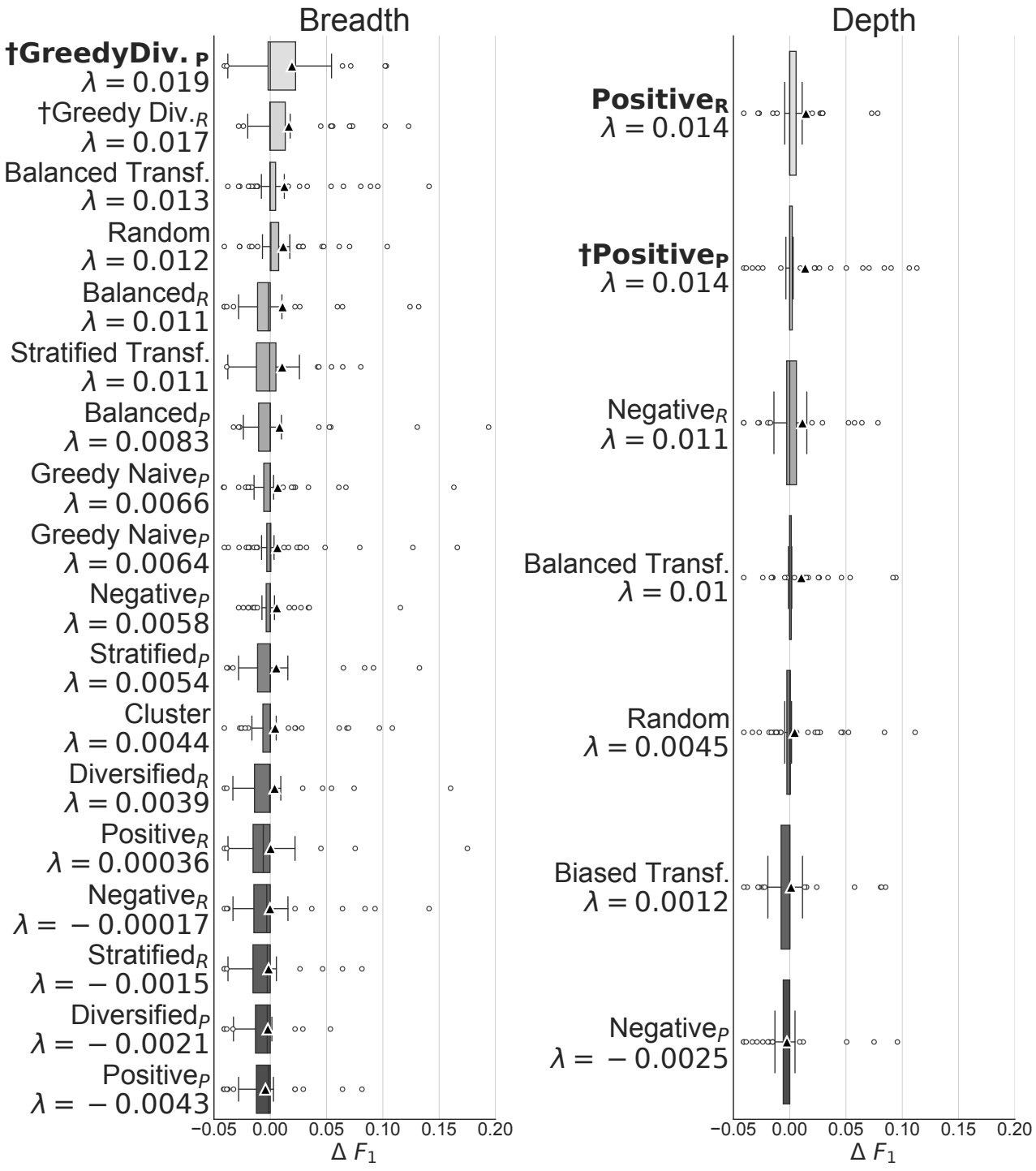


Figure 9.8: Gains/losses for queries selected using sampling strategies with breadth-first exploration method (left) and depth-first exploration method (right). Mean values (λ) are reported using ▲ and are presented beneath each label. Statistical significance ($p < 0.05$) is indicated by †. The most effective model for each exploration method is highlighted in **bold**.

9.4.2 Distribution of Selected Queries

RQ2.6

How does sampling affect the distribution of *selected query effectiveness* in the Query Transformation Chain framework?

To answer **RQ2.6**, we investigate the gains and losses of selected query effectiveness (compared to the original queries) as chosen by the LTR models. Figure 9.8 presents the distributions of selected queries using the same LTR model trained with query variations obtained with different sampling strategies. For the breadth-first sampling strategies, the Greedy Diversified$_P$ and Greedy Diversified$_R$ models select significantly more effective queries than the original. Meanwhile, for the depth-first sampling strategies, the Positively Biased Precision model was the only one to select significantly more effective queries than the original. Overall, the combination of breadth-first exploration with the Greedy Diversity$_P$ sampling strategy leads to a LTR model which selects the most effective queries.

Next, the differences between sampling strategies applicable in both the breadth-first and depth-first exploration methods are compared. Figure 9.9 presents the differences between selected queries using the same sampling strategies, but different exploration methods. This figure illustrates that the depth-first sampling strategies which rely on evaluation for sampling produce better training queries for the LTR models than the same breadth-first strategies. However, there is very little difference between the two exploration models for the Balanced Transformation and Random sampling strategies. This finding indicates that the exploration method does, in fact, play a role in the effectiveness of the LTR model, not just the criteria for sampling. This comparison highlights the significant impact the exploration method used to traverse the query variation space has on the sampling strategy. The depth-first exploration method can significantly improve the effectiveness of selected queries and is a key contribution of this chapter.

9.4.3 Relationship Between Sampled and Selected Distributions

RQ2.7

Are there relationships between the set of *sampled queries* and the effectiveness of the *selected queries*?

To answer **RQ2.7**, we investigate correlations between the distributions of sampled queries and corresponding distributions of selected queries for each sampling strategy. The first relationship we investigate is between sampled queries and queries selected using a LTR model trained on the sampled queries. Figure 9.10 presents this relationship for the best and worst queries using the breadth-first and depth-first exploration methods. The two worst-performing models for both exploration methods were trained on queries sampled using the Negatively Biased Recall strategy. Selected queries are weakly positively correlated with the sampled queries (Figure 9.11, right). On the other hand, the queries selected by the best performing models for the two exploration methods are moderately correlated with

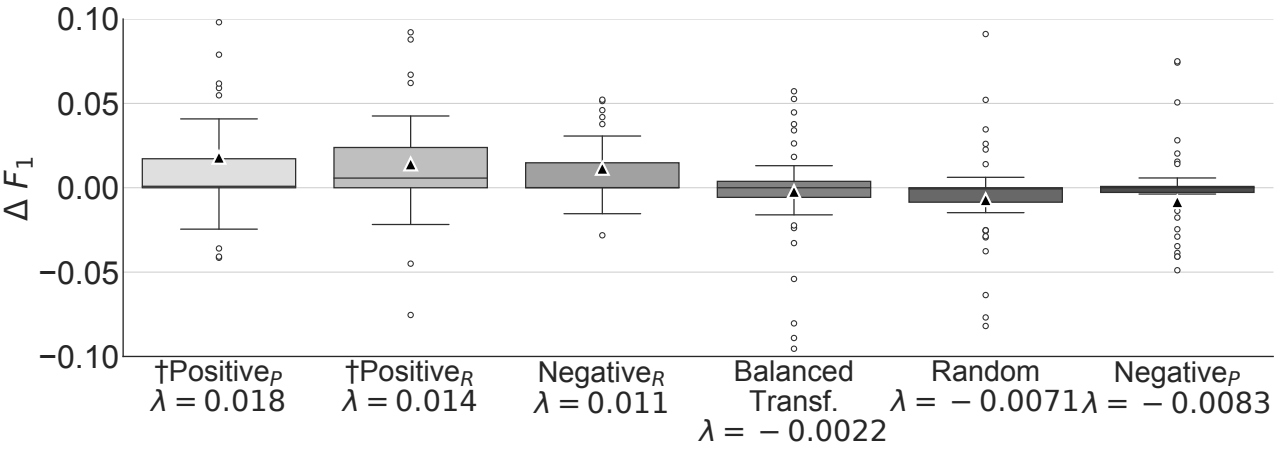


Figure 9.9: Differences between the depth-first exploration method and the breadth-first exploration method. Mean values (λ) in each distribution can be identified by a ▲ and are presented beneath each label. Statistical significance ($p < 0.05$) is indicated by †.

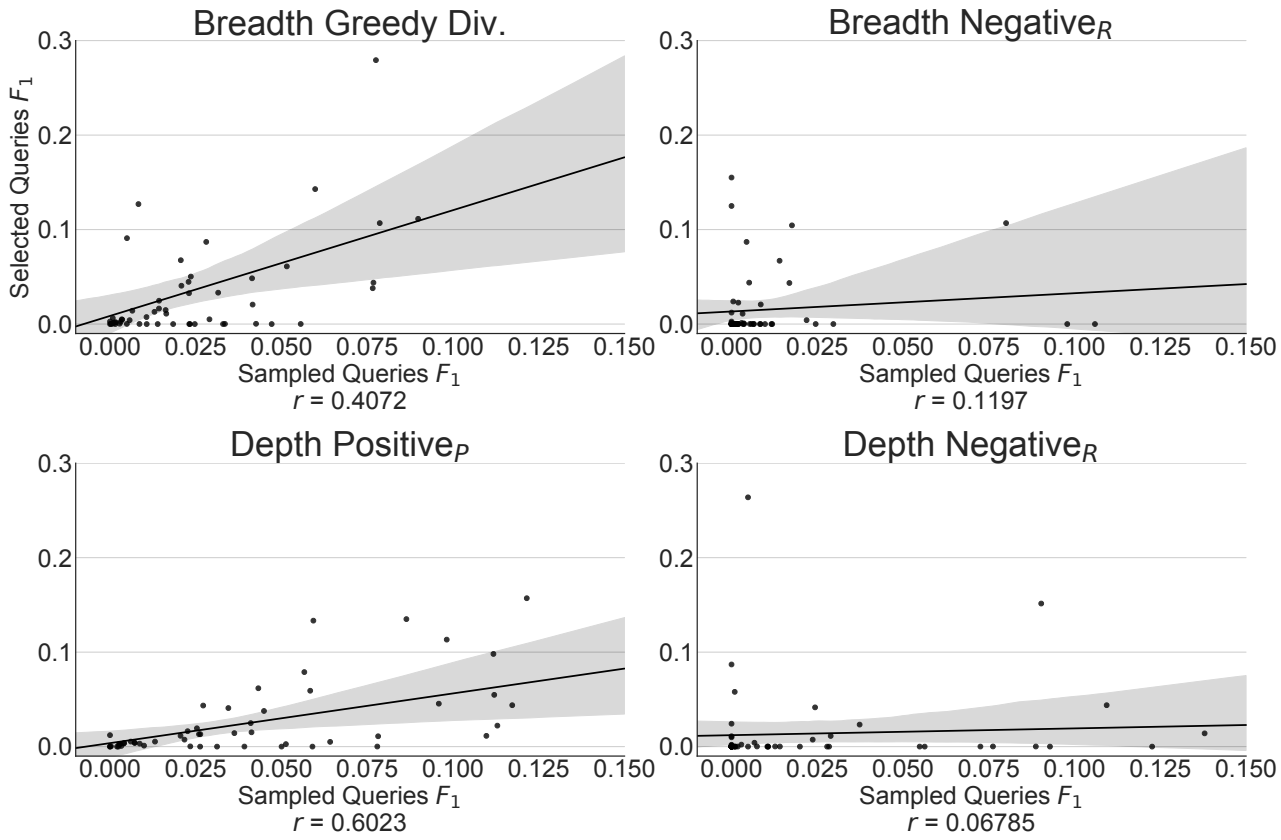


Figure 9.10: Relationship between the mean sampled query (i.e., the average value of queries from each sampled distribution; those in Figure 9.7) and selected queries (i.e., those in Figure 9.8) for the best and worst models for breadth-first and depth-first methods. The horizontal axis contains the average F_1 value of sampled queries. The vertical axis contains the F_1 value of selected queries. The linear relationship between these variables is presented by a solid black line. A central tendency with 95% confidence interval is overlaid beneath the line in grey. The Pearson's r correlation coefficient between sampled queries and selected queries is repeated.

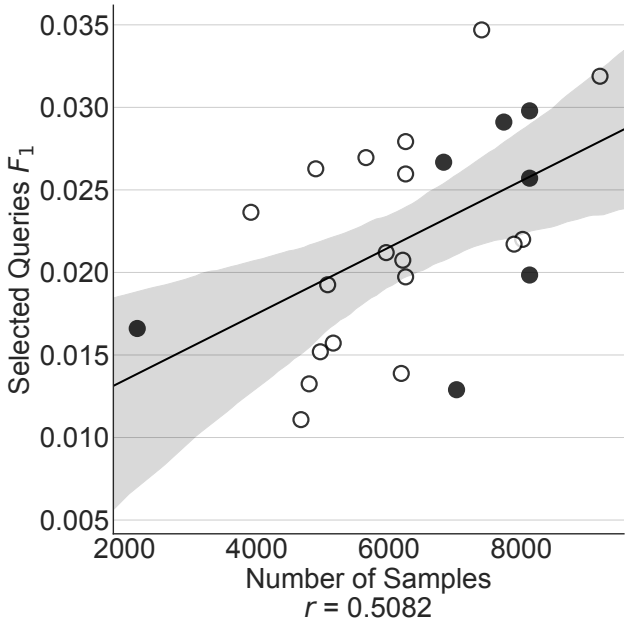


Figure 9.11: Relationship between all queries selected using a model trained on each sampling method (breadth-first methods coloured white, depth-first methods coloured black) and the number of samples used to train each model. The linear relationship between these variables is presented by a solid black line.

the sampled queries (Figure 9.11, left). These results suggest that the distribution of training data does impact the effectiveness of trained candidate selector models. The second relationship we investigate is between the number of sampled queries for each model and the effectiveness of queries selected using that model (Figure 9.11). A moderate positive correlation is observed, suggesting increasing the amount of training queries leads to LTR models which are better at selecting more effective queries.

In summary, there are relationships between the *sampled* queries and the *selected* queries. These relationships empirically suggest that (a) a well-distributed sampled set of queries, and (b) the number of queries sampled lead to more effective models.

9.5 Discussion

Although gains over the original queries by the best LTR models trained using different sampling strategies may seem small, some are statistically significant and consistent with our findings in the previous chapter. The context of these experiments is also important to recognise. For systematic review literature search, small decreases in the total number of non-relevant studies retrieved (i.e., several thousand) can have a significant impact on the total cost and time to complete reviews [133, 209].

For example, for a search that retrieves 5,000 studies of which 10 are relevant (typical of focused systematic reviews), an improvement in precision of 0.0005 leads to approximately a screening cost saving of AUD $5,000, not to mention considerable savings in time as well (estimates obtained inferring costs from Shemilt et al. [209]). These sampling experiments show that different sampling strategies significantly affect the selection of queries and that choosing how to sample queries to obtain

training data is highly important.

The breadth-first Greedy Diversified$_P$ sampling strategy is overall the most effective strategy for sampling queries. This sampling strategy led to a model which was able to select the highest performing queries overall automatically. When considering the depth-first exploration method, these experiments show that queries selected using models trained on the evaluation-based sampling strategies were more effective than the same breadth-first strategies. However, these models trained on depth-first sampled variations did not select more effective queries than the best model trained on breadth-first sampled variations. Meanwhile, the transformation-based and random samplers using the breadth-first and depth-first exploration methods led to models that selected similar positive effectiveness queries but are not statistically significant. This finding is somewhat unfortunate as the evaluation-based sampling strategies are more computationally intensive. However, it shows that to rank queries effectively, features are important and the distributions of queries regarding their effectiveness. That is, while exclusively sampling queries according to an evaluation measure (e.g., Positive Precision) can provide significant gains in effectiveness, other sampling strategies (e.g., Greedy Diversified) can provide even higher gains. Further, choosing a sampling strategy that provides more training examples is correlated with an effective candidate selector. When developing or choosing methods to sample queries, both query diversity and the number of samples should be considered to train an effective LTR model in the context of QTC.

9.6 Summary

To sample effective query variations for systematic review creation, we formalised two exploration methods for traversing the space of query variations and applied several sampling strategies within these exploration methods. A sampling of query variations in this domain is necessary because the Boolean queries used to search the literature for systematic reviews are verbose, complex, and the number of variations that these queries can produce is computationally infeasible to handle. While sampling cuts down on these computation costs, it leads to its own class of problems: different subsets of training data result in different effectiveness when training automatic models on these subsets. We empirically show these differences as we evaluate each sampling strategy. Our results suggest that sampling strategies that rely on the transformations or features of the query are the least effective for automatic training models. Moreover, the strategies that diversify the effectiveness of queries for sampling provide higher gains than those that rely on biasing effectiveness alone. While the gains of selected queries over the original queries may seem marginal, small changes to precision while maintaining recall lead to significant reductions in the total time and cost of the systematic review process [209].

Many professional domains also use Boolean queries to satisfy companies and institutions' information needs in various sectors. Typically, such a requirement falls under the purview of 'professional search' or 'eDiscovery' [151, 235]. Many of the sampling strategies and methods laid out in this work can be easily applied to other professional search domains such as patent or eDiscovery search,

which also require complex Boolean queries to satisfy a complex information need. The difficulty in formulating these complex Boolean queries can be offset by using automatic techniques to assist in the query formulation process. These automatic techniques often require training data. As the complexity of queries is high in these domains as well, applying these sampling strategies to train candidate selectors for use in QTC [192, 197] is a viable area for future work. Moreover, our sampling framework can be extended to add additional sampling strategies, for example, a class of sampling strategies that use word embeddings rather than features about queries (e.g., used for determining which words to expand or remove from queries).

Chapter 10

Tools to Support Query Refinement

This chapter describes tools that have arisen as a consequence of research into query automation refinement methods. These tools address specific problems in the pipeline of search strategy development relating to query refinement. Although the effectiveness of these tools has not been empirically evaluated (e.g., in a user study setting), some have been used to develop real queries for systematic reviews [45]. One of the research aims of this thesis is to demonstrate practical query automation tools to assist information specialists,

Tool Demonstration: Query Refinement

Examples of query automation tools for systematic review literature search to support information specialists in refining existng queries into more effective queries.

What follows is a description of two query automation tools that have been developed, and the motivation behind the implementation of the tool.

10.1 Description of Tools

The following tools are described in this chapter:

QueryVis Interface to support query visualisation and understanding. Provides a holistic perspective of a Boolean query, allowing modifications to be made to a query with more confidence.

QueryLens Assisted query refinement. Provides an interface on top of the Query Transformation Chain framework that generates candidate queries and selects the most likely query to improve effectiveness.

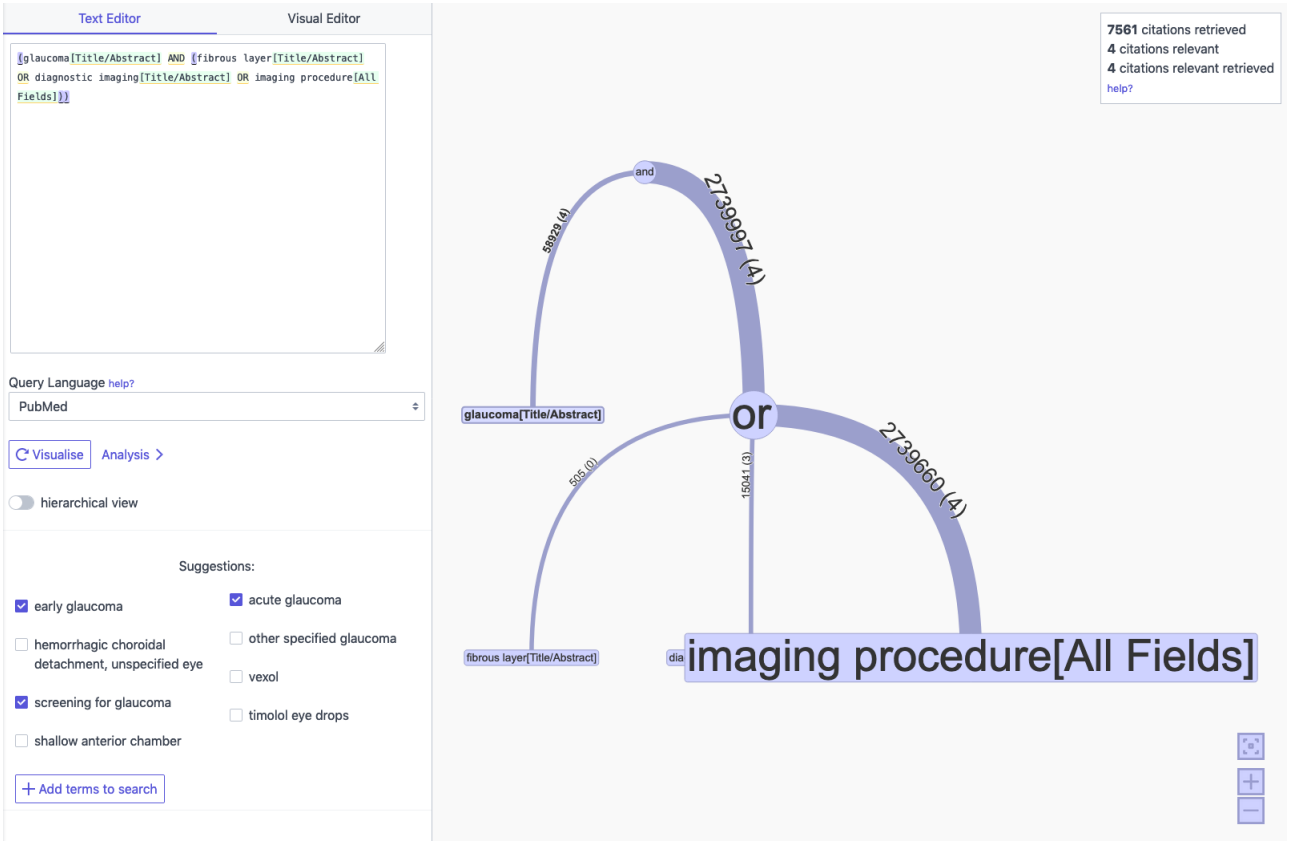


Figure 10.1: Manually refining a query using QueryVis. The top-left panel is where queries can be input to the tool. The bottom-left panel presents alternative terms found by the KeywordSuggestor tool. When keywords in the query are clicked, keyword suggestions fill the bottom-left panel.

10.1.1 QueryVis

QueryVis [202] (Figure 10.1) is a tool for visualising search strategies. QueryVis presents a query as a tree with information about the number of studies retrieved and the number of seed studies retrieved by each leaf node. These annotations along the edges of the tree assist information specialists by allowing them to gain a deep understanding of what a query retrieves and exactly how it retrieves it. Specifically, it allows the information to understand how many studies each clause retrieves (as opposed to how many studies overall the query retrieves) and how many seed studies each clause retrieves (as opposed to how many seed studies overall the query retrieves. This visualisation and annotation enable information specialists to decide what clauses they should focus on to refine or expand the query. QueryVis has been used extensively to construct search strategies for systematic review literature search, enabling information specialists to develop more effective queries in a shorter amount of time [45]. The KeywordSuggestion tool, as seen in Chapter 6 is integrated into QueryVis to allow information specialists to explore candidate expansion keywords for their query interactively.

The motivation behind this tool was originally to help visualise and understand the computational models described earlier in Part 1 and Part 2. However, it was later picked up in the systematic review community and seen as a useful tool for many there.

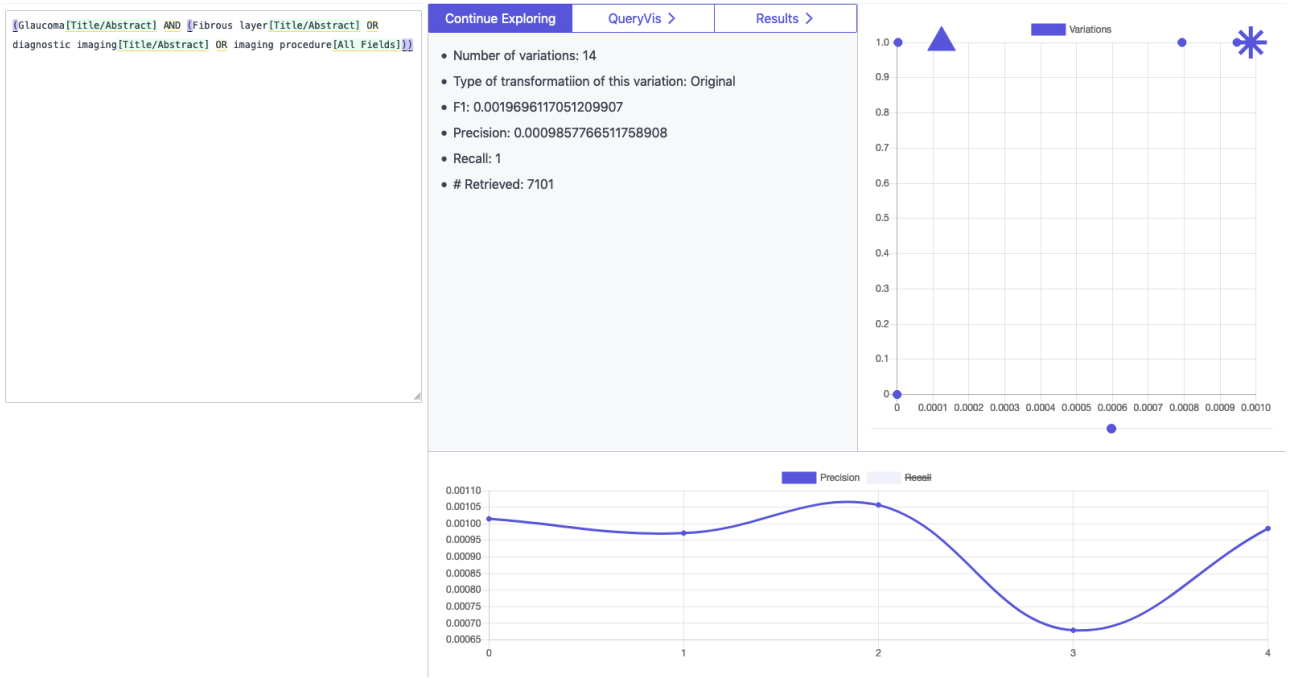


Figure 10.2: Automatically refining the query with QueryLens. The left panel is where queries can be input to the tool. The middle panel displays query statistics. The right panel plots all variations in terms of recall and precision. The sliding bar controls which variation is inspected. Clicking points in the plot also selects variations. The bottom plot shows how the query changes over time.

10.1.2 QueryLens

QueryLens [201] (Figure 10.2) is a tool that automatically generates variations for a query (by, e.g., adding or removing keywords, and fields, rewriting Boolean operators, or exploding MeSH headings). The middle panel provides statistics and evaluation results. The right panel plots each variation in precision-recall space. A point in this plot can be clicked to see information about the referring query in the middle panel. The variations produced by QueryLens are ranked as suggestions using a learning to rank model optimised for evaluation measures (e.g., precision, recall, F_β, etc.). The model used to produce suggestions is the candidate selector function used in the Query Transformation Chain framework, as described in the previous chapters.

The motivation behind this tool was to provide a system for information specialists to interact with an implementation of the theoretical framework described earlier in this part.

10.2 Case Study

This section describes a small case study for typical use cases of the previously described tools to demonstrate that these query automation tools address gaps in the search strategy development phase. The goal of this tool demonstration is to show how tools support information specialists in refining search strategies to be more effective.

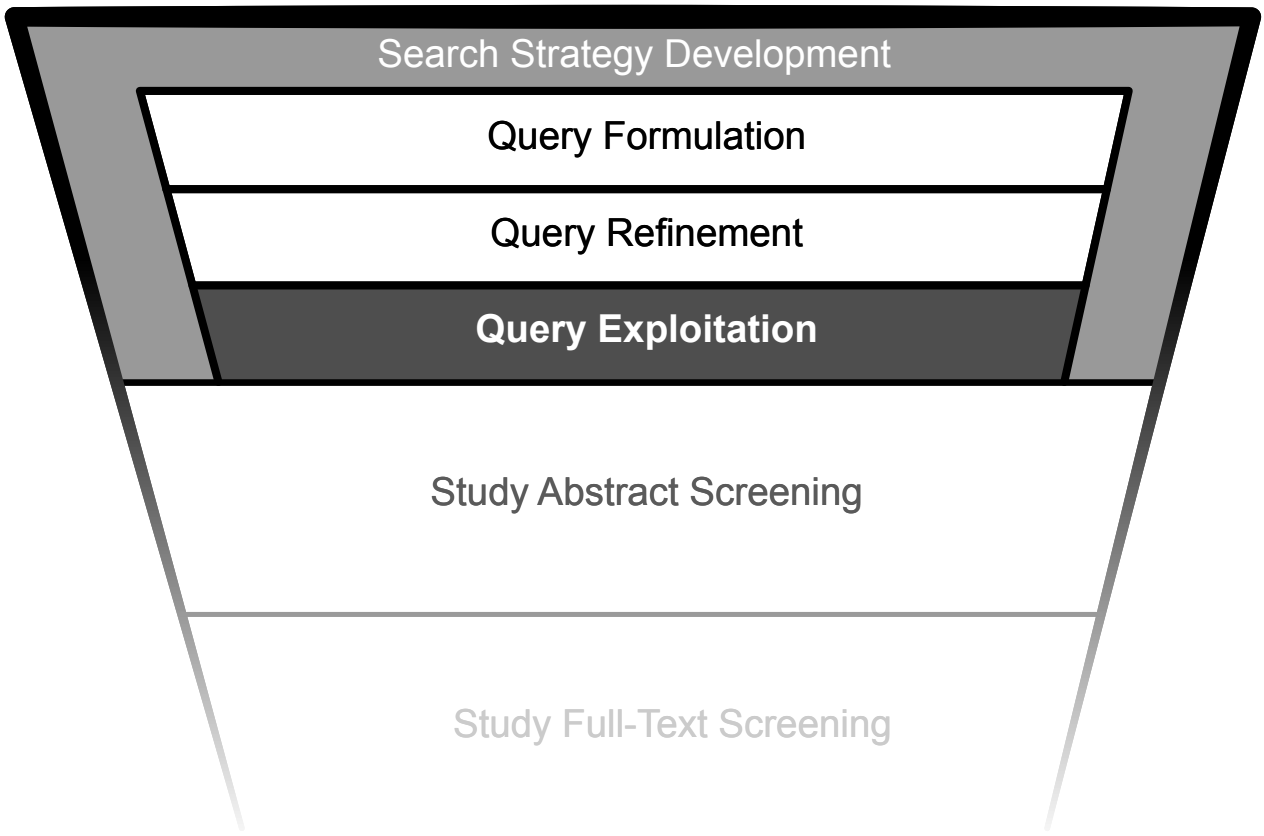
The QueryVis tool may be the first tool an information specialist will use once they have formulated an initial query. Here, they will see if they should remove any keywords that are retrieving far too

many studies relative to other keywords. The information specialist may also use QueryVis to expand a clause with other keywords, either manually, or using the KeywordSuggest tool embedded inside it.

Once satisfied with the query, the information specialist may then load their query into the QueryLens tool. Here, they automatically generate variations of their query that they may not have considered. This may lead the information specialist to identify a more effective refinement of their original query by interacting with the generation and selecting candidate queries provided by the QTC framework.

Part 3

Query Exploitation



How can the downstream phases of systematic review creation (specifically the study abstract screening phase) be expedited by exploiting queries alone? This is the kind of question that guides the research contained within Part 3. Indeed, this part of the thesis is interested in approaches that address the third research question of this thesis:

RQ3

Can the intrinsic characteristics of search strategies be exploited to further improve the effectiveness of systematic review literature search?

The chapters contained in this part seek to identify characteristics of the Boolean queries used for systematic review literature search for improving the effectiveness of search. Unlike the previous two parts in this thesis, this part does not present an overarching framework. Instead, it comprises two chapters that each investigate a different query exploitation method.

To this end, the first investigation of **RQ3**, in Chapter 11, seeks to extend the syntax of Boolean queries to support information specialists develop more refined searches (resulting in the retrieval of fewer studies). Specifically, this chapter investigates the use of fields that tighten the bound of studies that can be retrieved using the PICO framework.

The second investigation into this research question, in Chapter 12, seeks to exploit the syntax of Boolean queries to support both information specialists and researchers by ranking the retrieved studies. Specifically, helping information specialists to identify relevant documents to improve the search quickly, and helping researchers by allowing them to screen the most important studies first, thus allowing them to begin the following phases of systematic review creation earlier. In particular, a new ranking model is proposed that extends the intuitions of an earlier model of ranking for Boolean queries, coordination-level matching.

The lion's share of research in systematic review automation is focused on improving the abstract screening phase without considering the Boolean query (i.e., text mining, active learning, etc.). The following two chapters demonstrate considerable and significant gains in retrieval effectiveness that can be made by exploiting only the syntax of queries. The gains achieved through query exploitation may only serve to improve further the effectiveness of existing methods that do not consider the Boolean query (i.e., those listed in Chapter 2.2.3).

Both chapters present computational models that exploit queries to improve retrieval effectiveness. However, this part further differentiates itself from the previous two parts of this thesis, as no tools have yet been developed that implement the computational models presented in this part. Therefore, the tool demonstration chapter that has been present in Parts 1 and 2 is not included in this part. However, the code behind the chapters in this part has been published online. The code behind Chapter 11 is available at `https://github.com/ielab/SIGIR2017-SysRev-Collection`. The code behind Chapter 12 is available at `https://github.com/ielab/clf`.

Chapter 11

Integrating PICO into Retrieval

In this chapter, we evaluated the use of PICO annotations at both query and study level to restrict the number of retrieved studies for a given systematic review, while attempting to maintain the same recall achieved by the original search strategy (without PICO annotations). PICO is a popular framework used in medicine and clinical practice to formulate clinical questions. The framework promotes framing clinical questions according the four types of clinical information: (i) *population* the question refers to (e.g., males aged 20-50); (ii) *intervention* the population is administered with (e.g., weight loss drug); (iii) the criteria for *comparison or control* (e.g., controlled exercise regime); and (iv) *outcome* to measure (e.g., weight loss). Systematic review guidelines recommend using PICO in the development of search strategies, and most modern systematic reviews in medicine adhere to this guideline. While PICO is commonly used to formulate the search strategy (i.e., the conceptual approach), it is not used by the underlying retrieval system executing that strategy. This chapter aims to highlight that it should be, and guides investigation into **RQ3**:

RQ3

Can the intrinsic characteristics of search strategies be exploited to further improve the effectiveness of systematic review literature search?

11.1 Use of PICO to Search

An increasing amount of PICO annotations over biomedical research studies has become available. The Cochrane association (a global network of health professionals that provide access to high-quality medical information) is undergoing a sizeable manual annotation effort to create an extensive repository of PICO annotated MEDLINE articles[1], intending to make the content and data in systematic reviews more discoverable. Simultaneously, numerous automated methods have been developed that can annotate biomedical sentences with PICO categories with high accuracy. In this work, we use one such

[1] `http://community.cochrane.org/tools/data-management-tools/pico-annotation-project`

tool, called RobotReviewer [240]. RobotReviewer is a machine learning system that uses supervised distance supervision to train models that automate the extraction of PICO elements from systematic reviews. In a previous evaluation of RobotReviewer [240], it was found that the system extracted PICO annotations with a precision of 0.9 (top 3 annotations), outperforming other existing methods.

Search strategies aimed at finding studies to be included in systematic reviews are often formulated using the PICO framework. However, the information of which keywords and Boolean clauses in the search strategy refer to each PICO element is generally not included in the strategies themselves.

We next consider how to constrain the keywords (or Boolean clauses) to specific PICO elements. This could be achieved, for example, by appending operators to keywords that indicate which PICO element was elicited to decide the inclusion of the keyword itself in the search strategy. If this was the case, and if PICO annotations extracted from research studies where indexed as fields alongside the text and metadata of the studies themselves, then the matching between search strategies and research studies could be limited to matching keywords for the correspondent PICO annotations. For example, suppose that a search study identified *weight loss* as the intervention. If the query had no PICO annotations (as in current systems), then studies in which *weight loss* was the measurement outcome would be retrieved. Conversely, if the query included PICO annotations as a condition of the match (as in the method we investigate here), then studies in which *weight loss* was the measurement outcome would *not* be retrieved, and only studies for which the keywords *weight loss* have been annotated as an intervention would be retrieved. Note that keywords could have been annotated with more than a PICO element; similarly, some keywords may not be related to any of the PICO elements.

While the method proposed here may improve precision because the match is restricted to keywords used in the same (PICO) context in both the search strategies and the research studies, there is the definite possibility that it also harms recall. This is because of possible noise or errors introduced in the annotation of text for PICO (whether manually or automatically). Also, note that the construction of search strategies is an iterative process, and information specialists and researchers use initial searches to formulate the final search strategy. Thus, the conversion of existing search strategies with no PICO annotations to queries with PICO constraints may not retrieve relevant studies identified during search strategy formulation.

In the remainder of the chapter, we aim to empirically compare the effectiveness of using PICO elements in search strategies and research studies to compile a list of results to be screened for inclusion in systematic reviews.

11.2 Empirical Evaluation

To evaluate the effectiveness of the retrieval methods we use the Scells collection described in Section 2.3.1, as it contains labelled PICO annotations for queries. To extract PICO annotations from the research studies, we use RobotReviewer, version 3. Note that this version of RobotReviewer does not extract Control elements.

The collection was indexed using Elasticsearch version 5.1.1, including indexing studies with

respect to the PICO elements present in the text as separate fields. For each study, we indexed title, abstract and metadata (including publication date and medical subject heading terms) separately because search strategies may contain conditions that restrict the matching of keywords or Boolean clauses to only some of these elements.

We compared four approaches: (B) a baseline approach that use the original Boolean queries, (P) an approach that uses queries for which matches of specific keywords or Boolean clauses may be restricted to some PICO elements, and finally, (cP & fP) two approaches that both use the same PICO restrictions, but attempt to select the optimal PICO elements.

Our evaluation criteria consists of four levels of relevance since not all studies may have been retrieved by the queries we acquired. The studies are classified as such: excluded and not retrieved (*l1*), included and not retrieved (*l2*), excluded and retrieved (*l3*), and included and retrieved (*l4*) [205]. The evaluation described in this section is performed using *l3* and *l4*.

In the first approach (cP), we performed a 'coarse grain selection' of the PICO elements, in which, for each query, we found a combination of the PICO fields by iterating through each possible combination of fields and selecting the combination that lead to the lowest recall loss compared to the baseline. This shows the effect that maintaining only the PICO fields that least reduce recall has on precision. In the second approach (fP), we performed a 'fine grain selection' of the PICO elements, in which we use an oracle heuristic to remove the individual PICO elements (and not the entire fields) that caused a reduction in recall. This last approach is thus explicitly aimed at maintaining the same recall as the Boolean baseline, but improving precision with PICO, where possible.

During preliminary experiments we found that 13 queries matching on only PICO fields (P method) retrieve no studies: this may be due to the reasons outlined in Section 11.1. When the PICO-based strategy (P) did not retrieve any relevant studies but the baseline did, we removed the PICO annotations from the query (fallback).

As a retrieval task, we considered the task of retrieving studies for screening. The task aims to retrieve all relevant studies, while at the same time minimising non relevant studies; this thereby minimises the time researchers need to spend in reviewing the full text of studies. The balance between ensuring high recall and decreasing the number of not relevant studies to be screened is key for this task.

11.3 Results and Discussion

11.3.1 Analysis of PICO Annotations

Of the 26 million studies available in PubMed at the time, we found that RobotReviewer annotated with PICO 63% of the studies. All studies annotated with PICO contained a Population field; all studies except for one contained an Intervention field, and all studies except for four contained an Outcomes field. On average, we found that the Population field is 15.8 words in length, the Intervention field is 16.46 words in length, and the Outcomes field is 16.43 words in length. We found that many of the

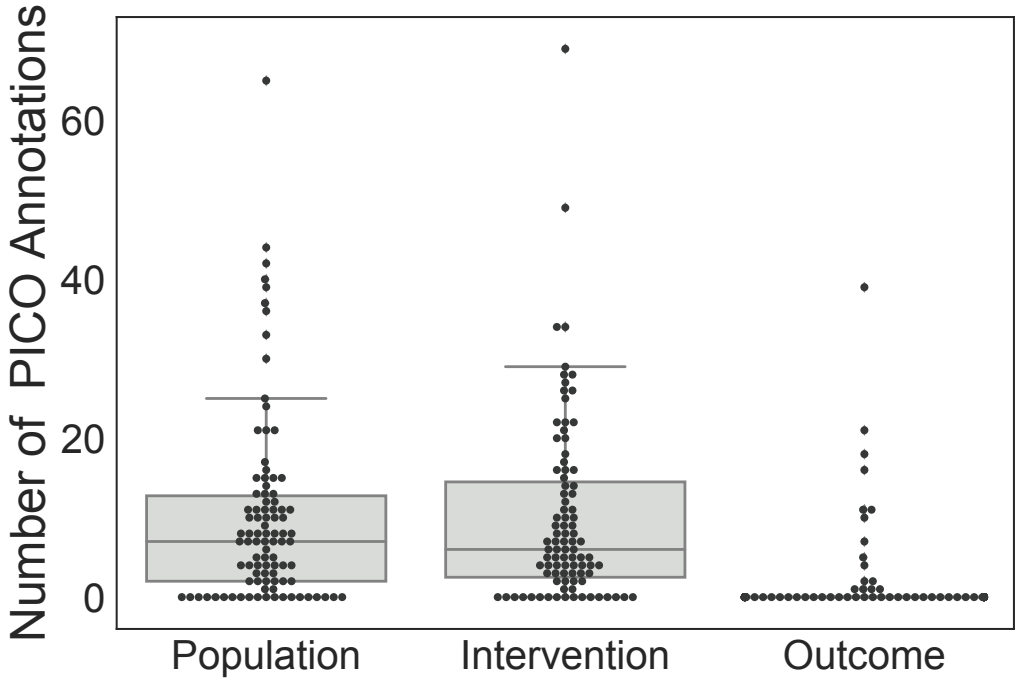


Figure 11.1: Distribution of the number of PICO annotations per query. 3 outliers have been omitted for Intervention (number of Intervention fields were 96, 113, and 380).

	Recall	Precision	F3	F1	F0.5	WSS
B	**0.7553**^{p}	0.0137^{pf}	0.0901^{pf}	0.0255^{pf}	0.0168^{pf}	0.0120^{pf}
P	0.6509^{bcf}	0.0215^{bc}	0.1128^{bc}	0.0375^{bc}	0.0258^{bc}	0.0206^{bc}
cP	0.7002^{pf}	0.0139^{pf}	0.0886^{pf}	0.0257^{pf}	0.0170^{pf}	0.0124^{pf}
fP	**0.7553**^{pc}	**0.0223**^{bc}	**0.1263**^{bc}	**0.0400**^{bc}	**0.0271**^{bc}	**0.0214**^{bc}

Table 11.1: Comparison of the effectiveness of the Boolean baseline (B), PICO based retrieval (P), coarsely selected PICO retrieval (cP), and finely selected PICO retrieval (fP). Statistical significance ($p < 0.01$) is denoted as ^{b} (wrt. Boolean) ^{p} (wrt. PICO), ^{c} (wrt. coarsely selected PICO) and ^{f} (wrt. finely selected PICO).

studies that had not been annotated with PICO comprised of title only (no abstract).

Of the 94 queries, all were annotated with PICO elements. On average, there was a similar number of Boolean clauses annotated with Population and Intervention fields (approximately 10.6 clauses per queries). However, there was a significantly lower number of clauses annotated as Outcome (2.3 on average). Figure 11.1 shows the distribution of the number of PICO annotations per query, across the three types of annotations. We found that the majority of queries contained a similar number of Population and Intervention annotations. Most queries have an Intervention annotation (80) and Population annotation (78), while only 18 queries contain an Outcome annotation.

11.3.2 Analysis of Retrieval Effectiveness

Table 11.1 reports the evaluation of the retrieval results obtained by the Boolean baseline (B) and the methods that exploit PICO annotations (P, cP, fP). For all measures, we computed statistical significance using a paired one-tail t-test. Overall, all PICO-based methods significantly improved

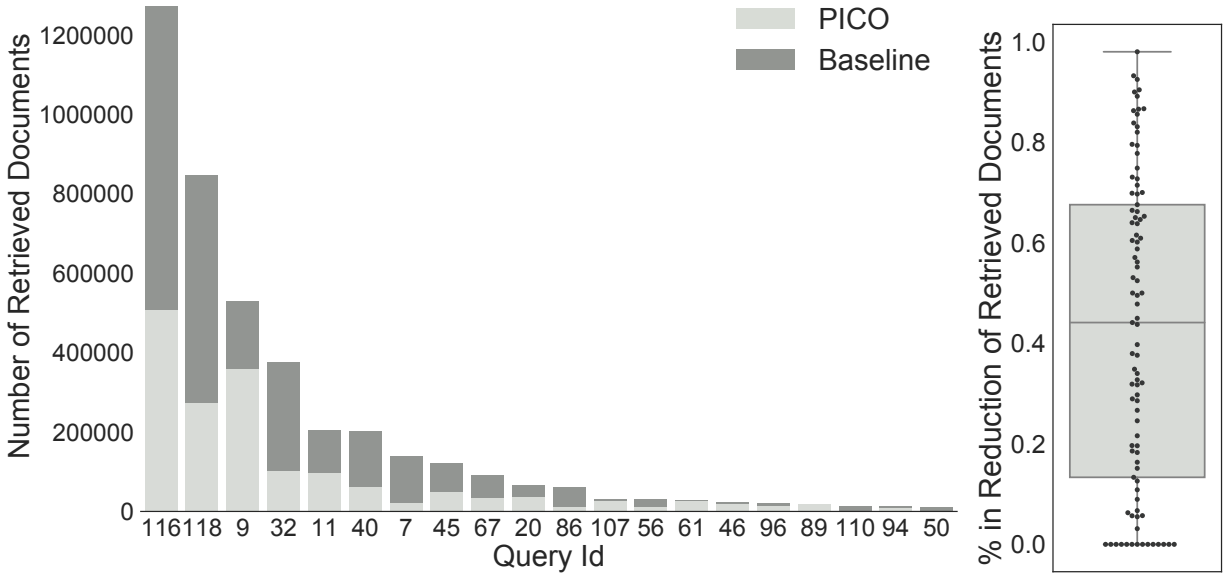


Figure 11.2: (Left) Number of studies retrieved using PICO search strategies overlaid over the number of studies retrieved using the baseline Boolean search strategies. The first 20 queries are shown; (Right) Percentage reduction of retrieved studies for Boolean search strategies versus PICO search strategies for every query.

precision over the baseline, except for cP, which exhibited no significant change in precision. The increase in precision for PICO-based methods translated to a saving of studies retrieved for screening, thus likely allowing potential time savings and cost reduction[2]. Figure 11.2 (left) illustrates these savings in the total number of studies retrieved by method P for the 20 queries with the highest number of relevant studies. Overall, method P retrieved 46.4% fewer studies to be screened than the baseline (B). Figure 11.2 (right) shows the amount of reduction P achieved for all queries in the collection. (A reduction of 0 indicates the use of *fallback* because the PICO query retrieved no studies). The savings in the number of studies to be screened is achieved in a simpler and more controllable way[3] (e.g., without resorting to parameter tuning) compared to previous studies that attempted to exploit PICO elements within the retrieval process, e.g. [24].

While the P method reduced recall compared to the baseline, a trade-off between a reduction in studies to be screened and the relevant studies to be retrieved can be obtained using fP. The results in Table 11.1 show that fP maintains the same recall as the baseline (by definition), while significantly increasing precision (42.58% reduction in the number of studies retrieved for screening).

By analysing the other evaluation measures reported in Table 11.1, we can observe that method fP provides the highest gains over the baseline for all measures, apart from recall which is unchanged.

While overall fP provides promising results, the automatic selection of the optimal PICO fields that guaranteed no loss in recall compared to the Boolean baseline is still an open challenge. Future work will investigate statistical predictors, e.g., query performance prediction, applied to the individual PICO elements in the Boolean queries clauses.

[2]For an estimation of cost reduction at the expense of increased bias due to the lower recall, refer to [209].

[3]The ability to carefully control and replicate the retrieval of studies in a systematic review has a key importance within protocols for systematic reviews compilation.

11.4 Summary

In this chapter, we investigated the effectiveness of exploiting PICO annotations in both search strategies (queries) and studies (documents) to narrow down the matches between keywords or Boolean clauses for retrieving studies for screening in systematic reviews.

The use of PICO was compared to a Boolean retrieval baseline that represents current search technology employed when performing retrieval for systematic review screening.

Our empirical evaluation showed the effectiveness of the PICO-based methods in reducing the number of false positives retrieved for screening (increase in precision). Despite this, PICO use showed a consequent decrease in recall compared to the Boolean baseline, unless the PICO query was modified to remove the PICO explicitly constraints that led to recall losses. However, the analysis of F_β-measure and WSS values suggested that, in general, recall losses were more than balanced by savings in terms of the number of non-relevant studies to be examined in the screening process. This outcome can drastically and immediately decrease the screening time of research studies for systematic reviews. Using an extensive test collection (both in documents and queries) and a tool to annotate abstracts with PICO automatically allowed us to show the large scale effects of searching using PICO elements matching in both queries and documents for systematic reviews screening. In contrast, previous studies have used significantly smaller data sets. These results have considerable implications for the future of medical systematic reviews. Firstly, we recommend that systematic review creation guidelines include annotating elements of the search strategy with PICO. Secondly, we highlight the need for an automatic tool to extract PICO elements from queries in existing systematic reviews: this would benefit the common process of updating existing reviews. As for the research question this chapter seeks to address,

RQ3

Can the intrinsic characteristics of search strategies be exploited to further improve the effectiveness of systematic review literature search?

We found that the field restriction characteristic of search strategies can be exploited to significantly reduce the total number of studies to screen, improving systematic review literature search effectiveness.

Chapter 12

Ranking Studies with Boolean Queries

This chapter proposes an extension to coordination level matching (CLM) by exploiting the query-document relationship with rank fusion. The background required to understand CLM and rank fusion are presented in Section 2.4.3 and Section 2.3.2 respectively. CLM is a ranking function originally proposed for Boolean queries that score documents using the occurrences of documents retrieved by different clauses of the query. The proposed extension, *coordination level fusion* (CLF), has many advantages over CLM that enable it to use multiple weighting schemes (rankers) and different fusion methods dependent on the Boolean clauses. We use CLF to rank studies in the screening prioritisation task of systematic reviews: this is the task of producing an ordering of studies from the likelihood of most relevant to least relevant. We further plan to study using a cut-off threshold tuned on training data to control when the screening of studies should be stopped based on the CLF retrieval score. The intuition behind CLF is the exploitation of the Boolean query structure to apply contemporary BOW-based ranking functions to Boolean queries. Studying this intuition is guided by **RQ3**:

RQ3

Can the intrinsic characteristics of search strategies be exploited to further improve the effectiveness of systematic review literature search?

The empirical results obtained on the CLEF Technology Assisted Review datasets [104, 107] show that CLF significantly outperforms existing state-of-the-art methods that consider similar settings, including the ranking method currently used in PubMed (a popular database to search for literature for systematic reviews).

12.1 Coordination Level Fusion

In this chapter, we propose Coordination Level Fusion (CLF), a novel method that extends traditional Coordination Level Matching (CLM) [126] by integrating rank fusion into the Boolean retrieval model by exploiting the semantic and syntactic aspects of the Boolean query.

CLM's intuition is that documents retrieved by many clauses should be considered more likely to be relevant. We note that this intuition is supported by axioms put forward in axiomatic analyses of ranking functions [76], and, more importantly for our work, it is similar to the intuition of rank fusion, namely, the *chorus effect*: the fact that "several retrieval approaches suggest that an item is relevant to a query" [237]. CLF leverages this intuition to further boost relevant documents higher up the ranking, using the agreement from multiple weighting schemes (rankers) *and* the agreement afforded by the structure of Boolean queries. Next, we describe the CLF method for ranking documents.

12.1.1 Producing a Ranking

We assume that a set R of rankings $r_1, r_2, \ldots, r_k$ is available for each atomic Boolean clause (i.e., a term in the Boolean query, see Figure 2.4) . These rankings could be produced by any weighting scheme available, e.g., IDF, BM25, etc. A ranking is an ordered list of documents: $r =< d_0, d_1, ..., d_k >$ with $s(d_i, r_j)$ representing the score of document d_i within ranking r_j. In CLF, these rankings are recursively fused, first at an atomic clause level, then at the level of (often nested) Boolean operators, until the highest level of the Boolean query is considered (typically represented by an AND operator): at this level, rankings are again fused together to produce a single, final ranking. This is achieved by applying the CLF fusion function to each document d as:

$$f_{CLF}(R,T,d) = \begin{cases} \sum_{r_j \in R} s(d, r_j) & \text{if } T = \texttt{AND} \\ |d \in R| \cdot \sum_{r_j \in R} s(d, r_j) & \text{if } T = \texttt{OR}/\texttt{Atomic} \end{cases} \tag{12.1}$$

where R is the set of rankings associated with the Boolean query clauses considered at the current level, and T is the type of Boolean operator applied. In this work, we consider T as either identifying an atomic clause or the `AND` and `OR` operators. The queries we consider do not have `NOT` clauses (therefore we do not have a fusion method for this operator). According to Equation 12.1, CLF performs CombSUM fusion [207] if the Boolean clause is `AND` ($T =$ `AND`). Likewise, CombMNZ fusion [207] is used when dealing with atomic clauses or the `OR` operator. Figure 12.1 visualises how fusion is performed for different Boolean clauses. When scoring exploded MeSH terms, the score provided by a weighting scheme is the summed score of each child in the subsumption (similar for phrases). Both CombSUM and CombMNZ boost the documents which multiple rankers estimate to be highly relevant (i.e., the chorus effect); however, CombMNZ at the `OR` and atomic levels is used to combat less accurate estimates of relevance (i.e., the dark horse effect). That is, documents, where only a single ranker estimates them as highly relevant, are not boosted.

12.1.2 Stopping Prediction

The task of stopping prediction in systematic review literature search is that: given a ranking of the set of documents retrieved by the Boolean query, at what position should screening stop? We model this task with an equivalent description: given a set of documents retrieved by a Boolean query, what

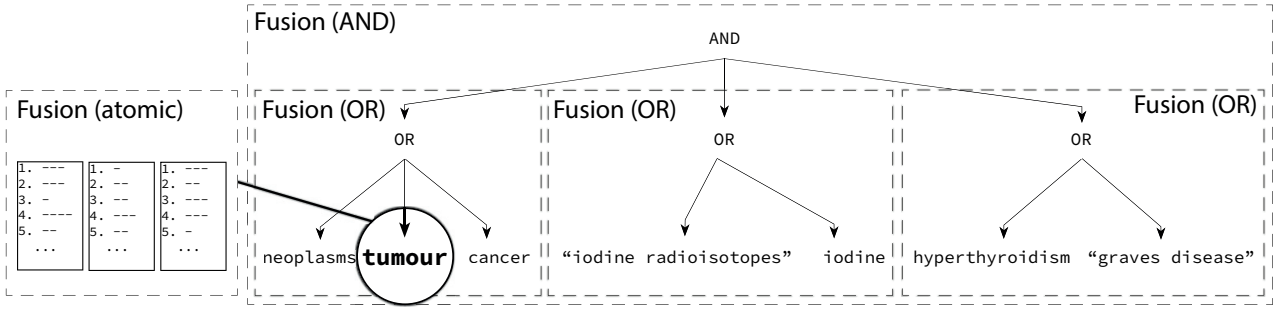


Figure 12.1: Bottom-up visualisation of the fusion of ranked lists using the CLF method. First one or more ranked lists of an atomic clause are fused, then the results of each Boolean clause are fused. Each clause that has fusion applied to is encapsulated in a dashed box. The nested clauses which it encapsulates are included inside it. Each applicable fusion method is labelled within each respective box. Note that all atomic clauses use the same range of weighting schemes: in this figure, only one is shown for space reasons.

```
(((oesophag*[All Fields] OR endocapsule[All Fields] OR microcam[All Fields] OR esophag*[All Fields] OR
 enteroscop*[All Fields] OR pillcam[All Fields] OR videocapsule*[All Fields]) AND ("Esophageal and
Gastric Varices"[Mesh Terms:noexp] OR (gastroesophag*[All Fields] OR oesophag*[All Fields] OR
oesophago gastric varix[All Fields] OR paraoesophag*[All Fields] OR oesophago gastric varic*[All
Fields] OR periesophag*[All Fields] OR perioesophag*[All Fields] OR esophag*[All Fields]))) AND
(23593613[pmid] OR 23029720[pmid] OR 22379346[pmid] OR 22346246[pmid] OR 22155754[pmid] OR 21814064[
pmid] OR 21624583[pmid] OR 21429016[pmid] OR 21372764[pmid] OR 21274889[pmid] OR 20490679[pmid] OR
20684186[pmid] OR 20682230[pmid] OR 20363433[pmid] OR 20135731[pmid] OR 20054320[pmid] OR 19809355[
pmid] OR 19743993[pmid]))
```

Figure 12.2: Example query formatted to be issued to PubMed for re-ranking. Constructing the query like above ensures only the documents specified (e.g., document number 23593613) are retrieved, and therefore re-ranked.

is the subset of documents which does not need to be screened? In this work, stopping prediction is performed by exploiting the scores of documents for each atomic term after fusion. Rather than setting a fixed cut-off on scores similar to participants in the CLEF TAR task [103], here a gain-based approach is used. Our approach is as follows: Given that researchers will screen documents starting at the first document and continuing to the next document for the entire list, they are accumulating gain from documents (equal to the document score) as they continue down the document list. Once enough gain from documents has been accumulated, they can stop screening. To model this, we use a κ parameter to control what percentage of the total gain a researcher can accumulate before stopping. Therefore, the stopping point becomes the position of the document in the ranked list, where the cumulative gain exceeds the total allowable gain. When κ is set to 1, no documents are discarded. In the task of screening prioritisation, where documents are assessed, κ is set to 1.

12.2 Experimental Setup

Empirical evaluation is conducted on the CLEF TAR 2017 and 2018 collections [104, 107]. These collections are chosen because they have many other methods that have been developed for the same tasks studied in this chapter. This enables a better comparison to previous methods. For the 2018

collection, evaluation is performed on topics from Task 2. Experiments are compared over two baselines: a ranking obtained by submitting queries directly to PubMed (explained in detail below), and a ranking obtained using CLM. The results of the CLF rankings are also compared to the rankings produced by the participants of the CLEF TAR task. Note that many of these participants do not rank directly according to the terms and structure of the Boolean query (while we do), and often consider the query as a bag-of-words, and incorporate terms from the title for re-ranking. Also, note that many participants used feedback from the relevance assessments and created active learning solutions. The comparisons between participants and our results only consider those reported not to use relevance assessments and do not use human intervention to rank (fully automatic, thus excluding active learning settings). In other words, we experiment considering the first round of retrieval.

All experiments are run using the QueryLab domain-specific Information Retrieval framework [191]. To obtain statistics for ranking documents, the documents retrieved by each query are fetched from PubMed and indexed by QueryLab. No stopwording or stemming is applied. The particular queries in this collection contain terms which are explicitly stemmed. Therefore, we use the PubMed Entrez API [189] to identify the original terms in documents from the explicitly stemmed term (this backward approach to stemming is to allow information specialists fine-grained control over their search). The title, abstract, MeSH headings, and publication date of each PubMed document is stored in four separate fields. When a title was not available for a document, the book title field was used instead; if no book title was available, the field was left empty (this replicates how searching on the title field works in PubMed).

The following weighting schemes are used in our experiments to produce document rankings for an atomic clause: IDF, TF-IDF, BM25, InL2 of Divergence from Randomness, PubMed, term position, text score, publication date, and document length. The PubMed weighting scheme uses a state-of-the-art learning to rank system of Pubmed [77]. The best match ranking system of PubMed uses a three-stage ranking system: first, documents are retrieved using the Boolean query; then, documents are ranked using BM25; finally, top-ranked documents are re-ranked using LambdaMART trained on click data, using document features such as document length, publication date, and past usage. Note that the PubMed best match ranker can *only* rank documents given a term or phrase, *not* a Boolean query. After the first stage, the Boolean query is translated into a bag-of-words type of query, similar to those seen in web search (it is often the case that the query translation results in fewer documents retrieved). Therefore, by embedding the PubMed ranker into CLF, the query translation step may be skipped entirely. The term position weighting scheme is defined as the relative position of a term in a document (0 if the term does not appear in the document). Publication date scores documents higher if the document is newer (accounting for recency, linearly). Document length scores documents higher the longer the document is. Text score weights documents by the fields a term appears in: for example, a document is scored higher if a term appears in the title and the body than if it appears only in the body. When queries are submitted to PubMed, they are modified to restrict them to only the PMIDs reported in the CLEF topic file (in order to account for minor discrepancies in retrieval after different periods, see Figure 12.2 for an example), and set the retrieval mode in PubMed to 'relevance'

in order to obtain a ranked list of documents by relevance (instead of the default ranking by publication date). Before fusion for any clause, ranked lists are normalised using min-max normalisation. Z-score and softmax normalisation were also considered, however through early empirical testing, min-max normalisation provided the consistently higher effectiveness compared to z-score and softmax. When there are ties in the ranking, the document with a more recent publication date is ranked higher. The different modifications made to CLF used in this chapter are taxonimised below:

CLM – The basic form of coordination level matching using the approach described above.

CLF+PubMed – CLF, using the PubMed ranker via the PubMed Entrez API.

CLF+weighting – CLF, using the weighting schemes described in the paragraph above (excluding the PubMed weighting scheme).

CLF+weighting+PubMed – CLF, using all of the weighting schemes from **CLF+weighting** in addition to the PubMed ranker from **CLF+PubMed**.

CLF+weighting+qe – CLF, using all the weighting schemes from **CLF+weighting**, but with a naïve query expansion method using terms from the topic titles and terms specific to DTA systematic reviews (obtained from an information specialist). Here, two additional Boolean `OR` clauses are constructed, each containing terms from the title and DTA specific terms respectively. Terms from the title have stopwording and Porter stemming applied.

CLF+weighting+PubMed+qe – CLF, using all of the weighting schemes from **CLF+weighting**, in addition to the PubMed ranker from **CLF+PubMed**, and the approach to query expansion from **CLF+weighting+qe**.

12.2.1 Evaluation

Evaluation is performed differently depending on the task. For the screening prioritisation task, rank-based measures are used. For comparison between the CLEF TAR participants (of which we acquired the runs), the MAP measure is included. The nDCG measure is included as a more realistic model of user behaviour. Reciprocal rank (RR) is used to demonstrate the effectiveness of systems in an active learning scenario (to show how soon the first relevant document would be shown and an update to the ranking potentially triggered). Precision after R documents (Rprec) is used to show the theoretically best possible precision obtainable in the stopping task, along with last relevant (Last Rel) that reports at what rank position the very last relevant document was shown. Participant runs are chosen for comparison if they are a fully automatic, unsupervised method, which does not use the training data or explicit relevance feedback, and do not set a threshold (as categorised in the TAR overview papers [104, 107]). Note that the tables in the CLEF TAR overview papers contain errors regarding these aspects; instead, each participant's papers were considered to determine which runs to compare our methods to individually directly. For the stopping prediction task, several

standard set-based measures are used: precision, recall, $F_{\beta=\{0.5,1,3\}}$, total cost, and reliability [58]. Reliability is a loss measure (i.e., where smaller values are better) specifically designed for the TAR task. It has two components: $loss_r = 1 - (\text{recall})^2$ and $loss_e = (n/(R+100) * 100/N)^2$, where n is the number of documents retrieved, N is the size of the collection, and R is the total number of relevant documents. Therefore, Reliability $= loss_r + loss_e$. Participants runs are chosen if they are fully automatic, supervised or unsupervised (thus we consider approaches that used training data), do not use explicit relevance feedback, and do set a threshold. Runs are evaluated using `trec_eval` or the evaluation scripts provided by the CLEF TAR organisers, where applicable.

When used for predicting when to stop screening, κ is tuned on training queries using a grid search to determine the best value. The parameter space searched in these experiments is $\{0.05, 0.075, 0.1, 0.2, 0.3, 0.4, 0.5, 0.75, 0.9, 0.95\}$. Note that κ can be set at a clause-level; therefore, it can be adaptive based on the clause. We leave learning an adaptive κ for future work, and here we fix κ to a set value across all clauses.

12.3 Results

12.3.1 Screening Prioritisation

Tables 12.1 and 12.2 present the results of the screening prioritisation task for the 2017 and 2018 CLEF TAR collections. Comparing CLM to CLF (without query expansion), CLF is statistically significantly better than CLM in all of the evaluation measures presented in both 2017 and 2018 tables (using a two-tails t-test where $p < 0.05$). Comparing the CLM and CLF methods to the state-of-the-art PubMed ranking, CLM is often statistically significantly worse than the PubMed ranker, whereas some CLF-based methods can perform statistically significantly better than the PubMed method. Next, the best performing CLF method (**CLF+weighting+PubMed+qe**) and the best performing CLEF participant method for each year is compared. For 2017 topics, the best-performing methods are Sheffield-run-2 (documents ranked with TF-IDF vector space model using terms from the topic title and terms extracted from the Boolean query), and Sheffield-run-4 (same as Sheffield-run-2 except a PubMed stopword list is used) [5]. The CLF method does not perform statistically significantly better than these two methods in any evaluation measure considered (however in all measures apart from MAP and last relevant, CLF is better). For 2018 topics, the best performing method is Sheffield-general-terms (same as Sheffield-run-4 from 2017, however terms specifically designed to identify systematic reviews are added to the query) [4]. Comparing this method to CLF, the CLF method performs statistically significantly better in RR (and has gains in all evaluation measures apart from last relevant). Overall, CLF can obtain the highest MAP for 2018 topics, and the highest overall nDCG, RR, and Rprec for both 2017 topics and 2018 topics performing statically significantly better than the state-of-the-art PubMed ranker.

	MAP	nDCG	RR	Rprec	Last Rel
PubMed	0.1597	0.5378	0.4292	0.1786	2974.00
CLM	0.0483*†	0.3941*†	0.1344*†	0.0415*†	3763.76*
CLF+PubMed	0.1313	0.5129	0.3722	0.1387	3119.06
CLF+weighting	0.1494	0.5247	0.4213	0.1696	3307.76
CLF+weighting+PubMed	0.1643	0.5422	0.4028	0.1754	3048.10
CLF+weighting+qe	0.1960	0.5735	0.5326	0.2239	3301.73
CLF+weighting+PubMed+qe	0.2165*	**0.5939***	**0.6037***	**0.2302**	3028.03
Sheffield-run-1	0.1700	0.5404	0.3644	0.1788	2678.33
Sheffield-run-2	**0.2183**	0.5930	0.5085	0.2190	2441.70
Sheffield-run-3	0.1986	0.5770	0.4700	0.2115	2404.96
Sheffield-run-4	0.2179	0.5937	0.5099	0.2185	2382.46*
ECNU-run1	0.0905*†	0.4517*†	0.1849*†	0.0907*†	3633.16*
QUT-bool	0.1293	0.4221*	0.3465	0.1535	1972.20*†
QUT-pico	0.1197	0.4067*	0.3088	0.1565	**1873.53***†

Table 12.1: Results for CLEF TAR 2017. The first row of results is obtained by issuing queries to PubMed, the next set of rows is are results of the various configurations of CLF, and the last set of rows are the relevant runs from participants for that year. Two-tailed t-test between the PubMed ranker and the other methods with $p < 0.05$ is indicated by $*$ and $p < 0.01$ by †.

	MAP	nDCG	RR	Rprec	Last Rel
PubMed	0.1918	0.5971	0.5085	0.2131	**3479.40**
CLM	0.0483*†	0.4413*†	0.1338*	0.0316*†	7194.76
CLF+PubMed	0.1734*†	0.5938	0.4942	0.2002	6363.13
CLF+weighting	0.2012	0.6186	0.5331	0.2139	6061.06
CLF+weighting+PubMed	0.2363*	0.6390	0.5289	0.2435	5937.93
CLF+weighting+qe	0.2397*	0.6501	0.5969	0.2662*†	5931.13
CLF+weighting+PubMed+qe	**0.2722***†	**0.6767***†	**0.6649**	**0.2882***†	5743.26
ECNU-TASK2-RUN1-TFIDF	0.1415*	0.5682	0.4212	0.1862	7173.00
sheffield-general-terms	0.2584*	0.6495*	0.4723	0.2779*	5519.20
sheffield-query-terms	0.2243	0.6184	0.4012	0.2425	5736.70

Table 12.2: Results for CLEF TAR 2018. Presentation of results and statistical significance is indicated the same was as in Table 12.1.

12.3.2 Stopping Prediction

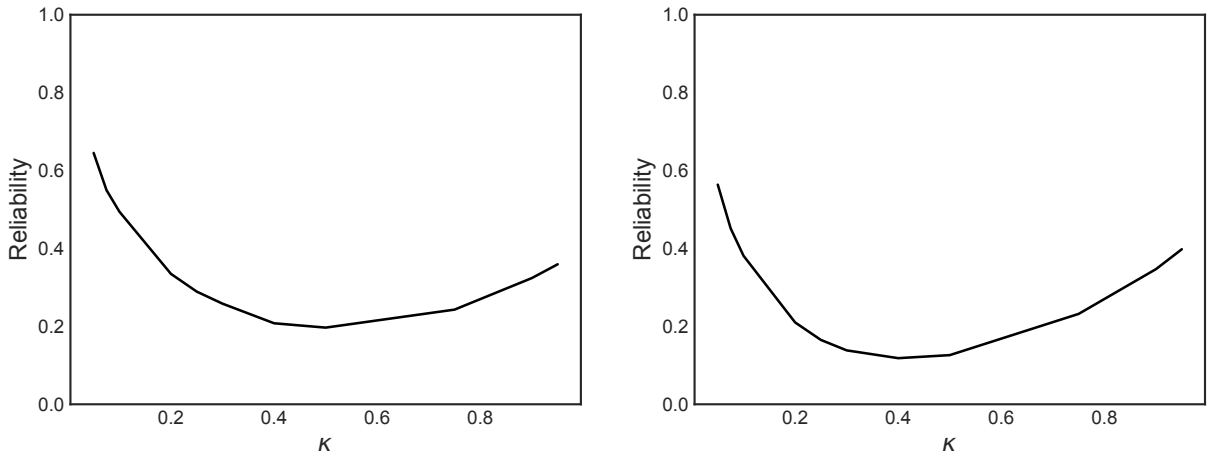
Tables 12.3 and 12.4 present the results of the stopping prediction task using the cut-off parameter κ. A κ value of 0.4 through parameter tuning on training data was found to provide the least loss in Reliability and was therefore chosen for the test queries for both 2017 and 2018. Results of the parameter tuning process on the training portion of the CLEF 2017 and 2018 topics are presented in Figure 12.3. The CLF method used in this task was **CLF+weighting+PubMed+qe** as it obtained the highest performance on the screening task. Examining first Table 12.3, CLF obtains the highest precision, F_1, $F_0.5$, F_3, and lowest loss in reliability. CLF also obtains the second-lowest total cost and maintains both a low total cost and reliability for this set of queries. Losses in recall are within a tolerable threshold [58].Table 12.4, reveals similar results to the 2017 topics. Significant improvements

	Precision	Recall	F_1	$F_{0.5}$	F_3	Total Cost	Reliability
No stopping	0.0415	**1.0000**	0.0752	0.0505	0.2345	3918.70	0.5441
CLF/0.4	**0.1040***†	0.7836*†	**0.1545***†	**0.1186***†	**0.3286***†	1324.63*†	**0.1259***†
ecnu-run2	0.0397	0.7075*†	0.0696	0.0478	0.2085	**1000.00***†	0.4445
ecnu-run3	0.0399	0.7164*†	0.0700	0.0480	0.2102	**1000.00***†	0.4433
sis.t1	0.0461*†	0.9868	0.0834*†	0.0561*†	0.2544*†	3435.03*†	0.4453*†
sis.t1.5	0.0482*†	0.9727*	0.0865*†	0.0585*†	0.2596*†	3165.56*†	0.3843*†
sis.2	0.0517*†	0.9531*†	0.0919*†	0.0626*†	0.2684*†	2824.6667*†	0.3309*†
sis.t2.5	0.0577*†	0.9382*†	0.1007*†	0.0695*†	0.2815*†	2536.80*†	0.2724*†

Table 12.3: Results of CLF for stopping prediction for CLEF TAR 2017. The first row are the results from the original queries, the second row is when CLF with $\kappa = 0.4$. Two-tailed t-test between the original results and the other methods with $p < 0.05$ is indicated by $*$ and $p < 0.01$ by †.

	Precision	Recall	F_1	$F_{0.5}$	F_3	Total Cost	Reliability
No stopping	0.0471	**1.0000**	0.0851	0.0573	0.2622	4640.23	0.3981
CLF/0.4	**0.1225***†	0.8582*†	**0.1827***†	**0.1400***†	**0.3794***†	**1140.06***†	0.4330*†

Table 12.4: Results of CLF for stopping prediction for CLEF TAR 2018. The first row are the results from the original queries, the second row is when CLF with $\kappa = 0.4$. Significance is indicated the same as in Table 12.3.



(a) 2017 topics.

(b) 2018 topics.

Figure 12.3: Tuning the κ parameter on the training portions of the 2017 (left) and 2018 (right) CLEF TAR topics. Lowest value for both plots is 0.4.

over the original queries in terms of precision, F_1, $F_{0.5}$, F_3, and total cost, with a tolerable reduction in recall can be observed. However, the Reliability of this set of queries is higher (thus worse). Given that the total cost is low, this indicates that the $loss_r$ component of Reliability does not decrease at the same rate as $loss_e$ increases for these topics. No participants contributed a comparable run to the 2018 TAR task; therefore, no comparisons to other systems can be made for this collection.

While there is a drop in recall, real monetary savings are associated with the increase in precision. Across the 2017 and 2018 topics, the CLF method provides savings between approximately AUD$5000 and AUD$12,000, according to estimates reported by McGowan et al. [133] when considering double screening.

12.4 Summary

In this chapter, a novel approach to ranking documents for systematic review literature search using rank fusion applied to coordination level matching was presented. The dubbed Coordination Level Fusion (CLF) method outperformed the current state-of-the-art for two different tasks. For the screening prioritisation task, CLF significantly outperformed the existing PubMed ranking system and participants that submitted comparable runs to the CLEF TAR tasks. The screening prioritisation task results demonstrate the applicability of CLF to systematic review literature search when prioritisation is considered, and suggest it may also be applied to obtain an effective early ranking in settings that consider active learning. CLF could significantly reduce the cost of screening with tolerable losses in recall for the stopping prediction task. The results of the stopping prediction task demonstrate the applicability of CLF to specific systematic reviews where total recall is not essential, such as in rapid reviews [130].

There are many aspects of CLF that require further investigation. First, we propose to study the effectiveness of CLF within an active learning setting. In this context, CLF can be used as the first ranker, before relevance feedback is collected. Then, feedback could be further weaved into CLF by devising and integrating weighting schemes. We also plan to investigate the use of CLF as a method for query performance prediction (e.g., as a post-retrieval predictor using reference lists [212], or as a candidate selection function in query transformation chain frameworks [197]). In terms of extending CLF, the weighting schemes themselves can be weighted (i.e., one weighting scheme may have more importance over others); e.g., using the linear combination fusion method [237] which assigns weights to each ranker being fused. The problem then is learning the weight to assign to each weighting scheme (ranker) used for rank fusion. Rather than using fusion methods like CombMNZ, it is foreseeable to use a different combination of weights for each Boolean clause considered. As for the research question this chapter seeks to address,

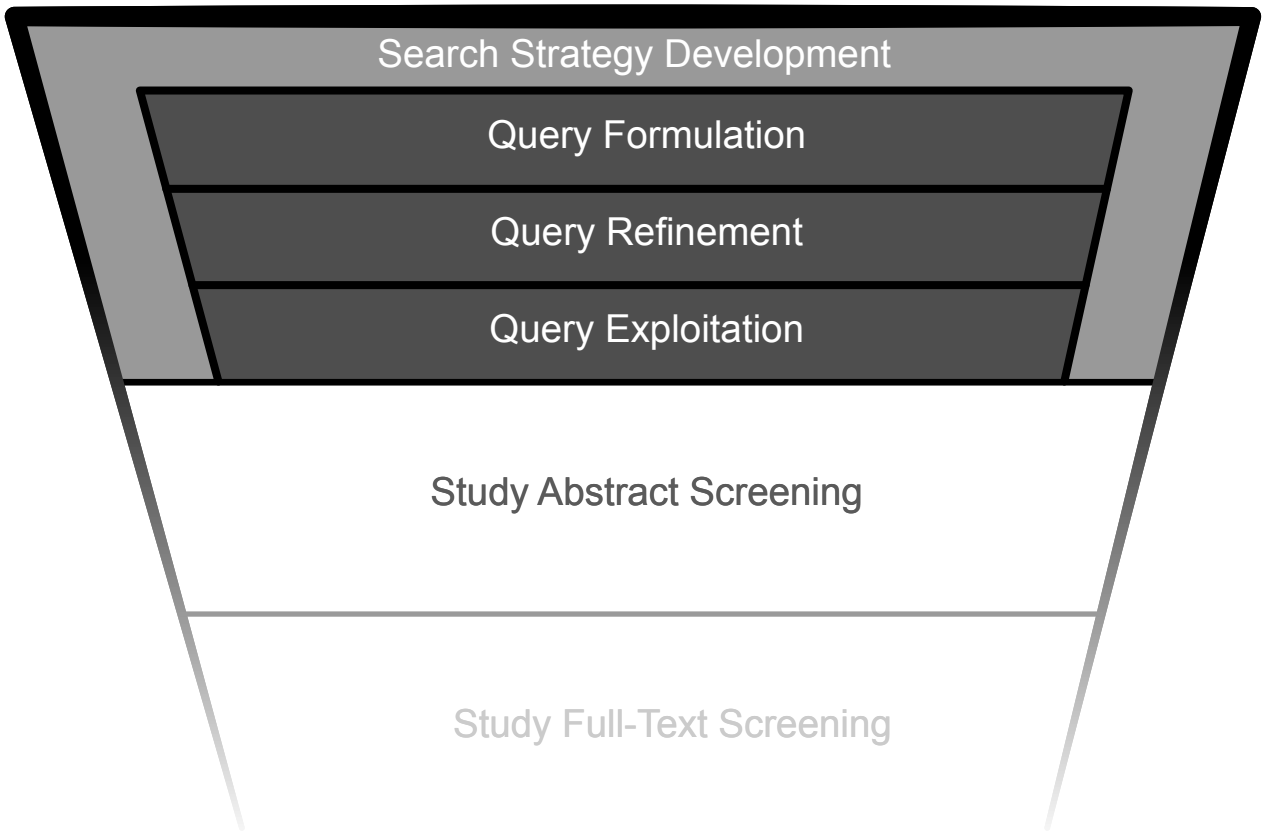
RQ3

Can the intrinsic characteristics of search strategies be exploited to further improve the effectiveness of systematic review literature search?

We found that the syntax of a Boolean query can be exploited to produce rankings of studies that are more effective than methods that use BOW queries.

Part 4

Conclusion



Chapter 13

Research Overview

The primary research focuses of this thesis were: (i) the investigation of methods to support the *development* of search strategies, (ii) ways to *exploit* the search strategy for screening, and (iii) tools to *assist* those who work with search strategies. To this end, this thesis was split into three parts, each covering one of the three research questions that addressed different aspects of the research focuses:

RQ1

Can search strategies be automatically formulated and if so, are they comparable in effectiveness to search strategies manually formulated by information specialists?

RQ2

Can manually formulated search strategies be automatically refined to produce more effective search strategies?

RQ3

Can the intrinsic characteristics of search strategies be exploited to further improve the effectiveness of systematic review literature search?

Parts 1 and 2 of this thesis also included a demonstration of tools that have been developed, often as a by-product of the computational methods proposed that address one of the research questions. These tools were a large component of this thesis, and were exemplified by small case studies that demonstrated the use of the tools by an information specialist:

Tool Demonstration

Examples of query automation tools for systematic review literature search to support information specialists in developing more effective queries.

This chapter provides an overview for each of the three main parts and the key findings related to each of the three research questions. What follows is an overview of the three main parts, summarising the findings and contributions from each.

13.1 Formulation of Boolean Queries

The first part of this thesis presented two methodologies for query formulation used by information specialists to develop Boolean queries for systematic review literature search. The chapters within this part provided computational adaptations of these methodologies to permit the automatic formulation of Boolean queries. The two methodologies (and their computational adaptations) were the conceptual and objective approaches. The conceptual approach is currently the most common way for information specialists to develop queries. Meanwhile, the objective approach is not as widely used, but is gaining traction within the systematic review community. As the conceptual method was demonstrated to rely more on the information specialist's intuitions and knowledge to develop queries, no computational adaptation could perfectly capture this subjectivity. However, as the objective method is more clearly defined in terms of methodology, the associated computational adaptation more closely matched the approach.

The computational adaptations of both of these query formulation methodologies allow for a large-scale comparison between them. As such, this part of the thesis also contained a comparison between the two computational adaptations, and compared queries formulated by each of the computational adaptations to manually formulated queries. The two computational adaptations also relied on seed studies in order to fine-tune the query formulation processes. A study was performed that investigated how sensitive to variation these two computational adaptations were to seed studies. The next section discusses these findings in more detail, with respect to the results of the study undertaken in Chapter 5.

13.1.1 Findings

Comparison between conceptual and objective. The first finding of Part 1 relates to the comparison between the computational adaptations of the conceptual and objective methods. Firstly, each adaptation was instantiated using terms or terms and phrases. There were minor differences between these two instantiations for the conceptual method, with the inclusion of phrases slightly increasing precision-oriented measures and reducing recall-oriented measures. This trend was also present for most of the objective instantiations. Between all the conceptual and objective instantiations, the objective method obtained significantly higher retrieval effectiveness over the conceptual method. However, this higher effectiveness came at a cost as the trade-off between precision and recall was higher (i.e., the gap between precision and recall) for the objective method, despite the higher effectiveness.

Comparison between computational adaptations and original. Both computational adaptations were unable to achieve similar retrieval effectiveness as the original queries (i.e., those formulated

manually by humans). All instantiations of both adaptations demonstrated a significantly lower precision. All but one instantiation demonstrated a significantly lower recall. The small case-study that was undertaken to refine the queries derived from the computational adaptations manually did show that the effectiveness of queries could be increased through query reduction alone. For the conceptual method, the recall increased with a small drop in precision. For the objective method, the precision increased with a small drop in recall. Both the precision and recall were not increased; thus, the automatically formulated queries were not as effective as the original queries even with manual refinement.

Sensitivity to seed studies. The computational adaptations of the conceptual and objective methods were highly sensitive to the initial seed studies. Most topics from each instantiation varied wildly in retrieval effectiveness depending on the set of seed studies used. This sensitivity has implications beyond the work contained in this thesis, extending to both manual approaches. Information specialists should be aware that the seed studies used to derive a query will have a considerable impact on the effectiveness of their resulting query.

13.1.2 Future Work

Perhaps an obvious next step with this work is to apply the Query Transformation Chain (QTC) framework from Part 2 to the automatically formulated queries derived within Part 1. At first glance, this may seem trivial. However, several factors must be considered before one undertakes this step and motivates why it was not undertaken within this thesis. Indeed, it would be naïve to attempt to simply apply the existing QTC framework to formulated queries and expect considerable gains in effectiveness automatically.

1. Firstly, consider the impact of seed studies on the variability of query effectiveness. Does one cherry-pick only the best query from a set of iterations of seed studies? Perhaps one can derive a metric to measure the quality of seed studies to inform which should be used for query formulation. However, this still does not accurately simulate the actual formulation and refinement processes. Instead, to fairly compare the effectiveness of a query's automatic formulation and refinement to a manually formulated query, the original seed studies are required. As there currently does not exist a test collection with this data, a fair comparison is not possible.

2. Secondly, consider that the training data for the QTC framework is automatically generated by sampling variations of high-quality queries. The experiments of Chapter 9 demonstrated that there already exists a sizeable negative example of an imbalance in training data that can lead to less effective models. As the automatically formulated queries are of low effectiveness to begin with, this negative example imbalance will be even larger, leading to less effective candidate selection, or requiring vastly more training data (which is already extremely expensive to create). Improving the effectiveness of the computational adaptations is required to obtain higher quality training data and effective QTC candidate selection models.

3. Finally, the existing QTC candidate selection models exhibit the same trade-off in precision and recall as the automatic formulation methods because they are optimising for a target evaluation measure. For example, consider the Term/Recall/MeSH instantiation of the objective computational adaptation. This method achieved the highest recall, but the lowest precision. One approach to refine a query constructed using the Term/Recall/MeSH method is to increase precision. However, it is still unclear how to develop a QTC selection model to conditionally optimise for one evaluation measure while preventing another's lowering. As a result, the existing framework will likely produce a new query with higher precision, but a lowered recall. Likewise, if a candidate selection model optimised for recall is applied, it is likely that the precision will decrease even further. Perhaps the most direct way of incorporating conditional optimisation could be to use or develop a measure that more closely models the search task. That is, a measure that correlates strongly with what an information specialist would consider acceptable. Such a measure would likely also be improved by incorporating seed studies, much like the query formulation models in Part 1.

13.1.3 Contributions

The contributions of Part 1 include:

1. **Computational adaptations of the existing manual conceptual and objective methodologies for search strategy development.** These adaptations support the automation of query formulation.

2. **A comparison of the two computational adaptations to original, human formulated, queries and a comparison of the two computational adaptations between themselves.** We found that queries can indeed be automatically formulated; however, these automatically formulated queries were not as effective as those manually formulated by information specialists. However, with minimal query refinement, the automatically formulated queries can become comparable to those formulated manually. The computational objective method was shown to be more effective than the computational conceptual method.

3. **A comprehensive analysis of how seed studies affect the sensitivity of the two computational adaptations.** This analysis showed that seed studies could indeed improve the effectiveness of both computational adaptations. However, the improved effectiveness is dependent on exactly which seed studies are used. That is, while gains in effectiveness are possible over not using seed studies, the use of seed studies can introduce a high amount of variability in the effectiveness of queries.

4. **The implementation of tools to assist information specialists to develop more effective queries.** These tools comprise (i) AutoFormulate, a tool that formulates queries for information specialists based on the computational methods developed in Part 1, (ii) KeywordSuggest, a tool for the identification of semantically related keywords for use as search terms in a query,

and (iii) AutoDoc, a tool for supporting the documentation of queries as search strategies for publishing alongside a systematic review, aiding in the reproducibility of the review.

13.2 Refinement of Boolean Queries

The second part of this thesis presented the QTC framework for fully automatic Boolean query refinement for systematic review literature search. The chapters within provided a theoretical overview of the framework, experimental evaluation of the framework, and an investigation into the sampling of training data on the framework's effectiveness. It was empirically demonstrated that the QTC framework produced more effective queries for systematic review literature search and that these queries could be automatically selected.

13.2.1 Findings

More effective queries. More effective queries can be found by exploring variations of high-quality search strategies using relevance assessments to guide the exploration. The effectiveness of query variations was in the range of 100%-800% higher than original queries. These results suggest that information specialists formulate queries *conservatively*. They must balance the number of retrieved seed studies with the total number of retrieved studies. This balancing act is difficult because it requires the information specialist to *make a prediction* about how many total relevant studies there are, whilst ensuring the total number of studies to screen is within an acceptable limit. What is meant by conservative query formulation is that information specialists are over-estimating the number of studies required to screen to find the majority of relevant studies. Another related finding as a result of this conservatism is that the automatically generated variations of queries obtained a higher *recall* than the original queries (note that it is often the case, and indeed expected, that more relevant studies will be identified through downstream activities such as snowballing). This finding means that while over-estimating the number of studies required to screen, they are also under-estimating the number of relevant studies that exist.

Automatically selecting more effective queries. Once established that more effective queries were possible, this led the investigation into the second finding: the *automatic* selection of more effective queries. To accomplish this, machine learning models were trained to optimise for different evaluation measures. These models demonstrated reasonable success, and it was shown that models optimised for different evaluation measures did indeed select query variations that were more effective in the respective measure. However, this led to a trade-off: the optimisation of one measure led to decreases in other measures. Despite this, the selected queries were often more effective than the original queries, given specific evaluation measures.

Impact of sampling on automatic selection. One major limitation with the investigation in Chapter 8 (Evaluation of Query Transformation Chain), however, was how training data was generated

(i.e., through generating query variations). Because this generation process produces exponential query variations, a sampling process needed to be applied. Chapter 9 (Sampling Query Variations) set out to investigate whether the way training data was sampled had an impact on the effectiveness of candidate selection models in the QTC framework. It was found that this certainly was the case and that the candidate selector's ability to identify better queries was dependent on the training data. While the strategy for sampling training data chosen in Chapter 8 did not negatively impact the candidate selector model, a better sampling strategy could have been chosen that likely would have improved the effectiveness of candidate selection further.

13.2.2 Future Work

The QTC framework comprises several elements: (i) the generation and sampling of training data, (ii) efficient and informative feature extraction, and (iii) candidate selection models. Each of these elements may be explored in greater detail, although perhaps the most worthwhile in terms of gains in effectiveness could be improved candidate selection models. These models were mentioned previously in the related work for Part 1. It is envisioned that models that can optimise for precision without degrading recall will be the most beneficial. One could go about this by modelling the trade-off in precision and recall as an optimisation problem.

Another area of future work related to the tools developed as part of both query refinement and query formulation is user studies. Such studies would allow us to measure the computational method's effectiveness from a user perspective and the user acceptance of the tools. In addition to evaluating these aspects of tools, it would also be useful to study whether information specialists can identify if a query is more effective than another query given limited information, for example, can queries be ranked by effectiveness by an information specialist given only the queries themselves?. These kinds of user studies may drive the development of new measures for predicting query effectiveness, for example, as a candidate selection function in the query transformation chain framework.

13.2.3 Contributions

The contributions of Part 2 include:

1. **Query Transformation Chain (QTC): A comprehensive framework for the generation and selection of more effective Boolean query variations for systematic review literature search.** This framework supports the automation of query refinement.

2. **An investigation into the QTC framework's effectiveness in generating and selecting more effective Boolean queries for systematic review literature search.** We found that more effective queries are capable of being refined out of existing high-quality queries. Not only were more effective queries possible, but it was found that these queries could be automatically identified.

3. **An investigation into how training data is sampled for training QTC candidate selection models and how training data impacts these models' effectiveness.** We found that differ-

ences in how training data was sampled led to significant differences in model performance and that training data that was sampled in a way that maximised diversity led to the most effective model.

4. **The implementation of tools to assist information specialists automatically refine queries to be more effective.** These tools comprise (i) QueryVis, a visualisation and understandability tool to assist information specialists to formulate better queries, and (ii) QueryLens, a tool that provides an interactive interface that allows a human in the loop to generate and select candidate variations of queries using the QTC framework.

13.3 Exploitation of Boolean Queries

The third part of this thesis presented two methods for exploiting Boolean queries to improve search effectiveness in systematic review literature search. The first chapter in this part investigated exploiting the implicit PICO framework Boolean queries are often developed within. This chapter demonstrated that enhanced searches using PICO fields could significantly reduce the number of studies retrieved while maintaining recall. The second chapter in this part investigated exploiting the implicit structure of Boolean queries to produce an effective ranking of studies. The CLF ranking method proposed in this chapter significantly outperformed many state-of-the-art methods for ranking studies, including PubMed itself, without using BOW queries (as almost all other methods do).

13.3.1 Findings

Exploitation of PICO. The first finding was that the retrieval effectiveness of queries could be improved simply by exploiting existing characteristics about how queries are already developed. Specifically, through semantically restricting which parts of a study should be matched by a query, the number of irrelevant studies was considerably reduced (i.e., improved precision). This finding is related to the conservative query formulation of Part 2.

Exploitation of Syntax. The second finding was that effective rankings of studies could be achieved by exploiting the existing syntax of Boolean queries. This result was achieved using the CLF ranking function. This ranking function fused rankings from different clauses of a Boolean query in a bottom-up fashion, resulting in a single ranked list of studies. The CLF method achieved state-of-the-art performance for the screening prioritisation and stopping prediction tasks on the CLEF TAR 2017 and 2018 test collections.

13.3.2 Future Work

The investigations in this part only considered exploiting the intrinsic characteristics of Boolean queries for systematic review literature search. By intrinsic aspects, the characteristics of Boolean queries are given: no new functionalities were added to the queries. While these exploitations lead to considerable

improvements in retrieval effectiveness, they raise the question: how can the characteristics of Boolean queries be *extended* (rather than exploited) to improve retrieval effectiveness further? One way to go about this could be to develop new Boolean operators that exploit the semantic relationships within studies. This method is a clear extension of the work in Chapter 11 (Integrating PICO into Retrieval), except instead of merely restricting matches, one may exploit relationships (e.g., an operator that matches studies where two specified interventions are tested against each other, or an intervention and a control). Another way to go about this could be to add syntax that limits the total number of studies retrieved on a per-clause basis. This method is a clear application of the work in Chapter 12 (Ranking Studies with Boolean Queries) that would allow such an operator to exist.

13.3.3 Contributions

The contributions of Part 3 include:

1. **Two proposed methods for exploiting intrinsic aspects to Boolean queries for systematic review literature search.** These methods exploit aspects of queries that already exist, i.e., no new functionality is added to the queries.

2. **An evaluation of the two methods for exploiting those intrinsic aspects.** In Chapter 11, it was found that the use of PICO fields can significantly improve precision while maintaining recall, drastically reducing the total number of irrelevant studies needed to screen. In Chapter 12, it was found that the proposed ranking method could outperform several state-of-the-art methods and that a cut-off could be derived from the ranking to reduce the total number of studies to screen while maintaining an acceptable level of recall.

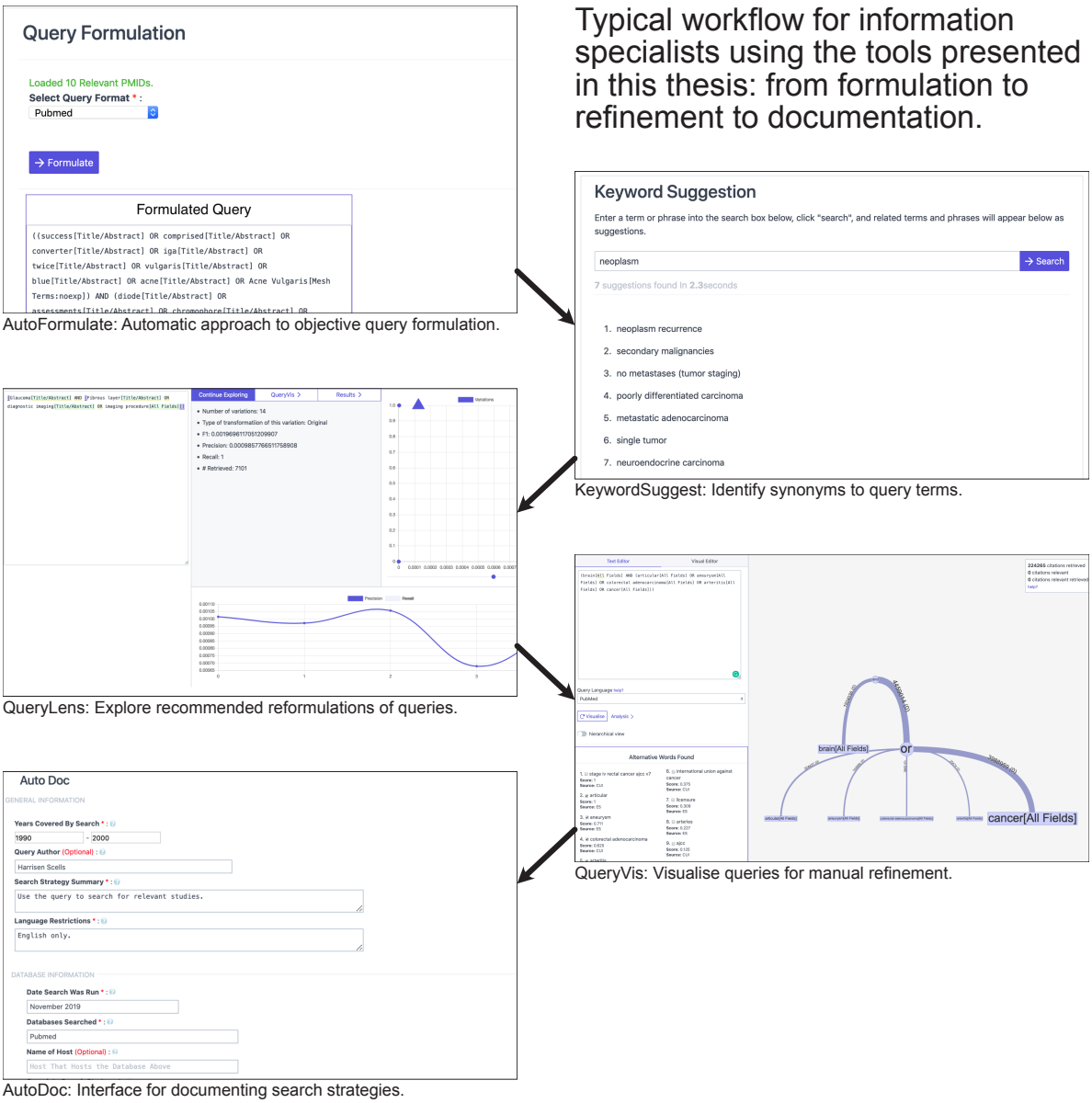
13.4 Tooling

Automation tools are becoming increasingly crucial to the creation of timely systematic reviews [45]. Some of the primary contributions of this thesis are several query automation tools made freely available for information specialists to use. Some of these tools are realisations of the theoretical methods put forth by this thesis, such as interfaces to the automatic query formulation and refinement methods presented in Chapters 4 and 7, respectively. Figure 13.1 presents an overview of all the tools, in the workflow they are intended to be used in, that have been produced throughout this thesis, with a short description of each.

The tools that have been developed are:

1. **AutoFormulate**: An interface to the automatic objective method presented in Chapter 4. This tool produces queries as a starting point for further human refinement.

2. **KeywordSuggest**: An interface for identifying synonyms for query terms. An information specialist would use this tool for expanding their query, especially in cases where they may not be familiar with the domain they are developing a query.



AutoFormulate: Automatic approach to objective query formulation.

KeywordSuggest: Identify synonyms to query terms.

QueryLens: Explore recommended reformulations of queries.

QueryVis: Visualise queries for manual refinement.

AutoDoc: Interface for documenting search strategies.

Figure 13.1: Overview of the systematic review query automation tools that have been developed as part of this thesis. The left column represents tools that relate to query formulation. The right column represents tools that relate to query refinement.

3. **QueryLens**: An interface for exploring recommended reformulations of a given query. This tool would be used for finding variations of a query that may otherwise not have been identified by humans.

4. **QueryVis**: An interface for further refining a query, through the query's visualisation as a tree. The annotations on the visualisation assist information specialists in identifying query terms or entire clauses that do not contribute to the effective retrieval of studies.

5. **AutoDoc**: An interface to assist information specialists to document their search strategies for publishing. Often search strategies are under-reported and contain errors. This tool aims to overcome these problems by validating that a search strategy contains all necessary aspects.

13.5 Summary

Automation methods and tools will continue to play a large role in the creation of systematic reviews. There is also a clear path for new automation methods as systematic reviews continue to evolve, supporting the clinicians and policy makers who rely on them. Automation will also continue to directly support those who create systematic reviews, such as information specialists and medical researchers. This thesis has demonstrated that there is still much to be done in automating the creation of systematic reviews, that query automation can have a considerable effect on reducing the costs of systematic review creation, and that there is still room for query automation to grow, in methods and tools, in the systematic review automation space.

Bibliography

[1] Review manager web (RevMan web), 2019.

[2] J. G. Adeva, J. P. Atxa, M. U. Carrillo, and E. A. Zengotitabengoa. Automatic text classification to support systematic reviews in medicine. Expert Systems with Applications, 41(4):1498–1508, 2014.

[3] T. Agoritsas, A. Merglen, D. S. Courvoisier, C. Combescure, N. Garin, A. Perrier, and T. V. Perneger. Sensitivity and predictive value of 15 PubMed search strategies to answer clinical questions rated against full systematic reviews. Journal of Medical Internet Research, 14(3):e85, 2012.

[4] A. Alharbi, W. Briggs, and M. Stevenson. Retrieving and ranking studies for systematic reviews: University of Sheffield's approach to CLEF eHealth 2018 Task 2. In CEUR Workshop Proceedings: Working Notes of CLEF 2018: Conference and Labs of the Evaluation Forum, volume 2125. CEUR Workshop Proceedings, 2018.

[5] A. Alharbi and M. Stevenson. Ranking abstracts to identify relevant evidence for systematic reviews: The university of sheffield's approach to CLEF eHealth 2017 task 2. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[6] A. Alharbi and M. Stevenson. A dataset of systematic review updates. In Proceedings of the 42nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1257–1260, New York, NY, USA, 2019. Association for Computing Machinery.

[7] I. E. Allen and I. Olkin. Estimating time to conduct a meta-analysis from number of citations retrieved. Journal of the American Medical Association, 282(7):634–635, 1999.

[8] A. Anagnostou, A. Lagopoulos, G. Tsoumakas, and I. P. Vlahavas. Combining inter-review learning-to-rank and intra-review incremental training for title and abstract screening in systematic reviews. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[9] S. Ananiadou, B. Rea, N. Okazaki, R. Procter, and J. Thomas. Supporting systematic reviews using text mining. Social Science Computer Review, 27(4):509–523, 2009.

[10] I. Androutsopoulos, G. D. Ritchie, and P. Thanisch. Natural language interfaces to databases–an introduction. Natural language engineering, 1(1):29–81, 1995.

[11] A. R. Aronson. Effective mapping of biomedical text to the UMLS Metathesaurus: The MetaMap program. In Proceedings of the AMIA Symposium, page 17. American Medical Informatics Association, 2001.

[12] D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[13] J. A. Aslam, E. Kanoulas, V. Pavlu, S. Savev, and E. Yilmaz. Document selection methodologies for efficient and effective learning-to-rank. In Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 468–475, 2009.

[14] J. A. Aslam and M. Montague. Models for metasearch. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 276–284, 2001.

[15] L. Azzopardi and G. Zuccon. Economic Models of Interaction. Computational Interaction, page 311, 2018.

[16] L. M. Bachmann, R. Coray, P. Estermann, and G. Ter Riet. Identifying diagnostic studies in MEDLINE: Reducing the number needed to read. Journal of the American Medical Informatics Association, 9(6):653–658, 2002.

[17] S. Balaneshin-kordan and A. Kotov. Optimization method for weighting explicit and latent concepts in clinical decision support queries. In Proceedings of the 6th ACM International Conference on the Theory of Information Retrieval, pages 241–250, 2016.

[18] N. Balasubramanian, G. Kumaran, and V. R. Carvalho. Exploring reductions for long web queries. In Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 571–578, 2010.

[19] L. Bauld. The impact of smokefree legislation in England: Evidence review. 2011.

[20] A. L. Beam, B. Kompa, I. Fried, N. P. Palmer, X. Shi, T. Cai, and I. S. Kohane. Clinical concept embeddings learned from massive sources of medical data. Computing Research Repository, abs/1804.01486, 2018.

[21] M. Bendersky, D. Metzler, and W. B. Croft. Effective query formulation with multiple information sources. In Proceedings of the 12th International Conference on Web Search and Web Data Mining, pages 443–452, 2012.

[22] R. Benham, J. S. Culpepper, L. Gallagher, X. Lu, and J. Mackenzie. Towards efficient and effective query variant generation. In Proceedings of the 1st Biennial Conference on Design of Experimental Search & Information Retrieval Systems, pages 62–67, 2018.

[23] G. Bordogna and G. Pasi. Application of fuzzy set theory to extend boolean information retrieval. In Soft Computing in Information Retrieval, pages 21–47. 2000.

[24] F. Boudin, J.-Y. Nie, and M. Dawes. Clinical information retrieval using document and PICO structure. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 822–830, 2010.

[25] W. M. Bramer, D. Giustini, G. B. de Jonge, L. Holland, and T. Bekhuis. De-duplication of database search results for systematic reviews in EndNote. Journal of the Medical Library Association, 104(3):240, 2016.

[26] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. 1998.

[27] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. Efficient query evaluation using a two-level retrieval process. In Proceedings of the 12th ACM International Conference on Information and Knowledge Management, pages 426–434, 2003.

[28] D. A. Buell. A general model of query processing in information retrieval systems. Information Processing & Management, 17(5):249–262, 1981.

[29] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In Proceedings of the 22nd International Conference on Machine Learning, pages 89–96, 2005.

[30] C. J. Burges, R. Ragno, and Q. V. Le. Learning to rank with nonsmooth cost functions. In Advances in Neural Information Processing Systems, pages 193–200, 2007.

[31] P. B. Burns, R. J. Rohrich, and K. C. Chung. The levels of evidence and their role in evidence-based medicine. Plastic and reconstructive surgery, 128(1):305, 2011.

[32] W. A. Calo and N. T. Brewer. HPV vaccine requirements, opt-outs and providers' support: Key studies missing from a recent systematic review. Human Vaccines & Immunotherapeutics, 16(1):128–130, 2020.

[33] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: From pairwise approach to listwise approach. In Proceedings of the 24th International Conference on Machine Learning, pages 129–136, 2007.

[34] G. Capannini, C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, and N. Tonellotto. Quality versus efficiency in document scoring with learning-to-rank models. Information Processing & Management, 52(6):1161–1177, 2016.

[35] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 335–336, 1998.

[36] C. Carpineto and G. Romano. A survey of automatic query expansion in information retrieval. ACM Computuing Surveys, 44(1), 2012.

[37] I. Chalmers, K. Dickersin, and T. C. Chalmers. Getting to grips with archie cochrane's agenda. British Medical Journal, 305(6857):786, 1992.

[38] J. Chandler, M. Cumpston, T. Li, M. J. Page, and V. A. Welch. Cochrane Handbook for Systematic Reviews of Interventions. John Wiley & Sons, 2019.

[39] J. Chen, S. Chen, Y. Song, H. Liu, Y. Wang, Q. Hu, L. He, and Y. Yang. ECNU at 2017 eHealth task 2: Technologically assisted reviews in empirical medicine. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[40] P. A. Chirita, C. S. Firan, and W. Nejdl. Personalized query expansion for the web. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 7–14, 2007.

[41] W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. Journal of machine learning research, 6:1019–1041, 2005.

[42] W. Chu and Z. Ghahramani. Preference learning with Gaussian processes. In Proceedings of the 22nd International Conference on Machine Learning, pages 137–144, 2005.

[43] W. Chu and S. S. Keerthi. New approaches to support vector ordinal regression. In Proceedings of the 22nd International Conference on Machine Learning, pages 145–152, 2005.

[44] J. Clark. Systematic reviewing. In G. M. W. Suhail A. R. Doi, editor, Methods of Clinical Epidemiology. 2013.

[45] J. Clark, P. Glasziou, C. Del Mar, A. Bannach-Brown, P. Stehlik, and A. M. Scott. A full systematic review was completed in 2 weeks using automation tools: A case study. Journal of clinical epidemiology, 121:81–90, 2020.

[46] J. Clark, P. Glasziou, C. Del Mar, A. Bannach-Brown, P. Stehlik, and A. M. Scott. How to complete a full systematic review in 2 weeks: Processes, facilitators and barriers. Journal of Clinical Epidemiology, 2020.

[47] J. M. Clark, S. Sanders, M. Carter, D. Honeyman, G. Cleo, Y. Auld, D. Booth, P. Condron, C. Dalais, S. Bateup, et al. Improving the translation of search strategies using the Polyglot Search Translator: A randomized controlled trial. Journal of the Medical Library Association, 108(2):195, 2020.

[48] C. W. Cleverdon. The aslib cranfield research project on the comparative efficiency of indexing systems. In Aslib Proceedings, 1960.

[49] C. W. Cleverdon. The significance of the Cranfield tests on index languages. In Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 3–12, 1991.

[50] A. L. Cochrane et al. Effectiveness and Efficiency: Random Reflections on Health Services, volume 900574178. Nuffield Provincial Hospitals Trust London, 1972.

[51] A. Cohen, W. Hersh, K. Peterson, and P. Yen. Reducing workload in systematic review preparation using automated citation classification. Journal of the American Medical Informatics Association, 13(2):206–219, 2006.

[52] A. M. Cohen, C. E. Adams, J. M. Davis, C. Yu, P. S. Yu, W. Meng, L. Duggan, M. McDonagh, and N. R. Smalheiser. Evidence-based medicine, the essential role of systematic reviews, and the need for automated text mining tools. In Proceedings of the 1st ACM International Health Informatics Symposium, pages 376–380, 2010.

[53] A. M. Cohen and N. R. Smalheiser. UIC/OHSU CLEF 2018 task 2 diagnostic test accuracy ranking using publication type cluster similarity measures. In CEUR Workshop Proceedings: Working Notes of CLEF 2018: Conference and Labs of the Evaluation Forum, volume 2125, 2018.

[54] R. J. Cook and D. L. Sackett. The number needed to treat: A clinically useful measure of treatment effect. Biomedical Journal, 310(6977):452–454, 1995.

[55] W. S. Cooper, F. C. Gey, and D. P. Dabney. Probabilistic retrieval based on staged logistic regression. In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 198–210, 1992.

[56] G. V. Cormack and M. R. Grossman. Autonomy and reliability of continuous active learning for technology-assisted review. arXiv preprint arXiv:1504.06868, 2015.

[57] G. V. Cormack and M. R. Grossman. Engineering quality and reliability in technology-assisted review. In Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 75–84, 2016.

[58] G. V. Cormack and M. R. Grossman. Engineering quality and reliability in technology-assisted review. In Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 75–84, 2016.

[59] G. V. Cormack and M. R. Grossman. Technology-assisted review in empirical medicine: Waterloo participation in CLEF eHealth 2017. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[60] G. V. Cormack and M. R. Grossman. Technology-assisted review in empirical medicine: Waterloo participation in CLEF eHealth 2018. In CEUR Workshop Proceedings: Working Notes of CLEF 2018: Conference and Labs of the Evaluation Forum, 2018.

[61] V. R. Cornelius, M. J. Perrio, S. A. Shakir, and L. A. Smith. Systematic reviews of adverse effects of drug interventions: A survey of their conduct and reporting quality. Pharmacoepidemiology and Drug Safety, 18(12):1223–1231, 2009.

[62] D. Cossock and T. Zhang. Subset ranking using regression. In Proceedings of the 19th Annual Conference on Learning Theory, pages 605–619, 2006.

[63] K. Crammer and Y. Singer. Pranking with ranking. In Advances in Neural Information Processing Systems, pages 641–647, 2002.

[64] F. Crestani. Exploiting the similarity of non-matching terms at retrieval time. Information Retrieval Journal, 2(1):27–47, 2000.

[65] W. B. Croft. Boolean queries and term dependencies in probabilistic retrieval models. Journal of the American society for Information Science, 37(2):71–77, 1986.

[66] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 299–306, 2002.

[67] L. De Vine, G. Zuccon, B. Koopman, L. Sitbon, and P. Bruza. Medical semantic similarity with a neural language model. In Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, pages 1819–1822, 2014.

[68] D. Demner-Fushman and J. Lin. Answering clinical questions with knowledge-based and statistical techniques. Computational Linguistics, 33(1):63–103, 2007.

[69] G. M. Di Nunzio. A study of an automatic stopping strategy for technologically assisted medical reviews. In Proceedings of the 40th European Conference on Information Retrieval, pages 672–677, 2018.

[70] G. M. Di Nunzio, F. Beghini, F. Vezzani, and G. Henrot. An interactive two-dimensional approach to query aspects rewriting in systematic reviews. IMS unipd at CLEF eHealth task 2.

In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[71] G. M. Di Nunzio, G. Ciuffreda, and F. Vezzani. Interactive sampling for systematic reviews. IMS unipd at CLEF 2018 eHealth task 2. In CEUR Workshop Proceedings: Working Notes of CLEF 2018: Conference and Labs of the Evaluation Forum, 2018.

[72] M. Dillon, J. Ulmschneider, and J. Desper. A prevalence formula for automatic relevance feedback in Boolean systems. Information Processing & Management, 19(1):27–36, 1983.

[73] P. Donmez and J. G. Carbonell. Optimizing estimated loss reduction for active sampling in rank learning. In Proceedings of the 25th International Conference on Machine Learning, pages 248–255, 2008.

[74] S. Dumais and L. M. TREC. A status report. In Proceedings of the 1st Text REtrieval Conference, pages 500–207, 1992.

[75] K. Elbedweihy, S. N. Wrigley, and F. Ciravegna. Evaluating semantic search query approaches with expert and casual users. In The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, pages 274–286, 2012.

[76] H. Fang and C. Zhai. An exploration of axiomatic approaches to information retrieval. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 480–487, 2005.

[77] N. Fiorini, K. Canese, G. Starchenko, E. Kireev, W. Kim, V. Miller, M. Osipov, M. Kholodov, R. Ismagilov, S. Mohan, et al. Best Match: New relevance search for PubMed. PLoS biology, 16(8):e2005343, 2018.

[78] M. Fiszman, D. Demner-Fushman, F.-M. Lang, P. Goetz, and T. C. Rindflesch. Interpreting comparative constructions in biomedical text. In Biological, Translational, and Clinical Language Processing, pages 137–144, 2007.

[79] E. A. Fox and J. A. Shaw. Combination of multiple searches. NIST special publication SP, 243, 1994.

[80] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. Journal of machine learning research, 4(Nov):933–969, 2003.

[81] N. Fuhr. Optimum polynomial retrieval functions based on the probability ranking principle. ACM Transactions on Information Systems, 7(3):183–204, 1989.

[82] A. Gates, M. Gates, M. Sebastianski, S. Guitard, S. A. Elliott, and L. Hartling. The semi-automation of title and abstract screening: A retrospective exploration of ways to leverage Abstrackr's relevance predictions in systematic and rapid reviews. BMC Medical Research Methodology, 20:1–9, 2020.

[83] F. C. Gey. Inferring probability of relevance using the method of logistic regression. In Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 222–231, 1994.

[84] S. Golder, Y. Loke, and H. M. McIntosh. Poor reporting and inadequate searches were apparent in systematic reviews of adverse effects. Journal of Clinical Epidemiology, 2008.

[85] T. Greenhalgh and R. Peacock. Effectiveness and efficiency of search methods in systematic reviews of complex evidence: Audit of primary sources. Biomedical Journal, 331(7524):1064–1065, 2005.

[86] G. Guyatt, J. Cairns, D. Churchill, D. Cook, B. Haynes, J. Hirsh, J. Irvine, M. Levine, M. Levine, J. Nishikawa, et al. Evidence-based medicine: A new approach to teaching the practice of medicine. Journal of the American Medical Association, 268(17):2420–2425, 1992.

[87] C. Hauff, L. Azzopardi, and D. Hiemstra. The combination and evaluation of query performance prediction methods. In Proceedings of the 31st European Conference on Information Retrieval, pages 301–312, 2009.

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

[89] E. Hausner, C. Guddat, T. Hermanns, U. Lampert, and S. Waffenschmidt. Development of search strategies for systematic reviews: Validation showed the noninferiority of the objective approach. Journal of clinical epidemiology, 68(2):191–199, 2015.

[90] E. Hausner, S. Waffenschmidt, T. Kaiser, and M. Simon. Routine development of objectively derived search strategies. Systematic reviews, 1(1):19, 2012.

[91] B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. In String Processing and Information Retrieval, 11th International Conference, pages 229–248, 2004.

[92] B. He and I. Ounis. Query performance prediction. Information Systems, 31(7):585–594, 2006.

[93] D. Hiemstra, S. Robertson, and H. Zaragoza. Parsimonious language models for information retrieval. In Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 178–185, 2004.

[94] J. Higgins, R. Churchill, D. Tovey, T. Lasserson, and J. Chandler. Update on the MECIR project: Methodological expectations for Cochrane intervention. Cochrane Methods, page 2, 2011.

[95] K. Hofmann, S. Whiteson, and M. de Rijke. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. Information Retrieval, 16(1):63–90, 2013.

[96] N. Hollmann and C. Eickhoff. Ranking and feedback-based stopping for recall-centric document retrieval. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[97] W. Hsu, W. Speier, and R. K. Taira. Automated extraction of reported statistical analyses: Towards a logical representation of clinical trial literature. In AMIA Annual Symposium Proceedings, volume 2012, page 350, 2012.

[98] V. H. Innovation. Covidence systematic review software. 2016.

[99] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. ACM Transactions on Information Systems, 20(4):422–446, 2002.

[100] Y. Jiang, C. Lin, W. Meng, C. Yu, A. M. Cohen, and N. R. Smalheiser. Rule-based deduplication of article records from bibliographic databases. Database-the Magazine of Electronic Database Reviews, 2014.

[101] Jimmy, G. Zuccon, and B. Koopman. Choices in knowledge-base retrieval for consumer health search. In Advances in Information Retrieval, pages 72–85, 2018.

[102] T. Joachims. Optimizing search engines using clickthrough data. In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 133–142, 2002.

[103] V. Kalphov, G. Georgiadis, and L. Azzopardi. Sis at clef 2017 ehealth tar task. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, volume 1866, pages 1–5, 2017.

[104] E. Kanoulas, D. Li, L. Azzopardi, and R. Spijker. CLEF 2017 technologically assisted reviews in empirical medicine overview. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[105] E. Kanoulas, D. Li, L. Azzopardi, and R. Spijker. CLEF 2019 technology assisted reviews in empirical medicine overview. In CEUR Workshop Proceedings: Working Notes of CLEF 2018: Conference and Labs of the Evaluation Forum, volume 2380, 2019.

[106] E. Kanoulas, S. Savev, P. Metrikov, V. Pavlu, and J. Aslam. A large-scale study of the effect of training set characteristics over learning-to-rank algorithms. In Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1243–1244, 2011.

[107] E. Kanoulas, R. Spijker, D. Li, and L. Azzopardi. CLEF 2018 technology assisted reviews in empirical medicine overview. In CEUR Workshop Proceedings: Working Notes of CLEF 2018: Conference and Labs of the Evaluation Forum, 2018.

[108] S. Karimi, S. Pohl, F. Scholer, L. Cavedon, and J. Zobel. Boolean versus ranked querying for biomedical systematic reviews. BMC Medical Informatics & Decision Making, 10(1):1, 2010.

[109] S. Karimi, J. Zobel, S. Pohl, and F. Scholer. The challenge of high recall in biomedical systematic search. In Proceedings of the 3rd International Workshop on Data and Text Mining in Bioinformatics, pages 89–92, 2009.

[110] A. Khwileh, A. Way, and G. J. F. Jones. Improving the reliability of query expansion for user-generated speech retrieval using query performance prediction. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, pages 43–56. 2017.

[111] S. Kim, D. Martinez, L. Cavedon, and L. Yencken. Automatic classification of sentences to support evidence based medicine. BMC bioinformatics, 12(2), 2011.

[112] Y. Kim, J. Seo, and W. B. Croft. Automatic boolean query suggestion for professional search. In Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2011.

[113] B. Koopman, L. Cripwell, and G. Zuccon. Generating clinical queries from patient narratives: A comparison between machines and humans. In Proceedings of the 40th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 853–856, 2017.

[114] B. Koopman, G. Zuccon, P. Bruza, L. Sitbon, and M. Lawley. An evaluation of corpus-driven measures of medical concept similarity for information retrieval. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management, pages 2439–2442, 2012.

[115] G. Kumaran and V. R. Carvalho. Reducing long queries using query quality predictors. In Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 564–571, 2009.

[116] K. L. Kwok. A new method of weighting query terms for ad-hoc retrieval. In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 187–195, 1996.

[117] A. Laupacis, D. L. Sackett, and R. S. Roberts. An assessment of clinically useful measures of the consequences of treatment. New England journal of medicine, 318(26):1728–1733, 1988.

[118] J. Lavis, H. Davies, A. Oxman, J.-L. Denis, K. Golden-Biddle, and E. Ferlie. Towards systematic reviews that inform health care management and policy-making. Journal of health services research & policy, 10:35–48, 2005.

[119] G. E. Lee. A study of convolutional neural networks for clinical document classification in systematic reviews: Sysreview at CLEF eHealth 2017. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[120] G. E. Lee and A. Sun. Seed-driven document ranking for systematic reviews in evidence-based medicine. In Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 455–464, 2018.

[121] M. M. Leeflang, J. J. Deeks, Y. Takwoingi, and P. Macaskill. Cochrane diagnostic test accuracy reviews. Systematic reviews, 2(1):82, 2013.

[122] H. Li, H. Scells, and G. Zuccon. Systematic review automation tools for end-to-end query formulation. In Proceedings of the 43rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 25–30, 2020.

[123] P. Li, Q. Wu, and C. J. Burges. Mcrank: Learning to rank using multiple classification and gradient boosting. In Advances in Neural Information Processing Systems, pages 897–904, 2008.

[124] T.-Y. Liu et al. Learning to rank for information retrieval. Foundations and Trends in Information Retrieval, 3(3):225–331, 2009.

[125] D. Locke, G. Zuccon, and H. Scells. Automatic query generation from legal texts for case law retrieval. In Proceedings of the 13th Asia Information Retrieval Symposium, pages 181–193, 2017.

[126] R. Losee. Probabilistic retrieval and coordination level matching. Journal of the American Society for Information Science, 38(4):239–244, 1987.

[127] C. Lucchese, F. M. Nardini, R. Perego, and S. Trani. The impact of negative samples on learning to rank. In Proceedings of the 1st International Workshop on LEARning next gEneration Rankers, 2017.

[128] B. J. Maguire and P. J. Guérin. A living systematic review protocol for COVID-19 clinical trial registrations. Wellcome Open Research, 5, 2020.

[129] I. J. Marshall, J. Kuiper, and B. C. Wallace. RobotReviewer: Evaluation of a system for automatically assessing bias in clinical trials. Journal of the American Medical Informatics Association, 23(1):193–201, 2016.

[130] I. J. Marshall, R. Marshall, B. C. Wallace, J. Brassey, and J. Thomas. Rapid reviews may produce different results to systematic reviews: A meta-epidemiological study. Journal of clinical epidemiology, 109:30–41, 2019.

[131] I. J. Marshall, A. Noel-Storr, J. Kuiper, J. Thomas, and B. C. Wallace. Machine learning for identifying randomized controlled trials: An evaluation and practitioner's guide. Research synthesis methods, 9(4):602–614, 2018.

[132] D. Martinez, S. Karimi, L. Cavedon, and T. Baldwin. Facilitating biomedical systematic reviews using ranked text retrieval and classification. In Proceedings of the 13th Australasian Document Computing Symposium, 2008.

[133] J. McGowan and M. Sampson. Systematic reviews need systematic searchers (IRP). Journal of the Medical Library Association, 93(1):74, 2005.

[134] K. A. McGraw, M. J. Anderson, et al. Analysis of the reporting of search strategies in Cochrane systematic reviews. Journal of the Medical Library Association, 97(1):21, 2009.

[135] R. Mehrotra and E. Yilmaz. Representative & informative query selection for learning to rank using submodular functions. In Proceedings of the 38th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 545–554, 2015.

[136] D. Metzler and W. B. Croft. Linear feature-based models for information retrieval. Information Retrieval Journal, 10(3):257–274, 2007.

[137] T. Mikolov, G. Corrado, K. Chen, and J. Dean. Efficient Estimation of Word Representations in Vector Space. In Proceedings of the 2013 International Conference on Learning Representations, pages 1–12, 2013.

[138] G. A. Miller. WordNet: An Electronic Lexical Database. MIT press, 1998.

[139] A. Minas, A. Lagopoulos, and G. Tsoumakas. Aristotle university's approach to the technologically assisted reviews in empirical medicine task of the 2018 CLEF eHealth lab. In CEUR Workshop Proceedings: Working Notes of CLEF 2018: Conference and Labs of the Evaluation Forum, 2018.

[140] S. Mirhosseini, G. Zuccon, B. Koopman, A. Nguyen, and M. Lawley. Medical free-text to concept mapping as an information retrieval problem. In Proceedings of the 19th Australasian Document Computing Symposium, pages 93–96, 2014.

[141] M. Miwa, J. Thomas, A. O'Mara-Eves, and S. Ananiadou. Reducing systematic review workload through certainty-based screening. Journal of Biomedical Informatics, 51:242–253, 2014.

[142] A. Moffat, P. Thomas, and F. Scholer. Users versus models: What observation tells us about effectiveness metrics. In Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, pages 659–668, 2013.

[143] A. Moffat and J. Zobel. Rank-biased precision for measurement of retrieval effectiveness. ACM Transactions on Information Systems, 27(1):2, 2008.

[144] V. M. Montori, N. L. Wilczynski, D. Morgan, and R. B. Haynes. Optimal search strategies for retrieving systematic reviews from Medline: Analytical survey. Biomedical Journal, 330(7482):68, 2005.

[145] J. Mothe and L. Tanguy. Linguistic features to predict query difficulty. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 7–10, 2005.

[146] C. Norman, M. Leeflang, and A. Névéol. Data extraction and synthesis in systematic reviews of diagnostic test accuracy: A corpus for automating and evaluating the process. In AMIA Annual Symposium Proceedings, volume 2018, page 817, 2018.

[147] C. Norman, M. Leeflang, and A. Névéol. LIMSI@ CLEF eHealth 2018 task 2: Technology assisted reviews by stacking active and static learning. In CEUR Workshop Proceedings: Working Notes of CLEF 2018: Conference and Labs of the Evaluation Forum, 2018.

[148] C. R. Norman, M. M. Leeflang, R. Porcher, and A. Névéol. Measuring the impact of screening automation on meta-analyses of diagnostic test accuracy. Systematic reviews, 8(1):243, 2019.

[149] C. Norman12, M. Leeflang, and A. Névéol. Limsi@ clef ehealth 2017 task 2: Logistic regression for automatic article ranking. In CEUR Workshop Proceedings: Working Notes of CLEF 2019: Conference and Labs of the Evaluation Forum, 2017.

[150] B. Nussbaumer-Streit, I. Klerings, G. Wagner, T. L. Heise, A. I. Dobrescu, S. Armijo-Olivo, J. M. Stratil, E. Persad, S. K. Lhachimi, M. G. Van Noord, et al. Abbreviated literature searches were viable alternatives to comprehensive searches: A meta-epidemiological study. Journal of clinical epidemiology, 102:1–11, 2018.

[151] D. W. Oard, J. R. Baron, B. Hedin, D. D. Lewis, and S. Tomlinson. Evaluation of information retrieval for E-discovery. Artificial Intelligence and Law, 18(4):347–386, 2010.

[152] D. Odijk and A. Schuth. Online Learning to Rank for Recommender Systems. In Proceedings of the 11th ACM Conference on Recommender Systems, page 348, Aug. 2017.

[153] A. O'Mara-Eves, J. Thomas, J. McNaught, M. Miwa, and S. Ananiadou. Using text mining for study identification in systematic reviews: A systematic review of current approaches. Systematic reviews, 4(1):5, 2015.

[154] H. Oosterhuis and M. de Rijke. Differentiable Unbiased Online Learning to Rank. In Proceedings of the 27th International ACM Conference on Information and Knowledge Management, pages 1293–1302, 2018.

[155] M. Ouzzani, H. Hammady, Z. Fedorowicz, and A. Elmagarmid. Rayyan—a web and mobile app for systematic reviews. Systematic Reviews, 5(1):210, 2016.

[156] C. D. Paice. Soft evaluation of Boolean search queries in information retrieval systems. Information Technology: Research and Development, 3(1):33–41, 1984.

[157] J. Palotti, H. Scells, and G. Zuccon. Trectools: An open-source python library for information retrieval practitioners involved in trec-like campaigns. In Proceedings of the 42nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1325–1328, 2019.

[158] E. Partners. DistillerSR [Computer program]. Ottawa, Canada: Evidence Partners, 2011.

[159] R. A. Pazos R, J. J. González B, M. A. Aguirre L, J. A. Martínez F, and H. J. Fraire H. Natural language interfaces to databases: An analysis of the state of the art. Recent Advances on Hybrid Intelligent Systems, pages 463–480, 2013.

[160] T. Pedersen, S. V. Pakhomov, S. Patwardhan, and C. G. Chute. Measures of semantic similarity and relatedness in the biomedical domain. Journal of biomedical informatics, 40(3):288–299, 2007.

[161] P. Pirolli and S. Card. Information foraging in information access environments. In Proceedings of the SIGCHI 1995 Conference on Human Factors in Computing Systems, pages 51–58, 1995.

[162] V. Plachouras, I. Ounis, C. J. van Rijsbergen, and F. Cacheda. University of glasgow at the web track: Dynamic application of hyperlink analysis using the query scope. In Proceedings of the 12th Text REtrieval Conference, volume 3, pages 636–642, 2003.

[163] S. Pohl, A. Moffat, and J. Zobel. Efficient extended boolean retrieval. IEEE Transactions on Knowledge and Data Engineering, 24(6):1014–1024, 2011.

[164] A.-M. Popescu, A. Armanasu, O. Etzioni, D. Ko, and A. Yates. Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In Proceedings of the 20th International Conference on Computational Linguistics, 2004.

[165] M. F. Porter. An algorithm for suffix stripping. Programming, 14(3), 1980.

[166] A. Prasad, M. Kaur, and M.-Y. Kan. Neural ParsCit: A deep learning based reference string parser. Journal on Digital Libraries, 19:323–337, 2018.

[167] T. Qin, X.-D. Zhang, M.-F. Tsai, D.-S. Wang, T.-Y. Liu, and H. Li. Query-level loss functions for information retrieval. Information Processing & Management, 44(2):838–855, 2008.

[168] T. Radecki. Fuzzy set theoretical approach to document retrieval. Information Processing & Management, 15(5):247–259, 1979.

[169] T. Radecki. A probabilistic approach to information retrieval in systems with Boolean search request formulations. Journal of the American Society for Information Science, 33(6):365–370, 1982.

[170] K. Rashmi and R. Gilad-Bachrach. Dart: Dropouts meet multiple additive regression trees. In Proceedings of the 18th International Conference on Artificial Intelligence and Statistics, pages 489–497, 2015.

[171] J. Rathbone, M. Carter, T. Hoffmann, and P. Glasziou. Better duplicate detection for systematic reviewers: Evaluation of Systematic Review Assistant-Deduplication Module. Systematic reviews, 4(1):6, 2015.

[172] S. Reeves, I. Koppel, H. Barr, D. Freeth, and M. Hammick. Twelve tips for undertaking a systematic review. Medical teacher, 24(4), 2002.

[173] M. L. Rethlefsen, A. M. Farrell, L. C. O. Trzasko, and T. J. Brigham. Librarian co-authors correlated with higher quality reported search strategies in general internal medicine systematic reviews. Journal of clinical epidemiology, 68(6):617–626, 2015.

[174] S. Robertson. A new interpretation of average precision. In Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 689–690, 2008.

[175] S. E. Robertson. The probability ranking principle in IR. Journal of documentation, 33(4):294–304, 1977.

[176] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford, et al. Okapi at TREC-3. Nist Special Publication Sp, 109:109, 1995.

[177] J. Rocchio. Relevance feedback in information retrieval. The Smart retrieval system-experiments in automatic document processing, pages 313–323, 1971.

[178] R. Rodriguez-Esteban and I. Iossifov. Figure mining for biomedical research. Bioinformatics, 25(16):2082–2084, 2009.

[179] F. B. Rogers. The development of MEDLARS. Bulletin of the Medical Library Association, 52(1):150, 1964.

[180] S. Rose, D. Engel, N. Cramer, and W. Cowley. Automatic keyword extraction from individual documents. In Text Mining: Applications and Theory, pages 1–20. 2010.

[181] T. Russell-Rose and P. Gooch. 2dSearch: A visual approach to search strategy formulation. In Proceedings of the 1st Biennial Conference on Design of Experimental Search and Information Retrieval Systems, 2018.

[182] G. Salton, E. A. Fox, C. Buckley, and E. M. Voorhees. Boolean query formulation with relevance feedback. Technical report, Cornell University, 1983.

[183] G. Salton, E. A. Fox, and H. Wu. Extended boolean information retrieval. Technical report, Cornell University, 1982.

[184] G. Salton, E. A. Fox, and H. Wu. Extended boolean information retrieval. Communications of the ACM, 26(11):1022–1036, 1983.

[185] G. Salton, E. Voorhees, and E. A. Fox. A comparison of two methods for Boolean query relevancy feedback. Information processing & management, 20(5-6):637–651, 1984.

[186] G. Salton, A. Wong, and C.-S. Yang. A vector space model for automatic indexing. Communications of the ACM, 18(11):613–620, 1975.

[187] M. Sampson and J. McGowan. Errors in search strategies were identified by type and frequency. Journal of Clinical Epidemiology, 59(10):1057–e1, 2006.

[188] M. Sampson, K. G. Shojania, C. Garritty, T. Horsley, M. Ocampo, and D. Moher. Systematic reviews can be produced and published faster. Journal of clinical epidemiology, 61(6):531–536, 2008.

[189] E. Sayers. A general introduction to the e-utilities. Entrez Programming Utilities Help [Internet]. Bethesda: National Center for Biotechnology Information, 2010.

[190] H. Scells, L. Azzopardi, G. Zuccon, and B. Koopman. Query variation performance prediction for systematic reviews. In Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1089–1092, 2018.

[191] H. Scells, D. Locke, and G. Zuccon. An information retrieval experiment framework for domain specific applications. In Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1281–1284, 2018.

[192] H. Scells and G. Zuccon. Generating better queries for systematic reviews. In Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 475–484, 2018.

[193] H. Scells and G. Zuccon. Searchrefiner: A query visualisation and understanding tool for systematic reviews. In Proceedings of the 27th International Conference on Information and Knowledge Management, pages 1939–1942, 2018.

[194] H. Scells and G. Zuccon. Ielab at the open-source IR replicability challenge 2019. In CEUR Workshop Proceedings: Working Notes of CLEF 2019: Conference and Labs of the Evaluation Forum, pages 57–61, 2019.

[195] H. Scells and G. Zuccon. You can teach an old dog new tricks: Rank fusion applied to coordination level matching for ranking in systematic reviews. In Proceedings of the 42nd European Conference on Information Retrieval, pages 399–414, 2020.

[196] H. Scells, G. Zuccon, A. Deacon, and B. Koopman. QUT ielab at CLEF eHealth 2017 technology assisted reviews track: Initial experiments with learning to rank. In CEUR Workshop

Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[197] H. Scells, G. Zuccon, and B. Koopman. Automatic boolean query refinement for systematic review literature search. In Proceedings of the 28th World Wide Web Conference, pages 1646–1656, 2019.

[198] H. Scells, G. Zuccon, and B. Koopman. A comparison of automatic boolean query formulation for systematic reviews. Information Retrieval Journal, pages 1–26, 2020.

[199] H. Scells, G. Zuccon, and B. Koopman. A computational approach for objectively derived systematic review search strategies. In Proceedings of the 42nd European Conference on Information Retrieval, pages 385–398, 2020.

[200] H. Scells, G. Zuccon, and B. Koopman. Sampling query variations for learning to rank to improve automatic boolean query generation in systematic reviews. In Proceedings of the 29th World Wide Web Conference, pages 3041–3048, 2020.

[201] H. Scells, G. Zuccon, B. Koopman, and J. Clark. Automatic search strategy reformulation interface for systematic reviews. In Proceedings of the 2019 Cochrane Colloquium, 2019.

[202] H. Scells, G. Zuccon, B. Koopman, and J. Clark. Visualising systematic review search strategies to assist information specialists. In Proceedings of the 2019 Cochrane Colloquium, 2019.

[203] H. Scells, G. Zuccon, B. Koopman, and J. Clark. Automatic boolean query formulation for systematic review literature search. In Proceedings of the 29th World Wide Web Conference, pages 1071–1081, 2020.

[204] H. Scells, G. Zuccon, B. Koopman, A. Deacon, L. Azzopardi, and S. Geva. Integrating the framing of clinical questions via PICO into the retrieval of medical literature for systematic reviews. In Proceedings of the 26th International Conference on Information and Knowledge Management, pages 2291–2294, 2017.

[205] H. Scells, G. Zuccon, B. Koopman, A. Deacon, S. Geva, and L. Azzopardi. A test collection for evaluating retrieval of studies for inclusion in systematic reviews. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1237–1240, 2017.

[206] C. Schardt, M. B. Adams, T. Owens, S. Keitz, and P. Fontelo. Utilization of the PICO framework to improve searching PubMed for clinical questions. BMC medical informatics and decision making, 7(1):16, 2007.

[207] J. A. Shaw and E. A. Fox. Combination of multiple searches. NIST SPECIAL PUBLICATION SP, pages 105–105, 1995.

[208] D. Sheldon, M. Shokouhi, M. Szummer, and N. Craswell. LambdaMerge: Merging the results of query reformulations. In Proceedings of the 4th International Conference on Web Search and Data Mining, pages 795–804, 2011.

[209] I. Shemilt, N. Khan, S. Park, and J. Thomas. Use of cost-effectiveness analysis to compare the efficiency of study identification methods in systematic reviews. Systematic reviews, 5(1):140, 2016.

[210] I. Shemilt, A. Simon, G. Hollands, T. Marteau, D. Ogilvie, A. O'Mara-Eves, M. Kelly, and J. Thomas. Pinpointing needles in giant haystacks: Use of text mining to reduce impractical screening workload in extremely large scoping reviews. Research Synthesis Methods, 5(1):31–49, 2014.

[211] A. Shtok, O. Kurland, and D. Carmel. Predicting query performance by query-drift estimation. In Proceedings of the 31st European Conference on Information Retrieval, pages 305–312, 2009.

[212] A. Shtok, O. Kurland, and D. Carmel. Query performance prediction using reference lists. ACM Transactions on Information Systems, 34(4):19, 2016.

[213] M. Simon, E. Hausner, S. F. Klaus, and N. E. Dunton. Identifying nurse staffing research in Medline: Development and testing of empirically derived search strategies with the PubMed interface. BMC medical research methodology, 10(1):76, 2010.

[214] G. Singh, I. Marshall, J. Thomas, and B. Wallace. Identifying diagnostic test accuracy publications using a deep model. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, volume 1866, 2017.

[215] J. Singh and L. Thomas. IIIT-H at CLEF eHealth 2017 task 2: Technologically assisted reviews in empirical medicine. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[216] A. Singhal et al. Modern information retrieval: A brief overview. IEEE Data Engineering Bulletin, 24(4):35–43, 2001.

[217] L. Soldaini, A. Cohan, A. Yates, N. Goharian, and O. Frieder. Retrieving medical literature for clinical decision support. In Proceedings of the 37th European Conference on Information Retrieval, pages 538–549, 2015.

[218] L. Soldaini and N. Goharian. Quickumls: A fast, unsupervised approach for medical concept extraction. In Medical Information Retrieval Workshop, 2016.

[219] K. Spark-Jones. Report on the need for and provision of an'ideal'information retrieval test collection. Computer Laboratory, 1975.

[220] C. Stansfield, A. O'Mara-Eves, and J. Thomas. Reducing systematic review workload using text mining: Opportunities and pitfalls. Journal of the European Association for Health Information and Libraries, 11(3):8–10, 2015.

[221] C. Stansfield, J. Thomas, and J. Kavanagh. 'Clustering' documents automatically to support scoping reviews of research: A case study. Research synthesis methods, 4(3):230–241, 2013.

[222] S. Subpaiboonkit, X. Li, X. Zhao, H. Scells, and G. Zuccon. Causality discovery with domain knowledge for drug-drug interactions discovery. In Advanced Data Mining and Applications - 15th International Conference, pages 632–647, 2019.

[223] R. Summerscales, S. Argamon, J. Hupert, and A. Schwartz. Identifying treatments, groups, and outcomes in medical abstracts. In Proceedings of the 6th Midwest Computational Linguistics Colloquium, 2009.

[224] R. L. Summerscales, S. Argamon, S. Bai, J. Hupert, and A. Schwartz. Automatic summarization of results from clinical trials. In Proceedings of the 2011 IEEE International Conference on Bioinformatics and Biomedicine, pages 372–377, 2011.

[225] M. Taylor, J. Guiver, S. Robertson, and T. Minka. Softrank: Optimizing non-smooth rank metrics. In Proceedings of the 1st International Conference on Web Search and Web Data Mining, pages 77–86, 2008.

[226] J. Thomas and J. Brunton. EPPI-Reviewer: Software for research synthesis. 2007.

[227] P. Timsina, J. Liu, and O. El-Gayar. Advanced analytics for the automation of medical systematic reviews. Information Systems Frontiers, 18(2):237–252, 2016.

[228] T. Tomokiyo and M. Hurst. A language model approach to keyphrase extraction. In Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment, pages 33–40, 2003.

[229] M. T. Torres and C. E. Adams. RevManHAL: Towards automatic text generation in systematic reviews. Systematic reviews, 6(1):27, 2017.

[230] G. Tsafnat, A. Dunn, P. Glasziou, and E. Coiera. The automation of systematic reviews. 2013.

[231] G. Tsafnat, P. Glasziou, M. K. Choong, A. Dunn, F. Galgani, and E. Coiera. Systematic review automation technologies. Systematic reviews, 3(1):74, 2014.

[232] H. Turtle and W. B. Croft. Inference networks for document retrieval. In Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1–24, 1989.

[233] Ö. Uzuner, B. R. South, S. Shen, and S. L. DuVall. 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. Journal of the American Medical Informatics Association, 18(5):552–556, 2011.

[234] A. H. van der Vegt, G. Zuccon, and B. Koopman. Learning inter-sentence, disorder-centric, biomedical relationships from medical literature. In AMIA Fall Symposium, 2019.

[235] S. Verberne, J. He, U. Kruschwitz, B. Larsen, T. Russell-Rose, and A. P. De Vries. First international workshop on professional search (ProfS2018). In Joint Proceedings of the First International Workshop on Professional Search, pages 1431–1434, 2018.

[236] S. Verberne, M. Sappelli, D. Hiemstra, and W. Kraaij. Evaluation and analysis of term scoring methods for term extraction. Information Retrieval Journal, 19(5):510–545, 2016.

[237] C. C. Vogt and G. W. Cottrell. Fusion via a linear combination of scores. Information retrieval Journal, 1(3):151–173, 1999.

[238] E. M. Voorhees. Query expansion using lexical-semantic relations. In Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 61–69, 1994.

[239] B. C. Wallace, I. J. Dahabreh, C. H. Schmid, J. Lau, and T. A. Trikalinos. Modernizing the systematic review process to inform comparative effectiveness: Tools and methods. Journal of comparative effectiveness research, 2(3):273–282, 2013.

[240] B. C. Wallace, J. Kuiper, A. Sharma, M. B. Zhu, and I. J. Marshall. Extracting PICO sentences from clinical trial reports using supervised distant supervision. Journal of Machine Learning Research, 17(132):1–25, 2016.

[241] B. C. Wallace, C. H. Schmid, J. Lau, and T. A. Trikalinos. Meta-Analyst: Software for meta-analysis of binary, continuous and diagnostic data. BMC medical research methodology, 9(1):80, 2009.

[242] B. C. Wallace, K. Small, C. E. Brodley, J. Lau, and T. A. Trikalinos. Deploying an interactive machine learning system in an evidence-based practice center: Abstrackr. In Proceedings of the 2nd ACM International Health Informatics Symposium, pages 819–824, 2012.

[243] B. C. Wallace, T. A. Trikalinos, J. Lau, C. Brodley, and C. H. Schmid. Semi-automated screening of biomedical citations for systematic reviews. BMC bioinformatics, 11(1):55, 2010.

[244] W. Waller and D. H. Kraft. A mathematical model of a weighted Boolean retrieval system. Information Processing & Management, 15(5):235–245, 1979.

[245] L. Wang, J. Lin, and D. Metzler. A cascade ranking model for efficient ranked retrieval. In Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development in Information, page 105, 2011.

[246] T. D. Wilson. On user studies and information needs. Journal of documentation, 37(1):3–15, 1981.

[247] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl. Sampling matters in deep embedding learning. In Proceedings of the IEEE International Conference on Computer Vision, pages 2840–2848, 2017.

[248] H. Wu, T. Wang, J. Chen, S. Chen, Q. Hu, and L. He. Ecnu at 2018 ehealth task 2: Technologically assisted reviews in empirical medicine. Methods-a Companion to Methods in Enzymology, 4(5):7, 2018.

[249] Q. Wu, C. J. Burges, K. M. Svore, and J. Gao. Adapting boosting for information retrieval measures. Information Retrieval Journal, 13(3):254–270, 2010.

[250] S. Wu. Data Fusion in Information Retrieval, volume 13 of Adaptation, Learning, and Optimization. 2012.

[251] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li. Listwise approach to learning to rank: Theory and algorithm. In Proceedings of the 25th International Conference on Machine Learning, pages 1192–1199, 2008.

[252] J. Xu and H. Li. Adarank: A boosting algorithm for information retrieval. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 391–398, 2007.

[253] A. Yoshii, D. Plaut, K. McGraw, M. Anderson, and K. Wellik. Analysis of the reporting of search strategies in Cochrane systematic reviews. Journal of the Medical Library Association, 97(1):21, 2009.

[254] C. T. Yu and G. Salton. Precision weighting—an effective automatic indexing method. Journal of the ACM, 23(1):76–88, 1976.

[255] Z. Yu and T. Menzies. Data balancing for technologically assisted reviews: Undersampling or reweighting. In CEUR Workshop Proceedings: Working Notes of CLEF 2017: Conference and Labs of the Evaluation Forum, 2017.

[256] Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In Proceedings of the 26th International Conference on Machine Learning, pages 1201–1208, 2009.

[257] Y. Zhao, F. Scholer, and Y. Tsegay. Effective pre-retrieval query performance prediction using similarity and variability evidence. In Proceedings of the 30th European Conference on Information Retrieval, pages 52–64, 2008.

[258] Y. Zhou and W. B. Croft. Query performance prediction in web search environments. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 543–550, 2007.

[259] S. Zhuang and G. Zuccon. Counterfactual online learning to rank. In Proceedings of the 42nd European Conference on Information Retrieval, pages 415–430, 2020.

[260] G. Zuccon, L. A. Azzopardi, and K. van Rijsbergen. The quantum probability ranking principle for information retrieval. In Proceedings of the 2nd International Conference on the Theory of Information Retrieval, pages 232–240, 2009.

[261] G. Zuccon, B. Koopman, P. Bruza, and L. Azzopardi. Integrating and evaluating neural word embeddings in information retrieval. In Proceedings of the 20th Australasian Document Computing Symposium, pages 1–8, 2015.

[262] G. Zuccon, B. Koopman, A. Nguyen, D. Vickers, and L. Butt. Exploiting medical hierarchies for concept-based information retrieval. In Proceedings of the 17th Australasian Document Computing Symposium, 2012.