

# INFORMATION RETRIEVAL FOR THE DIGITAL HUMANITIES

A dissertation presented by  
**Tim Gollub**

to the  
Faculty of Computer Science and Media  
of the  
Bauhaus-Universität Weimar

in partial fulfillment of the requirements for the academic degree of  
**Dr. rer. nat.**

Weimar, Germany  
January 2022

Advisor: Prof. Dr. Benno Stein  
Reviewer: Prof. Dr. Norbert Fuhr  
Date of oral exam: July 1, 2022

TO KATRIN  
— *WISH YOU WERE HERE.*

# Acknowledgments

During all the years as a research assistant, I had the pleasure to work in a group full of kind and talented colleagues, many of them being good friends today. I'd like to say thank you to Sven Meyer zu Eissen, Martin Potthast, Maik Anderka, Nedim Lipka, Peter Prettenhofer, Henning Wachsmuth, and Matthias Hagen, who accompanied me in the early years at the webis group. Thanks also to Dennis Hoppe, Tsvetomira Palarkarska, Martin Trenkmann, and Steven Burrows, as well as to my current fellows Michael Völske, Johannes Kiesel, Khalid Al-Khatib, Magdalena Wolska, Shabaz Syed, Wei-Fan Chen, Roxanne El Baff, Yamen Ajjour, Janek Bevendorff, and Matti Wiegmann.

Furthermore, I'd like to say thank you to the awesome media science people I was happy to learn to know during our joint digital humanities research project. Their way to look at things opened up a new world for me. In this respect, I am especially grateful to Henning Schmidgen and Franziska Klemstein, for their time and the patience they had with me.

Of course, the biggest thank you goes out to my PhD supervisor Benno Stein, who only gave me the possibility to work in such a great team for such a long time. Benno, I owe so much to you, and learned so much in all the years. Thank you! Your way of managing the group is remarkable, and became a role model for me in many respects of my life.

Last but not least, I'd like to thank Norbert Fuhr, who was so kind to assume the role as my second thesis reviewer. For me, it is a great pleasure to have such a highly reputed professor on board.

*Tim Gollub, October 2021*



# Abstract

## INFORMATION RETRIEVAL FOR THE DIGITAL HUMANITIES

In ten chapters, this thesis presents information retrieval technology which is tailored to the research activities that arise in the context of corpus-based digital humanities projects.

The presentation is structured by a conceptual research process that is introduced in Chapter 1. The process distinguishes a set of five research activities: research question generation, corpus acquisition, research question modeling, corpus annotation, and result dissemination. Each of these research activities elicits different information retrieval tasks with special challenges, for which algorithmic approaches are presented after an introduction of the core information retrieval concepts in Chapter 2.

A vital concept in many of the presented approaches is the keyquery paradigm introduced in Chapter 3, which represents an operation that returns relevant search queries in response to a given set of input documents. Keyqueries are proposed in Chapter 4 for the recommendation of related work, and in Chapter 5 for improving access to aspects hidden in the long tail of search result lists.

With pseudo-descriptions, a document expansion approach is presented in Chapter 6. The approach improves the retrieval performance for corpora where only bibliographic meta-data is originally available. In Chapter 7, the keyquery paradigm is employed to generate dynamic taxonomies for corpora in an unsupervised fashion.

Chapter 8 turns to the exploration of annotated corpora, and presents scoped facets as a conceptual extension to faceted search systems, which is particularly useful in exploratory search settings. For the purpose of highlighting the major topical differences in a sequence of sub-corpora, an algorithm called topical sequence profiling is presented in Chapter 9.

The thesis concludes with two pilot studies regarding the visualization of (re)search results for the means of successful result dissemination: a metaphoric interpretation of the information nutrition label, as well as the philosophical bodies, which are 3D-printed search results.



# Abstract (in German)

## INFORMATION RETRIEVAL FOR THE DIGITAL HUMANITIES

In zehn Kapiteln stellt diese Arbeit Information-Retrieval-Technologien vor, die auf die Forschungsaktivitäten korpusbasierter Digital-Humanities-Projekte zugeschnitten sind.

Die Arbeit strukturiert sich an Hand eines konzeptionellen Forschungsprozess der in Kapitel 1 vorgestellt wird. Der Prozess gliedert sich in fünf Forschungsaktivitäten: Die Generierung einer Forschungsfrage, die Korpusakquise, die Modellierung der Forschungsfrage, die Annotation des Korpus sowie die Verbreitung der Ergebnisse. Jede dieser Forschungsaktivitäten bringt unterschiedliche Information-Retrieval-Aufgaben mit besonderen Herausforderungen mit sich, für die, nach einer Einführung in die zentralen Information-Retrieval-Konzepte in Kapitel 2, algorithmische Ansätze vorgestellt werden.

Ein wesentliches Konzept der vorgestellten Ansätze ist das in Kapitel 3 eingeführte Keyquery-Paradigma. Hinter dem Paradigma steht eine Suchoperation, die als Antwort auf eine gegebene Menge von Eingabedokumenten relevante Suchanfragen zurückgibt. Keyqueries werden in Kapitel 4 für die Empfehlung verwandter Arbeiten, in Kapitel 5 für die Verbesserung des Zugangs zu Aspekten im Long Tail von Suchergebnislisten vorgeschlagen.

Mit Pseudo-Beschreibungen wird in Kapitel 6 ein Ansatz zur Document-Expansion vorgestellt. Der Ansatz verbessert die Suchleistung für Korpora, bei denen ursprünglich nur bibliografische Metadaten vorhanden sind. In Kapitel 7 wird das Keyquery-Paradigma eingesetzt, um auf unüberwachte Weise dynamische Taxonomien für Korpora zu generieren.

Kapitel 8 wendet sich der Exploration von annotierten Korpora zu und stellt Scoped Facets als konzeptionelle Erweiterung von facettierten Suchsystemen vor, die besonders in explorativen Suchszenarien nützlich ist. Um die wichtigsten thematischen Unterschiede und Entwicklungen in einer Sequenz von Sub-Korpora hervorzuheben, wird in Kapitel 9 ein Algorithmus zum Topical Sequence Profiling vorgestellt.

Die Arbeit schließt mit zwei Pilotstudien zur Visualisierung von Such- bzw. Forschungsergebnissen als Mittel für eine erfolgreiche Ergebnisverbreitung: eine metaphorische Interpretation des Information-Nutrition-Labels, sowie die philosophischen Körper, 3D-gedruckte Suchergebnisse.



## **Ehrenwörtliche Erklärung**

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Teile der Arbeit, die bereits Gegenstand von Prüfungsarbeiten waren, sind ebenfalls unmissverständlich gekennzeichnet.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt. Ich versichere, dass ich nach bestem Wissen die reine Wahrheit gesagt und nichts verschwiegen habe.

Weimar, 13. Januar 2022

---

Tim Gollub



# Contents

<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	Information Retrieval for the Digital Humanities (IR4DH) . . . . .	1
1.2	Research Questions and Contributions . . . . .	5
1.3	Publication Record . . . . .	9
1.4	Thesis Structure . . . . .	12
<b>2</b>	<b>INFORMATION RETRIEVAL BACKGROUND</b>	<b>13</b>
2.1	The Query Response Paradigm (QRP) . . . . .	14
2.2	Retrieval Models . . . . .	15
2.3	Relevance Computation . . . . .	17
2.4	Limitations of the QRP . . . . .	21
2.5	Extensions to the QRP . . . . .	26
<b>3</b>	<b>FROM KEYWORDS TO KEYQUERIES: CONTENT DESCRIPTORS FOR THE WEB</b>	<b>29</b>
3.1	Introduction . . . . .	29
3.2	Related Work . . . . .	31
3.3	Approach . . . . .	32
3.4	Evaluation . . . . .	35
3.5	Summary . . . . .	37
<b>4</b>	<b>SUPPORTING SCHOLARLY SEARCH WITH KEYQUERIES</b>	<b>39</b>
4.1	Introduction . . . . .	39
4.2	Related Work . . . . .	40
4.3	Baselines and Approach . . . . .	43
4.4	Combining Approaches . . . . .	47
4.5	Evaluation . . . . .	47
4.6	Summary . . . . .	52
<b>5</b>	<b>BEYOND PRECISION@10: CLUSTERING THE LONG TAIL OF WEB SEARCH RESULTS</b>	<b>55</b>
5.1	Introduction . . . . .	56
5.2	Related Work . . . . .	57
5.3	Approach and Evaluation . . . . .	59
5.4	Summary . . . . .	64

<b>6 PSEUDO DESCRIPTIONS FOR META-DATA RETRIEVAL</b>	<b>67</b>
6.1 Introduction . . . . .	68
6.2 Related Work . . . . .	69
6.3 Approach . . . . .	75
6.4 Evaluation . . . . .	78
6.5 Summary . . . . .	84
<b>7 DYNAMIC TAXONOMY COMPOSITION VIA KEYQUERIES</b>	<b>87</b>
7.1 Introduction . . . . .	87
7.2 Related Work . . . . .	91
7.3 Approach . . . . .	94
7.4 Evaluation . . . . .	101
7.5 Summary . . . . .	111
<b>8 EXPLORATORY SEARCH PIPES WITH SCOPED FACETS</b>	<b>113</b>
8.1 Introduction . . . . .	113
8.2 Related Work . . . . .	115
8.3 Approach . . . . .	116
8.4 Evaluation . . . . .	120
8.5 Summary . . . . .	121
<b>9 TOPICAL SEQUENCE PROFILING</b>	<b>123</b>
9.1 Introduction . . . . .	124
9.2 Related Work . . . . .	126
9.3 Approach . . . . .	127
9.4 Case Study . . . . .	130
9.5 Summary . . . . .	133
<b>10 CONCLUSION</b>	<b>135</b>
10.1 A holistic view on the DH research process . . . . .	135
10.2 (Re)search Result Presentation . . . . .	136
<b>REFERENCES</b>	<b>141</b>

# 1

## Introduction

This thesis is about information retrieval tasks that arise in digital humanities research. This first chapter serves as an introduction to the topic. It presents a set of five research activities that commonly arise in digital humanities projects, the information retrieval tasks these activities elicit, and how I approached the information retrieval tasks in the publications that are accumulated in the following chapters. My publication record and an overview of the thesis' structure conclude the chapter.

### 1.1 INFORMATION RETRIEVAL FOR THE DIGITAL HUMANITIES (IR4DH)

The digital humanities (DH) are concerned with the application of digital infrastructure and computational methods to the disciplines of the humanities (cf. Berry and Fagerjord, 2017). DH projects commonly center around the acquisition and analysis of a digital corpus, i.e., a collection of digitized or born-digital cultural artifacts (like books, paintings, sculptures, or recordings) which are related to a specific context. Among the most prominent digital humanities projects are digitization initiatives like Google Books<sup>1</sup>, the Europeana<sup>2</sup>, or the Perseus project<sup>3</sup>, as well as computational methods applied to these resources, like cultural analytics (Michel et al., 2011; Kozlowski et al., 2018; Manovich, 2020) and distant reading (Moretti, 2013; Underwood, 2017).

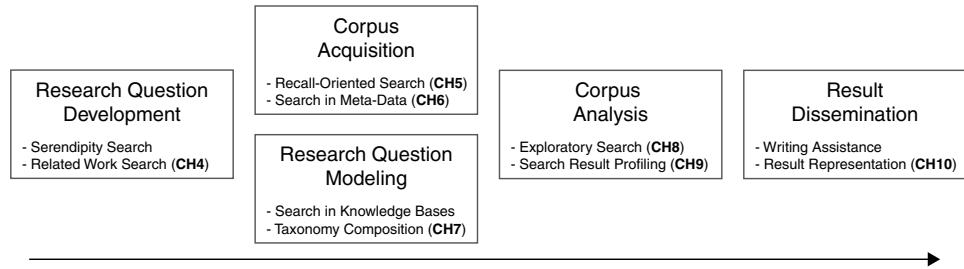
Information retrieval (IR) in turn is a research field concerned with satisfying the information needs of users with the aid of computer systems (cf.

---

<sup>1</sup><https://books.google.com>

<sup>2</sup><https://www.europeana.eu>

<sup>3</sup><http://www.perseus.tufts.edu>



**FIGURE 1.1:** A conceptualization of the digital humanities research process with outlined information retrieval tasks. The chapter numbers in parenthesis indicate where in the thesis research contributions concerning the respective retrieval task are presented.

Manning et al., 2008). In the present case of IR for DH (IR4DH), users happen to be DH researchers, and their information needs are elicited by their research activities in DH projects. The five DH research activities addressed in this thesis are illustrated in Figure 1.1. From (1) the development of an initial research question, the activities comprise (2) the acquisition of a research corpus, (3) the development of a research model, (4) the annotation and analysis of the corpus with respect to the model, and, ultimately, (5) the synthesis and dissemination of the obtained research results. In the following paragraphs, each research activity is discussed, with a focus on the special IR tasks it elicits. Note that in accordance with information retrieval terminology, which is introduced in Chapter 2 and extended in Chapter 3, the elements of a corpus are referred to as *documents* in this thesis, irrespective of their specific media type.

The most fundamental DH research activity is the *development of an initial research question*, which is depicted in the left most box of Figure 1.1. The goal of research question development is to find an interesting yet open research question that can be answered with the analysis of an acquirable corpus. An example of a potentially relevant research question may be: “Which organisms had been used in life science experiments at the end of the 19th century?” Coming up with a relevant research question is a creative hence elusive process, where accident and sagacity play important roles. An IR task devoted to actively creating the conditions for unexpected discoveries and innovation events is called *serendipity search* (Carr, 2015; Reviglio, 2018). Once a potential research question has been formulated, a thorough survey of related work is good scientific practice, in order to check whether the question is indeed still open, but also to find related research to refer to. Supporting the effective retrieval of related work is the goal of the second IR task which is specific to research question development, namely *related*

*work search*. Related work search is point of discussion in Chapter 4.

DH research questions, like the example above, contain two kinds of information: corpus-related and model-related information. Corpus-related information pertain to the properties relevant documents must have, and are hence leveraged for *corpus acquisition*, the second research activity in Figure 1.1. The goal of corpus acquisition is to find, collect and edit a representative corpus that allows to derive an answer to the posed research question. E.g., the example research question above suggests to acquire a representative corpus of life science research papers from around 1900, from which descriptions of experiments on organisms could be mined. An issue for corpus acquisition by the means of search in digital libraries and archives is that often a minority of popular documents seizes the top search results for a majority of relevant queries, making less prominent documents hard to retrieve (Azzopardi and Vinay, 2008). *Recall-oriented search* is an IR-task that aims at diminishing this retrievability issue, e.g. by providing a subject-based information access in addition to relevance-based result lists. Recall-oriented search is point of discussion in Chapter 5. A further retrieval problem arises if only bibliographic meta-data about each document is available. Due to the sparsity of linguistic features, common algorithms for determining the relevance of a document for a search query fail to produce useful results (Efron et al., 2012). This problem arises especially for the acquisition of historic or non-linguistic corpora. The IR task devoted to improve retrieval performance in such cases, e.g. by adding linguistic features from external sources, is called *search in meta-data*. Search in meta-data is point of discussion in Chapter 6.

The activity *research question modeling* is depicted below corpus acquisition in Figure 1.1. Research question modeling leverages the model-related information of the research question. The goal of research question modeling is to specify the concepts and relations brought up by the developed research question, and to devise formal methods for annotating instances of those concepts and relations in a corpus. Considering the example research question from above, a respective model ought to specify the two concepts <experiment> and <organism>, as well as the relation “organism <used in> experiment”. Respective annotation methods could be in the form of annotation guidelines devised for consistent human annotation, but increasingly, the application of computational methods becomes the preferred practice (Berry, 2012). A valuable resource for model generation are knowledge bases, such as DBpedia oder Wikidata. Considering again our example, a knowledge base may be applied to derive a taxonomy of organism names and synonyms. With such an external taxonomy, the de-

tection of organism mentions reduces, in terms of complexity, from an open concept recognition task to a closed concept detection task. Hence, *search in knowledge bases* (Bast et al., 2016; Jimmy et al., 2018) is a retrieval task worth considering in the context of DH. Another benefit of using corpus-agnostic models retrieved from knowledge bases is that, when applied to a corpus, the absence of specific concepts and relations can be attested. Still, in other occasions, research questions may ask for a taxonomy which summarizes the concepts which are in fact present in a corpus. One possibility to compose such a data-driven taxonomy is based on search engine technology. The respective IR task has been called *dynamic taxonomy composition* (Sacco and Tzitzikas, 2009). Dynamic taxonomy composition is point of discussion in Chapter 7.

When both a corpus and a model have been established, an answer to the formulated research question can be obtained by *corpus analysis*, the fourth research activity in Figure 1.1. Corpus analysis comprises the annotation of the corpus according to the established model, as well as statistical analysis. Considering the example question about organisms in life science experiments, an answer could take the form of a list, which names all organisms found in the context of experiment descriptions together with respective occurrence statistics. Confronted with such an answer, an obvious next step is to explore the answer, be it for verifying the validity of the underlying annotations, or for the generation of more specific follow-up research questions. The IR-task devoted to the exploration of annotated corpora is called *exploratory search* (White and Roth, 2009). Exploratory search is point of discussion in Chapter 8. In the specific but common case where a research question asks for trends within a sequence of corpus subsets (e.g., whether trends of organism-use can be observed over time), algorithms for the comparative IR task *search result profiling* can be applied. Search result profiling is point of discussion in Chapter 9.

The last research activity in Figure 1.1 is *result dissemination*, which includes all efforts towards the publication of achieved research results. IR-tasks designed to support the publication process are search technologies for *writing assistance* such as Netspeak (Potthast et al., 2010), and also *result presentation*, an IR-task concerned with the question of how search results can be communicated to the user effectively. Result presentation is point of discussion in the thesis' conclusion.

## 1.2 RESEARCH QUESTIONS AND CONTRIBUTIONS

The overall goal of my research is to invent and develop information retrieval technologies that respond to the research questions of humanities scholars. This section summarizes the contributions I made towards this goal by presenting the specific research questions my contributions address. I further hope that with this thesis, I raise awareness that search technology can do more to DH than supporting lookup searches in digital libraries: Search engines can be the technological backbone of digital humanities projects that ultimately delivers an answer to the research questions posed.

### 1.2.1 What makes IR4DH different from Web search (CH 3)?

A premise for the development of IR4DH technology is to develop a decent understanding of the information needs DH researchers have, and how these needs differ from the information needs supported by standard Web search engines.

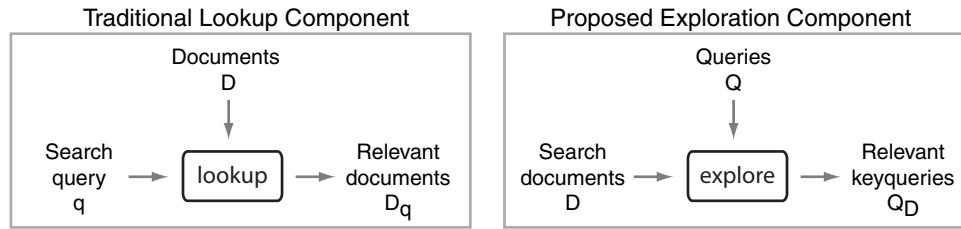
Most search engines for the Web, like DuckDuckGo<sup>4</sup> or Google<sup>5</sup>, are designed to retrieve documents which can give an explicit answer to the user's information need. The underlying assumption of Web search is that the user is interested in the best existing answers that have been published, and the task of the search engine is to point the user to these documents. If, for example, a user enters the search query "organisms in 19th century life science experiments", a Web search engine is supposed to retrieve state-of-the-art documents where the authors elaborate on organism use in life science experiments back in the 19th century. On the contrary, in the case of IR4DH, the assumption is that no document exists upfront which satisfies the user, but it is the overall goal of the user to create one with the aid of information retrieval technology. Hence, the essential task in IR4DH must be to retrieve, as exclusive and comprehensively as possible, the evidence that is needed to synthesize an answer to a given information need. In addition, means for the analysis and exploration of this potentially huge result set must be provided.

My conceptual contribution to address this requirement is to introduce an additional component to the traditional search engine architecture: an exploration component, which operates conversely to the traditional lookup component. It takes as input a set of documents, and returns, in response, a ranked list of relevant queries called the *keyqueries* of the input documents. The converse functionality of the two components is illustrated in Figure 1.2,

---

<sup>4</sup><https://duckduckgo.com>

<sup>5</sup><https://google.com>



**FIGURE 1.2:** Comparison of the traditional lookup component (left) and the proposed complementary exploration component (right). My research contributions combine both components in different ways to solve specific retrieval problems.

and discussed in depth in Chapter 3. As will become apparent from the following research questions, by wiring up the exploration and the lookup component in different ways, interactive retrieval scenarios arise which approach the IR tasks introduced in Section 1.1.

### 1.2.2 To which extend can keyqueries complement citation-based related work search (CH 4)?

Identifying related work is an integral part of research question development. Often, an initial set of related work is already at hand or can be retrieved by issuing a lookup query to a scholarly search engine. Given such an initial set, a practical approach to further extend the set is to exploit citation information in the documents by means of collaborative filtering. An alternative possibility is to make use of the documents' content. In the work presented in Chapter 4, we present an algorithm for related work search based on the keyqueries of the given initial document set, i.e., the search queries which retrieve the initial document set when issued as a lookup query. Given the keyqueries for the initial set, further related work is recommended by making a lookup search with the obtained keyqueries, and then recommending the relevant results which are not already part of the initial set. The conducted experiments show that the effectiveness of the approach is on par with citation based methods, but with substantially different recommendation results. Most effective is a combination of both approaches, which substantiates the claim that keyquery-based related work search complements citation-based approaches.

### **1.2.3 How to improve access to information that resides solely in the long tail of search results (CH 5)?**

A further possibility to apply the keyquery paradigm is to filter the long tail of a search result list for documents which are textually different from those at the top of the search result list. This application is presented in Chapter 5 in the context of recall-oriented search. To remove the search results which „shadow“ potentially interesting outliers from the tail, keyqueries obtained for the documents in the result list head are added to the base query as not expressions. This way, only documents relevant for the base query but not for queries addressed already in the result list head are retained. With the aid of this long tail filtering, serendipitous discoveries in the long tail of a search result list can be fostered and, by clustering the retained tail documents, the spectrum of available subtopics can be presented with high recall. The experiments conducted reveal that our keyquery-based approach is able to significantly reduce the shadowing effect while it retains the intended outliers.

### **1.2.4 How to exploit the prevalence of large external corpora like the Web for search in meta-data (CH 6)?**

Chapter 6 is about search in meta-data repositories with sparse linguistic information. To improve retrieval performance in such collections, the presented approach suggests to augment the meta-data of each document with a pseudo-description. The pseudo-description is composed by exploiting the search result snippets obtained from issuing the available linguistic meta-data as lookup search query to a reference search engine (e.g. a Web search engine). To prevent that irrelevant information is added to the document representation, the decision up to which rank search results are taken over into the pseudo-description is relevance-dependent. The experiments conducted on several TREC corpora reveal that the search performance with pseudo-descriptions achieves a significant improvement over meta-data only search.

### **1.2.5 How to dynamically compute taxonomies over corpora on the basis of search queries (CH 7)?**

DH research questions sometimes ask for an overview over the contents of a corpus. E.g., given a corpus of life science publications from the 19th century, it may be interesting to learn about its major themes. In Chapter 7, a keyquery-based approach that computes dynamic taxonomies over corpora

is presented for this purpose. The algorithm takes as input a corpus and a fan-out parameter  $k$ , which determines the maximum number of classes on each taxonomy level. In each iteration of taxonomy composition, a set of up to  $k$  keyqueries is determined which together retrieve as many documents of the parent class with as little overlap as possible. Empirical evaluations with large scientific corpora demonstrate the efficiency and effectiveness of dynamic taxonomy composition with keyqueries.

### **1.2.6 How to design faceted search interfaces which support both drill-down and exploration (CH 8)?**

Once a corpus has been annotated with respect to a specific model, the modeled concepts can be made available in the user interface of a search engine as facets. E.g., a search engine indexing a corpus of life science experiments could feature facets for concepts like organisms, dates of publication, or a topic taxonomy. Commonly, faceted search interfaces are designed in such a way that users can only progressively narrow down search into smaller result sets by selecting facet terms. In Chapter 8, we argue that for the exploration of annotated corpora, it is at least equally important to be able to broaden a search by facet term selection in order to refocus and explore related but different result sets. To give the user the ability to decide whether to drill-down or to explore, the concept of scoped-facets is introduced in combination with faceted search pipes. A user study that compares a prototypical implementation of scoped facets with a traditional faceted search system as baseline reveals that for complex exploration tasks, the user experience with scoped facets is clearly superior.

### **1.2.7 How to convey the most important topical differences in a sequence of document collections (CH 9)?**

Facets can be used to divide a corpus into sub collections. If the values of a facet are cardinally scaled, a sequence of sub collections is formed. A frequently available example of such a facet is the publication date of the documents in a corpus. Chapter 9 deals with the question of how to generate a profile for a given sequence of document collections that allows users to quickly survey the sequence's most characteristic topics<sup>6</sup>. The presented algorithm is designed to compose a minimal list of topics that is representative for every individual sub-collection, and at the same time, highlights

---

<sup>6</sup>The publication focuses on topics but in principal, any other facet, e.g. authors, could be chosen for the profile

those topics which fluctuate most across the sequence. The potential of topical sequence profiling for the analysis of annotated corpora is demonstrated with a case study on a sequence of conference proceedings.

### 1.2.8 Does data physicalization provide added value for the dissemination of humanities research results (CH 10)?

In the conclusion of this thesis, the question of how to effectively disseminate the research results encoded in annotated corpora to a wider public is raised. Besides visualization techniques like the information nutrition label, where we propose to communicate information by means of metaphoric symbols, the physicalization of information is an attractive possibility to showcase humanities research in the public. With the philosophical bodies, a pilot study devised to get first insights about data physicalization for the dissemination of DH research results is presented. Though more research is required to come to extensive conclusions, we learned that showcasing data physicalizations in exhibitions is an effective means to grasp visitors' attention.

## 1.3 PUBLICATION RECORD

The following list gives an overview over the research papers that are accumulated in this thesis. The first column shows in which chapters content of the respective publication is integrated.

**TABLE 1.1: Overview of peer-reviewed publications this thesis is based on. For each publication, the chapter(s) in which content is reused is given, as well as the publication venue and type, the extend of publication reuse, and the original number of pages.**

Ch.	Reference	Venue	Type	Reuse	Pages
2	Stein et al. (2017)	ESNAM	chapter	reprint	9
	<i>Benno Stein, Tim Gollub, and Maik Anderka. Encyclopedia of Social Network Analysis and Mining (ESNAM), chapter Retrieval Models.</i>				
3	Gollub et al. (2013a)	SIGIR	conference	reprint	4
	<i>Tim Gollub and Matthias Hagen and Maximilian Michel and Benno Stein. From Keywords to Keyqueries: Content Descriptors for the Web.</i>				
3	Fuhr et al. (2012)	IR Journal	journal	mention	22
	<i>Norbert Fuhr, Marc Lechtenfeld, Benno Stein, and Tim Gollub. The Optimum Clustering Framework: Implementing the Cluster Hypothesis.</i>				
4	Hagen et al. (2016)	ECIR	conference	reprint	12

*Matthias Hagen, Anna Beyer, Tim Gollub, Kristof Komlossy, and Benno Stein.  
Supporting Scholarly Search with Keyqueries.*

5	Stein et al. (2011)	CIKM	conference	reprint	4
<i>Benno Stein, Tim Gollub, and Dennis Hoppe. Beyond Precision@10: Clustering the Long Tail of Web Search Results.</i>					
5	Stein et al. (2012)	CIKM	conference	excerpt	4
<i>Benno Stein, Tim Gollub, and Dennis Hoppe. Search Result Presentation Based on Faceted Clustering.</i>					
5	Gollub et al. (2016a)	AIRS	conference	excerpt	12
<i>Tim Gollub, Matthias Busse, Benno Stein, and Matthias Hagen. Keyqueries for Clustering and Labeling.</i>					
5	Lipka et al. (2020)	USPTO	patent	mention	18
<i>Nedim Lipka, Tim Gollub, Eunyee Koh. Organizing electronically stored files using an automatically generated storage hierarchy.</i>					
6	Gollub et al. (2018a)	ICTIR	conference	reprint	8
<i>Tim Gollub, Erdan Genc, Nedim Lipka, and Benno Stein. Pseudo Descriptions for Meta-Data Retrieval.</i>					
7	Gollub et al. (2014)	JCDL	conference	reprint	10
<i>Tim Gollub, Michael Völske, Matthias Hagen, and Benno Stein. Dynamic Taxonomy Composition via Keyqueries.</i>					
7	Völske et al. (2014)	WOSP	conference	excerpt	4
<i>Michael Völske, Tim Gollub, Matthias Hagen, and Benno Stein. A Keyquery-Based Classification System for CORE.</i>					
7,9	Gollub and Stein (2010)	GFKL	conference	mention	8
<i>Tim Gollub and Benno Stein. Unsupervised Sparsification of Similarity Graphs.</i>					
8	Gollub et al. (2019)	ICTIR	conference	reprint	4
<i>Tim Gollub, Leon Hutans, Tanveer Al Jami, and Benno Stein. Exploratory Search Pipes with Scoped Facets.</i>					
9	Gollub et al. (2016b)	TIR	conference	reprint	5
<i>Tim Gollub, Nedim Lipka, Eunyee Koh, Erdan Genc, and Benno Stein. Topical Sequence Profiling.</i>					
10	Gollub et al. (2018b)	NewsIR	conference	excerpt	3
<i>Tim Gollub, Martin Potthast, and Benno Stein. Shaping the Information Nutrition Label.</i>					
10	Schmidgen et al. (2021)	ZfDG	journal	excerpt	14
<i>Henning Schmidgen, Benno Stein, Tim Gollub, Michael Braun, Jan Willmann. Philosophische Körper. Von digitalem Text zu greifbaren Material. (german)</i>					

In addition to the publications covered in the thesis, I contributed towards the issues of reproducibility and communality in the scientific practice of computer science research. The contributions pertain to our evaluation as a service platform called TIRA<sup>7</sup>, as well as the organization of various shared tasks, including most prominently the Clickbait Challenge<sup>8</sup>.

**TABLE 1.2: Overview of peer-reviewed publications not included in the thesis.**

Reference	Venue	Type	Pages
Potthast et al. (2019)	IR series	chapter	41
<i>Martin Potthast, Tim Gollub, Matti Wiegmann, and Benno Stein. TIRA Integrated Research Architecture.</i>			
Potthast et al. (2018a)	CoRR	journal	6
<i>Martin Potthast, Tim Gollub, Matthias Hagen, and Benno Stein. The Clickbait Challenge 2017: Towards a Regression Model for Clickbait Strength.</i>			
Potthast et al. (2018b)	COLING	conference	10
<i>Martin Potthast, Tim Gollub, Kristof Komlossy, Sebastian Schuster, Matti Wiegmann, Erika Patricia Garces Fernandez, Matthias Hagen, and Benno Stein. Crowdsourcing a Large Corpus of Clickbait on Twitter.</i>			
Qu et al. (2018)	HComp	conference	10
<i>Jiani Qu, Anny Marleen Hijßbach, Tim Gollub, and Martin Potthast. Towards Crowdsourcing Clickbait Labels for YouTube Videos.</i>			
Hopfgartner et al. (2018)	JDIQ	journal	32
<i>Frank Hopfgartner, Allan Hanbury, Henning Müller, Ivan Eggel, Krisztian Balog, Torben Brodt, Gordon V. Cormack, Jimmy Lin, Jayashree Kalpathy-Cramer, Noriko Kando, Makoto P. Kato, Anastasia Krithara, Tim Gollub, Martin Potthast, Evelyne Viegas, and Simon Mercer. Evaluation-as-a-Service for the Computational Sciences: Overview and Outlook.</i>			
Potthast et al. (2014)	CLEF	journal	32
<i>Martin Potthast, Tim Gollub, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. Improving the Reproducibility of PAN’s Shared Tasks: Plagiarism Detection, Author Identification, and Author Profiling.</i>			
Gollub et al. (2013b)	CLEF	journal	20
<i>Tim Gollub, Martin Potthast, Anna Beyer, Matthias Busse, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. Recent Trends in Digital Text Forensics and its Evaluation.</i>			
Gollub et al. (2012b)	TIR	conference	5
<i>Tim Gollub, Benno Stein, Steven Burrows, and Dennis Hoppe. TIRA: Configuring, Executing, and Disseminating Information Retrieval Experiments.</i>			
Gollub et al. (2012a)	SIGIR	conference	4
<i>Tim Gollub, Benno Stein, and Steven Burrows. Ousting Ivory Tower Research: Towards a Web Framework for Providing Experiments as a Service.</i>			

<sup>7</sup><https://www.tira.io/>

<sup>8</sup><https://webis.de/events/clickbait-challenge/shared-task.html>

## 1.4 THESIS STRUCTURE

The remainder of the thesis is structured as follows. In Chapter 2, an introduction to core information retrieval concepts is given, including the important notions of exploratory search and interactive information retrieval. In Chapter 3, the keyquery paradigm is introduced which constitutes my conceptual contribution for solving the special IR tasks that arise in DH research projects. In Chapters 4 to 9, my algorithmic contributions towards solving specific IR4DH tasks are presented together with empirical findings. Chapter 10 concludes with a presentation of two pilot studies concerning the dissemination of (re-)search results by the means of visualization and physicalization.

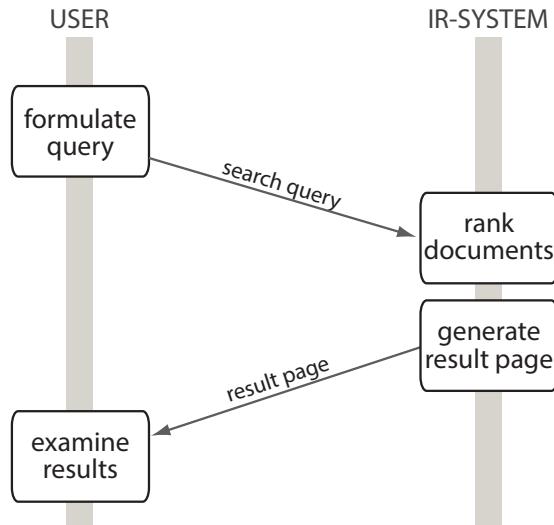
# 2

## Information Retrieval Background

In order to store and share information we produce documents. Books, images, videos, audio recordings, websites – in information retrieval research all of these are collectively considered documents that contain information. And there are a lot of documents. According to Völske et al. (2021), the indexed World Wide Web on its own comprises approximately 60 billion pages. This massive amount of documents is an invaluable resource when it comes to solving our everyday tasks. We “only” have to find the right ones at the right time.

The research field information retrieval (IR) is concerned with the development of systems that allow users to effectively find the documents that solve their information needs, especially within large document collections such as the World Wide Web (cf. Manning et al., 2008). A system approaching this task is called an information retrieval system or, in short, IR system. The most fundamental way to interact with an IR system is by the means of the query response paradigm, which is presented in Section 2.1. Given a user’s search query, an IR system computes the relevance of the indexed documents and return them in a result page. How the relevance of a document for a query can be determined is subject of Sections 2.2 and 2.3.

Though the query response paradigm is sufficient in many cases, user-centered IR research revealed that in some important situations, the paradigm can hardly solve the users’ information need. Section 2.4 gives an introduction to user-centered IR research, then focuses on the limitations of the query response paradigm. Conceptual extensions to the query response paradigm, namely interactive and dynamic information retrieval, as well as faceted search, are the subject in Section 2.5.



**FIGURE 2.1:** Sequence model of the query-response paradigm. The user first formulates an information need in terms of a search query. The IR-system then ranks the indexed documents by relevance and generates a result page. The user finally examines the result page.

## 2.1 THE QUERY RESPONSE PARADIGM (QRP)

IR systems can be characterized by the user interaction model they implement. User interaction models provide a protocol for the communication between a user and an IR system. The most fundamental user interaction model most widely applied in commercial information retrieval systems is the query response paradigm (White and Roth, 2009) or adhoc model, which models interaction with an IR system as two user activities and two intermediate system activities. This interaction sequence is depicted in Figure 2.1. Given a user with an information need, the user first expresses the information need linguistically in terms of a search query  $q$ , e.g. as a list of keywords or as a natural language question. To respond to the query, the IR system first ranks the documents by relevance to the query, and then assembles the most relevant subset  $D_q$  to a result page. Concerning the design of the result page, the most prevalent approach is to present  $D_q$  in terms of a list of documents excerpts (snippets), each containing a link to the respective document. To determine the order of the snippets in the result list, the probability ranking principle (PRP) is typically applied (Robertson, 1997): Given a query  $q$ , the ranking of  $D_q$  according to their probabilities of being relevant to  $\mathcal{I}$  leads to the optimum retrieval performance. The examination of the result page and individual documents constitutes the second user activity, which is performed until the user finds one or multiple documents

$d \in D_q$  that satisfy  $\mathcal{I}$  or reaches the end of the personal examination scope (typically less than 30 results (iProspect, 2008)).

## 2.2 RETRIEVAL MODELS

Retrieval models implement the system side of a particular user interaction model. The core task of any retrieval model is hence to assess the relevance of a document  $d$  for a search query  $q$  with the aid of a linguistic theory. The historical development of well-known retrieval models is illustrated in Figure 2.2. In the literature, a distinction between *empirical models*, *probabilistic models*, and *language models* is often made, which is mainly rooted in the linguistic theory they operationalize but also has historical reasons.

1. Empirical models, sometimes referred to as vector space models, focus on the document representation (Salton and McGill, 1983). Both documents and queries are considered as high-dimensional vectors in the Euclidean space, where a compatible representation is presumed: a particular document term or query term is always associated with the same dimension, whereas the term importance is specified by a weight. Usually, the cosine of the angle between two such vectors or simply their dot product is used to quantify their similarity; in particular, the concept of similarity is put on a level with the concept of relevance. Empirical models can be distinguished with regard to the dimensions that are considered (features that are chosen) and how these dimensions (features) are weighted.
2. Probabilistic models strive for an explicit modeling of the concept of relevance. Bayesian statistics is applied to derive an estimate of the probability that a document is relevant for a given information need. Most probabilistic models employ conditional probabilities to quantify document relevance given the occurrence of a term.
3. Language models are based on the idea of language generation as it is used in speech recognition systems. A language-based retrieval model is computed specifically for each document in a collection and is usually term-based. Given a query  $q$ , document ranking happens according to the generation probability of  $q$  under the language model of the respective document.

From each of the three modeling paradigms selected representatives are in the following characterized along with the respective publications.

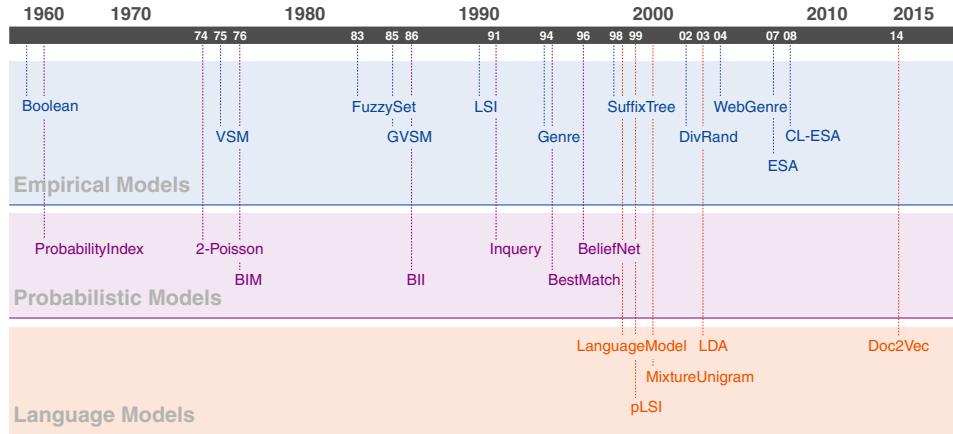


FIGURE 2.2: Historical development of retrieval models, organized according to three paradigms: empirical models, probabilistic models, and language models.

The Boolean retrieval model uses binary term weights, and a query is a Boolean expression with terms as operands. Drawbacks of the Boolean model include its simplistic weighting scheme, its restriction to exact matches, and that no document ranking is possible. The Vector Space Model (VSM) and its variants consider documents and queries as embedded in the Euclidean space (see above). Key challenge for these kinds of models is the term weighting. Salton et al. (1975) proposed the  $tf \cdot idf$ -scheme, which combines the term frequency  $tf$  (the number of term occurrences in a document) with the inverse document frequency  $idf$  (the inverse of the number of documents that contain this term). The Latent Semantic Indexing (LSI) model was developed to improve query interpretation and semantic-based matching (Deerwester et al., 1990). E.g., a document  $d$  should match a query even if the user specified valid synonyms that do not occur in  $d$ . The LSI model attempts to achieve such effects by projecting documents and queries into a “semantic space”, which is constructed by a singular value decomposition of the term-document-matrix. The Explicit Semantic Analysis (ESA) model was introduced to compute the semantic relatedness of natural language texts (Gabrilovich and Markovitch, 2007). The model represents a document  $d$  as a high-dimensional vector whose dimensions quantify the pairwise similarities between  $d$  and the documents of some reference collection such as Wikipedia. Potthast et al. (2008) demonstrated how the ESA principles are applied to develop an effective cross-language retrieval approach, the so-called CL-ESA model. In contrast to most retrieval

models, the Suffix Tree Model represents a document  $d$  not as a vector of index terms but as a compressed trie containing all suffixes (i.e., suffixes of all lengths) of a text  $d$ . As a consequence, the collocation information of  $d$  is preserved, which may render the model superior for particular retrieval tasks (Meyer zu Eissen et al., 2005).

Under the Binary Independence Model (BIM) the documents are ranked by decreasing probability of relevance (Robertson and Sparck-Jones, 1976). The model is based on two assumptions which allow for a practical estimation of the required probabilities: documents and queries are represented under a Boolean model, and, the terms are modeled as occurring independently of each other. The Best Match (BM) model computes the relevance of a document to a query based on the frequencies of the query terms appearing in the document and their inverse document frequencies (Robertson and Walker, 1994). Three parameters tune the influence of the document length, the document term frequency, and the query term frequency in the model. The Best Match model belongs to the most effective retrieval models in the Text Retrieval Conference (TREC) series.

The Language Modeling approach to information retrieval was proposed by Ponte and Croft (1998); the idea is to rank documents by the generation probabilities for a given query (see above). The algorithmic core of the model is a maximum likelihood estimation of the probability of a query term under a document's term distribution. The Latent Dirichlet Allocation (LDA) model is a sophisticated generative model in the context of probabilistic topic modeling (Blei et al., 2003). Under this model it is assumed that documents are composed as a mixture of latent topics, where each topic is specified as a probability distribution over words. The mixture is generated by sampling from a Dirichlet distribution. More recently, Le and Mikolov (2014) introduced Paragraph Vector, also known as the Doc2Vec model, which learns continuous distributed vector representations for documents using a neural network classifier.

### 2.3 RELEVANCE COMPUTATION

Despite the large variety of retrieval models that have been developed, computing the relevance  $\rho$  of a document for a query usually boils down to a multiplication of two feature vectors: (1) a feature vector  $\mathbf{q}$  representing query  $q$ , and (2) a feature vector  $\mathbf{d}$  representing document  $d$ :

$$\rho(q, d) = \mathbf{q}^T \cdot \mathbf{d} = \sum_{i=1}^{|\mathbf{q}|} \mathbf{q}_i \cdot \mathbf{d}_i$$

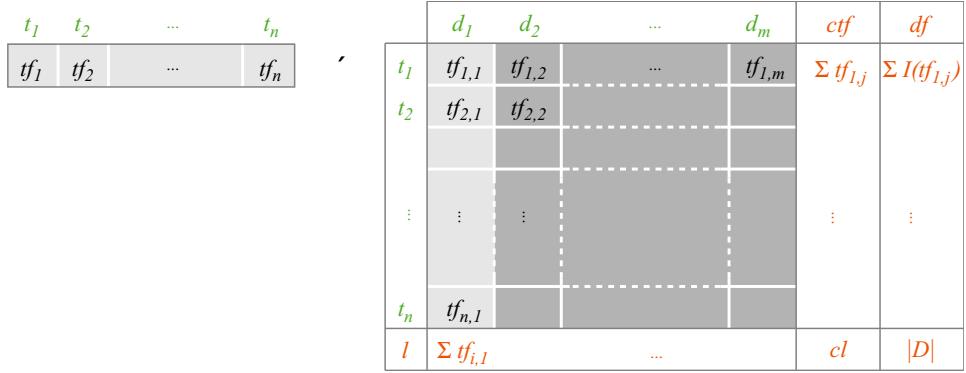


FIGURE 2.3: Basic model for relevance computation: the term frequency vector of a query (left) is combined with the term frequency vectors of the documents  $d_j$  from a document collection  $D$  (shaded matrix on the right).

What distinguishes retrieval models from each other is the feature set that they employ for representing queries and documents, as well as the computation rule used to calculate the respective feature weights. In the following, the feature sets and the computation rules for four retrieval models are outlined, starting from the basic  $tf$ -Model to the more sophisticated models  $tf \cdot idf$ , BM25, and ESA.

*tf-Model*. The  $tf$ -Model (*term frequency* model) is a variant of the Vector Space Model that uses the vocabulary of the given document collection  $D$  as feature set. The dimension  $i$  of a feature vector is associated with a specific term  $t_i$  that occurs in  $D$ . As feature weight, the frequency  $tf$  by which  $t_i$  appears in a specific query or document is taken:

$$\mathbf{q}_i^{tf} = tf(t_i, q)$$

$$\mathbf{d}_i^{tf} = tf(t_i, d)$$

Stacking the  $tf$  feature vectors of all documents  $d \in D$  as columns into a matrix gives the so called *term-document-matrix*, a data structure from which many retrieval models can be derived. The term-document-matrix (along with additional statistics used later on, shown in red) is depicted in Figure 2.3 as shaded area on the right-hand side. On the left-hand side, a canonical query vector is shown. Multiplying the query vector with the term-document-matrix results in a vector containing the relevance scores for all documents in  $D$ . Note that in practice, to facilitate instant search results even over trillions of documents, IR systems maintain an inverted index  $\mu : V \rightarrow 2^D$  that maps the vocabulary  $V$  of search terms to the subset of documents that are relevant for a term. Given a search query, the inverted

index allows to efficiently compute  $\rho(q, d)$  for all  $d \in D$  as it can be used to exclude all documents from the computation that are not at least partially relevant for the query. A particular property of the  $tf$ -Model is that the relevance computation for a document  $d$  is independent of the collection  $D$  from which  $d$  is taken. However, the model has certain weaknesses that the following models try to alleviate.

*$tf \cdot idf$ -Model*. An extension of the  $tf$ -Model is the  $tf \cdot idf$ -Model, where  $idf$  stands for *inverse document frequency*. The linguistic intuition behind this extension is that the occurrence of a rare query term in a document is a better indicator for relevance than the occurrence of a frequent term. Considering the query “math for computer science” as an example, the occurrence of the term “for” in a document provides, in comparison to the occurrence of “math” or “computer science”, only little evidence about the relevance of a document—a fact which is not exploited in the  $tf$ -Model. A document containing the query term “for” ten times is considered as relevant as a document containing the query term “math” ten times. To address this deficit, the  $tf \cdot idf$ -Model incorporates the document frequency  $df$  of a term into the feature weight computation for documents. The document frequency denotes the number of documents in  $D$  that contain a term. In Figure 2.3,  $df$  is illustrated as an additional column of the term-document-matrix. To formalize the computation of  $df$  the indicator function  $I(\cdot)$  is used, which yields 1 in case  $tf(t_i, d_j) > 0$  and 0 otherwise. Since the influence of a term should decrease with its document frequency, an *inverse document frequency* factor is added to the  $tf$  feature weight computation. Note that several variations for this factor have been proposed. The original formula by Salton et al. (1975) reads as follows:

$$\mathbf{d}_i^{tfidf} = tf(t_i, d) \cdot \log \frac{|D|}{df_i}$$

*BM25-Model*. The BM25-Model is a further advancement of the  $tf \cdot idf$ -Model. The development of the model was driven by the observation that the  $tf \cdot idf$ -Model (1) is biased towards long documents, and (2) that it insufficiently favors documents containing all query terms—compared to documents containing only a subset of the query terms. To account for the first observation, the BM25-Model introduces a length normalization factor to the feature weight computation. The length of a document is considered as the sum of its term frequencies. In Figure 2.3, the document length  $l$  is illustrated as an additional row of the term-document-matrix. The sum over all  $l \in D$  gives the overall collection length  $cl$ . The idea of the length

normalization factor is to calculate the average length  $\hat{l}$  of the documents,  $\hat{l} = cl/|D|$ , and to penalize documents that are longer than the average, while rewarding shorter documents. To account for the second observation, a term frequency normalization factor is introduced. Given the above example query “math for computer science”, the goal of this factor is to consider a document containing both “math” and “computer science” once as more relevant than a document containing one of the terms twice, even if the terms have equal document frequency. The BM25 approach applies a “logarithmic-shaped” function to the term frequency value in order to limit the contribution of a single term to the overall relevance score. The general form of this function is  $\frac{tf}{tf+c}$ , where  $c$  is a constant. The final BM25 formula for the computation of feature weights, which incorporates both normalization factors into a single expression, is the product of extensive empirical evaluation efforts:

$$\mathbf{d}_i^{bm25} = \frac{tf(t_i, d) \cdot (k_1 + 1)}{tf(t_i, d) + k_1 \cdot (1 - b + b \cdot \frac{l(d)}{\hat{l}})} \cdot \log \frac{|D| - df_i + 0.5}{df_i + 0.5}$$

For the two parameters of the function, values of  $k_1 = [1.2, 2.0]$  and  $b = 0.75$  are considered standard choices. The two normalization factors (length normalization and term frequency normalization) are balanced by the parameter  $b$ . The last factor of the formula is the BM25 variant of the inverse document frequency.

*ESA-Model.* The ESA-Model represents a class of retrieval models that do not employ terms as features but concepts or topics (hence called “topic models”). Other retrieval models of this kind are LSI and LDA. Topic models aim to further improve the assessment of relevance by taking the semantic relatedness of terms into account. The intuition is that if a document contains terms related to the query terms, like “statistics” or “calculus” which are related to the query term “math”, or “programming” and “algorithm” which are related to “computer science”, the relevance of this document should be raised. To operationalize this idea, topic models represent queries and documents by a feature vector of topics and provide a means to compute the relevance of a topic for a query or document. In the case of ESA, topics are drawn randomly from the set of Wikipedia articles, and the  $tf \cdot idf$ -Model is employed to represent each drawn article  $a$  as a term based feature vector  $\mathbf{a}$ . The assumption underlying this approach is that each feature vector  $\mathbf{a}$  will contain high  $tf \cdot idf$  scores for semantically related terms. To compute the relevance of a topic for a document or query, the cosine similarity between the  $tf \cdot idf$  representation of the topic and the

$tf \cdot idf$  representation of the query or document is used:

$$\mathbf{q}_i^{esa} = \cos(\mathbf{q}^{tfidf}, \mathbf{a}_i^{tfidf})$$

$$\mathbf{d}_i^{esa} = \cos(\mathbf{d}^{tfidf}, \mathbf{a}_i^{tfidf})$$

## 2.4 LIMITATIONS OF THE QRP

In addition to research on retrieval models, an important aspect for advancing IR research is to study and model the users of existing IR systems. User models operate on different scale, and aim to describe either (1) the broader sociological, cultural, and operational context or task into which information retrieval is embedded, (2) cognitive processes in users' minds that lead to the creation, change, and satisfaction of information needs, as well as (3) the different types of information needs.

Concerning (1) the broader context into which information retrieval is embedded, major information seeking and information behavior models have been proposed by Bates (1989), Ellis (1989) (extended later by Meho and Tibbo (2003)), Kuhlthau (1993), Byström and Hansen (2005), and, as part of a cognitive information seeking and retrieval theory, by Ingwersen (1996). For example, the berrypicking user model proposed by Bates (1989) describes searching as iterative moving, by alternations of the search query, through an information space which dynamically evolves as the user encounters new information. Further user models are referred to in the context of exploratory search further below.

A seminal (2) cognitive user model is Belkin's Anomalous States of Knowledge (ASK) model (Belkin, 1980). According to Belkin, users are engaged with making sense of the world using their current state of knowledge. Whenever a situation cannot be resolved with the current state of knowledge, users enter an *anomalous state of knowledge*, which triggers the creation of an information need the user wants to satisfy in order to escape from the anomalous state.

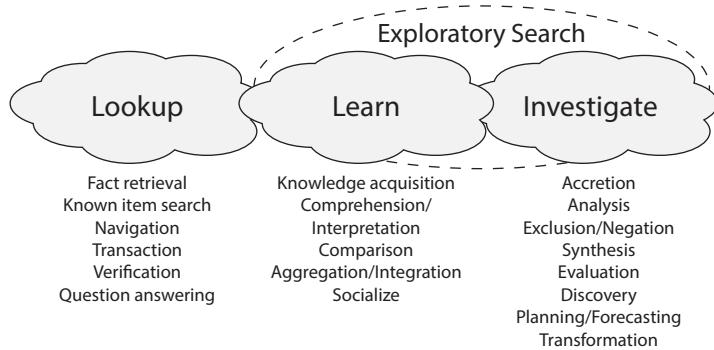
The taxonomy of web search by Broder (2002) can be interpreted as another user model that distinguishes (3) the different types of information needs. According to Broder, information needs can be classified into navigational, transactional, and informational needs. His model had been refined by Rose and Levinson (2004), taken up by Marchionini (2006), and further extended by Russell et al. (2009). The latter two explicated exploratory information needs as searches that are intended to discover something or to learn about a topic area.

Task Attribute	Task Level
<i>Learning and investigation as goal</i>	Work
<i>Target is multiple items</i>	Search
<i>Occurs over time</i>	Seeking
<i>Accompanied by sensemaking, decision making or other cognition</i>	Work
<i>Uncertainty is involved</i>	All
<i>Ill-structured problem</i>	Work
<i>"Not too easy"</i>	Seeking
<i>Multi-faceted</i>	Work
<i>General problem, not specific</i>	Work
<i>Dynamic, evolution during search</i>	Seeking
<i>Open-ended problem</i>	Seeking

TABLE 2.1: Exploratory search attributes synthesized by Wildemuth and Freund (2012) on the basis of an analysis of 51 exploratory search research papers. Each attribute is assigned to one of the three task levels proposed by Byström and Hansen (2005). The attribute “uncertainty” pertains to all three levels.

Though aspects of exploratory information needs are studied for a long time under labels such as subject search, general search tasks, decision tasks, or open-ended tasks (Diriye et al., 2010), exploratory search became an explicit subject of IR research around 2005, with notable events having been the Exploratory Search Interfaces (XSI) workshop in 2005, the ACM SIGIR workshop on Evaluating Exploratory Search Systems in 2006, as well as the seminal book publication “Exploratory Search: Beyond the query-response paradigm” by White and Roth in 2009. The driving force of these initiatives was the insight, that the query response paradigm is not capable of effectively supporting search tasks with specific “exploratory” attributes. In 2012, Wildemuth and Freund (2012) inductively analyzed 51 exploratory search research papers for the key attributes used therein to describe, specify, or design search tasks which are not well supported by the query response paradigm. What their set of synthesized attributes, which is outlined in the first column of Table 2.1, shows is that, although the list is diverse, the attributes coincide well with situations that frequently arise in DH research. Hence, having a decent understanding of research done in the context of exploratory search is important for IR4DH research as well. In the following paragraphs, the exploratory search attributes in Table 2.1 are presented in more detail.

*Learning and investigation as goal*, which is the first attribute in Table 2.1, has been proposed by Marchionini (2006) as defining characteristic of exploratory search. In his well-received user model, Marchionini distinguishes three kinds of search goals: lookup, learning, and investigation,



**FIGURE 2.4: Search activities according to Marchionini (2006). Especially learning and investigation are pertinent to exploratory search.**

and highlights that the latter two are especially pertinent to exploratory search (see Figure 2.4). Users with lookup goals strive for discrete, well-structured information objects that are retrievable with a single carefully specified query and require minimal result set examination. Lookup goals are therefore well supported by query-response IR systems. In contrast, users with learning or investigation goals strive for sets of documents (*target is multiple items*) that are retrievable only through multiple query-response iterations occasionally occurring over long time periods. The opinion that exploratory information needs can be distinguished from lookup needs by the number of anticipated documents and the time or query-response turns allotted for retrieving them is advocated also by other researchers, including Kules and Capra (2008), Qu and Furnas (2008), and Golovchinsky et al. (2012).

For successful learning, Marchionini continues, cognitive processing and interpretation of the documents, including activities such as comparison and making qualitative judgments, are required. For successful investigation, users have to take on a skeptical attitude towards the retrieved documents and critical assessment by the means of analysis, synthesis, and evaluation is required to accomplish investigative goals. Especially Vakkari (2010), but also Diriye et al. (2010) and White and Roth (2009) support the opinion that cognitive demanding user activities such as *sensemaking* or *decision making* are inherent to exploratory search, and that the answer to an exploratory information need is created, not found.

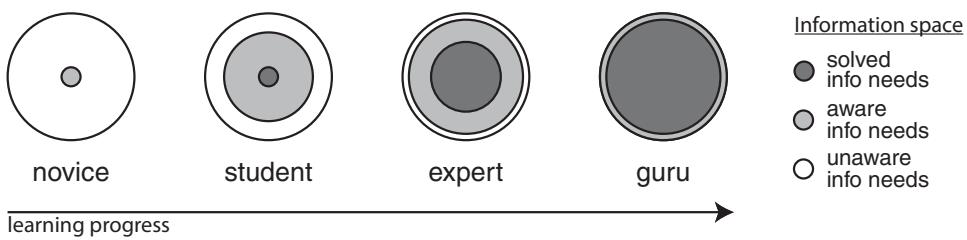
A further attribute that has been used frequently, albeit in different senses, to define exploratory search is *uncertainty*, which has been attributed to the user's information need, the search procedure to apply, the topic of interest, as well as to the amount of available relevant information. Con-

cerning uncertainty as a characteristic of the user's information need, White and Roth (2009) pick up the notion of ill-defined or *ill-structured* search goals from the information seeking literature. According to White and Roth, exploratory information needs correspond to ill-defined search goals that arise as a product of users' diversive curiosity. Opposed to specific curiosity, which is users' well-structured desire for a particular piece of information similar to Marchionini's definition of lookup goals, diversive curiosity represents the more general seeking of stimulation or novelty. In line with this view, Aula and Russell (2008) state that "a search is deemed exploratory when the searcher has a very abstract search goal". Because of the low a priori determinability of the task's requirements (cf. Byström and Hansen, 2005), ill-structured problems are challenging to solve with query-response IR systems.

Concerning uncertainty with respect to the search procedure to apply, Aula and Russell (2008) and Diriye et al. (2010) propose that a high procedural complexity of a search task, which means a large number of involved subtasks and steps, is a defining element of exploratory search. Also Kules and Capra (2008) state that search tasks must be *not too easy*, i.e. procedural complex and uncertain, to qualify for an exploratory search task. Procedural complexity is closely related to the notion of *multi-faceted* search tasks, which refer to information needs including multiple aspects or a number of concepts (Kintsch, 1998). Especially White and Roth emphasize the multi-faceted nature of a search task as being a key attribute for exploratory search.

Uncertainty with respect to the topic of interest is typical for early stages of *general* (in the sense of conceptually broad) learning and investigation tasks as defined by Marchionini (2006). As illustrated in Figure 2.5, for a novice to a topic domain, its specific concepts and terminology are unknown, preventing effective query formulation for query-response IR systems. As White and Roth (2009) state, users need to learn about the topic in order to understand how to achieve their goal, and therefore engage in exploratory searches. This view is also reflected in Kuhlthau (1993)'s user model of longitudinal searches. Kuhlthau divides the search process of users into six stages, with early stages being characterized by high uncertainty that is reduced by the means of exploration. Figure 2.5 also illustrates that as learning progresses, information needs evolve and change. This *dynamic* evolution of the problem context has been pointed out as being characteristic for exploratory search scenarios by Kules and Capra (2008), White and Roth (2009), and Ruotsalo et al. (2013) among others.

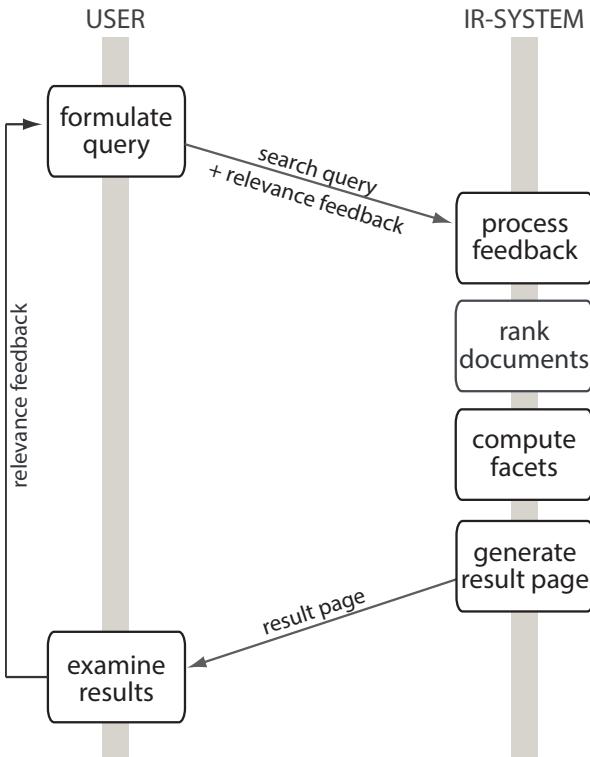
Finally, uncertainty with respect to the amount of available relevant in-



**FIGURE 2.5:** Typical stages of learning and investigation tasks as defined by Marchionini (2006). As learning progresses, the perceived information needs of users evolve. Especially as a novice, unfamiliarity with the domain specific concepts prevents an effective use of query-response IR systems.

formation has been proposed to elicit exploratory search. Especially for investigation tasks such as negative searches (Garfield, 1970), where users seek out to prove the absence of information, or exception searches (Wilson et al., 2010), aiming at finding outliers, being certain that all relevant documents have been retrieved is critical. To continuously increase retrieval recall, users engage in an *open-ended* exploration process (White and Roth, 2009). In this respect, Marchionini (2006) notes that the focus on retrieval recall is not well supported by query-response IR systems, which are typically highly tuned towards precision in the first result page.

To summarize, the exploratory search attributes in Table 2.1 have been assigned to the different task levels proposed by Byström and Hansen (2005). Byström and Hansen argue that every search with an IR system is performed as an attempt to solve (parts of) an information need, which in turn is part of a plan for solving a higher level, information intensive work task. Any attribute that characterizes an overall work task therefore resides on the work task level, attributes describing properties of information needs on the information seeking level, and attributes describing individual searches on the information search or retrieval level. Taken together, on the working level, exploratory search tasks have been defined in the literature as general, multi-faceted learning or investigation tasks which require sense making, decision making, or other cognitive demanding activities in order to be solved. DH research projects are usually prime examples for this kind of task. On the seeking level, exploratory information needs have been defined as being procedurally complex and dynamically evolving. To satisfy them, an open-ended, long-term effort is required. Ultimately, on the search level, exploratory search queries have been proposed to target at retrieving multiple relevant documents.



**FIGURE 2.6:** Illustration of iterative user interaction models. The query contains information from past searches to improve document ranking. The result list is analyzed to provide faceted search functionality.

## 2.5 EXTENSIONS TO THE QRP

In contrast to the heterogeneity of attributes and task levels chosen to define exploratory search, all reviewed research concludes that exploratory search tasks are supported superficially at most by the query response paradigm, and that more sophisticated models are needed. The major limitation of the query response paradigm is that it does not provide for situations where users could not satisfy their information need from examining the result page. To address this issue, the query response paradigm has been extended by iterative user interaction models. Taking up the perspective of the berrypicking user model, iterative user interaction models understand searching with an IR system as an iterative dialog between the user and the IR system. E.g., in the interactive information retrieval model (IIR) (Fuhr, 2008), result pages are regarded as containing binary choices that the IR system provides (like in a menu), and decisions the user makes (e.g. visiting a document, clicking a query recommendation) serve as clues to determine the optimal subsequent result page. This perspective is taken also in the

dynamic information retrieval model. According to Yang et al. (2016), dynamic information retrieval is even a progression from IIR in that it “deals with the complexity of user interaction by operating over multiple states”. The states may represent multiple queries in a search session, multiple sessions in a user’s search history, different users in a search log, and so on.

The protocol of iterative user interaction models is depicted, in forms of a sequence diagram, in Figure 2.6. In contrast to the query response paradigm, searches are not independent from each other. Through adding relevance feedback information from the examination activity to the search query, past searches influence the result pages of follow up searches. Relevance feedback is called implicit, if the IR system tries to infer the relevance of a document from user interactions with the result page (e.g. the user clicking on a result). Relevance feedback is called explicit, if the user interface provides functionality (e.g. a “more like this” button) to mark a document as being relevant. To extend the query representation  $\mathbf{q}$  with information about document relevance  $D_r$  and irrelevance  $D_{nr}$ , the Rocchio algorithm (with  $\alpha$ ,  $\beta$  and  $\gamma$  as weights) is a classic approach (Rocchio, 1971).

$$\mathbf{q}' = \alpha \mathbf{q} + \frac{\beta}{|D_r|} \sum_{d \in D_r} \mathbf{d} - \frac{\gamma}{|D_{nr}|} \sum_{d \in D_{nr}} \mathbf{d}$$

In practice, irrelevance information is hardly beneficial and hence often neglected. A downside of the Rocchio algorithm is that the query representation grows substantially in terms of non-zero vector entries, as all terms that appear in  $D_r$  (and  $D_{nr}$ ) are considered. On top of that, a large fraction of the added terms will not contribute to a better query, and in the worst case, lead to topic drift. As a solution to these problems, in Chapter 3, the keyquery paradigm is introduced, which suggests to compute a set of relevant search queries as an alternative sparse representation of  $D_r$ , which then can be used for query expansion.

A further implication of iterative user interaction models is that obtaining feedback information from the user becomes, apart from serving the most relevant results, a second objective for the design of the search result page. Whereas for relevance feedback, a highly diversified result page covering different interpretations of the query is desirable, applying the probability ranking principle is optimal for result page relevance. Iterative user interaction models aim at balancing these two incentives. In his probability ranking principle for interactive information retrieval, Fuhr (2008) models the expected benefit of a result list and suggests to maximize this quantity instead of relevance. More recently, Li et al. (2016) model the balanc-

ing problem as a multi-armed bandit problem, where optimizing for relevance feedback represents the player's exploration option, and optimizing for maximum relevance represents the exploitation option.

In addition to information about the relevance of individual documents, a further possibility to guide the user in the interactive retrieval process is by the provision of facets (Tunkelang, 2009). Facets are sets of meta-data values which the user can select to refine the current query. Faceted search excels most when the documents retrieved by a query already feature a sophisticated set of structured metadata that can be presented as facets, such as in e-commerce (e.g. size, price, brand) or academic/library search (e.g. author, journal, publication type and year) settings. For Web documents, metadata facets such as content type, content length, or host domain can be generally provided, but the relevance of these facets is commonly very limited. To provide more relevant facets, information extraction, search result clustering, or text classification can be applied. E.g., named entity recognition can be applied to populate facets such as persons, places, or organizations. Genre classification, readability scores, and sentiment analysis constitute further possibilities to extract facets from un- or semi-structured documents such as Web documents. To provide facets that are specific to the user's information need, the discovery of relevant facets (and their values) from search results has been proposed. To this end, Kong and Allan (2014) and also Friedrich et al. (2015) search for lexical and HTML patterns in the top search results. Also, query recommendations derived from the query logs of a search engine can be presented as a facet. Interactive retrieval with facets is subject of debate again in Chapter 8.

# 3

## From Keywords to Keyqueries: Content Descriptors for the Web

This chapter introduces the core concept of the thesis, which is applied by many of the algorithmic contributions introduced in later chapters. The concept is called *keyquery*, and serves as dynamic content descriptor for documents. Keyqueries are defined implicitly by the index and the retrieval model of a reference search engine: keyqueries for a document are the minimal queries that return the document in the top result ranks. Besides applications in the fields of information retrieval and data mining, keyqueries have the potential to form the basis of a dynamic classification system for future digital libraries—the modern version of keywords for content description.

To determine the keyqueries for a document, an exhaustive search algorithm is presented along with effective pruning strategies. For applications where a small number of diverse keyqueries is sufficient, two tailored search strategies are proposed. The conducted experiments emphasize the role of the reference search engine and show the potential of keyqueries as innovative document descriptors for large, fast evolving bodies of digital content such as the Web.

### 3.1 INTRODUCTION

A content descriptor is a word or a short phrase that expresses the central topical aspect or the domain of a document. Typical examples can be found in the ACM’s classification system: categories like “Information Search and Retrieval”, general terms like “Algorithms” or “Experimentation”, as well

as keywords like “keyquery, content description, query formulation” place a document into the classification system.

With the keyquery paradigm, we propose an additional modern means of document content description: search queries. The underlying idea is that those queries that return a given document in their top ranks for some reference search engine “describe” the document’s content well enough to stand out from the indexed collection. If the query is maximally general (i.e., no subset of the query has the same property), we call it a *keyquery* for the document. The validity check of a query’s suitability as a document’s content descriptor is straightforward: submission to the reference search engine.

In information retrieval systems, keyqueries can serve multiple purposes. For example, as query recommendations to find related resources for specified documents (see Chapter 4), or to implement query expansion and relevance feedback techniques (see Chapter 5). For data mining tasks such as clustering, a “bag of queries” document representation can be based on the keyqueries of a corpus, and the relevance scores of the search engine can be used as the feature weights. As the concept of search queries nowadays is well understood by web users, keyqueries might even be used as cluster labels—especially when clustering search results.

In the context of digital libraries, keyqueries constitute an interesting alternative to *keywords*, which typically are manually chosen free form content descriptors. The advantage of keyqueries is that their descriptiveness can be automatically tested using the library’s search engine. Whenever a candidate is not descriptive enough, additional terms can be suggested or automatically added. Carried out consequently, the keyquery approach includes validity checks for the whole library on every new insertion of a document, and automatic re-establishment when necessary. This way, keyqueries can form the basis of a dynamic and automatically maintained classification system for digital libraries (see Chapter 7). Especially for fast evolving large-scale bodies of digital content, where manually maintained classification systems need enormous manpower to stay up to date, dynamic classification with keyqueries is an effective alternative.

In the following section, we review literature related to the concept of keyqueries, and give pointers to the state of the art in various potential applications. Section 3.3 provides algorithms for finding the keyqueries of a document, and explores options for pruning the search space. In Section 3.4, we report on our experiments with keyqueries of scientific papers using different reference search engines. Our findings reveal that the concept of keyqueries is sound and effectively applicable.

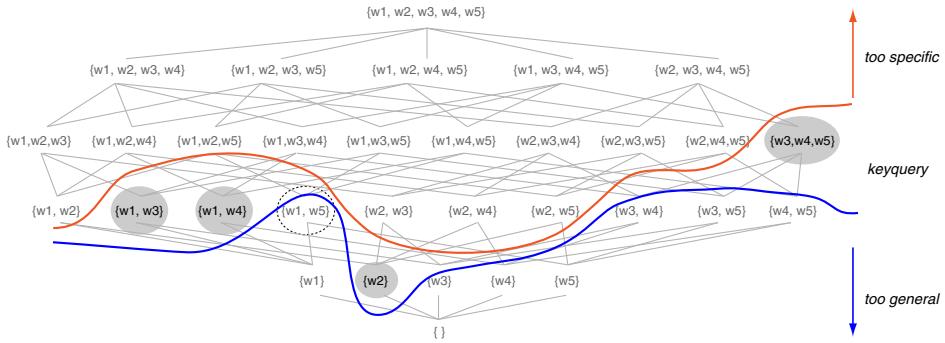


FIGURE 3.1: The search space  $\mathcal{Q}$  for a five-word vocabulary, divided into the three subspaces *too generic*, *keyqueries*  $\mathcal{Q}^*$ , and *too specific*.

### 3.2 RELATED WORK

The state-of-the-art technique for the automated generation of content descriptors is keyword or keyphrase extraction. Actually, we chose the term *keyquery* to relate to these two concepts. In particular, as will be discussed in detail in Section 3.3, we use keyword extraction in a subroutine to efficiently find a small subset of diverse keyqueries. For the respective experiments presented in Section 3.4, we employ the TextRank algorithm (Mihalcea and Tarau, 2004). TextRank is an unsupervised keyword and keyphrase extraction technique that represents a document as a graph and determines a keyword ranking by applying the PageRank algorithm.

A dynamic classification system based on keyqueries requires an efficient means to store and update the keyqueries. An adequate data structure has been proposed by Pickens et al. (2010): the reverted index. As its name suggests, the reverted index is related to the inverted index used in search engines, but it stores queries in document postlists—instead of documents in term postlists. In their experiments on query recommendation and relevance feedback, Pickens et al. populate the reverted index with unigram queries and suggest the usage of  $n$ -gram queries or queries from query logs for further performance improvements. Keyqueries now contribute a third sophisticated alternative with a sound theoretical justification.

Once keyqueries are stored in a reverted index, the next step towards a dynamic classification system is to derive a hierarchical structuring of the queries. Bonchi et al. introduce this task as topical query decomposition (Bonchi et al., 2008), since the queries of a common class should represent “coherent, conceptually well-separated topics.” In their work, Bonchi et al. achieve good results by using a set cover algorithm with red-blue metric for the problem. Our approach to compose dynamic classification systems

with keyqueries is presented in Chapter 7.

Inspiring related work exists also in the field of data mining: In our optimum clustering framework (OCF) (Fuhr et al., 2012), we suggest to represent documents as “bags of queries”, and to use the relevance scores or retrieval ranks for each query as the feature weights. Keyqueries fit very naturally into the proposed scheme. That they might be even more appropriate than arbitrary queries in an OCF-style clustering approach can be motivated from a finding of Azzopardi and Vinay (2008): in their analysis of document retrievability with search engines, they observed that simple bag-of-words representations with  $tf \cdot idf$ -based weights are biased towards a small number of documents. These documents appear in the top results for significantly more of the basic index terms than other documents. In the context of the optimum clustering framework, a set of arbitrary queries would thus come with a bias, obviously harming the overall effectiveness. Instead, a comparably unbiased feature set can be formed by including the same number of keyqueries from each document. The documents’ retrievability is then approximately equal, and the feature weight distribution unbiased.

### 3.3 APPROACH

In this section, we formalize the keyquery concept and present an exhaustive search algorithm to find all keyqueries for a given document. We also explore possibilities to reduce the search space and present two heuristic search strategies that are tailored to scenarios where a few diverse keyqueries suffice.

#### 3.3.1 Keyqueries

Given the vocabulary  $W_d = \{w_1, w_2, \dots, w_n\}$  of a document  $d$ , let  $\mathcal{Q}_d$  denote the family of search queries that can be formulated from  $W_d$  without word repetitions; i.e.,  $\mathcal{Q}_d$  is the power set of  $W_d$ ,  $\mathcal{Q}_d = 2^{W_d}$ . Note that no distinction is made with respect to the ordering of the words in a query. If it is clear from the context, we omit the subscripts and just use  $W$  and  $\mathcal{Q}$  to denote the vocabulary and the potential queries from  $d$ .

A query  $q \in \mathcal{Q}$  is a *keyquery* for  $d$  with respect to a reference search engine  $S$  iff: (1)  $d$  is among the top- $k$  results returned by  $S$  on  $q$ , and (2) no subset  $q' \subset q$  returns  $d$  in its top- $k$  results when submitted to  $S$ . The parameter  $k$  controls the level of keyquery generality and is usually set to some small integer. In our experiments, we set  $k$  equal to 10.

Figure 3.1 shows a visualization of  $\mathcal{Q}$ , which is adapted from work done by Hagen and Stein (2010) on query formulation. The keyquery search space  $\mathcal{Q}$  divides into three subspaces. The subspace of too-general-queries that do not retrieve the desired document in the top- $k$  results, the subspace  $\mathcal{Q}^*$  of keyqueries, and the subspace of too-specific-queries all of which are supersets of a shorter keyquery. An interesting query in the example is  $\{w_1, w_5\}$  (third row): it is still too general but cannot be extended without becoming too specific.

### 3.3.2 Exhaustive Search

Note that an exhaustive search in  $\mathcal{Q}$  is required in order to find all keyqueries for a given document. Previous work in the field of query formulation used an adapted version of the Apriori algorithm (Agrawal and Srikant, 1994) to identify queries not returning too many results (Hagen and Stein, 2011). The Apriori algorithm stems from the field of frequent itemset mining and is considered one of the top ten data mining algorithms (Wu et al., 2007). It is more efficient than enumerating  $\mathcal{Q}$  in an arbitrary level-wise order. As frequent itemset mining is very similar to our keyquery setting we also employ a tailored Apriori variant.

The algorithm of Apriori for keyquery identification is given in Pseudocode 1. The algorithm starts with submitting all one-word queries to the reference search engine (line 1), and evaluates whether the queries are either keyqueries or too general (lines 2–3). Note that it is assumed that the search engine always retrieves the document at some rank in the result list since all query words are present in the document. If the query combining all the candidate terms in  $C_1$  is still too general (lines 4–5), all shorter queries have to be too general as well, and the algorithm can immediately terminate returning  $\mathcal{Q}^*$ . The main loop of the algorithm (lines 7–15) iterates through the search space in a level-wise manner, i.e., all two-word keyquery candidates first, then the three-word keyquery candidates, etc. During this process, the search space is pruned whenever possible: all queries which are supersets of identified keyqueries can be omitted. The algorithm returns  $\mathcal{Q}^*$  if no new candidate query  $q_{\text{cand}}$  can be formulated.

### 3.3.3 Search Space Reduction

Since the search space  $\mathcal{Q}$  grows exponentially in the size of the vocabulary, an effective strategy to prune  $\mathcal{Q}$  is to reduce the vocabulary ahead of a search. We call strategies that reduce the vocabulary *vertical pruning strategies*, whereas *horizontal pruning strategies* constrain the maximum length of

---

**Pseudocode 1** The Apriori algorithm for keyquery search
 

---

**Input:** document  $d$  with vocabulary  $W$   
**Output:** the family  $\mathcal{Q}^*$  of keyqueries

```

1: for all  $w \in W$  do submit( $w$ )
2:  $\mathcal{Q}^* \leftarrow \{w : w \in W \text{ and } w \text{ is keyquery}\}$ 
3:  $C_1 \leftarrow \{w : w \in W \text{ and } w \text{ too general}\}$ 
4: submit( $\bigcup_{w \in C_1} w$ )
5: if  $\bigcup_{w \in C_1} w$  too general then stop and output  $\mathcal{Q}^*$ 

6:  $i \leftarrow 1$ 
7: while  $C_i \neq \emptyset$  do
8:   for all  $q, q' \in C_i$  do
9:     if  $|q \cap q'| = i - 1$  then  $q_{\text{cand}} \leftarrow q \cup q'$ 
10:    if  $(q_{\text{cand}} \setminus w) \in C_i$  for all  $w \in q_{\text{cand}}$  then
11:      if  $q_{\text{cand}} \notin C_{i+1}$  and  $q_{\text{cand}} \notin \mathcal{Q}^*$  then
12:        submit( $q_{\text{cand}}$ )
13:        if  $q_{\text{cand}}$  too general then  $C_{i+1} \leftarrow C_{i+1} \cup q_{\text{cand}}$ 
14:        if  $q_{\text{cand}}$  is keyquery then  $\mathcal{Q}^* \leftarrow \mathcal{Q}^* \cup q_{\text{cand}}$ 
15:     $i \leftarrow i + 1$ 
16: output  $\mathcal{Q}^*$ 
```

---

the keyqueries to be found. In the following, we explore pruning strategies of both kinds.

To understand the basic characteristics of contemporary content descriptors, we analyze the 2012 version of the ACM Computing Classification System.<sup>1</sup> Our review reveals that only three parts-of-speech types are typically used for content descriptors: nouns and noun combinations (e.g., *web search*), adjectives (e.g., *personalized search*), and conjunctions (e.g., retrieval models *and* ranking). An appropriate vertical pruning strategy can thus ignore all words that do not belong to any of these three classes. Furthermore, adjectives are only considered when they appear in combination with a noun. For the documents used in our experiments (cf. Section 3.4), this vertical pruning results in a search space reduction to about  $\sqrt{2^n}$ , where  $n$  is the original vocabulary size.

We also infer a horizontal pruning strategy from the length of the ACM content descriptors. The average descriptor length is about three words, and up to eight words in extreme cases. Hence, we suggest to not consider longer queries.

### 3.3.4 Heuristic Search Strategies

For certain applications such as query recommendation, it is often not necessary to compute all keyqueries of a document; two or three keyqueries

---

<sup>1</sup><http://dl.acm.org/ccs.cfm>

with no or only a small term overlap could suffice. In such scenarios, heuristic search strategies can simply proceed in a depth-first manner. For this purpose, the TextRank algorithm (Mihalcea and Tarau, 2004) provides an interesting basis. TextRank ranks each word of a document by applying the PageRank procedure to a document’s graph model: nodes represent words and edges connect words that occur next to each other in the text. From the TextRank scores for words and the graph representation, we derive two heuristic search strategies. Note that the vertical pruning described above (vocabulary reduction) is always applied.

*Rank-Driven Search.* An initially empty query is successively expanded by that word from  $\{w : w \in W, w \notin q\}$  with the highest score until the document  $d$  is returned among the top- $k$  results. As the derived query  $q$  might be too specific, it may need to be minimized. A straightforward solution is to successively remove each word from  $q$  and to test whether  $d$  is still returned in the top- $k$  (if not, keep the word and try the next one). The reduced  $q$  then is guaranteed to be a keyquery. To find another keyquery with different terms, start again with  $W = W \setminus \{q^*\}$  until no keyquery can be found or sufficiently many have been returned.

*Graph-Driven Search.* A second heuristic search strategy can be based on the TextRank graph. The first term added to the initially empty query  $q$  again is the highest scoring word  $w \in W$ . But then  $q$  is extended by adding from all words *adjacent* to  $w$  in the TextRank graph the word  $w^*$  with the highest score. If no adjacent word exists, the word with the highest score from  $\{w : w \in W, w \notin q\}$  is used, similar to rank-driven search. The extension of  $w^*$  works analogously until a search with  $q$  has the document  $d$  among the top- $k$  results. Again,  $q$  is minimized as above by trying to remove keywords, and the process restarts with  $W = W \setminus \{q^*\}$  until either enough keyqueries have been found or no more are possible.

Graph-driven search obviously favors phrases in the keyquery generation and should work especially well for search engines that include proximity features and allow phrasal search. However, graph-driven search is also applicable with basic engines, then probably losing much of its potential.

### 3.4 EVALUATION

Our empirical evaluation is conducted in the context of scientific paper collections and focuses on the following two questions.

1. How many queries have to be submitted to determine one keyquery

**TABLE 3.1:** Keyquery statistics for two document sets and three reference search engines.

Search engine	Dataset	Queries	Success ratio	Keyquery length	Retrieval rank	Results
Google	SIGrand	18.8	60%	7.3	3.1	3.2 M
	SIGcited	10.6	54%	5.7	4.2	3.5 M
Lucene (no boost)	SIGrand	4.4	100%	3.2	4.1	89
	SIGcited	14.4	100%	3.6	4.9	161
Lucene (citations)	SIGrand	5.5	100%	3.4	4.0	88
	SIGcited	3.4	100%	2.8	3.5	234

for a paper against a contemporary search engine?

2. Does citation count influence the number of submitted queries or the keyqueries' lengths?

The first query addresses the efficiency or “costs” of keyquery computation in general (typically query submissions require non-negligible amounts of time), while the second query aims at evaluating a potential factor influencing efficiency (Google often shows highly cited papers in the top ranks).

Our experimental study is conducted on all the papers published at the SIGIR and CIKM conferences in the years 1999–2012. To address our first question, we sample the document set SIGrandom containing 50 random papers published at SIGIR. To address our second question, we use the document set SIGcited of the 50 SIGIR papers having the highest citation numbers in the ACM Digital Library (DL). The three reference search engines are Google as representative of current web search and two variants of a local Lucene indexing the 3796 papers (phrasal search enabled): with and without a logarithmic citation boost on the actual relevance score (i.e., the score is multiplied with the  $\log_2$  of the paper’s citation count obtained from the ACM DL). The granularity parameter is set to  $k = 10$ . Since Google imposes usage restrictions, we constrain the number of queries that can be submitted per document to 128 for all the engines. This would allow exhaustive search with seven words but in the experiments presented here, we focus on the graph-driven heuristic and evaluate finding one keyquery per document. The results of our experiments are shown in Table 3.1. All values are averaged over the document sets.

Using Google, a keyquery is found for about 54–60% of the documents in the collections (success ratio). This does not mean that the documents cannot be found using Google, but that within our restricted budget of

128 query submissions no keyquery could be identified. As expected, the average number of required queries in case of success is lower for highly cited papers than for random ones (10.6 vs. 18.8). Keyqueries for highly cited papers are also shorter (5.7 vs. 7.3). The documents themselves are returned higher in the ranking for random papers which is not that surprising as the longer keyqueries are more specific; the total result list lengths are of comparable magnitude (3.5 million). Given the size of Google’s search index, the observed keyquery length is small; it is comparable to manually created content descriptors in the ACM DL. However, the success ratio of only 54–60% supports the finding of Azzopardi and Vinay (2008) that retrievability is often an issue.

The low success ratio using Google cannot be further explored due to the black-box characteristic of the search process. It is thus interesting to compare the findings with the two Lucene instances that we can control completely. As expected, the small size of the index yields a perfect success ratio with Lucene and also enables more efficient construction (less query submissions) and shorter keyqueries. Not surprisingly, the number of results of a keyquery is much lower for Lucene than for Google.

Comparing the two document sets on the two Lucene instances yields an interesting observation. Without citation boost, it is much more difficult to find keyqueries for the highly cited papers than for a random one (14.42 vs. 4.44 queries). A possible explanation is that the highly cited papers are part of large research branches with many papers on similar topics—the highly cited papers probably being the most influential ones. Without boosting highly cited papers, it is much more difficult to retrieve them from the rest. Random papers on the other hand often do not have that many other papers on similar topics such that keyqueries are easier to find. The average keyquery result list length also supports this explanation as the result lists are longer for SIGcited keyqueries (161.34 vs. 88.56). When the citation boost is included, the SIGcited papers are much more easy to find (11 queries less than without boost) and ranked higher in even longer keyquery result lists. The random papers instead require a little more effort than before but their keyquery characteristics remain comparable.

### 3.5 SUMMARY

With the concept of keyqueries we introduce a dynamic means to address the fast evolving bodies of digital content in our society. The range of potential applications in information retrieval, data mining, or digital library tasks underline the innovation and relevance of our idea: relying on the

acceptance, the agreed semantics, and the approved indexing of keyword-based search engines, the state-of-the-art search technologies become a viable replacement for manually constructed content descriptors. Our experiments show that keyqueries can be effectively constructed for the majority of the analyzed documents when using a graph-driven search, and that the role of the reference search engine as context provider can be exploited to favor content with specific properties.

As for future work, it would be very interesting to study basic characteristics of keyqueries for larger document corpora and to further investigate the retrievability issues we observed for our restricted budget with Google. In the digital library setting, also the actual acceptance of keyqueries with human users could be evaluated. Other interesting research directions are the potential applications of keyqueries as suggestions to find related documents or as cluster labels in search scenarios. These applications will be presented in subsequent chapters.

Last but not least, note that the shortest possible keyquery for this chapter with respect to the SIGIR-CIKM test corpus used in our experiments is very simple: keyquery.

# 4

## Supporting Scholarly Search with Keyqueries

One of the retrieval tasks which arise during research question development is related work search, i.e., identifying relevant papers about a given topic. In this chapter, we focus on the scenario where a researcher can come up with a few relevant papers (e.g. from a lookup search or suggested by a colleague), and then wants to find further related publications. Our proposed approach to the problem is based on the concept of keyqueries: formulating keyqueries from the input papers and suggesting other papers that can be found with these queries.

We compare our approach to three baselines that also represent the different ways of how humans search for related work: (1) a citation-graph-based approach focusing on cited and citing papers, (2) a method formulating queries from the paper abstracts, and (3) the “related articles”-functionality of Google Scholar. The effectiveness is measured in a Cranfield-style user study on a corpus of 200,000 papers. The results indicate that our novel keyquery-based approach is on a par with the strong citation and Google Scholar baselines but with substantially different results—a combination of the different approaches yields the best results.

### 4.1 INTRODUCTION

We tackle the problem of automatically supporting a scholar’s search for related work. Given a research task, the term “related work” refers to papers on similar topics. Scholars collect and analyze related work in order to get a better understanding of their research problem and already exist-

ing approaches; a survey of the strengths and weaknesses of related work forms the basis for placing new ideas into context. In this chapter, we show how the concept of keyqueries, introduced in Chapter 3, can be employed to support search for related work.

Search engines like Google Scholar, Semantic Scholar, or CiteSeerX provide a keyword-based access to their paper collections. However, since researchers usually have limited knowledge when they start to investigate a new topic (cf. Figure 2.5), it is difficult to find all the related papers with self-formulated queries against such interfaces. Such queries help to identify a few initial papers, but to find further papers, researchers usually bootstrap their search from information in these initial papers.

Every paper provides two types of information useful for finding related work: content and metadata. Content (title, abstract, body text) is a good resource for query terms. Metadata (bibliographic records, references) can be used to follow links to referenced papers. Recursively exploring the literature via queries or citations to and from some initial papers is common practice, although rather time-consuming. Support is provided by methods that automate the above procedure: graph-based methods exploit the citation network, and content-based methods can generate queries.

This chapter addresses the research questions of whether keyqueries are useful for identifying related work, and whether they complement other standard approaches. Assuming that the top results returned by a document’s keyqueries cover similar topics, the concept of keyqueries seems promising for identifying related papers. Moreover, it is conceivable that keyqueries can identify papers that graph-based methods miss.

The contributions with respect to our research question are threefold: (1) We develop a keyquery-based method identifying related work. (2) For the evaluation, we implement three strong baselines representing standard approaches: the graph-based Sofia Search by Golshan et al. (2012), the query-based method by Nascimento et al. (2011), and the “related articles”-feature of Google Scholar. (3) We conduct a Cranfield-style user study to compare the different approaches.

## 4.2 RELATED WORK

Methods for identifying related work can be divided into citation-graph-based and content-based approaches. Only few of the content-based methods use queries, such that we also investigate query formulation techniques for similar tasks.

### 4.2.1 Identifying Related Research Papers

Many variants of related work search are known: literature search, citation recommendation, research paper recommendation, etc. Some try to find references for a written text, others predict further “necessary” references given a subset of a paper’s references. In our setting, the task is to find related papers according to a given set of papers. This represents the everyday use case of enlarging an initial related work research.

#### RELATED WORK SEARCH

Given: An input list  $\mathcal{I} = \langle d_1, d_2, \dots, d_n \rangle$  of papers.

Task: Find an output list  $\mathcal{O} = \langle d'_1, d'_2, \dots, d'_m \rangle$  of related papers.

Given the initial knowledge of a scholar specified as the list  $\mathcal{I}$  of input papers (could also be just one), most approaches to RELATED WORK SEARCH retrieve *candidate papers* in a first step. In a second step, the candidates are ranked to generate the final output list  $\mathcal{O}$ . More than 80 approaches are known for the problem of identifying related papers (Beel et al., 2013). We concentrate on the recent and better performing approaches and classify them by the employed candidate retrieval method: *content* and/or *citations* can be used.

**Citation-Graph-Based Methods** The network of citations forms the *citation graph*. If  $d$  cites  $d'$ , we call  $d$  a *citing paper* of  $d'$ , and  $d'$  a *cited paper* of  $d$ . Several approaches apply collaborative filtering (CF) using the adjacency matrix of the citation graph as the “rating” matrix (Sugiyama and Kan, 2013; Caragea et al., 2013). A limitation of CF is that it has problems for poorly connected papers, known as the “cold-start-problem” (Wang and Blei, 2011). Ekstrand et al. (2010) thus explore the additional application of link ranking algorithms like PageRank, that can also be applied stand-alone (Küçüktunç et al., 2013). Since the citation graph tends to be noisy and sparse (Caragea et al., 2013), by design, graph-based approaches favor frequently cited papers (Wang and Blei, 2011). Methods that only use the citation graph easily miss papers that are rarely cited (e.g., very recent ones).

As a graph-based baseline, we select Sofia Search (Golshan et al., 2012). It very closely mimics the way how humans would identify candidates from the citation graph. Starting from an initial set of papers, the approach follows all links to cited and citing papers up to a given recursion depth or until a desired number of candidates is found. Note that this procedure

conforms with CF methods and link ranking to some extent. Papers citing similar papers are linked via the cited papers, such that the CF candidates are included. Setting the recursion depth large enough, also all interesting results from PageRank random walk paths will be found—except the low-probability “clicks” on some random non-linked papers. Thus, Sofia Search forms a good representative of graph-based approaches.

**Content-Based Techniques** Content-based approaches utilize the paper content to find related work. Translation models are used to compute the probability of citing a paper based on (1) citation contexts (Huang et al., 2012), (2) the content of potential references (Lu et al., 2011), or (3) via an embedding model (Tang et al., 2014). Similar ideas are based on topic models: LDA was combined with PLSA to build a topic model from texts and citations (Nallapati et al., 2008). Later improvements use only citation contexts instead of full texts (Kataria et al., 2010; Tang and Zhang, 2009). Drawbacks of translation- and topic-model-based approaches are the long training phase and that re-training is necessary whenever papers dealing with new topics are added to the collection. Besides such efficiency aspects, topic and translation models do not resemble human behavior, and they cannot be used with the keyword query interfaces of existing scholarly search engines. We thus choose a query-based baseline to represent the content-based approaches.

While there are complete retrieval models for recommending papers for a given abstract using metadata and content-based features (Bethard and Jurafsky, 2010), we prefer a standard keyword query baseline since it can be used against any scholarly search interface. Nascimento et al. (2011) propose such a method: given a paper, ten word bigrams are extracted from the title and the abstract and submitted as separate queries. Note that queries containing only two words are very general and return a large number of results. Still, Nascimento et al.’s idea is close to human behavior and forms a good content-based baseline.

**Combined Approaches** Several methods combine citations and content. For instance, by querying with sentences from a given paper and then following references in the search results (He et al., 2010, 2011). However, submitting complete sentences leads to very specific queries returning very few results only; a drawback we will avoid in our query formulation. Other combined approaches use topic models for citation weighting (El-Arini and Guestrin, 2011), or for overcoming the cold-start-problem of papers without ratings in online reference management communities like CiteULike or

Mendeley (Wang and Blei, 2011). The CiteSight system (Livne et al., 2014) is supposed to recommend references while writing a manuscript using both graph- and content-based features to recommend papers the author cited in the past, or cited papers from references the author already added. Since our use case is different and considering only cited papers appears very restrictive, we do not employ this approach.

As a representative of the combined approaches, we use Google Scholar’s feature “related articles” to form an often used and very strong baseline. Even though the underlying algorithms are proprietary, it is very reasonable that content-based features (e.g., text similarity) and citation-based features (e.g., number of citations) are combined.

### 4.3 BASELINES AND APPROACH

After describing the three selected baselines in more detail, we introduce our novel keyquery-based approach and a straightforward interleaving scheme for combining the results of different methods.

#### 4.3.1 Baselines

The baselines are chosen from the literature to mimic the strategies scholars employ: formulating queries, following citations, and Google Scholar’s “related articles.”

As representative of the content-based strategies, we select the method by Nascimento et al. (2011). The approach submits as its ten queries the ten distinct consecutive bigrams of non-stopword terms from a paper’s title and abstract that have the highest normalized *tf*-weights. For each query, the top-50 results are stored. The combined candidate set is then ranked according to the *tf*-weighted cosine similarity to the input paper. Note that the candidate retrieval and ranking are designed for only one input paper. We adapt the process to our use case by applying the retrieval phase for every input paper individually, and then ranking the combined result sets of all input papers by the highest similarity to any of the input papers. The reason for taking the highest similarity and not the average is the better performance in pilot experiments. We also tried to combine the titles and abstracts to a single meta-paper in case of more than one input paper. However, taking the highest similarity for individual input papers achieved the best performance.

As a representative of the citation graph approaches, we select Sofia Search (Golshan et al., 2012). For each input paper  $d$ , both cited and

citing papers are added to a candidate set  $\mathcal{C}$ . This routine is iterated using the candidates as new starting points until either enough (or no more) candidates are found, or a specified recursion depth is reached (typically 2 or 3). Again, we rank the candidates by their highest similarity to any of the input papers.

Our third baseline is formed by Google Scholar’s “related articles” feature. For a given research paper, a link in the Google Scholar interface yields a list of about 100 related articles. We collect all these related articles for each input paper individually, treating the underlying retrieval model as a black box. Since Google Scholar already presents ranked results, we do not re-rank them. In case of more than one input paper, we use a simple interleaving strategy: first the first rank for the first input paper, then the first rank for the second input paper, then the first rank for the third input paper, etc., then the second rank for the first input paper, etc. In case that a ranked result is already contained in the merged list, it is not considered again.

### 4.3.2 New Keyquery-Based Approach

Our new approach combines the concepts of keyqueries and query covers (Hagen and Stein, 2011) by generalizing the original single-document notion for keyqueries to sets of documents. A query  $q$  is defined to be a *keyquery* for a document set  $D$  with respect to a reference search engine  $S$ , if it fulfills the following conditions: (1) every  $d \in D$  is in the top- $k$  results returned by  $S$  for  $q$ , (2)  $q$  has at least  $l$  results, and (3) no subset  $q' \subset q$  returns every  $d \in D$  in its top- $k$  results. The generality of a keyquery is defined by the parameters  $k$  and  $l$  (e.g., 10 or 50). We argue that the top-ranked results returned by a keyquery for  $D$  cover topics similar to the documents themselves—rendering keyqueries a promising concept for identifying related work.

The power set  $\mathcal{Q} = 2^W$  forms the range of queries that can be formulated from the extracted keyphrases  $W$  of a document set  $D$ . A query  $q$  is said to *cover* the terms it contains. The more terms  $w \in W$  are covered by  $q$ , and the more documents  $d \in D$  are among  $q$ ’s top results, the better the query describes the topics represented by  $D$ ’s vocabulary  $W$ . The covering property (Hagen and Stein, 2011) states that (1) in a proper set  $Q$  of queries for  $W$ , each term  $w \in W$  should be contained in at least one query  $q \in Q$  (i.e., the queries “cover”  $W$  in a set-theoretic sense) and (2)  $Q$  should be *simple* (i.e.,  $q_i \not\subseteq q_j$  for any  $q_i, q_j \in Q$  with  $i \neq j$ ), to avoid redundancy. The keyquery cover problem we tackle hence is:

**KEYQUERY COVER**

Given: (1) A vocabulary  $W$  extracted from a set  $D$  of documents.

(2) Levels  $k$  and  $l$  describing keyquery generality.

Task: Find a simple set  $Q \subseteq 2^W$  of queries that are keyquery for every  $d \in D$  with respect to  $k$  and  $l$  and that together cover  $W$ .

The parameters  $k$  and  $l$  are typically set to 10, 50, or 100 but it will not always be possible to find a covering set of queries that are keyqueries for all documents in  $D$ . In such a case, we strive for queries that are keyqueries for a  $|D| - 1$  subset of  $D$ .

**Solving KEYQUERY COVER** Our approach has four steps (see Pseudocode 2). First, all keywords  $w \in W$  are removed that return less than  $l$  results when submitted to the search engine  $S$  (lines 1–2); queries including such terms can not be keyqueries.

In a second step, the remaining terms  $w \in W$  are iterated and added to an intermediate candidate query  $q$  (lines 3–5). If  $q$  is a keyquery for all papers  $d \in D$ , it is added to the set  $Q$  of keyqueries and  $q$  is emptied (line 6). Then the next query is formed from the remaining terms, etc., until the last term  $w \in W$  has been processed.

After this first iteration, not all terms  $w \in W$  are necessarily covered by the set  $Q$  of keyqueries. In this case,  $q$  is not empty (line 7) and we again go through the terms  $w \in W$  (line 8) and consecutively add terms  $w$  to  $q$ —as long as no keyquery is found and simplicity of  $Q$  is not violated (line 9). According to the keyquery definition, keyqueries that are already in  $Q$  must not be contained in the candidate query  $q$ . We hence omit terms  $w \in W$  that would cause  $q$  to contain a keyquery  $q' \in Q$ .

After this iteration, we do not further deepen the search but output  $Q$ , although still not all terms  $w \in W$  may be covered. This heuristic serves efficiency reasons and our experiments show that often all possible keywords are covered.

**Solving RELATED WORK SEARCH** The pseudocode of our algorithm using keyquery covers for related work search is given as Pseudocode 3. Using the KEYQUERY COVER algorithm as a subroutine, the basic idea is to first try to find keyqueries for all given input papers, and to add the corresponding results to the set  $C$  of candidates. If there are too few candidates after this

---

**Pseudocode 2** Solving KEYQUERY COVER

---

**Input:** Sets  $D$  of documents and  $W$  of keywords, keyquery generality parameters  $k$  and  $l$

**Output:** Set  $Q$  of keyqueries covering  $W$

- 1: **for all**  $w \in W$  **do**
- 2:     **if**  $w$  returns less than  $l$  search results **then**  $W \leftarrow W \setminus w$
- 3:      $q \leftarrow \emptyset$
- 4:     **for all**  $w \in W$  **do**
- 5:          $q \leftarrow q \cup \{w\}$
- 6:         **if**  $q$  is keyquery for all  $d \in D$  **then**  $Q \leftarrow Q \cup \{q\}$ ,  $q \leftarrow \emptyset$
- 7:     **if**  $q \neq \emptyset$  **then**
- 8:         **for all**  $w \in W$  **do**
- 9:             **if**  $\exists q' \in Q : q' \subset q \cup \{w\}$  **then**  $q \leftarrow q \cup \{w\}$
- 10:             **if**  $q$  is keyquery for all  $d \in D$  **then**  $Q \leftarrow Q \cup \{q\}$ , **break**
- 11: **return**  $Q$

---

step, KEYQUERY COVER is solved for combinations with  $|\mathcal{I}| - 1$  input papers, then with  $|\mathcal{I}| - 2$  input papers, etc.

The vocabulary combination (line 3) is based on the top-20 keyphrases per paper extracted by KP-Miner (El-Beltagy and Rafea, 2009), the best unsupervised keyphrase extractor for research papers in SemEval 2010 (Kim et al., 2010). The terms (keyphrases) in the combined vocabulary list  $W$  are ranked by the following strategy: First, all terms that appear in all papers ranked according to their mean rank in the different lists. Below these, all terms contained in  $(|D| - 1)$ -sized subsets ranked according to their mean ranks, etc.

In the next steps (lines 4–6), the KEYQUERY COVER-instance is solved for the subset  $D$  and its combined vocabulary  $W$ . The found candidate papers are added to the candidate set  $C$ . In case that enough candidates are found, the algorithm stops (line 7). Otherwise, some other input subset is used in the next iteration. If not enough candidates can be found with keyqueries for more than one paper, the keyqueries for the single papers form the fall-back option (also applies to single-paper inputs).

In our experiments, we will set  $k, l = 10$  and  $c = 100 \cdot |\mathcal{I}|$  (to be comparable, also the baselines are set to retrieve 100 candidate papers per input paper).

---

**Pseudocode 3** Solving RELATED WORK SEARCH

---

**Input:** List  $\mathcal{I}$  of input papers, number  $c$  of desired related papers, keyquery generality parameters  $k$  and  $l$

**Output:** Set  $\mathcal{C}$  of candidate related papers

- 1: **for**  $i \leftarrow |\mathcal{I}|$  down to 1 **do**
- 2:   **for all**  $D : D \in 2^{\mathcal{I}}, |D| = i$  **do**
- 3:      $W \leftarrow$  combine vocabularies of documents in  $D$
- 4:      $Q \leftarrow$  KEYQUERY COVER( $D, W, k, l$ )
- 5:      $C \leftarrow$  combine at most top- $l$  results of each  $q \in Q$
- 6:      $\mathcal{C} \leftarrow \mathcal{C} \cup C$
- 7:     **if**  $|\mathcal{C}| \geq c$  **then break**
- 8: **return**  $\mathcal{C}$  ranked by highest similarity to any document in  $\mathcal{I}$

---

## 4.4 COMBINING APPROACHES

To combine different RELATED WORK SEARCH algorithms (e.g., Google Scholar’s related articles and our keyquery approach), we use a simple interleaving procedure. First, the results of the individual approaches are computed and ranked as described above. We then interleave the ranked lists by first taking the first rank from the first approach, then the first rank from the second approach and so on, then the second rank from the first approach, etc. Already contained papers are not added again.

## 4.5 EVALUATION

To experimentally compare the algorithms, we conduct a Cranfield-style experiment on 200,000 computer science papers. Topics and judgments are acquired in a user study.

### 4.5.1 Experimental Design

In general, there are two different approaches to evaluate algorithms for related work search. A widely used method is to take the reference lists of papers as ground truth for the purpose of evaluation. Some of the references of a single input paper are hidden and it is measured whether an algorithm is able to re-identify these. The second possible method uses relevance judgments assigned by scholars. These judgments then state whether a recommended paper is relevant to a specific topic or not. Since the first method is rather biased towards the citation graph and also not really representing the use case we have in mind, we choose the second approach and utilize human relevance judgments from a carefully designed user study.

**Paper Corpus** We crawled a corpus of computer science papers starting from the 35,000 papers published at 20 top-tier conferences like SIGIR, CHI, CIKM, ACL, STOC, and iteratively included cited and citing papers until the desired size of 200,000 papers was reached. In the crawling process, not all papers had a full text available (for 57% of the corpus, we have an associated full text). In this case, only the abstracts and metadata were obtained when possible, but often not even abstracts were available and the respective papers were not included. On average, only 7.0 of the 13.9 references in a paper and 6.8 of the 9.5 citations to a paper could be crawled. Thus, only about 60% of the cited/citing papers are contained in our corpus. Not surprisingly, especially older papers often were not available. The papers included in our corpus have been published in the years 1962–2013 (more than 75% published after 2000). Starting from the 20 seed conferences, we included papers from about 1,000 conferences/workshops and 500 journals.

**Experimental Setup** Sofia Search is run on the citation information contained in the metadata of the papers. Note that due to the non-availability of 40% of the cited/citing papers, our corpus might not be optimally suited for Sofia Search. Still, this was not intended in the corpus design and could not be avoided—it rather represents the realistic Web scenario, but should be kept in mind when analyzing Sofia Search’s performance.

In order to run the query-based baseline and our keyquery cover algorithm, we indexed the corpus papers using Lucene 5.0 while treating title, abstract, and body text as separate fields. In case that no full text is available in the corpus, only title and abstract are indexed. The retrieval model is BM25F (Robertson et al., 2004) with different boost factors: the title is the most important field, followed by the abstract and then the body text.

Whenever a corpus paper with only title and abstract is used as an input paper for our keyquery-based algorithm, the keyphrase extraction is done on these two fields only and text similarity of a candidate paper is also only measured against these two fields (similar to the Sofia Search and Nascimento et al. baselines).

**User Study Design** Topics (information needs) and relevance judgments for a Cranfield-style analysis of the related work search approaches are obtained from computer science students and scholars (other qualifications do not match the corpus characteristics). A study participant first specifies a topic by selecting a set of input papers (could just be one) and then judges the found papers of the different approaches with respect to their relevance. These judgments are the basis for our experimental comparison.

In order to ensure a smooth work flow, we have built a web interface that also allows a user to participate without being on site. The study itself consists of two steps with different interfaces. In the first step, a participant is asked to enter a research task they are familiar with, and describe it with one or two sentences. Note that we request a familiar research task because expert knowledge is later required in order to judge the relevance of the suggested papers. After task description, the participants have to enter titles of input papers related to their task. While the user is entering a title, a background process automatically suggests title auto completions from our corpus. Whenever a title was not chosen from the suggestions but was manually entered, it is again checked whether the specified paper exists in our corpus or not. If the paper cannot be found in the collection, the user is notified to enter another title. After this two-phase topic formation (written description + input papers), the participants have to name at least one paper that they expect to be found (again with the help of auto completion). Last, the users should describe how they have chosen the input and expected papers to get some feedback of whether for instance Google Scholar was used which might bias the judgments.

After a participant has completed the topic formation, the different recommendation algorithms are run on the input papers and the pooled set of the top-10 results of each approach is displayed in random order for judgment (i.e., at most 40 papers to judge). For each paper, the fields title, authors, publication venue, and publication year are shown. Additionally, links to fade in the abstract and, if available in our corpus, to the respective PDF-file are listed. Thus, the participants can check the abstract or the full text if needed.

The participants assessed two criteria: relevance and familiarity. Relevance was rated on a 4-point scale (highly relevant, relevant, marginally, not relevant), while familiarity was a two-level judgment (familiar or not). Combining the two criteria, we can identify good papers not known to the participant before our study. This is especially interesting since we asked for research topics the participants are familiar with, and in such a scenario, algorithms identifying relevant papers not known before are especially interesting.

**Characteristics of the User Study** In total, 13 experienced scholars and 7 graduate students have “generated” and judged 42 topics in our study—in a previous study we had another 25 topics with single-paper inputs only (Hagen and Glimm, 2014). The scholars typically chose topics related to one of their paper projects, while the graduate students chose topics from

**TABLE 4.1:** (Left) Performance values achieved by the different algorithms averaged over all topics. (Right) Percentage of top-10 rank overlap averaged over all topics.

Algorithm	nDCG@10	prec @10	rece@50	recur@10		Sofia	Nasc.	Google	KQC
Sofia Search	0.60	0.59	0.33	0.20	Sofia Search	100	35	17	43
Nascimento	0.58	0.56	0.34	0.16	Nascimento	35	100	15	36
Google Scholar	0.60	0.59	<b>0.43</b>	<b>0.21</b>	Google Scholar	17	15	100	18
KQC	<b>0.62</b>	<b>0.60</b>	0.37	0.16	KQC	43	36	18	100
KQC+Google	<b>0.65</b>	0.63	<b>0.48</b>	0.21					
KQC+Sofia	0.61	0.60	0.39	0.19					
KQC+Sofia+Google	<b>0.65</b>	<b>0.64</b>	<b>0.48</b>	<b>0.24</b>					

their Master’s theses. On average, the topic creation took about 4 minutes while the judgment took about 27 minutes. For 23 topics, one or two input papers are given, while for 19 topics even three or up to five input papers were specified. For most topics (80%), two or more expected papers were entered. The number of different papers returned by the four algorithms varies highly. For the topic with the most different results, the participant had to judge 37 papers, while the topic with the least different results required only 13 judgments. On average, a participant had to judge 28 papers. In total, 31% of the results were judged as highly relevant, 22% as relevant, 24% as marginally relevant, and 23% as not at all relevant to the topic. Interestingly, 79% of the papers that are judged as relevant and 59% of the highly relevant papers were unfamiliar to the user before the study. We will evaluate the algorithms with respect to their ability of retrieving unexpected good results when discussing the experimental results.

#### 4.5.2 Experimental Results

We employ the standard retrieval effectiveness measures of nDCG (Järvelin and Kekäläinen, 2002) and precision for the top-10 results, and recall for evaluating the ability to retrieve the expected and also the relevant unexpected papers. The experimental results are reported in Table 4.1 (Left). The top four rows contain the results for the four individual algorithms, while the bottom three rows contain the most promising combinations that can also be evaluated due to the pooling strategy. In the top- $k$  results of our combination scheme described in Section 4.4, only results from the top- $k$  of the combined algorithms are included.

**General Retrieval Performance** We measure nDCG using the 4-point scale: high as 3, relevant as 2, marginal as 1, and not relevant as 0, and precision considering highly relevant and relevant as the relevant class. The

mean nDCG@10 and  $prec@10$  over all topics are given in the second and third columns of Table 4.1 (Left). The top-10 of our new keyquery-cover-based approach KQC are the most relevant among the individual methods but on par with Google Scholar and Sofia Search; both differences are not significant according to a two-sided paired t-test ( $p = 0.05^1$ ). The overall best result is achieved by the KQC+Sofia+Google combination that significantly outperforms its components. Another observation is that most methods significantly outperform the Nascimento et al. baseline.

**Recall of Expected Papers** To analyze how many papers were retrieved that the scholars entered as expected in the first step of our study, we use the recall of the expected papers denoted as  $rec_e$ . Since the computation of  $rec_e$  is not dependent on the obtained relevance judgments, we can compute recall for any  $k$ . We choose  $k = 50$  since we assume that a human would often not consider many more than the top-50 papers of a single related work search approach. The fourth column of Table 4.1 (Left) shows the average  $rec_e@50$ -values for each algorithm over all topics. The best  $rec_e@50$  among the individual methods is the 0.43 of Google Scholar. The combinations including KQC and Google Scholar achieve an even better result of 0.48. An explanation for the advantage of Google Scholar—beyond the probably good black-box model underlying the “related articles” functionality—can be found in the participants’ free text fields of topic formation. For several topics, the study participants stated that they used Google Scholar to come up with the set of expected papers and this obviously biases the results.

**Recall of Unfamiliar but Relevant Papers** We also measure how many “gems” the different algorithms recommend. That is, how many highly or relevant papers are found that the user was not familiar with before our study. Providing such papers is a very interesting feature that might eventually help a researcher to find “all” relevant literature on a given topic. Again we measure recall, but since we have familiarity judgments for the top-10 only, we measure the recall  $rec_{ur}@10$  of unexpected but relevant papers listed in the rightmost column of Table 4.1 (Left). Again, the combination KQC+Sofia+Google finds the most of the unfamiliar but relevant papers.

**Result Overlap** Since no participant had to judge 40 different papers, the top-10 results cannot be completely distinct; Table 4.1 (Right) shows the per-

---

<sup>1</sup>The effective  $p$ -value may be higher due to the multiple comparisons problem.

centage of overlap for each combination. On average, the top-10 retrieved papers of two different algorithms share 2–4 papers, meaning that the approaches retrieve a rather diverse set of related papers. This again suggests combinations of different approaches as the best possible system and combining the best query-based method (our new KQC), Google Scholar, and Sofia Search indeed achieves the best overall performance. Having in mind the “sparsity” of our crawled paper corpus’ citation graph (only about 60% of the cited/citing papers could be crawled), Sofia Search probably could diversify the results even more in a corpus containing all references. The combination of the three systems in our opinion also very well models the human way of looking for related work, such that the KQC+Sofia+Google combination could be viewed as very close to automated human behavior.

**A Word on Efficiency** The most costly part of our approach is the number of submitted queries. In our study, about 79 queries were submitted per topic; results for already submitted queries were cached such that no query was submitted twice (e.g., once when trying to find a keyquery for all input papers together, another time for a smaller subset again). On the one hand, 79 queries might be viewed more costly than the about 27 queries submitted by the Nascimento et al. baseline ( $10 \cdot |\mathcal{I}|$  queries), or the about 30 requests submitted for the Google Scholar suggestions ( $11 \cdot |\mathcal{I}|$  requests; one to find an individual input paper, ten to retrieve the 100 related articles). Also Sofia Search on a good index of the citation graph is much faster than our keyquery-based approach. On the other hand, keyqueries could be pre-computed at indexing time for every paper such that at retrieval time, only a few postlists from a reverted index (Pickens et al., 2010) have to be merged. This would substantially speed up the whole process, rendering a reverted-index-based variant of our keyquery approach an important step for deploying the first real prototype.

## 4.6 SUMMARY

In this chapter, we have presented a keyquery-based approach to related work search. The addressed common scenario is a scholar who has already found a handful of papers in an initial research, and wants to find “all” the other related papers—often a rather tedious task. Our problem formalization of RELATED WORK SEARCH is meant to provide automatic support in such situations. As for solving the problem, our new keyquery-based technique focuses on the content of the already found papers, complementing most

of the existing approaches that exploit the citation graph, only. Our overall idea is to get the best of both worlds (i.e., queries and citations) from appropriate method combinations.

And in fact, in our effectiveness evaluations of a Cranfield-style experiment on a collection of about 200,000 computer science papers, the combination of keyqueries with the citation-based Sofia Search and Google Scholar’s related article suggestions performed best on 42 topics (i.e., sets of initial papers). The top-10 results of each approach were judged by the expert who suggested the topic. Based on these relevance judgments, we have evaluated the different algorithms and identified promising combinations based on the rather different returned results of the individual approaches, amongst which keyqueries slightly outperformed Sofia Search and Google Scholar suggestions.



# 5

## Beyond Precision@10: Clustering the Long Tail of Web Search Results

This chapter addresses recall-oriented search, an IR task that arises in the context of corpus acquisition. In order to improve the recall of aspects or subtopics which are covered by the search result list for a query, we report on selected analyses and propose new concepts. Our findings in a nutshell are: (1) *Don't compete with a search engine's top hits.* In response to a query, we presume search engines to return an optimal result list in the sense of the probabilistic ranking principle: documents that are expected by the majority of users are placed on top, and form the result list head. We argue that, with respect to the top results, it is not beneficial to replace this established form of result presentation. (2) *Improve document access in the result list tail.* Documents that address the information need of "minorities" appear at some position in the result list tail. Especially for ambiguous and multi-faceted queries, we expect this tail to be long, with many users appreciating different documents. In this situation, web search result clustering can improve subtopic recall by reorganizing the long tail into topic-specific clusters. (3) *Avoid shadowing when constructing cluster labels.* We show that most of the cluster labels that are generated by current clustering technology occur within the snippets of the result list head—an effect which we call *shadowing*. The value of such labels for topic organization and navigation within a clustering of the entire result list is limited. We propose and analyze a filtering approach to significantly alleviate the label shadowing effect.

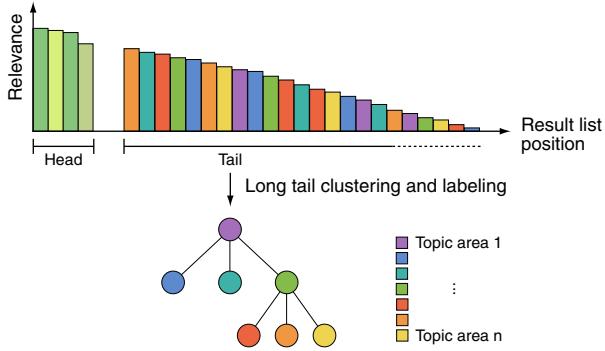
## 5.1 INTRODUCTION

Web search is the task of finding a document in the World Wide Web in order to satisfy a user's information need that is specified as a query. Deriving the "true" information need from a query is a challenge, and search engines often retrieve millions of documents from which only a fraction is relevant for the user. To reduce the negative impact of irrelevant documents, search engines apply a result presentation strategy, which can be characterized as either *relevance-based* or *diversity-based*.

The objective of the relevance-based strategy, which is the predominant strategy at this time, is to serve those information needs that are most likely associated with the query. The results are organized as a ranked list. The relevance-based strategy is extremely effective if a user can spot a desired document among the top ranks. If not, the user has to resort to a sequential search in the result list tail, which is ineffective since no topic-specific structuring is provided. Two user studies complement these observations impressively: A study from 2010 reviews the top five results of 1 000 queries sampled from the query log of a major search engine and reports that more than 90% of these queries are served excellent by all major search engines (Zaragoza et al., 2010). A study from 2008 reveals that only 8% of the users are willing to skim through more than three result pages, which corresponds to less than 30 search results (iProspect, 2008).

On the other hand, the objective of the diversity-based strategy is to serve multiple information needs "in parallel". A well-known representative of this strategy is web search result clustering (Carpinetto et al., 2009). Search engines that implement this strategy group similar documents into clusters and try to construct descriptive cluster labels to guide users during their search. If the clustering is effective and the labeling is expressive, diversity-based search engines can serve a multitude of information needs associated with a query equally well. Users with uncommon information needs or multi-faceted informational queries especially benefit from the structured information access provided by the clustering.

In this chapter, we propose to exploit both result presentation strategies by combining the head of a ranked result list with a clustering of those documents found in the tail and not topically covered by the head. Figure 5.1 illustrates the paradigm. Notice that this *competence partitioning* strategy is fundamentally different from the current practice of web search result clustering engines: by resorting to the result list head, we exploit the "wisdom" of the search engine's retrieval model. By clustering a filtered result list tail, we prevent a user from sifting through many redundant result pages.



**FIGURE 5.1: Result presentation by competence partitioning; combine the ranked result list head with a clustering of the result list tail.**

The remainder of this paper is organized as follows. Section 5.2 gives a focused overview of the state of the art, while Section 5.3 contributes analytical insights and methodological elements of our approach: (1) A critical analysis of the role that result clustering can play for web search, (2) a quantification of the label shadowing effect, and (3) a filtering approach to exploit the idea of long tail clustering. Compared to existing result clustering approaches, we report on less shadowing in the discovered clusters.

## 5.2 RELATED WORK

One of the first applications of document clustering was to improve retrieval based on the cluster hypothesis stating that "closely associated documents tend to be relevant to the same requests" (Rijsbergen, 1979). Later, clusters were also used as a browsing interface to explore and organize document collections like in the famous Scatter/Gather algorithm (Cutting et al., 1992). The numerous document clustering algorithms can be classified into three classes (Carpinetto et al., 2009): data-centric, description-aware, and description-centric algorithms.

### 5.2.1 Data-centric algorithms

Data-centric algorithms give top priority to the clustering, while the formation of cluster labels happens afterwards and has no effect on the partitioning. The labels are usually derived from a cluster's mathematical representation, e.g., a centroid under a bag-of-words model. Hence, the generated labels form a sequence of relevant but rather unrelated words, often lacking understandability. Examples for such systems are WebCat (Giannotti et al., 2003) and AIsearch (Stein and Meyer zu Eissen, 2004).

### 5.2.2 Description-aware algorithms

Description-aware algorithms try to circumvent the labeling issue of data-centric approaches by ensuring that the construction of cluster labels produces results that are comprehensive and meaningful to a user. One way to achieve this goal is to use algorithms that assign documents to clusters based on a single feature—so-called monothetic clustering—and to then use this feature as a label. One example is the Suffix Tree Clustering (Zamir and Etzioni, 1998) that exploits frequently recurring phrases as similarity features. First, base clusters are discovered by shared single frequent phrases utilizing suffix trees. Second, the base clusters are merged by their phrase overlap. However, since the merging step is based on the single-linkage criterion, the combined phrases of merged clusters forming the cluster labels often still tend to be unrelated and therefore misleading for a user. SnakeT (Ferragina and Gullì, 2004) enrich the labels obtained by STC using phrases from a predefined ontology. In Lipka et al. (2020), we patented a hierarchical online clustering algorithm that labels its clusters by the title of the Wikipedia article that is most similar to the cluster centroid. Still, the cluster analysis precedes and dominates the labeling task.

### 5.2.3 Description-centric algorithms

Description-centric algorithms consider the cluster labels as the crucial elements of a clustering. They assume that if a cluster cannot be described by a meaningful label, it is probably of no value to a user. The description precedes the assignment of a document to a cluster. Description-centric algorithms mainly tackle the use case of clustering web search results, with Lingo being one of the pioneering examples (Osiński et al., 2004)—now part of Carrot2, an open source framework for search result clustering.<sup>1</sup> A singular value decomposition of a frequent term matrix, generated from the search result list, is used to extract orthogonal vectors assumed to represent distinct topics in the snippets. The documents are then assigned to the extracted topic clusters using the Vector Space Model.

With a similar goal, Weiss revisits the data-centric  $k$ -means algorithm and adjusts it to a description-centric version: descriptive  $k$ -means (Weiss, 2006). First,  $k$ -means with  $tf \cdot idf$  features is run. From each cluster centroid, frequent phrases are extracted as potential cluster labels, the clustering itself is withdrawn. To populate the extracted cluster labels with documents, the algorithm searches for documents that are relevant for a cluster label under

---

<sup>1</sup><https://www.carrot2.org>

the Vector Space Model.

In 2016, we published a keyquery-based variant of descriptive k-means, which formulates the clustering problem as the reverse problem of query-based search as follows (Gollub et al., 2016a): Given a set of documents, find a set of diverse search queries that together retrieves the document set when submitted to a reference search engine. Along with their retrieved documents as cluster contents, the queries then form a labeled clustering. This formulation of the clustering problem implies that the potential clusters of a document are given by the keyqueries for which it is retrieved. From the algorithms presented in Gollub et al. (2016a), a keyquery-based variant of descriptive k-means showed the best performance: The quality of the obtained clusterings is on par with descriptive k-means, whereas the quality of the keyquery-based cluster labels is perceived significantly better in the user study conducted. Furthermore, the dynamic taxonomy composition approach presented in Chapter 7 can be understood as a keyquery-based, description-centric clustering algorithm.

### 5.3 APPROACH AND EVALUATION

In the following we discuss the competence partitioning strategy as illustrated in Figure 5.1, and introduce a tailored clustering approach, which we refer to as *LongTailClustering*.

#### 5.3.1 Towards Better Search Result Clustering

Relevance-based search engines answer navigational as well as the popular non-navigational queries almost perfectly. For such kinds of information needs, web search result clustering and related technology cannot compete. Aside from the navigational overhead (selecting, focusing, and browsing a cluster) the current clustering approaches struggle with the labeling problem as well as with runtime issues (Carpineto et al., 2009). One might argue that providing a comprehensive cluster containing all relevant results is always appreciated. But experience shows that often a fraction of the documents is sufficient to satisfy an information need. Once a relevant result is at the user’s disposal, alternative and more robust techniques exist to retrieve additional relevant documents: (1) search engines allow for searching similar documents given a particular result, (2) rephrasing the original query inspired by a relevant result became an accepted search strategy (Jones and Klinkner, 2008), and (3) the web itself as a hyperlinked document collection is a great resource for a guided search.

Given the impressive performance of relevance-based search engines, the outlined picture might raise the question whether web search result clustering is of any use. In Section 5.1, we identified the accessibility of the result list tail as the major weakness of ranked result lists. If we show that the result list tail is valuable and covers further query aspects, a perspective towards better search result clustering emerges.

In order to quantify the potential of the result list tail, we study two datasets: TREC Web Track 2009/2010 (Clarke et al., 2009, 2010) and AMBIENT (Carpinetto and Romano, 2008). Each dataset consists of a set  $T$  of search topics, with 100 and 44 elements, respectively. For each topic  $t \in T$ , a number of subtopics is provided. TREC features on average 4.6 subtopics, whereas AMBIENT provides 18 subtopics on average. Every subtopic focuses on a different aspect of  $t$ . We form the set  $S_t$  as the union of  $t$ 's subtopics, where we only consider the noun phrases of each subtopic description. Topic terms as well as stopwords present in the noun phrases are discarded. E.g., the topic  $t = \text{"fahrenheit"}$  of the AMBIENT dataset comprises subtopics such as "scale temperature", "michael moore movie", and "band". A standard search engine is queried with each topic  $t \in T$  in order to obtain a ranked list  $D_t = d_1, \dots, d_N$  of the top  $N$  results, where each  $d \in D_t$  denotes a result snippet. A subtopic of  $t$  is said to be *covered* by a snippet  $d$ , if at least two of its terms occur in  $d$ . In the special case of a single term subtopic, the single term has to appear in  $d$ . The set of all subtopics of  $t$  that are covered by a snippet  $d$  is denoted as  $\text{coverage}(d, S_t)$ . We are interested in the subtopic recall for a result list  $D_t$  at rank  $R$ , which is defined as the fraction of subtopics covered by the first  $R$  results (Zhai et al., 2003):

$$\text{SubTopicRecall}@R = \frac{|\cup_{i=1}^R \text{coverage}(d_i, S_t)|}{|S_t|}, \quad (5.1)$$

with  $d_i \in D_t$ . Figure 5.2 illustrates the average subtopic recall for all topics in  $T$  of the AMBIENT dataset. The TREC Web Track dataset reveals equivalent characteristics. The diagram shows that on average the first  $N=100$  results for a topic cover 66.6% of all subtopics. If we define the result list head to comprise the first ten results, it covers 24.5% of all subtopics; additional 42.1% of the subtopics are covered by the result list tail, rendering the tail a valuable information source. In summary, if web search result clustering can be tailored to organize just the remaining subtopics, it will perfectly complement the result list head and enable a more efficient and effective result analysis. The key to such a complementing behavior lies in the alleviation of the shadowing effect.

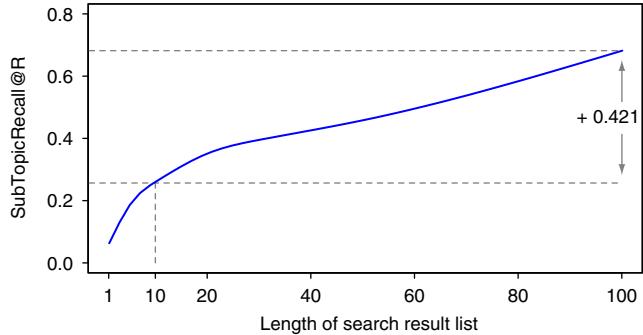


FIGURE 5.2: Average subtopic recall for the topics of the AMBIENT dataset. For each topic, we retrieve  $N=100$  search results from eTools (<http://www.etoools.ch>)

### 5.3.2 The Shadowing Effect

To generate a clustering that complements the head of a result list, the topics of documents that are to be clustered must differ from the topics already covered by the result list head. We refer to the effect of undesired topic repetition as *shadowing*. To quantify the shadowing effect, we perform state-of-the-art result clustering with Lingo on three different kinds of result lists: (a) the unmodified result list  $D_t$  to resemble the standard scenario of existing clustering search engines such as Carrot2, (b) the tail of  $D_t$ , as a naive application of our idea, and (c) a subset of the tail of  $D_t$ , which is filtered with respect to the topics of the head. In the following we set the number of generated clusters per clustering (= per topic  $t$ ) to ten; the respective cluster labels for a clustering are denoted as  $L_t$ . Cluster labels  $l \in L_t$  are phrases such as “Fahrenheit 9/11” or “Wellington based Web Design Collective”. Then, the shadowing at rank  $R$  can be expressed by replacing in Equation (5.1) the set of subtopics  $S_t$  with the set of cluster labels  $L_t$ :

$$\text{Shadowing@R} = \frac{|\cup_{i=1}^R \text{coverage}(d_i, L_t)|}{|L_t|},$$

with  $d_i \in D_t$ . Figure 5.3 illustrates the result of this analysis. The shadowing effect when clustering result list (a) is obvious: the Shadowing@10 is 0.417, i.e., about four cluster labels occur in the snippets of the head of the result lists. One might expect that clustering just the tail gives a reasonable smaller shadowing, which is not the case as can be seen in the results for result list (b). The reason for this behavior is that the topics from the head reappear throughout the entire result list. For our two datasets, up to 18% of the search result snippets in the tail cover the subtopics from the head. Finally, the filtered result list (c) reduces the shadowing effect by 31.6%, en-

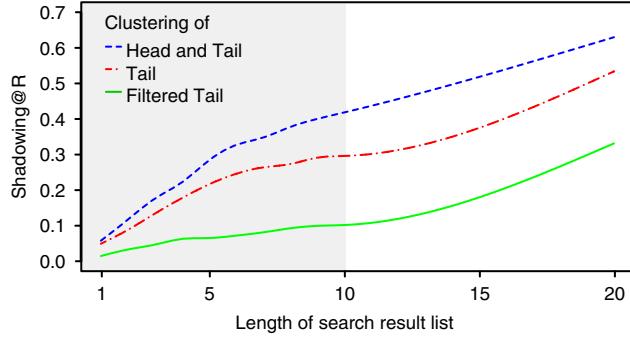


FIGURE 5.3: Average shadowing for the topics of the AMBIENT dataset. Three different result sets are considered for the cluster analysis

tailing a Shadowing@10 of only 0.1. Thus, nine out of ten cluster refer to subtopics present only in the tail.

### 5.3.3 Long Tail Filtering of Search Results

We now present a basic procedure to filter the result list tail. The aim of the filtering is to exclusively eliminate all results which cover topics from the result list head. We expect to satisfy the following two demands: (1) a reduction of the shadowing effect, and (2) a stable or increased subtopic recall after the filtering. Our procedure is detailed below.

---

#### Pseudocode 4 LongTailClustering

---

```

Input: query
Parameters: headSize, n
Output: tailClustering

1. ANALYZEHEAD(query)
   resultList = SEARCH(query)
   headTopics = ANALYZE(resultList, headSize)
2. CLUSTERTAIL(query, headTopics)
   augmentedQuery = query  $\wedge$   $\neg$ headTopics
   filteredTail = SEARCH(augmentedQuery)
   tailClustering = CLUSTER(filteredTail, n)
return tailClustering

```

---

*LongTailClustering* is a generic procedure, and the choices of the search engine **SEARCH**, the topic analyzer **ANALYZE**, and the clustering algorithm **CLUSTER** are user-specific. We experimented with a variety of configurations and see further research opportunities for this task. At the time of writing, the best overall results are achieved with the following setup: First, given a query, we receive from the meta-search engine eTools a result list of size  $N = 200$ . In the context of our evaluation, this search engine turned out to provide a higher snippet quality than did Bing or Yahoo. The quality of the

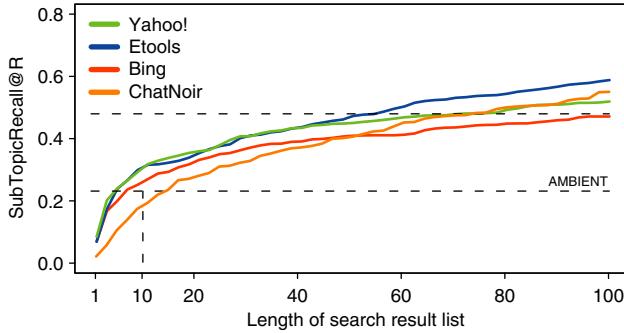
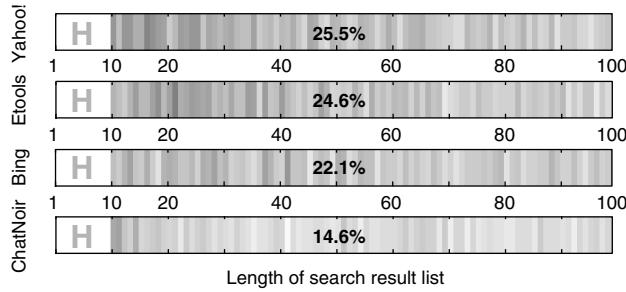


FIGURE 5.4: Average subtopic recall for the queries of the AMBIENT corpus. For each query 100 search results are retrieved from the four search engines Bing, Yahoo!, Etools, and ChatNoir.

snippets is of high importance, since we are interested in finding topics in short texts. For this topic analysis, we extract all noun phrases (Barker and Cornacchia, 2000) of maximum length four from the first ten results of the list. Each extracted noun phrase is treated as a topic that has to be filtered from the result list tail in the second step. Alternatively, only those noun phrases which constitute a keyquery for the respective snippet could be considered. In order to filter the result list tail, an augmented query is formed, which combines the original query with the extracted noun phrases prefixed by the NOT operator. For instance, the augmented query for "fahrenheit" starts with "fahrenheit -"scale temperature" -"taiwanese boy band" - "video game"". An augmented query can either be used to filter the original result list directly, or to send a new request to the search engine. In the latter case, the wisdom of the search engine developers is exploited again. For our experiments with the TREC Web Track and AMBIENT topics, better results are achieved with the former method. For the final clustering, we apply Lingo to the first 90 results of the filtered result list. Given the evaluation results in Gollub et al. (2016a), clusterings with superior labels should be achievable with our more recent keyquery-based variant of k-means.

As already reported and illustrated in Figure 5.3, the applied *LongTail-Clustering* procedure reduces the shadowing effect significantly.

In a final experiment, we investigate whether our filtering procedure retains the subtopics that do not appear in the head. For that, we substitute the original result list tail by its filtered version and measure the subtopic recall at rank 100. We find, that although 53.8% of the search results are eliminated in the course of *LongTailClustering*, the filtered result list still reaches a subtopic recall at 100 of 0.64, i.e., contains 96.4% of the subtopics from the original result list (original recall was 0.66). Hence, our procedure success-



**FIGURE 5.5:** Average shadowing for the queries of the AMBIENT corpus, caused by the search engines Yahoo!, Etools, Bing, and ChatNoir. A bar indicates the shadowing at the result list positions 10...100. The darker a bar the more shadowing occurred at that position.

fully implements our idea of moving the focus of web search result clustering to the subtopics that appear exclusively in the result list tail.

## 5.4 SUMMARY

A main achievement of our research is that we release web search result clustering from being considered direct competitor to ranked result lists. Instead, we see cluster analysis as a complementary presentation tool to improve the accessibility of relevant documents in the long tail of result lists. This way, web search result clustering goes in line with other advanced search tools such as related search or query suggestions. We have provided the measures SubTopicRecall@R and Shadowing@R to quantify our considerations, and we give empirical evidence for the claimed effects.

The shadowing effect is studied also in Stein et al. (2012), where we first analyzed the subtopics in the search results of four different search engines. For all queries of the AMBIENT corpus, the top 100 results of each search engine were studied. The obtained average subtopic recall is illustrated in Figure 5.4, which reveals that across all search engines, more than half of all subtopics appear only in the tail. This result substantiates our belief that the result list tail is a valuable resource. Concerning the shadowing effect, Figure 5.5 shows the results of our analysis. At each search result position, the shadowing effect is encoded as a gray scale value between 0.0 (no shadowing) and 1.0 (always shadowed). I.e., the gray scale value reflects the relative frequency of shadowing events at a specific rank. Up to 25.5% of the search results retrieved by Yahoo! are shadowed. Compared to all other search engines, ChatNoir reveals the lowest shadowing of only 14.6%. Nonetheless, all studied search engines confirm the hypothesis that

subtopics covered by the result list head reappear frequently throughout the tail.

With *LongTailClustering*, this chapter presents a procedure that is tailored to the implications of these findings: *LongTailClustering* is able to cope with the shadowing effect and to increase the density of subtopics that are exclusive for the result list tail. With the keyquery-based variant of k-means in Gollub et al. (2016a), we contribute an algorithm for the CLUSTER step which produces comprehensive cluster labels.



# 6

## Pseudo Descriptions for Meta-Data Retrieval

An IR task which arises in the context of corpus acquisition is search in meta-data. Search in meta-data is challenging due to the sparsity of the available textual information. To alleviate the sparsity problem, our contribution evolves from the existing document expansion paradigm and proposes pseudo-descriptions as a new paradigm. Instead of encoding paradigmatic term relations implicitly in an expansion vector, we generate an explicit cohesive text field for meta-data records that describes the entity associated with the record. In contrast to document expansions, pseudo-descriptions allow to reveal *why* a certain document is considered relevant although the original meta-data does not contain the query terms. Moreover, they are easier to operationalize and facilitate the use of sophisticated retrieval features such as phrase search and query term proximity. To generate pseudo-descriptions, we propose a relevance-dependent strategy that depends on the search engine result pages obtained from issuing the meta-data as a search query to a designated reference search engine. To demonstrate the validity of the pseudo-description paradigm, we experiment with different TREC collections where we withhold the content information to simulate a meta-data retrieval scenario. Though retrieval with full content information remains superior, our approach achieves significant retrieval performance improvements over meta-data only search.

## 6.1 INTRODUCTION

Search in meta-data collections is a common information retrieval scenario. The scenario is given whenever the entities of a collection are not linguistic but designate products, persons, or multi-media, or if the entities are linguistic but not available to the search engine provider. Prominent providers of meta-data search engines are e-commerce sites indexing product information, or libraries and archives indexing historic bibliographic records. From a search engine developer's point of view, one the one hand, meta-data collections are pleasant to work with in comparison to "standard" collections such as the Web: They usually can be obtained easily, are free of charge, professionally maintained, provided in a single standardized format, and compact, rendering both indexing and query processing efficient.

### 6.1.1 Retrieval Challenges

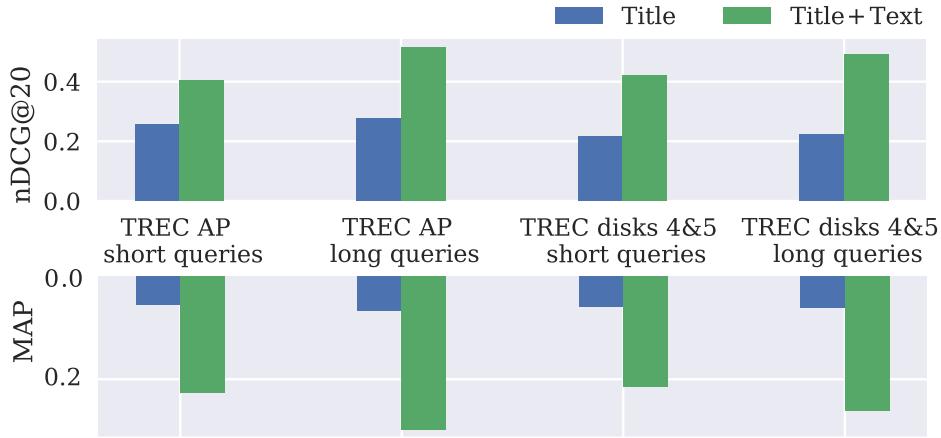
The compactness of meta-data enables efficient search, but, on the other hand, negatively impacts retrieval effectiveness. Due to the sparsity of lexical clues, meta-data retrieval often suffers noticeably from the so called vocabulary mismatch problem and the headroom problem (Efron et al., 2012).

The vocabulary mismatch problem arises since standard retrieval models ignore paradigmatic correlations between terms. Terms are treated as orthogonal basis vectors of an n-dimensional vector space, in which documents  $d$  and queries  $q$  are represented as vectors  $\mathbf{q}$  and  $\mathbf{d}$ . The relevance score  $RSV(q, d)$  is commonly computed by a multiplication of the two vectors, and hence zero if no terms match:

$$RSV(q, d) = \mathbf{q}^\top \mathbf{d} \quad \text{with } \mathbf{q}, \mathbf{d} \in \mathbb{R}^{n \times 1}$$

The less lexical information is provided in a meta-data collection, the more likely vocabulary mismatches will harm retrieval effectiveness.

The headroom problem arises since standard retrieval models rely on relative term frequency information to determine the result rank of a document. In collections with sparse lexical clues such as meta-data records, the term frequencies are likely to have Boolean character, i.e., terms occur either not or once. As a consequence, small variations in the document length determine the relevance ranking, which is, as pointed out by Efron et al. (2012), often not sensible. Figure 6.1 illustrates how limited lexical information harms the search effectiveness in the collections used in our experiments. If not the full content but only meta-data (titles) is indexed, the effectiveness in terms of nDCG@20 drops by about 50%. For performance



**FIGURE 6.1:** Retrieval effectiveness of Elastic Search with and without content information for two news collections. The nDCG@20 and MAP drop significantly if no content information is available.

measures that take more search results into account, such as MAP, the drop is yet higher.

### 6.1.2 Expansion Approaches

To tackle the problems that arise from sparse lexical information, query and document expansion approaches have been proposed. As detailed in Section 6.2, based on computed paradigmatic term relationships, these approaches add terms to the initial vector representations  $\mathbf{q}$  or  $\mathbf{d}$  to alleviate the vocabulary mismatch problem, and apply term weight smoothing to alleviate the headroom problem. The computed expansion vectors  $\mathbf{q}'$  or  $\mathbf{d}'$  are then factored into the relevance function  $RSV(q, d)$ , typically using a weighted sum. E.g., the relevance scoring for a document expansion then reads:

$$RSV_{\mathbf{d}'}(q, d) = \mathbf{q}^\top (w_1 \mathbf{d} + w_2 \mathbf{d}') \quad (6.1)$$

## 6.2 RELATED WORK

Since we introduce pseudo-descriptions as an alternative paradigm to document expansion for search in meta-data, no existing approaches for their generation exist. Therefore, in this section, we point out relations to other information retrieval tasks, and review existing work on document expansion.

### 6.2.1 Related IR Tasks

For the generation of pseudo descriptions for a given meta-data collection, we propose a process in Section 6.3 that comprises two principle steps.

In the first step, a pool of linguistic units from which a pseudo description can be compiled has to be retrieved for each meta-data record from a reference collection. If the lexical information in the meta-data records are short, e.g. only titles or product names, this step constitutes a classical information retrieval task including the generation of search result snippets (Bast and Celikik, 2014; Turpin et al., 2007). If the lexical information is more verbose, care has to be taken to not include too many irrelevant terms into the query. In this case, this first step becomes related to query-by-example retrieval, where, as reported in Chapter 4, keyword or keyquery extraction can be used to obtain appropriate query terms for an example document (Nascimento et al., 2011; Hagen et al., 2016). A further connection can be drawn to candidate retrieval for plagiarism detection, where a multitude of queries is commonly generated to retrieve potential source documents for different sections of a suspicious document (Hagen et al., 2017). If temporal information is available in the meta-data records, aspects of temporal information retrieval become relevant for this first step (Kanhabua et al., 2015).

In the second step, a subset of the candidate pool has to be selected and compiled into a text field. This task is similar to multi-document summarization (Ma et al., 2016; Mani et al., 2017). However, the objective is not a coherent, redundant-free text but a text that, if turned into a vector representation, approximates the result of applying a document expansion approach. In Section 6.3, we present an algorithm tailored to this objective.

### 6.2.2 Document Expansion Review

To devise an algorithm for the generation of pseudo descriptions, it is helpful to know about the existing document expansion approaches. As outlined in Section 6.1, the goal of document expansion is to alleviate the vocabulary mismatch problem and the headroom problem by incorporating paradigmatic term relations as a vector  $\mathbf{d}'$  into the relevance score computation. As one of the first attempts in this regard, the Generalized Vector Space Model (GVSM) presented by Wong et al. (1985) can be counted. The GVSM incorporates term relations by plugging a  $n \times n$  term relation matrix  $\mathbf{G}$  into the relevance function:

$$\text{RSV}_{\text{GVSM}}(q, d) = \mathbf{q}^\top \mathbf{G} \mathbf{d} \quad \text{with} \quad \mathbf{G} \in \mathbb{R}^{n \times n}$$

1985	2000	2005	2010	2015
Wong GVSM	Ponte Language Model	Singhal Spoken Doc. Retrieval	Tao Language Modeling	Efron Microblog Metadata
<u>Local Neighborhood Models</u>	Liu k-means	Wei LDA	Tsatsaronis WordNet	Yi LDA, Comp. Study
<u>Topic Models</u>	Berger EM		Karimzadehgan MI, Axiomatic Analysis	Egozi ESA
<u>Translational Models</u>			Ganguly LDA, Spoken Doc. Retrieval	Dalton Freebase
			Zuccon word2vec	Waitelonis LD-GVSM
				Ensan Semantic Linking
				Sherman Wikipedia

**FIGURE 6.2:** Document expansion approaches grouped by the problem perspective taken and arranged over time. Each publication is denoted by its first author and listed in the references. The keywords stated below the author outline characterize properties of the respective work.

Each element  $t_{ij} \in G$  denotes how strong the term  $t_j$  correlates with the term  $t_i$ . The GVSM can be interpreted as a document expansion approach, if  $\mathbf{G}$  is first multiplied by  $\mathbf{d}$  to obtain the expansion vector  $\mathbf{d}' = \mathbf{G} \mathbf{d}$ , which is then used in the relevance function  $RSV_{\mathbf{d}'}$  in Equation 6.1. Likewise, a query expansion approach arises when first multiplying  $\mathbf{q}^T$  with  $\mathbf{G}$ , which highlights the close relationship of the two expansion paradigms. Interestingly, document expansion papers commonly report increased retrieval performance when applying both query and document expansion. In terms of the GVSM, this corresponds to raising  $\mathbf{G}$  to the power of two, which intuitively uncovers second-order term relations, i.e., terms that co-occur with the same terms (such as synonyms).

Since the publication of the GVSM, a large amount of document expansion approaches have been proposed until recently. An overview of the approaches is illustrated in Figure 6.2. Though most of the approaches are framed within the Language Model by Ponte and Croft (1998), we will show here that it is possible to express most of them also within simple GVSM algebra. In the following paragraphs, we review document expansion approaches that exploit a lexical document collection. The key difference of the approaches is how the values in  $\mathbf{G}$  are determined. Please note that other approaches exist that rely on structured knowledge bases such as Wordnet (Tsatsaronis and Panagiotopoulou, 2009) or LinkedData (Ensan and Bagheri, 2017; Dalton et al., 2014; Waitelonis et al., 2015). However, it is not immediately clear how pseudo descriptions featuring a cohesive text can be obtained from these sources. Also note that, in order to keep the equations simple and general, normalization steps that are needed to exactly resemble the cited approaches are omitted in the following equations. In particular, every multiplication that is supposed to compute the cosine similarity of two vectors assumes that the vectors are  $L_2$ -normalized, first. For matrix rows, columns, and vectors that are supposed to be probability

distributions,  $L_1$ -normalization is assumed.

In their original paper, Wong et al. propose to use term co-occurrence statistics obtained from the document collection  $D = \{d_1, \dots, d_m\}$  itself as estimates for the entries of  $\mathbf{G}$ . In follow-up work, also the use of external collections (later denoted as  $E$ ) for this purpose, especially Wikipedia (Egozi et al., 2011; Sherman and Efron, 2017), has been proposed. Co-occurrence statistics can be computed by multiplying the normalized term-document matrix  $\mathbf{D}$  of  $D$  with its transpose (Manning et al., 2008), letting the computation of the document expansion vector become:

$$\mathbf{d}'_{\text{cooc}} = \mathbf{D} \mathbf{D}^\top \mathbf{d} \quad \text{with } \mathbf{D} \in \mathbb{R}^{n \times m}$$

As an alternative, computing the mutual information of the terms within  $D$  has been proposed in the context of translational document expansions models (Karimzadehgan and Zhai, 2010, 2012).

A further avenue of research on document expansion applies topic modeling to  $D$ . Topic models represent documents as a distribution over  $k$  topics, which in turn are distributions over  $n$  terms. Early topic based document expansion approaches apply cluster analysis to  $\mathbf{D}$  to obtain a set of  $k$  topic clusters, and then represent  $\mathbf{d}$  by its cosine similarity to the cluster centroids (Liu and Croft, 2004). If the  $k$  cluster centroids are stacked as a  $n \times k$  matrix  $\mathbf{C}$ , document expansion with clustering can be expressed as:

$$\mathbf{d}'_{\text{topic}} = \mathbf{C} \mathbf{C}^\top \mathbf{d} \quad \text{with } \mathbf{C} \in \mathbb{R}^{n \times k}$$

Canonically, document expansion with the explicit topic model ESA (Egozi et al., 2011) can be expressed. ESA treats articles from Wikipedia as explicit topics, and uses the vector representations of the article texts to compile a set of  $k$  topic vectors, which then make up the matrix  $\mathbf{C}$  in the equation above. Furthermore, also document expansion with word embeddings (Zucco et al., 2015) can be represented this way. In this case,  $\mathbf{C}$  contains the embeddings for each of the  $n$  terms over the  $k$  latent dimensions. To compute related terms,  $\mathbf{d}$  is first multiplied by  $\mathbf{C}^\top$  to obtain a latent representation for  $\mathbf{d}$ , which is then normalized and multiplied by  $\mathbf{C}$  to obtain a vector with cosine similarities of all  $n$  terms.

Though in principle, also document expansion with the latent topic model LDA (Ganguly et al., 2013; Wei and Croft, 2006; Yi and Allan, 2009) could be applied this way, the representation of documents under LDA is conventionally not determined by their similarity to the topic vectors but through statistical inference:

$$\mathbf{d}'_{\text{LDA}} = \mathbf{C} \text{ lda}(\mathbf{C}, \mathbf{d}) \quad \text{with} \quad \mathbf{C} \in \mathbb{R}^{n \times k}$$

In the equation above,  $\text{lda}(\mathbf{C}, \mathbf{d})$  is a function that infers a representation of  $\mathbf{d}$  within the latent topic space spanned by  $\mathbf{C}$ . A further function useful to express the final stream of research on document expansion reviewed in this section is the function  $\text{top}_r(\cdot)$ , which sets all but the  $r$  highest values of a vector to zero.

The function  $\text{top}_r(\cdot)$  is needed in the context of local neighborhood document expansion models (Efron et al., 2012; Sherman and Efron, 2017; Tao et al., 2006). Approaches of these kind interpret the multiplication of  $\mathbf{d}$  with the term-document matrix  $\mathbf{D}^T$  as a retrieval step, where  $\mathbf{d}$  serves as search query which is issued against  $D$ . The result is a vector of relevance scores over  $D$ . While co-occurrence models use all relevance scores to compute the final expansion vector, local neighborhood models pick only the  $r$  most relevant documents:

$$\mathbf{d}'_{\text{local}} = \mathbf{D} \text{ top}_r(\mathbf{D}^T \mathbf{d}) \quad \text{with} \quad \mathbf{D} \in \mathbb{R}^{n \times m}$$

With regards to the generation of pseudo descriptions discussed in the next section, the local neighborhood approach is conceptually closest to the approach we present for the generation of pseudo descriptions. We therefore use this approach as our baseline.

### 6.2.3 The Pseudo Description Paradigm

Common to all document expansion approaches is the fact that the lexical information that gives rise to the paradigmatic relations encoded in the expansion vectors is not retained explicitly. However, this lexical information may be valuable to have for a meta-data search engine: It would allow to provide search result snippets that reveal why a document is considered relevant although the original meta-data does not contain the query terms. If the document turns out to be indeed relevant for the user, the provided search result snippet may feature interesting extra information about the referred entity. If the document is not relevant, the user may identify the false paradigmatic relation in the snippet and is saved from following false assumptions. For this reason, we propose to progress from the expansion paradigm in this paper and propose a new paradigm for improving meta-data retrieval: pseudo descriptions.

In contrast to the expansion paradigm, the goal of pseudo descriptions is to compile for each meta-data document  $d$  a cohesive text field  $p$  which

(1) is a collected but limited set of linguistic units that are relevant for the entity referred to in the meta-data, and which (2) at the same time, encodes the paradigmatic term relations that are needed to alleviate the vocabulary mismatch and the headroom problem. Due to the cohesive text, in the best case, pseudo descriptions may even increase search effectiveness compared to expansion approaches since proximity features are now available for relevance scoring. From a search engine developer’s point of view, pseudo descriptions can be handled just like any other text field. Advanced expansion technology is not required. Each pseudo description  $p$  is represented canonically to  $d$  as a vector  $\mathbf{p}$ , and a multi-field query is used for the relevance computation:

$$\text{RSV}_p(q, d) = w_1 \mathbf{q}^\top \mathbf{d} + w_2 \mathbf{q}^\top \mathbf{p} \quad (6.2)$$

Pseudo-descriptions are, for the reasons mentioned above, appealing for meta-data search, but the question is if they can be produced in such a way that the two desired properties are met. To this end, in Section 6.3, we propose to exploit the search engine result page (SERP) that is returned when issuing meta-data as a search query to a reference search engine as an ad-hoc approach. Similar to pseudo-relevance feedback methods for query expansion, we assume that the SERP for the top documents represents a valid description of the meta-data’s entity. To tailor the SERP to the approximation of important paradigmatic relations, we disclose a relevance-dependent SERP generation strategy where the relevance score of a reference document controls the amount of linguistic units this document contributes to the SERP.

In Section 6.4, we report on a series of experiments we carry out to evaluate whether pseudo-descriptions can reach the effectiveness of document expansion approaches. The experiments are based on different TREC collections where we withhold the content information to simulate a meta-data retrieval scenario. As reference collections, the TREC collections themselves as well as Wikipedia and the CommonCrawl are used. Though, not surprisingly, retrieval with full content information is superior to both paradigms pseudo-descriptions and document expansion, our relevance-dependent approach for the former paradigm consistently reaches the document expansion baseline, and occasionally improves on it. In Section 6.5, we conclude with a discussion of open research questions and possibilities to further improve the generation of pseudo descriptions.

surrogate motherhood

**Judge Accepts Baby M Settlement**

Date: March 2, 1988

A [judge] Wednesday granted **surrogate** mother Mary Beth Whitehead-Gould unsupervised visits once a week with the child she bore under a \$10,000 contract and urged a truce in the [Baby M] legal battle.

Source: **Surrogate** Mom Gets Unsupervised Visitation With Child

FIGURE 6.3: Mockup of a meta-data search engine with pseudo descriptions. Shown is the first result for TREC query 70, “surrogate motherhood”. The result snippet features parts of the document’s pseudo-description. The red boxes surround terms that appear in the document’s title (=meta-data).

### 6.3 APPROACH

Pseudo-descriptions constitute an alternative paradigm to document expansion for alleviating the vocabulary mismatch and the headroom problem in the context of meta-data retrieval. In contrast to document expansion approaches, not a term vector  $d'$  is computed that encodes paradigmatic term relations, but term relations are encoded as an additional linguistic meta-data field  $p$  (see Equation 6.2 in Section 6.2.3). The main benefit of such a field is that the linguistic units that give rise to assume certain paradigmatic term relations are available to the meta-search engine for result snippet generation. This benefit is illustrated in Figure 6.3, which shows a search result for one of the TREC queries used in our experiments. Though the title of the retrieved meta-data document does not feature any of the query terms, its pseudo description features the query terms, allowing the search engine (1) to conclude that the document is relevant, and (2) to make the evidence used for this conclusion transparent for the user (in Figure 6.3, bold terms in the pseudo description excerpt indicate query term matches).

The question addressed in the remainder of this section is how pseudo descriptions should be computed for a meta-data collection  $D$  on the basis of an external reference collection  $E$ . As already mentioned in Section 6.2.1, we propose to model the generation of pseudo descriptions as a two step process: a candidate retrieval and a candidate selection process.

#### 6.3.1 Candidate Retrieval

In the first step, linguistic units that are relevant for a meta-data record  $d \in D$  have to be retrieved from an external reference collection  $E$ . Being

in an information retrieval context, a natural way to obtain relevant linguistic units is to employ a search engine indexing the reference collection for this purpose. Relevant linguistic units are obtained by formulating a search query on the basis of the meta-data record, submitting this search query to the reference search engine, and taking the document snippets of the returned search result page (SERP) as the pool  $S$  of relevant linguistic units:

$$S = \text{serp}_{l,u}(E, \text{top}_r(\mathbf{E}^\top \mathbf{d})) \quad \text{with} \quad \mathbf{E} \in \mathbb{R}^{n \times m}$$

In the equation above, the meta-data query of  $d$ , denoted as  $\mathbf{d}$ , is first multiplied by  $\mathbf{E}^\top$  to obtain the relevance scores for all reference documents in  $E$ . Analogous to local neighborhood document expansion, the function  $\text{top}_r(\cdot)$  is then applied to filter out all but the  $r$  most relevant documents. Instead of now aggregating the top  $r$  document vectors relative to their relevance scores to obtain an expansion vector  $\mathbf{d}'$ , we use the SERP-generation function  $\text{serp}(\cdot)$  of the reference search engine to obtain relevant linguistic units on sub-document level. By default, the reference search engine we employ for our experiments, Elastic Search, uses the Lucene Unified highlighter, which “breaks the text into sentences and uses the BM25 algorithm to score individual sentences as if they were documents in the corpus”. The parameters  $l$  and  $u$  of the  $\text{serp}$  function specify the target length of a single result snippet and the target amount of snippets that should be returned for each relevant document, respectively. As a consequence of this snippet generation approach, the linguistic units obtained feature subsets of the meta-data query terms within the context of one or a few sentences, making these linguistic units very appropriate for our scenario. In Figure 6.3, the meta-data query terms are surrounded by red boxes to highlight how the pseudo description connects the user’s search query (“surrogate motherhood”) with the meta-data title (“Judge Accepts Baby M Settlement”). Note that, by just joining all of the (up to)  $r * u$  returned linguistic units  $S = \{s_{1,1}, \dots, s_{r,u}\}$  together, a pseudo description  $p_{\text{adhoc}} = \text{join}(S)$  can be produced already after this first step. The resulting vector representation of  $p_{\text{adhoc}}$  is the sum over the vector representations  $s_{i,j}$  of all linguistic units in  $S$ :

$$\mathbf{p}_{\text{adhoc}} = \sum_{i=1}^r \sum_{j=1}^{u_i} \mathbf{s}_{i,j}$$

We refer to this one-step approach as *ad-hoc* approach in our experiments, where we request  $u = 1$  snippet with a target size of  $l = 250$  characters for the top  $r = 10$  reference documents.

### 6.3.2 Candidate Selection

A limitation of the ad-hoc approach is that the generated pseudo descriptions  $p_{\text{adhoc}}$  are not tailored to approximate a particular document expansion vector  $\mathbf{d}'$ , and hence may misrepresent existing paradigmatic term relations. To remedy this short coming, we propose a candidate selection procedure as the second step of the pseudo description generation. To derive a first algorithm for this second step, we choose local neighborhood document expansion as our reference document expansion approach that we try to approximate. To recap, for a document  $d$ , local neighborhood document expansion first issues  $d$  as a search query to a search engine indexing the reference collection  $E$ . The vector representations  $\mathbf{e}$  of the  $r$  most relevant documents in  $E$  are then summed up proportional to their relevance score  $\text{RSV}(d, e)$ . This can be, now including a normalization parameter  $z$ , written as:

$$\mathbf{d}'_{\text{local}} = \mathbf{E} \text{top}_r(\mathbf{E}^T \mathbf{d}) = \sum_{e_i \in \text{top}_r} \frac{\text{RSV}(d, e_i)}{z} \mathbf{e}_i \quad (6.3)$$

To approximate the vector  $\mathbf{d}'_{\text{local}}$  with a pseudo description, we make the simplifying assumptions that (1) the vector representation of every linguistic unit  $s_{i,j} \in S$  is, respecting its relative size  $l_{i,j}/|\mathbf{e}_i|$ , a fraction of the vector representation  $\mathbf{e}_i$  from which the linguistic unit is taken, and that (2) every linguistic unit has the same length  $l$ . Under these assumptions, the vector representation of  $p_{\text{adhoc}}$  can be rewritten as:

$$\mathbf{p}_{\text{adhoc}} = \sum_{i=1}^r \sum_{j=1}^{u_i} \mathbf{s}_{i,j} \stackrel{(1)}{\approx} \sum_{i=1}^r \sum_{j=1}^{u_i} \frac{l_{i,j}}{|\mathbf{e}_i|} \mathbf{e}_i \stackrel{(2)}{=} \sum_{i=1}^r u_i \frac{l}{|\mathbf{e}_i|} \mathbf{e}_i$$

Comparing the vector representations  $\mathbf{d}'_{\text{local}}$  with  $\mathbf{p}_{\text{adhoc}}$ , one can see that both vectors are sums over the  $r$  document vectors  $\mathbf{e}_i$ , and that the vectors would be equal if the fractions in the sums are equal, i.e., if  $\frac{\text{RSV}(d, e_i)}{z} = u_i \frac{l}{|\mathbf{e}_i|}$ . Since the number of snippets  $u_i$  taken from each reference document  $e_i$  for inclusion into a pseudo description can be controlled, we propose, towards the generation of a pseudo description  $p^*$  that is tailored to local neighborhood document expansion, to modify  $u_i$  such that the fractions above become equal:

$$u_i^* = \left[ \frac{\text{RSV}(d, e_i)}{z} \frac{|\mathbf{e}_i|}{l} \right]$$

In the equation, the brackets denote rounding to the next integer, which is required since the number of snippets taken for a reference document

must be a whole number. By using  $u_i^*$ , the final algorithm for our candidate selection algorithm, which we refer to as “relevance-dependent”, can be written as follows:

$$p^* = \text{join}(S^*) = \text{join} \left( \bigcup_{i=1}^r \bigcup_{j=1}^{u_i^*} s_{i,j} \right)$$

A final point to discuss is the normalization parameter  $z$ . Since local neighborhood document expansion has been proposed in the context of the Language Model, where any expansion vectors  $\mathbf{d}'$  are supposed to be probability distributions ( $L_1$  normalized), the parameter  $z$  is commonly chosen to be  $z = \sum_{e_i \in \text{top}_r} \text{RSV}(d, e_i)$ . However, we believe that this value is not appropriate in our context for the following reason: If for a document  $d$  only one marginally relevant document can be retrieved from the reference collection  $E$ , a  $L_1$ -normalizing value for  $z$  would have the effect that all of the snippets of this document end up in the pseudo description, likely having a negative effect on the overall retrieval performance. Conversely, if many highly relevant documents are retrieved, only few snippets would be taken from each of these documents, and significant paradigmatic term relations may be missed. Therefor, we propose to set  $z$  to be the maximum retrieval score that could be expected for the query representation of a meta-data document. This way, any marginally relevant document also contributes only marginally to a pseudo description, whereas any highly relevant document contributes many snippets. As the maximum expected value for a query highly depends on the retrieval model of the reference search engine, which might in detail be unknown, in our experiments, we set  $z$  to the maximum retrieval score that we observed for all queries submitted to a reference search engine.

## 6.4 EVALUATION

In this section, we report on a series of experiments we conducted to compare the retrieval performance of relevance-dependent pseudo descriptions  $p^*$  with the performance of both ad-hoc pseudo descriptions  $p_{\text{adhoc}}$  and local neighborhood document expansions  $\mathbf{d}'_{\text{local}}$ . Since the goal of  $p^*$  is to approximate  $\mathbf{d}'_{\text{local}}$ , achieving performance characteristics en par with  $\mathbf{d}'_{\text{local}}$  constitutes a positive evaluation result. Due to the availability of term proximity features when using pseudo descriptions, in the best case, small performance improvements can be anticipated.

**TABLE 6.1:** Evaluation results of the approaches  $d'_{\text{local}}$ ,  $p_{\text{adhoc}}$ , and  $p^*$  for two meta-data and four reference collections. Reported are nDCG@20 and MAP performance scores for the available short and long query versions.  $d'_{\text{local}}$  is regarded as baseline, and  $\uparrow$  /  $\downarrow$  indicate improvement / decline over the baseline.  $\dagger$  /  $\ddagger$  indicate statistical significance of a result according to the Wilcoxon signed-rank test at the levels  $p < 0.05$  /  $p < 0.01$ , respectively.

Collection	Reference Collection	Approach	nDCG@20		MAP	
			short	long	short	long
TREC	AP	-	0.405	0.516	0.226	0.298
		-	0.255	0.277	0.055	0.067
		$d'_{\text{local}}$	0.329	0.389	<b>0.151</b>	<b>0.173</b>
	without self	$p_{\text{adhoc}}$	0.319 $\downarrow$	0.381 $\downarrow$	0.108 $\downarrow$	0.137 $\downarrow$
		$p^*$	<b>0.335<math>\uparrow</math></b>	<b>0.418<math>\uparrow</math></b>	0.128 $\downarrow$	0.166 $\downarrow$
		$d'_{\text{local}}$	0.327	0.391	0.148	0.171
	AP Titles	$p_{\text{adhoc}}$	0.332 $\uparrow$	0.384 $\downarrow$	0.113 $\downarrow$	0.140 $\downarrow$
		$p^*$	<b>0.356<math>\uparrow</math></b>	<b>0.435<math>\uparrow</math></b>	<b>0.150<math>\uparrow</math></b>	<b>0.184<math>\uparrow</math></b>
		$d'_{\text{local}}$	0.301	0.348	<b>0.103</b>	<b>0.121</b>
TREC	Wikipedia	$p_{\text{adhoc}}$	0.299 $\downarrow$	0.348	0.082 $\downarrow$	0.103 $\downarrow$
		$p^*$	<b>0.315<math>\uparrow</math></b>	<b>0.361<math>\uparrow</math></b>	0.099 $\downarrow$	0.119 $\downarrow$
		$d'_{\text{local}}$	<b>0.330</b>	<b>0.397</b>	<b>0.112</b>	<b>0.135</b>
	Common-Crawl	$p_{\text{adhoc}}$	0.299 $\downarrow$	0.355 $\downarrow$	0.086 $\downarrow$	0.107 $\downarrow$
		$p^*$	<b>0.330</b>	0.382 $\downarrow$	0.103 $\downarrow$	0.126 $\downarrow$
		$d'_{\text{local}}$	0.426	0.492	0.217	0.261
DISKS 4&5	DISKS 4&5	-	0.217	0.222	0.060	0.060
		-	0.313	<b>0.354</b>	<b>0.130</b>	<b>0.145</b>
		$d'_{\text{local}}$	0.291 $\downarrow$	0.329 $\downarrow$	0.094 $\downarrow\dagger$	0.109 $\downarrow$
	Titles	$p_{\text{adhoc}}$	$\dagger$	$\dagger$	$\dagger$	$\dagger$
		$p^*$	<b>0.314<math>\uparrow\dagger</math></b>	0.353 $\downarrow$	0.110 $\downarrow\dagger$	0.130 $\downarrow$
		$d'_{\text{local}}$	0.304	0.350	0.121	0.137
TREC	DISKS 4&5	$p_{\text{adhoc}}$	0.291 $\downarrow\dagger$	0.329 $\downarrow$	0.093 $\downarrow\dagger$	0.109 $\downarrow$
		$p^*$	<b>0.333<math>\uparrow\dagger</math></b>	<b>0.379<math>\uparrow</math></b>	<b>0.126<math>\uparrow\dagger</math></b>	<b>0.149<math>\uparrow</math></b>
		$d'_{\text{local}}$	0.286	0.319	<b>0.100</b>	0.111
DISKS 4&5	Wikipedia	$p_{\text{adhoc}}$	0.296 $\uparrow\dagger$	0.306 $\downarrow$	0.086 $\downarrow\dagger$	0.095 $\downarrow$
		$p^*$	<b>0.308<math>\uparrow\dagger</math></b>	<b>0.332<math>\uparrow</math></b>	0.098 $\downarrow\dagger$	<b>0.113<math>\uparrow</math></b>
		$d'_{\text{local}}$	<b>0.313</b>	<b>0.372</b>	<b>0.111</b>	<b>0.131</b>
	Titles	$p_{\text{adhoc}}$	0.280 $\downarrow\dagger$	0.323 $\downarrow$	0.080 $\downarrow$	0.097 $\downarrow$
		$p^*$	0.302 $\downarrow\dagger$	0.357 $\downarrow$	0.098 $\downarrow$	0.121 $\downarrow$
		$d'_{\text{local}}$	0.280	0.319	0.121	0.137

**TABLE 6.2: nDCG@20 effectiveness comparison between disabled and enabled term proximity ranking.**  $\uparrow / \downarrow$  denotes improvement / decline and  $\dagger$  indicates statistical significance at the  $p < 0.05$

Collection	Proximity	Title and Text	$p^*$			
			Coll. w/o self	Coll. w/o titles	Wiki	Common- crawl
AP	Off	0.402	0.335	0.349	0.316	0.324
	On	0.405 $\dagger$	0.335 $\dagger$	0.356 $\dagger$	0.315 $\downarrow$	0.330 $\dagger$
DISKS 4&5	Off	0.420	0.309	0.324	0.305	0.298
	On	0.426 $\dagger\dagger$	0.314 $\dagger\dagger$	0.333 $\dagger\dagger$	0.308 $\dagger\dagger$	0.302 $\dagger\dagger$

#### 6.4.1 Datasets

We use two popular information retrieval benchmark collections for our experiments. The TREC AP collection (242,917 Associated Press news articles) and the combined collections on TREC DISKS 4&5 (472,521 news articles from Financial Times Limited, Foreign Broadcast Information Service, and Los Angeles Times; excluded due to missing title field: Congressional Records and Federal Register). To simulate a meta-data retrieval scenario, only the title field of the collections are made available to the meta-data search engines. To measure retrieval performance, we issue short and long versions of TREC queries 51-150 against the AP collection, and of TREC queries 351-450 against the TREC DISKS 4&5 collection.

As reference collections for the generation of pseudo descriptions and document expansion vectors, we reuse the two TREC collections in two variants: In variant one, “without self”, the complete collection is available but the document under consideration is in each case ignored when retrieved. In variant two, “without titles”, only the content field of the collections is available to simulate retrieval in an unstructured text collection with at least one highly relevant document. Furthermore, Wikipedia (2,104,323 articles)<sup>1</sup> is used as an encyclopedic reference collection, as well as the CommonCrawl (1,630,502,843 web pages)<sup>2</sup> to simulate, with more than a billion web pages, retrieval against the Web.

#### 6.4.2 Experimental Setup

To compute the retrieval performance of the three approaches  $d'_{local}$ ,  $p_{adhoc}$ , and  $p^*$  across all combinations of meta-data and reference collections, we proceed in three steps.

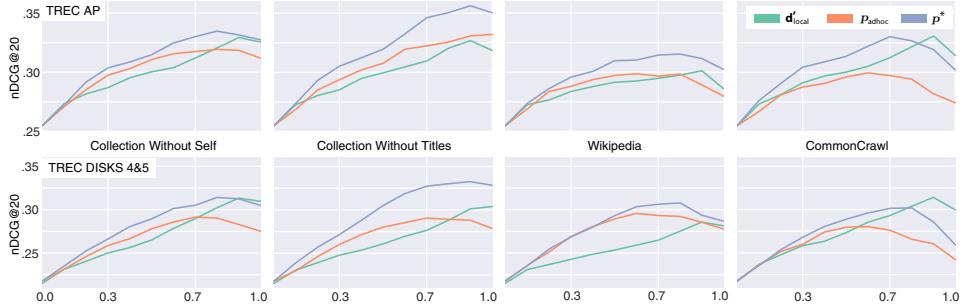
<sup>1</sup>Not considered were lists, disambiguation, and short (#words < 250) articles.

<sup>2</sup><http://commoncrawl.org/2015/12/november-2015-crawl-archive-now-available/>

**Step 1.** In the first step, all six reference collections (2 x TREC AP, 2x TREC DISKS 4&5, Wikipedia, and CommonCrawl) are indexed using the prevalent open source search engine Elastic Search with default settings (~BM25 retrieval model).

**Step 2.** In the second step, the titles of both meta-data collections are issued, without stopwords, as queries to each of the reference collections. To generate the expansion vectors  $\mathbf{d}'_{\text{local}}$ , the best performing settings reported by Tao et al. (Tao et al., 2006) are adopted, i.e., the term vectors of the  $r = 100$  most relevant documents were fetched for each query and summed up relative to their retrieval score using the  $L_1$ -normalizing version for the parameter  $z$  (cf. Equation 6.3). To generate the ad-hoc pseudo descriptions  $p_{\text{adhoc}}$ ,  $u = 1$  document snippet is requested from Elastic Search with a target size of  $l = 250$  characters for the top  $r = 10$  reference documents. To generate the relevance-dependent pseudo descriptions  $p^*$ ,  $u = 10$  candidate document snippets are initially requested from Elastic Search, also with a target size of  $l = 250$  characters for the top  $r = 10$  reference documents. If the relevance-dependent candidate selection algorithm suggests to add more snippets than returned for a document, all available snippets are just added to  $p^*$  without any compensation.

**Step 3.** In the third step, the generated pseudo descriptions and document expansions are separately added as additional field to the respective meta-data collection to yield “enriched” meta-data collections. For each of the two meta-data collections, three approaches have been applied to six different reference collections, giving a total of 36 enriched meta-data collections. Note that to keep the evaluation table concise, we do not report on the results across meta-data collections in this paper, e.g., not on results obtained for TREC AP meta-data with TREC DISKS 4&5 as the reference collection. Each enriched meta-data collections is again indexed with Elastic Search. To be able to process the document expansion vectors  $\mathbf{d}'_{\text{local}}$  properly, we implemented a custom plugin for Elastic Search. For the pseudo descriptions, no custom code is required, which highlights the operational advantage of our approach, again. To evaluate the retrieval performance of the enriched meta-data collections, the (short and long) TREC queries available for each meta-data collection are issued as multi-match query against the respective enriched meta-data search engines, and the resulting ranking is evaluated with the `trec_eval` script. Multi-match queries allow to combine retrieval scores obtained from multiple fields. In our case, retrieval scores for the title field as well as the added “enriched” field are combined using different field weights  $w_1$  and  $w_2$  (cf. Equations 6.1 and 6.2 in Section 7.1). To find optimal weights for every enriched meta-data collection,



**FIGURE 6.4:** Short query performance of the evaluated approaches for different values of  $\lambda \in [0, 1]$ . Each plot shows a different combination of meta-data and reference collection.

we set  $w_1 = 1 - \lambda$  and  $w_2 = \lambda$  (Jelinek-Mercer smoothing (Zhai and Lafferty, 2001)), and performed a grid search over  $\lambda$ .

### 6.4.3 Overall Results

The performance results of all approaches under optimal  $\lambda$  values are summarized in Table 6.1. On the top, results for the TREC AP collection are shown, the results for TREC DISKS 4&5 are shown below. As primary retrieval performance measure, nDCG@20 is reported, which has been found to correlate better than other measures with user preferences (Sanderson et al., 2010). In addition, performance in terms of MAP is reported. In the first row with results, the retrieval performance of a search engine indexing the full TREC collections are given as reference. In both cases, this (practically not available) search engine achieves the best performance, indicating that there is still room for improvement. Conversely, the second result row shows how a search engine indexing only the title information performs. In both cases, the title only search engine performs worst, indicating that every of the evaluated approaches is worth applying.

Comparing the performance of the three evaluated approaches among each other, it becomes apparent that the ad-hoc pseudo descriptions  $p_{adhoc}$  commonly perform slightly worse than the other two approaches. More importantly, the main finding of the experiment is that the performance of the relevance-dependent pseudo descriptions  $p^*$  is indeed en par with the performance of the local neighborhood document expansions  $d'_{local}$ . Except for the CommonCrawl reference collection and one further result, the nDCG@20 performance is even slightly better. Note that the significance assessments in Table 6.1 have to be treated with caution due to the multiple comparisons we do.

#### 6.4.4 nDCG vs. MAP

Studying the MAP performance scores in Table 6.1, one can observe that in 11 of the 16 cases, our approach  $p^*$  performs slightly worse than the document expansion approach  $d'_{\text{local}}$  for this performance measure. Compared to nDCG@20, MAP puts more emphasis on retrieval recall than on retrieval precision, giving rise to the hypothesis that our approach misses to encode some of the weak paradigmatic term relations needed to detect relevant documents with high recall. A possible reason for this is that fewer linguistic units are encoded in the vector representation of  $p^*$ . However, in most practical settings, a high retrieval precision as emphasized by nDCG@20 increases the user experience more than a high retrieval recall, and hence we argue in line with Fuhr (2018) that it is worth trading a small loss in terms of MAP against a small gain in nDCG@20.

#### 6.4.5 Influence of Reference Collections

Taking a closer look at the nDCG@20 performance changes under different reference collections, two observations are striking. First, our approach  $p^*$  performs better under the internal collections “without titles” than on the internal collections “without self”, whereas the opposite is true for the document expansion approach  $d'_{\text{local}}$ . We hypothesize that this phenomenon arises due to the different normalization parameters used in the two approaches. The maximum normalization used in our approach ensures that many snippets of the original document contained in the “without title” collections are included in  $p^*$ . For  $d'_{\text{local}}$ , the influence of the original document is, because of the  $L_1$  normalization, comparably low, preventing its full exploitation. The second interesting observation is that our approach is able to exploit Wikipedia better than  $d'_{\text{local}}$  does, whereas the opposite is true for the CommonCrawl. Our hypothesis here is that, in the case of Wikipedia, only small subsections of the retrieved documents really talk about the topic of the meta-data document, while the article itself is about a more general topic. Since, in contrast to the document expansions, pseudo descriptions consider linguistic units on sub-document level, it is possible that just these relevant subsections are taken. In the case of the CommonCrawl, we hypothesize that many relevant articles appear in this collection. Since  $d'_{\text{local}}$  takes more documents into account, it is likely that here, paradigmatic term relations can be inferred more robustly by this approach. To further investigate this phenomenon, a systematic exploration of the approaches’ parameters is required, which is, however, beyond the scope of this evaluation.

### 6.4.6 Lambda Plots

As noted in the experimental setup, the optimal value for the smoothing parameter  $\lambda$  is determined via grid search. In Figure 6.4, the performance development over  $\lambda$  is illustrated for the three evaluated approaches. Each plot refers to a specific combination of meta-data and reference collection. From the plots, it can be observed that the optimal  $\lambda$  is typically between 0.7 and 1.0 for the two best approaches. Further, across the value range,  $p^*$  is consistently better performing than the other approaches. Only with CommonCrawl as the reference collection,  $d'_{\text{local}}$  achieves better performances near the end of the value range. Towards the validity of our approach, this observation substantiates our belief that relevance-dependent pseudo descriptions are a robust alternative for local neighborhood document expansions.

### 6.4.7 Benefit of Term Proximities

One of the benefits of pseudo descriptions is that term proximity features can be used by the meta-data search engine for the relevance computation. As a final evaluation, in Table 6.2, the differences in retrieval performance when toggling the term proximity feature of Elastic Search on and off are presented for the full content search engine and the search engines indexing the pseudo descriptions  $p^*$ . In all but one case, using the proximity feature leads to slight retrieval performance increases. For the TREC DISKS 4&5 collection, the improvements are significant at the  $p < 0.05$  level according to the Wilcoxon signed-rank test. Again, we see this result as a substantiation of the validity of the pseudo description paradigm.

## 6.5 SUMMARY

The main contribution in this chapter is the introduction of pseudo descriptions as a new paradigm for improving search in meta-data. Towards conceptual and operational models for the generation of pseudo descriptions, we first reviewed existing document expansion approaches in terms of the Generalized Vector Space Model. Based on the gained insights, a general two-step procedure for the generation of meta-data description is presented, and we propose an algorithm that generates pseudo descriptions on the basis of search result pages while approximating local neighborhood document expansion. Related research questions include how non-lexical meta-data fields could be exploited, whether linguistic units other than search result snippets can be used as basis for pseudo descriptions, how other docu-

ment expansion approaches such as topic or translational models can be approximated, and whether local neighborhood document expansion can be approximated with less strong assumptions. The validity of our relevance-dependent pseudo description approach is evaluated in Section 6.4, where we find that our approach yield the performance characteristics of local neighborhood document expansion. An open research question in this regard is how beneficial the presentation of pseudo descriptions in search results is in practice. Though we observed a multitude of promising result snippets produced from pseudo descriptions in our experiments (like the one in Figure 6.3), a scientific answer to this question requires a sophisticated user study.



# 7

## Dynamic Taxonomy Composition via Keyqueries

This chapter presents an unsupervised framework for dynamic taxonomy composition, which can be used to model the topics of a corpus in a data-driven way, e.g. during the research question modeling activity of a DH project. The taxonomy classes of the presented approach correspond to keyqueries that are determined on the basis of a reference search engine.

The keyquery framework addresses important problems of static classification systems: overlarge classes and overly complex taxonomy structures. If, for instance, a leaf class grows to an indigestible size, keyqueries for the contained documents provide a suitable split mechanism. Since queries are well-known to digital humanities researchers from their daily web search experience, they increase the structural complexity in a transparent way. An empirical evaluation with an ACM and the CORE collection of scientific papers demonstrates the efficiency and effectiveness of our taxonomy composition framework.

### 7.1 INTRODUCTION

A classification system or taxonomy consists of a set of classes and a set of instances, where each instance is assigned to or tagged with some of the classes. In a hierarchical classification system, classes are usually arranged as a class tree, where the root is divided into subclasses until leaf classes are reached.<sup>1</sup> In this thesis, the term “taxonomy” is used to refer to all kinds

---

<sup>1</sup>The Colon Classification is a noteworthy exception.

of hierarchical classification systems that employ such a class tree. In the following, we present a framework for the unsupervised composition and maintenance of taxonomies for corpora in digital libraries.

### 7.1.1 Keyqueries as Taxonomy Classes

There are two principal approaches to compose a taxonomy for a given set of documents: (1) applying a hierarchical clustering algorithm to the documents and labeling the resulting clusters, and (2) compiling a set of candidate classes with labels and document assignments, and composing a taxonomy for the given documents from this candidate pool. With respect to the first option, the cluster labeling problem prevents, as pointed out in Chapter 5, an effective application to our use cases. We thus turn to the second option—dynamic taxonomy composition with a given set of classes. To obtain a set of classes with labels and document assignments, we identify the library’s query-based search engine as a powerful and flexible choice. We exploit the “wisdom” of the search engine by taking as classes the set of queries that can be formulated on the basis of the search engine’s vocabulary, and as class members their respective search results. In other words, we apply the keyquery concept to dynamic taxonomy composition.

The use of the library’s search engine as the source for classes has two big advantages compared to the use of other knowledge bases, but also one caveat that has to be handled in a reasonable way.

The first advantage is that both systems, the search engine and the taxonomy composition framework, are always in sync. Whenever the search engine indexes a new corpus document, the taxonomy composition will instantly feature the new document without the need for an additional integration process. If a document introduces a new topic, the set of classes is dynamically extended with the new topic. In this respect, we can think of the search engine as an implicit tagging mechanism which tags each document with its keyqueries. This automated tagging mechanism introduces a great deal of flexibility to the library management. Since the documents are automatically integrated and annotated with classes, human intervention can concentrate on documents for which no reasonable taxonomy class can be determined. In those cases, library experts can formulate appropriate classes, assign them to the documents, and force a re-indexing to commit the changes to the search engine (and hence to the taxonomy composition framework).

The second advantage of our approach is that search engine theory provides a well-defined concept for the integration of multiple knowledge re-

sources into a common representation, as well as for the combination of queries to a joint class: retrieval models. As introduced in Chapter 2, a retrieval model is the formalization of a linguistic theory that describes how to quantify the relevance of a document for a query, or, in this case, for a class. In sophisticated search engines, hundreds of features influence the relevance computation, and machine learning is used to find the optimum feature weights (Baeza-Yates and Ribeiro-Neto, 2011). There is an enormous amount of research that has been dedicated to the development of retrieval models for search engines, and this knowledge is now at our disposal. For instance, a retrieval model could be employed which indexes the existing meta-data about the documents with learned feature weights (Robertson et al., 2004). The retrieval model could itself classify the documents into a hierarchical classification system (Koller and Sahami, 1997), or tag the documents according to a vocabulary of subject headings (Tuarob et al., 2013). Even further, sophisticated retrieval models employ linguistic resources and thesauri like WordNet<sup>2</sup> or lexical taxonomies like Probbase<sup>3</sup> to infer further classes through synonym and hypernym relationships (Liu et al., 2012).

Though the presented advantages are appealing, there is one issue with the use of search engines in taxonomy generation. For a reasonable use of queries and their search results as classes, inconsistencies with the common perception of classes in a library classification system have to be avoided. To this end, in Section 7.3, we introduce five constraints for the dynamic composition of taxonomies on the basis of queries. Two class label constraints ensure that the queries used have the look and feel of conventional class labels, one class assignment constraint ensures that only those documents are assigned to a class for which relevance is substantial, and two class composition constraints, which control the taxonomy’s structure and introduce the notion of class generality.

### 7.1.2 Taxonomy Composition

Let  $D_u$  denote a set of documents and  $Q$  denote a set of candidate queries that adhere to the five constraints. The problem of taxonomy composition for  $D_u$  using classes from  $Q$  can then be stated as an optimization problem that has to be solved iteratively until leaf-classes are reached:

---

<sup>2</sup><http://wordnet.princeton.edu/>

<sup>3</sup><http://research.microsoft.com/en-us/projects/probase/>

$$\begin{aligned}
& \text{maximize} && \sum_{i=1}^{|Q|} x_i |D_i \cap D_u| \\
& \text{subject to} && \sum_{i=1}^{|Q|} x_i \leq k, \\
& \text{where} && x_i \in \{0, 1\}.
\end{aligned}$$

The objective is to find in each iteration the  $k$ -subset of  $Q$  that maximizes the recall with respect to  $D_u$ , subject to the constraint that the maximum fan-out  $k$  is never exceeded. The sets  $D_i$  in the formula refer to the documents which are assigned to the class  $Q_i$ , and the  $x_i$  are indicator variables where a value of one denotes that class  $Q_i$  is included in a particular  $k$ -subset. In Section 7.3.2, we present a time efficient greedy set-cover algorithm for this optimization problem.

In Section 7.4, we report on a series of experiments that demonstrate the feasibility of our approach for a digital library of 30,000 scientific papers. In the first experiment, we study the runtime characteristics of our approach by simulating a library which grows over the years. Our results show, that our approach can serve as an online library tool that dynamically composes a taxonomy for initial library subsets up to sizes of several thousand documents. In the second experiment, a taxonomy is computed for the whole library, and its profile is compared to a hand-crafted reference taxonomy for the library. In a third experiment, the effectiveness of dynamic taxonomy composition for document recommendation is studied. Compared to the reference taxonomy, our approach yields much higher recommendation precision while keeping recall at an acceptable rate. Finally, we present the dynamic taxonomy that we computed for the CORE dataset.

Our scientific contributions are fourfold. (1) We suggest dynamic taxonomy composition for various real-life use cases of classification systems in digital humanities contexts. (2) We suggest the retrieval model of query-based search engines as a powerful and flexible implicit tagging mechanism that can be used to infer a rich set of diverse classes for the library documents (cf. Section 7.3). (3) We present a greedy set-cover algorithm for the iterative composition of query-based taxonomies for arbitrary library subsets (cf. Section 7.3.2). (4) We demonstrate the efficiency and effectiveness of our approach in a case study with 30,000 scientific papers (cf. Section 7.4).

Related scientific literature is discussed in the following section.

## 7.2 RELATED WORK

In this section, we review research related to our dynamic taxonomy composition framework from the broader spectrum of information systems that provide a combination of querying and browsing facilities to their users. We describe the various approaches, point out the major differences to our approach, and give references to the state of the art. For a general introduction to classification theory and historical background information, we refer the interested reader to Arlene Taylor’s “The Organization of Information” (Taylor and Joudrey, 2009). An excellent discourse about the Web 2.0 and its implications for digital libraries is given by David Weinberger in his book “Everything is Miscellaneous” (Weinberger, 2007).

The spectrum of information systems we consider is spanned by search engines on the one end, and hierarchical classification systems on the other. Between these boundaries, there are extensions of the systems at either extreme that add browsing or querying facilities to their basis. In the following paragraphs, we traverse the spectrum of information systems from systems conceptually closer to query-based search engines to those that are closer to classification systems.

*Diversified Query Suggestions.* In most modern search engines for the Web, the user is provided a list of up to ten query suggestions (or autocompletions) while typing a query into the search box. The query suggestions are usually mined from the search engine’s query log (Wikipedia topics constitute an alternative source (Hu et al., 2013)), and contain the most popular queries that are syntactically similar to the entered query artifact. To increase the probability of a relevant suggestion, diversification of query suggestions has been proposed (Hu et al., 2013; Ma et al., 2010; Song et al., 2011). The aim of diversification is to compose a list of syntactically similar queries that are semantically diverse. Diversified query suggestions can hence be interpreted as a dynamically composed, flat classification system for the given query artifact. In contrast to our approach, candidate classes are not derived from the query artifact’s search results, but rather from its character sequence. Because of this difference, the two systems support different use cases and complement each other: Query suggestions help users find an initial query, while our approach could provide a hierarchical structuring of the results once retrieved.

*Diversified Query Recommendation.* Query recommendation refers to the task of providing, next to the search results, a list of queries that are semantically equivalent to the user’s submitted query. Semantic equivalence is typically measured by click through logs. From the set of candidate recommenda-

tions, the most popular (= most frequent in the query log) are chosen for presentation (Baeza-Yates et al., 2004). The rationale behind query recommendation is to educate the user about the “better” or more common ways of expressing an information need. As for query suggestions, diversification has been proposed also for query recommendations. Li et al. cluster queries into so-called query concepts (i.e., query groups of high semantic equivalence), and draw a single query from each concept as candidate recommendation to compile a diversified, non-redundant query recommendation list (Li et al., 2012). This kind of diversification is most useful for ambiguous queries, where diverse recommendations allow users to pick a query that specifies their information need more precisely. The problem of dynamic taxonomy composition is related, in that it can also be used to add a list of queries next to the search results. However, dynamic taxonomy composition strives for queries dividing the search results into more specific subclasses, and hence is best applied to non-ambiguous queries for the sake of exploration, recommendation, and serendipities.

*Query-Based Taxonomy Composition.* The task of composing a taxonomy for a given set of documents falls into two principal steps: the acquisition of classes and their arrangement to a taxonomy. For the first step, the majority of existing taxonomy composition approaches employ keyphrases extracted from the documents as the class set (Liu et al., 2012; Navigli et al., 2011; Poon and Domingos, 2010). Liu et al. submit individual keyphrases as queries to a search engine, but solely for the purpose of building a bag-of-words representation of the keyphrase classes (Liu et al., 2012). The use of queries as classes—as we propose it—can be regarded as an extension to the keyphrase strategy. Our framework comes with two advantages: First, the combination of keyphrases to joint classes has a well-defined interpretation: the underlying conjunctive query. Second, employing a topic model like ESA (Gabrilovich and Markovitch, 2007), it is possible to find keyphrases that do not explicitly appear in a document’s text.

Queries have been considered as classes by Chuang and Chien (2002) and Bonchi et al. (2008). In their work on query clustering, Chuang et al. use the query log of a search engine for class acquisition, and apply a multi-branch hierarchical clustering algorithm to compose a taxonomy for the queries. We see the clustering approach as a bottom-up alternative for the greedy set-cover algorithm we employ for taxonomy composition. A bottom-up strategy is efficient, when the task is to include the whole class set into the taxonomy. However, in our scenario, where only a subset of the classes is chosen for the taxonomy, a top-down approach like ours is much more efficient. A greedy set-cover algorithm similar to ours is employed by

Bonchi et al. in their work on query decomposition, but with different constraints. While we strive for an iterative decomposition into gradually more specific queries on every level, the goal of Bonchi et al.’s query decomposition is a flat classification system of queries that have small overlap and maximum precision, irrespective of their generality. A further difference is that topical query decomposition starts from an initial query, while our approach starts from a given library subset for which a single query might not exist.

As an alternative for the second principal step—taxonomy composition—Navigli et al. (2011) mine hypernym relationships from the documents’ texts to build up a graph of class relationships, and prune the graph to a tree in a post-processing step. Thanks to the flexibility of the retrieval models behind query-based search engines, our framework can integrate an approach like that: the retrieval model can simply assign to the library documents all known hypernym relationships. A query with a hypernym then retrieves the appropriate documents and can also be used by our greedy set-cover algorithm.

*Document Clustering.* Document clustering is the unsupervised complement to classification. It aims at grouping similar documents into one cluster. The objective of document clustering hence aligns well with the objective of dynamic taxonomy composition. The most well known approach to document clustering is Scatter/Gather (Cutting et al., 1992), which, like our approach, creates a cluster (class) hierarchy in an iterative process. In the Scatter phase of the two-step approach, documents are clustered by some cluster algorithm, and the resulting clustering is presented to the user. The user then selects the clusters of interest in the Gather phase, and the documents that belong to the selected clusters are used as input for the next iteration. This unsupervised discovery of classes has the potential to reveal semantic concepts that have no representation in the query space. But there are a couple of drawbacks often preventing document clustering from being effective. The most severe issue of clustering is that the problem of finding a meaningful label, from which the major cluster characteristics can be inferred, is not sufficiently solved (Stein et al., 2011).

Recent document clustering approaches, as pointed out in Chapter 5, thus turn to monothetic clustering, producing cluster labels from the terms that appear in each document of a cluster. However, we argue that this approach trades the discovery of sophisticated concepts against an ineffectual use of queries as labels. Furthermore, clustering acts only locally on the given document set and is not directly connected to the underlying search engine. Hence, no additional library documents can be efficiently retrieved

for a cluster, preventing complementation and serendipities. Finally, clustering constitutes a shift in the interaction paradigm that is hard to communicate to the user: Clicking on a cluster label reveals documents different from a search for the cluster label. Our proposed dynamic taxonomy composition sidesteps these issues by considering only classes with a representation in the query space. We also suggest the implementation of a feedback loop into our approach that drives the (manual) annotation of documents for which no proper taxonomy class can be found.

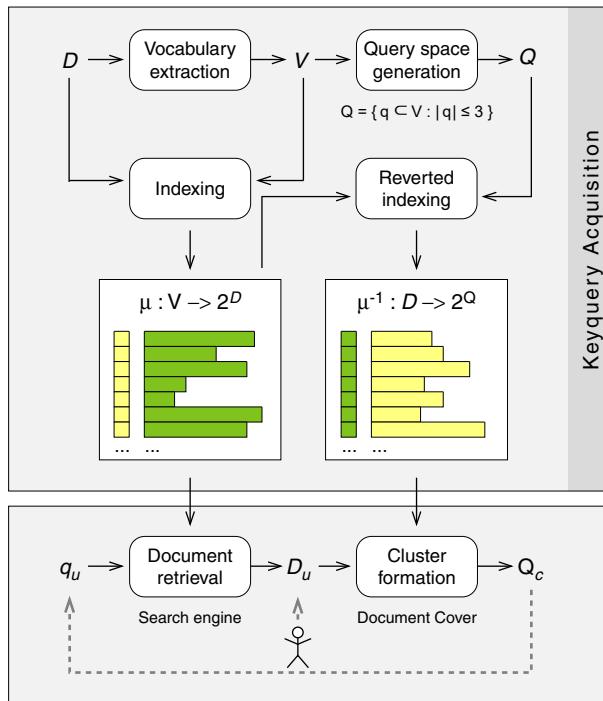
### 7.3 APPROACH

In this section, we formalize our approach and explain how to compute a keyquery-based taxonomy for a set of library documents. Our proposed strategy is outlined in Figure 7.1. The upper part shows the class acquisition process: tagging the library documents with their keyqueries. This process runs offline and generates the queries used as candidate classes in taxonomy composition. The lower part of the figure illustrates the online taxonomy composition process. Each level of the taxonomy is computed by solving the optimization problem introduced in Section 7.1.2. The computation of subsequent taxonomy levels is triggered by the user, who selects a query from the current taxonomy level. We present the two parallel processes in detail in the following, including five constraints that control the various subroutines.

#### 7.3.1 Class Acquisition

To compose a taxonomy for a set of library documents, a set of classes to draw from has to be acquired. Our approach derives this class set from the query-based search engine, a component commonly available in modern digital libraries (Henry, 2012). To facilitate the retrieval of documents through queries, the search engine extracts the vocabulary  $V$  from the library documents  $D$  with respect to  $V$  (left column in Figure 7.1). In the resulting data structure, the inverted index  $\mu$ , each term in  $V$  constitutes a key pointing to a postlist that contains all documents with a non-zero retrieval score for that term. If a query  $q \subset V$  is submitted to the search engine, the search engine takes each term from  $q$ , collects the documents in the respective postlists, and ranks them according to their aggregated retrieval scores (both sum and product are common for the aggregation).

For the computation of retrieval scores, a large variety of retrieval models



**FIGURE 7.1: Strategy overview:** How to for composing dynamic taxonomies for arbitrary library subsets. The upper part of the figure shows the offline component that continuously determines all keyqueries for the library documents and stores them in the reverted index. The lower part of the figure shows the online component that iteratively composes taxonomy levels  $Q_c$  for a user provided library subset  $D_u$ . The composition of sublevels is triggered by the user who selects a query  $q_u$  from the current taxonomy level  $Q_c$

are available (Manning et al., 2008). As discussed in Chapter 2, these range from simple bag-of-words models like Tf-Idf or BM25 to more sophisticated topic models like LDA (latent topics) or ESA (explicit topics). Note that the choice of the retrieval model is crucial to our approach, and has a large impact on the quality of the composed taxonomies.

To acquire a set of classes for taxonomy composition from the search engine, we propose to generate a suitable subset  $Q$  of all queries that can be formulated from  $V$ , submit these queries to the search engine, and tag the returned documents with the respective query as class (right column of Figure 7.1). An efficient access to the classes (or queries) for a document is provided by a reverted index  $\mu^{-1}$ . In the reverted index, each library document serves as the key for a postlist that contains all queries for which the document is relevant (Pickens et al., 2010).

The queries that can be formulated on the basis of  $V$  theoretically com-

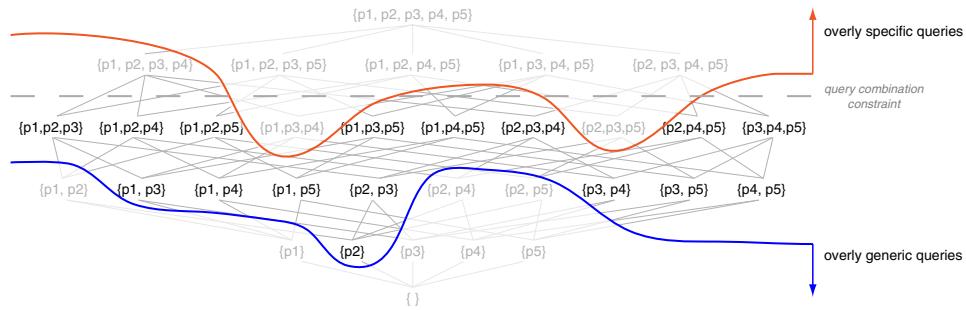
prise all  $2^{|V|}$  subsets of  $V$ . To prune the query set, we consider only those queries as classes that meet the common perception of class labels in conventional classification systems. Specifically, we introduce the following two class label constraints, which are adopted from the search space reduction strategies presented in Section 3.3:

1. *Part-of-speech constraint.* The parts of speech used for class labels in most classification systems are noun phrases (e.g., “information system” or “topic model”). We hence add to  $Q$  only queries that are combinations of noun-phrases.
2. *Query combination constraint.* A class label is usually made up of one noun phrase or two noun phrases joined by a conjunction (e.g., “digital libraries and archives”). In extreme cases, three phrases are joined. We hence discard from  $Q$  all queries that consist of more than three noun phrases.

Given the class label constraints, an upper bound on the number of queries can be stated as  $|Q| < \sum_{k=1}^3 \binom{|V|}{k}$ . Note that the actual number of queries will be lower, since the fraction of  $k$ -phrase queries that return a relevant document decreases for larger  $k$ . Taking up the view of the query space as the hypercube of phrase combinations, which is illustrated again in Figure 7.2, each level  $l$  of the hypercube contains all  $l$ -combinations of phrases from  $V$ . To populate the reverted index efficiently, recall that we explore the query space in a level-wise manner using the Apriori algorithm, starting with all single-phrase queries. For each subsequent level  $l$ , we consider only those  $l$ -phrase queries that can be generated by adding a phrase from  $V$  to an  $(l - 1)$ -phrase query that returns more than one result. This way, the query space is pruned whenever a query reaches the maximum degree of specificity.

In addition to the class label constraints, we introduce a class assignment constraint controlling the fraction of a query’s search results that are tagged with the query in the reverted index. The class assignment constraint ensures that the queries in a document’s postlist are all keyqueries for the document (i.e., that the document is returned in the top results):

3. *Relevance constraint.* Users expect that documents assigned to a class in a taxonomy contain content *about* the subject represented by the class label, and do not just mention the class label in the text. However, determining at which position in a query result list the mere occurrence of the query phrases turns into “aboutness” is not trivial. Most



**FIGURE 7.2:** The search space for a five-phrase vocabulary  $P = \{p_1, \dots, p_5\}$ . The two boundaries shown divide the search space into three subspaces of queries returning too few, the desired number, or too many documents (from top to bottom). The additional dashed line illustrates the query combination constraint prohibiting the use of more than three phrases in a query.

retrieval models assign retrieval scores greater than zero already if a document contains any of the query phrases. To facilitate a binary decision upon the membership of a document to a class, we suggest the introduction of a retrieval score threshold: The retrieval score must exceed the score of a fictitious document with average length and average click rate that contains all query terms once.

The use of a retrieval score threshold may appear a bit old-fashioned compared to the use of click through information from the search engine's query log. But we argue that our constraint would account for users' click behavior if the employed search engine takes click behavior into account for retrieval score computation. Furthermore, a score threshold circumvents the cold-start problem inherent to any pure query log-based method (a query log is empty at first).

Once the postlists of the reverted index are filled with the library documents' keyqueries, the online process can compose dynamic taxonomies for user-provided library subsets. Before we turn to the online process, however, we emphasize that the reverted index is not a static data structure. Whenever the search engine indexes a new document, the reverted index allocates a new postlist for the document and computes the respective keyqueries. If the new document adds new noun phrases to  $V$ , the query space  $Q$  is extended accordingly, and the search results for the new queries are incorporated into the reverted index. Through this tight interplay with the search engine, our taxonomy composition approach always reflects the current state of the library.

### 7.3.2 Taxonomy Composition

Whenever a user turns to the system with a set of library documents  $D_u$ , the taxonomy composition process responds with a set of queries  $Q_c$  that represents the first level of the taxonomy for  $D_u$ . If the user selects a query from  $Q_c$  for further subdivision, the same process is repeated for that query's result set as the new  $D_u$ . This gives rise to an iterative process, where the taxonomy composition adapts interactively to the user's information need.

In each iteration, we strive for a subdivision into gradually more specific complementation classes which cover all the documents  $D_u$  and contain further topically similar library documents. To reach these goals, we introduce the following class composition constraints:

4. *Fan-Out Constraint.* In conventional classification systems, the maximum number  $k$  of subclasses into which a class can fall is about ten. For example, every internal class in the Dewey Decimal Classification System is divided into exactly ten subclasses. For our intended presentation of the composed taxonomy in a grid on a computer screen, we argue that twelve subclasses constitutes a more appropriate choice, and in addition allots more space for serendipity classes. Twelve subclasses can be placed evenly in a grid of one, two, three, four, six, and twelve columns, and hence allows a space efficient presentation on many devices from mobile to large desktop screens. Note that for the same reason, popular responsive design frameworks like Bootstrap use a twelve-column grid.<sup>4</sup>
5. *Level of Generality Constraint.* One of the design goals for hierarchical classification systems is to compose a balanced taxonomy where each level constitutes a gradual specification of the parent classes (Taylor and Joudrey, 2009). Given a maximum fan-out of  $k$ , a balanced subdivision distributes the documents  $D$  of a parent class into disjunct subclasses of size  $t = |D|/k$ . Since the division into subclasses must be backed up by a semantic justification, the target subclass size  $t$  is only a rough reference point even in conventional classification systems. We thus add a margin  $m$  to  $t$ , and consider as candidates for the subclasses of a parent class all queries in  $Q$  with a search result size in the interval  $[t - m, t + m]$ . As a default, we suggest a margin of 50% of the target size.

The above level of generality constraint reduces the set of candidate queries that can be taken as subclasses in each iteration of the composition process.

---

<sup>4</sup><http://getbootstrap.com>

In Figure 7.2, the boundaries drawn into the query space indicate the separation of valid candidate queries from overly specific and overly generic ones.

To obtain the optimal set of subclasses in each iteration, we employ *Greedy Document Cover* given in Pseudocode 5. Using the reverted index, the algorithm first compiles the set of candidate queries  $Q_u$ . Each query from  $Q_u$  returns at least one of the documents from  $D_u$ , and satisfies the level of generality constraint.

---

**Pseudocode 5** Greedy Document Cover

---

**Input:** reverted index  $\mu^{-1}$ , document set  $D_u$ ,  
number of subclasses  $k$ ,  
target class size  $t$ , margin  $m$   
**Output:** taxonomy sublevel  $Q_c$

```
// Collect candidate queries for  $D_u$ :
1:  $Q_u \leftarrow \bigcup_{d \in D_u} \{ q \mid q \in \mu^{-1}(d), |D_q| \in [t \pm m] \}$ 

// Build a document cover  $D_c$  for  $D_u$ :
2:  $Q_c \leftarrow \emptyset$ 
3:  $D_c \leftarrow \emptyset$ 
4: while  $|Q_c| < k \wedge Q_u \neq \emptyset$  do
5:    $q^* \leftarrow \underset{q \in Q_u}{\operatorname{argmax}} \{|D_q \cap (D_u \setminus D_c)|\}$ 
6:    $Q_u \leftarrow Q_u \setminus q^*$ 
7:    $Q_c \leftarrow Q_c \cup q^*$ 
8:    $D_c \leftarrow D_c \cup D_q$ 
9: return  $Q_c$ 
```

---

The algorithm then finds an approximate solution for the optimization problem stated in Section 7.1.2 using a greedy set cover strategy (Vazirani, 2001). It initializes the sets  $D_c$  (the document cover) and  $Q_c$  (the queries representing the subclasses for  $D_u$ ) to the empty set. For up to  $k$  iterations, it then selects the query  $q^*$  from  $Q_u$  that covers the maximum number of documents not yet covered by previous queries, and adds it to the subclasses  $Q_c$ . This process is repeated until the maximum fan-out is reached, or no more candidate queries are available.

The queries in  $Q_c$ , when submitted to the search engine, return a family of subsets of  $D_u$ . These can be displayed to the user as the classification of the search results. Note that, depending on  $D_u$  (and the corresponding queries in the reverted index), it may not be possible to construct a cover of the entire set  $D_u$ . In this case, the remaining documents in  $D_u \setminus D_c$  may be added to a “miscellaneous” class. As previously noted, the algorithm is in-

tended to be applied iteratively to subsets of  $D_u$ , including the query result sets for the classes in  $Q_c$ . Since any label in the taxonomy is a valid query, retrieving the result set of a class for further partitioning is a simple matter of submitting the class label as a query to the search engine (illustrated through the arrow at the bottom of Figure 7.1).

### 7.3.3 Discussion

The constraints on candidate queries ensure that the set cover algorithm produces a reasonably well-balanced cover; at any level of the taxonomy, a given class can at worst be twice as large as any other. The reverted index must contain a sufficient number of queries satisfying the level of generality constraint for the documents in  $D_u$ . Otherwise, the dynamic classification cannot cover all of  $D_u$ , and will contain a large “miscellaneous” class.

Conversely, if the reverted index contains several disjoint queries near the upper bound of the required level of generality, the dynamic taxonomy may cover all of  $D_u$  in as few as  $\frac{2}{3}k$  iterations. In this case, additional queries can be added to  $Q_c$  to form serendipity classes. As a first basic approach, we pick serendipity classes randomly from the remaining candidate queries in  $Q_u$ .

For the online part of our system, it is prudent to consider the run-time complexity since the response time of the digital library retrieval system needs to be kept within reasonable limits for the sake of user experience. Compiling the set  $Q_u$  of candidate queries requires  $O(|D_u|)$  accesses to the reverted index, and at most  $O(|Q_u|)$  operations to select from the results those queries that fit the required specificity bounds. In order to solve the actual document covering, the algorithm requires  $O(|Q_u|)$  queries to the search engine (independently of the number of iterations), as well as  $O(|Q_u|)$  set-intersection operations per iteration.

Considering that this procedure is repeated iteratively for all subclasses in the dynamic taxonomy, the amount of computation may seem prohibitive for guaranteeing responsiveness at query time. However, only the initial document covering of  $D_u$  needs to be computed at query time. While the user studies the first taxonomy level, the computation of subsequent taxonomy levels can be initiated and the respective subclasses be cached. This introduces a tradeoff between storage and processing requirements at preprocessing versus at query time, and middle-ground solutions are conceivable. For instance, analysis of the query logs for the digital library’s search engine can motivate long-term caching for frequently used taxonomy classes, while rarely issued classes are computed online.

In this respect, taxonomy composition can be framed as an instance of a *slow search* problem, a class of search scenarios where traditional speed requirements are relaxed in favor of a high quality search experience (Teevan et al., 2013). Due to the top-down nature of the document cover algorithm, the taxonomy composition follows the user’s exploration of the information need, generating class subdivisions on demand.

#### 7.4 EVALUATION

Due to the combinatorial complexity of the search space, dynamic taxonomy composition is clearly a big data problem. Even for relatively small collections, the size of the query space quickly grows to a magnitude that requires the use of distributed processing techniques in order to be at all feasible. For the experiments reported in this section, we employ the MapReduce programming model (Dean and Ghemawat, 2008), as implemented by the Apache Hadoop software library.<sup>5</sup> Our experiments run on a 40-node cluster of off-the-shelf desktop machines, each equipped with four 2.4GHz CPU cores and 8–16 GB of RAM.

In order to evaluate the utility and feasibility of our proposed dynamic classification system, we first apply our approach to a dataset of 30,000 scientific papers—manually collected from the ACM digital library over the course of several years—which we refer to as Webis-CSP-Corpus. The corpus contains various metadata, including conference, year of publication, and the classification according to the ACM CCS taxonomy. With our experiments, we explore the following research questions:

1. What additional processing requirements are imposed by the dynamic classification system, both online and offline?
2. How useful are dynamic taxonomies in fulfilling a profiling-oriented information need?
3. How well do dynamic taxonomies support serendipity search or document recommendation? How do they compare to human-made taxonomies in this regard?

As a preprocessing step, we extract a set of keyphrases from each document in the collection, using the approach proposed by Barker and Cornacchia (2000), in which prominent head noun phrases function as keyphrases for a document. We use the log-linear part-of-speech tagger implementation by

---

<sup>5</sup><https://hadoop.apache.org/>

Toutanova et al. (2003) to identify candidate noun phrases. For the purpose of our experiments, the extracted keyphrases form the vocabulary  $V$  which gives rise to the queries in the search space. While we do not aim to replace traditional static classification systems entirely, they serve as baseline for the experiments described below.

### 7.4.1 Processing requirements

To map out the processing requirements for dynamic taxonomy composition, we generate the query space on corpus subsets of various sizes, and then compute a document cover for the set of all documents in the collection. For each year from 1990 through 2010, we take from the Webis-CSP-Corpus those documents published that year or earlier. Table 7.1 shows the number of documents, the number of keyphrases in the vocabulary and the resulting query space sizes for a sample of years, as well as the corresponding offline and online processing times. Note that the figures reported for the size of the query space  $Q$  refer to the pruned search space shown in Figure 7.2—the set of all possible queries constructed from the vocabulary is larger by several orders of magnitude.

As discussed in Section 7.3, offline processing involves acquiring a set of candidate queries from which to compose taxonomy classes. We implement class acquisition as a processing pipeline of MapReduce jobs which evaluate the Okapi BM25 retrieval function. By computing document relevance scores for the entire collection in parallel, this approach facilitates the large number of query evaluations needed to compile  $Q$ . If our system were implemented in a digital library setting, this type of batch processing could easily be integrated with the library’s other data preprocessing efforts. As the penultimate column of Table 7.1 shows, even using our modestly-sized Hadoop cluster, the investment in processing time is manageable, and much smaller than the time required for a taxonomy built by human experts.

Digital library users would expect their search results to be classified and displayed in a timely fashion. This constrains the time available to compute the document cover for the initial query result set  $D_u$ . Since starting up a MapReduce job involves significant overhead, the document cover for user queries needs to be handled differently. For this part of the experiment, we use an implementation of the document cover algorithm running on a single machine. The two indexes required for the computation are accessed via an efficient inverted index implementation developed in-house. The numbers in the last column of Table 7.1 report the time needed to compute the document cover for the entire collection on a single 2.4GHz CPU core. The 27

**TABLE 7.1:** Query space sizes and processing times for corpus subsets of different magnitudes.

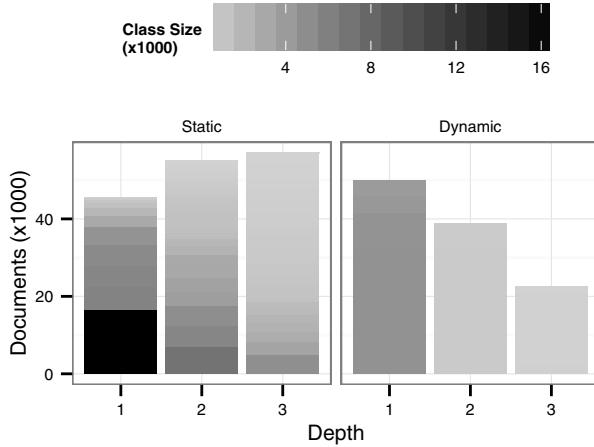
Year	Dataset sizes			Run time [h:m:s]	
	$D$	$V$	$Q$	$Q$	Cover
1990	4,784	3,670	174,605	6:17	4.9 s
1995	8,593	5,982	607,615	14:39	7.1 s
2000	12,901	8,431	1,677,217	35:21	11.3 s
2005	19,288	11,893	4,577,178	1:24:24	17.7 s
2010	28,950	15,807	13,516,167	3:09:00	27.2 s

seconds of processing time on the largest subset of the collection is arguably too long to guarantee a responsive user experience. However, the experiment represents a worst case, in that the document cover is computed for the entire collection. Whereas in the main use case for dynamic classification we envision—classification of search result sets—the set of documents to cover is much smaller. This impacts performance greatly, as the last column of Table 7.1 indicates. Each additional document to cover implies an access to the reverted index, yielding additional queries, each of which implies an additional access to the inverted index. In the reverted index for Webis-CSP-Corpus, each individual document is associated with an average of 10,000 queries, which supports the intuition that smaller subsets of the collection will be significantly less costly to process.

#### 7.4.2 Profiling

A user with a profiling-oriented information need (e.g., identifying all the classes of the papers from a specific conference) is best served by a classification system that subdivides the document set in a maximally informative way. We investigate a dynamic classification system generated for the documents in the Webis-CSP-Corpus, and compare it to the static, human-made ACM CCS classification system. Owed to its organic growth over time, we expect the static taxonomy to exhibit a lower information content than the dynamic classification system.

The ACM CCS taxonomy for the documents in the Webis-CSP-Corpus is three levels deep. The median number of subdivisions under a given CCS class is 11, 6 and 3, for the top, middle, and leaf level of the taxonomy. For the sake of comparability, we generate a dynamic taxonomy of the same depth, with the fan-out parameters set to the CCS median at each level. As we are comparing taxonomies for the entire corpus, the document set  $D_u$  input to the covering algorithm comprises the entire collection  $D$ .



**FIGURE 7.3: Distribution of class sizes in the human-made taxonomy, and a dynamic classification system generated from extracted keyphrases.**

Figure 7.3 shows the distribution of taxonomy class sizes for both the human-made and the dynamic classification system. The left chart shows taxonomy classes taken from the corpus documents' metadata. On the right, the classification system was generated by the document cover algorithm, using queries formulated from keyphrases extracted from the documents' text. In each chart, bar height shows the total number of documents covered per depth of the classification. Both classification systems assign more than one class label to some documents—hence the number of documents covered by the classification exceeds the total number of documents in the collection. The shading of the bars illustrates how class size is distributed among the branches of the taxonomy.

The figure demonstrates the high level of imbalance in the human-made taxonomy—especially on the highest level of classification. Here, the largest branch subsumes more than 16,000 documents, whereas any other class is at most one third that size. By contrast, the dynamic query-based classification system results in a taxonomy where the number of documents per class is uniform within any given level, and distinct across levels. Owed to the class size imbalance, the static taxonomy covers a much larger number of documents on the middle and leaf level. For instance, the static taxonomy contains a leaf class of comparable size to the top-level classes in the dynamic classification system.

To assess the usefulness of each classification system in a profiling-oriented task, we compute the entropy at each branch in the classification tree, as well as the number of redundant document-class assignments. Table 7.2 shows the results aggregated by hierarchy level. With entropy, we

**TABLE 7.2:** Comparison of entropy and redundancy for dynamic and human-made classification systems in the Webis-CSP-Corpus by hierarchy level.

Depth	Avg. Entropy	Classes per document				
		$\mu$	$\sigma$	Min	Median	Max
<b>Static</b>						
1	2.79	1.59	0.74	1	1	5
2	1.94	1.93	1.01	1	2	8
3	1.84	2.11	1.16	1	2	10
<b>Dynamic</b>						
1	3.46	1.93	1.01	1	2	7
2	2.58	2.19	1.47	1	2	13
3	1.58	1.93	1.32	1	1	19

measure the information content in bits of the class assignments, averaged over each level of classification. Entropy is measured at each branch in the taxonomy and corresponds to the probability of a randomly sampled document belonging to a given subclass below the branch—it is maximal if all subclasses are the same size. The dynamic taxonomy achieves higher entropy than the human-made classification system on the upper two levels of the hierarchy, whereas the human-made classification system carries more information at the leaf level. Both taxonomies are similar with respect to the number of classes per document. While outliers in the dynamic classification system tend to have higher redundancy than in the static one, the average document is assigned to two classes at each level of classification in both cases.

These results demonstrate that it is possible to generate a dynamic taxonomy with desirable structural properties, using only automatically extracted head noun phrases as class labels. The dynamic taxonomy outperforms the human-made reference taxonomy especially with respect to the information content of the higher-level subdivisions of the document set. However, the human-made taxonomy is superior in other regards, and we expect the best performance using a retrieval model that combines information from both sources—our described idea of backing up a traditional classification with a dynamic query-based component.

#### 7.4.3 Document Recommendation

In a final inquiry of the ACM dataset, we investigate the suitability of dynamic classification systems for document recommendation. In this scenario, the user has already retrieved a set of documents relevant to her in-

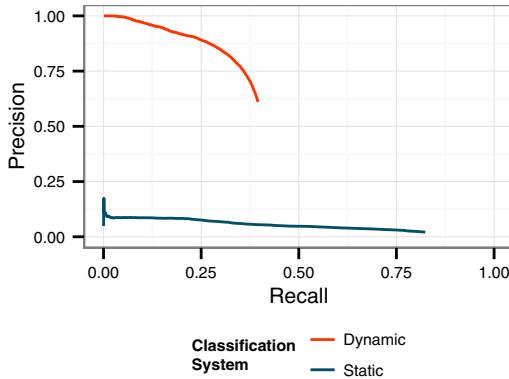


FIGURE 7.4: Average precision-recall curve, showing the performance of the dynamic and human-made classification systems at the document-recommendation task.

formation need. The retrieval system’s task is to present related relevant documents, helping the user uncover new aspects of the subject under investigation. Dynamic classification systems, being tailored to the user’s specific information need, should perform considerably better at this task than a static classification system that has to take the entire document collection into account.

To test this hypothesis, we perform the following experiment: First, we sample a random query  $q$  from the reverted index. This query represents the overall information need. Second, we randomly partition the result set of  $q$  into two subsets  $D^+$  and  $D^-$  of equal size, where  $D^+$  represents the known aspect of the information need, and  $D^-$  represents the unknown aspect (i.e., the set of documents that the retrieval system should recommend). Third, we construct a dynamic classification system for  $D^+$  using queries from  $Q \setminus q$ . Fourth, we measure the number of documents from  $D^-$  that are retrievable via the leaf classes of the classification system.

We model a user who scans all leaf classes for relevant documents. For the purpose of this experiment, the hypothetical user pursues an optimal strategy: in each step, the user examines the class that contains the maximum number of previously unseen documents from  $D^-$ . The user stops when all retrievable documents from  $D^-$  have been considered. For each taxonomy class the user considers, we measure the cumulative recall for hidden relevant documents, that is, the proportion of documents from  $D^-$  that the user has retrieved so far. We also measure the precision with respect to  $D^-$  at each step. In total, we perform forty runs of the above experiment, with different initial information needs. To put the performance of the dynamic classification system into context, we perform the same experiment

using the leaf classes of the static CCS taxonomy, using the same optimal search strategy as above.

Figure 7.4 shows a set of precision-recall curves depicting the performance averaged over all forty experiments, comparing the static and dynamic classification system. As the hypothetical user examines taxonomy classes, each curve plots the fraction of all documents in  $D^-$  that the user has seen (recall) against the fraction of previously unseen documents from  $D^-$  out of all retrieved documents (precision). The curves end at the point of maximum recall, where no further documents from  $D^-$  can be found in the taxonomy. Due to the much larger leaf classes, the human-made taxonomy achieves better maximum recall, at the cost of much lower precision. In the dynamic taxonomy, nearly all documents are relevant to the user’s information need, and out of these, close to half are related to the unknown aspect of the information need. Table 7.3 shows the distribution of maximum recall for  $D^-$  and precision at maximum recall, over all forty experiments. In other words, the numbers in the table show the distribution for the rightmost point in the precision recall curve for the hidden relevant documents. The table demonstrates a larger trend: while the large leaf classes in the static taxonomy allow for consistently higher recall, the ratio of relevant documents found is much better for the dynamic classification system.

#### 7.4.4 Application to CORE

In a follow-up evaluation published in Völske et al. (2014), we apply our approach to the CORE dataset. The CORE dataset provides a convenient real-world testbed for keyquery-based taxonomy composition. Out of the 850 000 total articles in the data dump, we limit our analysis to articles where English full-text is available. To further improve the quality of the dataset,

**TABLE 7.3: Distribution of maximum recall and precision at maximum recall for the human-made and the dynamic taxonomy in the document recommendation task.**

	$\mu$	$\sigma$	Min	Median	Max
<b>Static</b>					
Max. Recall	0.82	0.04	0.76	0.84	0.89
Max. Precision	0.02	0.01	0.01	0.02	0.05
<b>Dynamic</b>					
Max. Recall	0.40	0.11	0.26	0.34	0.62
Max. Precision	0.61	0.08	0.47	0.63	0.73

we try to filter out articles with parsing errors by discarding those whose average word length is more than two standard deviations from the mean of 6.5 characters. These filtering steps result in a set of about 450 000 articles, which form the foundation for the experiment presented below.

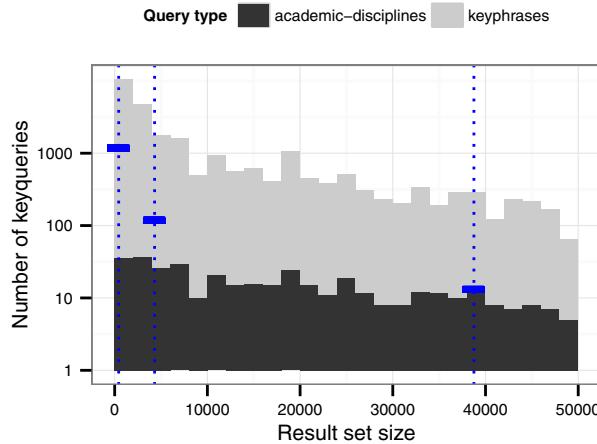
### KEYQUERY ACQUISITION

The major design decision that has to be made in the keyquery acquisition phase is the selection of a retrieval model for the reference search engine. Our retrieval model of choice for CORE is the explicit topic model ESA (Gabrilovich and Markovitch, 2007), because it possesses, in comparison to bag-of-words or latent topic models, outstanding properties for our task.

*Domain Knowledge.* ESA represents documents via their similarity to a set of Wikipedia articles. Since we know that the CORE dataset consists of academic papers from various disciplines, we can incorporate this domain knowledge by choosing the Wikipedia articles for ESA accordingly. We select all articles referenced on Wikipedia’s “List of Academic Disciplines” page, and we further enrich the set by articles whose titles appear in more than 100 CORE papers. In total, we extract 4 830 articles from the English Wikipedia dump for February 2, 2014.

*Query Representation.* The titles of the selected articles also form the vocabulary  $V$ , i.e. the query space  $Q$  is set equal to the topic space of ESA, which has an interesting effect when it comes to answering queries. Typically, to answer queries under a topic model, queries are treated as (tiny) documents in the vocabulary space. To identify relevant documents, the query is transferred from the vocabulary space to the topic space first, and its similarity to documents in the topic space is then used as relevance metric. The explicit topics of ESA allow us to pursue an alternative relevance assessment strategy which aligns well with the idea of queries as higher level concepts: Queries are treated as documents that already reside in the topic space, and relevance scores can be computed directly. This works since all query terms have—by vocabulary design—a well-defined representation in the topic space of ESA.

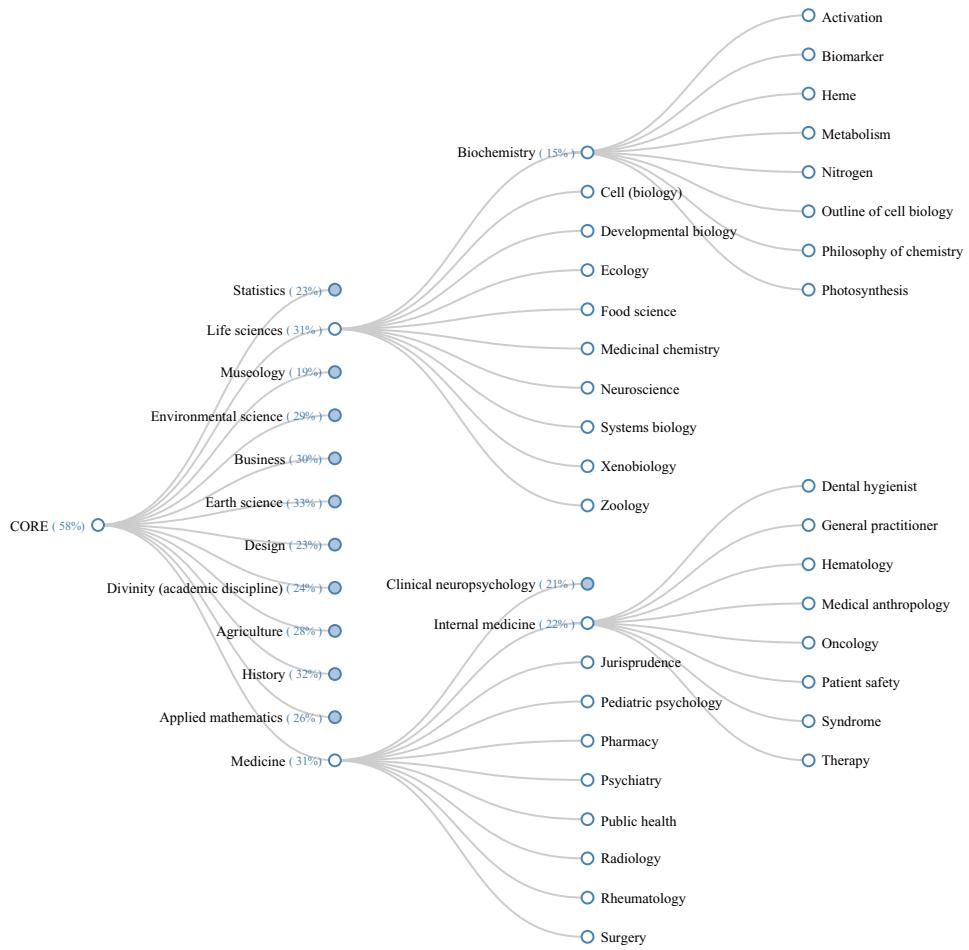
*Implicit Relevance.* As a final argument for the use of ESA, unlike bag-of-words models like TfIdf, ESA is capable of finding relevant documents that do not contain the query terms explicitly. Especially for general terms—like those one would expect on the first level of a taxonomy—this is an important property, since they are known to appear rarely explicitly in relevant documents (Sacco and Tzitzikas, 2009). With ESA, to be relevant for a query



**FIGURE 7.5:** Distribution of query result set sizes in the reverted index for the CORE dataset. The overlaid vertical lines show the optimal result set sizes for the generated taxonomy of depth 3; horizontal crossbars show the minimum number of queries needed at a given level

term, a document only needs to be similar to the respective Wikipedia article. To compute the similarity between documents and Wikipedia articles, we represent each document and each Wikipedia article under a BM25 retrieval model, and compute the dot product for each document-article pair. Given 450 000 CORE documents and 4 830 Wikipedia articles, this yields an 450 000-by-4 830 similarity matrix, where each of the column vectors contains the similarity distribution for one Wikipedia topic. Since these document-topic similarity distributions tend to be long-tailed, with many documents achieving a low similarity score, the sparsification criterion we presented in Gollub and Stein (2010) is applied, setting similarities which are not significantly above a computed expected similarity score to zero. For every CORE document, we consider the set of Wikipedia articles with non-zero similarity scores to be its keyqueries. We store the mapping  $\mu$  from each keyquery to documents in an inverted index. The mapping  $\mu^{-1}$  from each document to its keyqueries is stored in a reverted index for efficient access during cluster formation.

Figure 7.5 shows the frequency distribution of keyqueries in the reverted index. Given a fan-out parameter  $k = 12$  and a target depth of three for the taxonomy, keyquery-based taxonomy composition proceeds in a level-wise manner, first splitting the entire collection into 12 top-level classes, then incrementally subdividing subclasses up to the leaf level. For the chosen taxonomy parameters, the overlaid vertical bars in Figure 7.5 show the optimal result set sizes for the leaf, middle and top level, from left to right. The hori-



**FIGURE 7.6: Excerpt of a partial keyquery-based taxonomy generated for the CORE dataset. For each non-leaf class, the percentage of documents covered by its subtree is shown in brackets next to the class label.**

zontal cross bars superimposed on each level show the minimum number of queries needed at a given depth. Note that the “Academic Disciplines” topics alone suffice to fill out the top level of the taxonomy, whereas the lower levels require the additional articles selected from document keyphrases. As noted, the full set of Wikipedia topics forms the query space  $Q$  for the cluster-formation process.

#### TAXONOMY COMPOSITION

Composing the taxonomy with our MapReduce-based implementation of *Greedy Document Cover* for the entire CORE dataset on our 40-node cluster takes about 48 hours of wall-clock time. Taxonomy generation for an entire

digital library taking a long time is not a major issue—the taxonomy can be updated periodically by a nightly batch job. However, processing time is problematic in a live retrieval setting, where the document set returned by a user’s query is to be subdivided in real-time. As outlined above, we propose an on-demand generation of taxonomy levels as a possible solution: compute only those branches of the taxonomy that the user explores when surveying the search results.

## RESULTS

Figure 7.6 shows an excerpt of our taxonomy for the CORE dataset. The top level classes subdivide the collection with respect to the academic disciplines given as input with an average pairwise overlap of 12%. Through the use of tailored Wikipedia articles, the top level of our taxonomy meets the common expectations of a library classification system. The subclasses for two of the top-level and mid-level categories are shown in the middle and on the right of the figure, respectively. For the second and third level classes, we observe a clear semantic connection to the respective parent classes. However, our approach does not produce an obvious semantic connection between sibling classes; it’s not possible to guess the tenth sibling from studying the first nine. The percentage of documents covered by a given taxonomy subtree is shown in brackets next to each non-leaf label. Our keyquery-based classification system covers a total of 58% of the CORE subset considered in our experiment; the average coverage across all non-leaf taxonomy classes is 26%, indicating that despite the use of ESA as retrieval model, finding a classification at a high level of abstraction is a major challenge.

### 7.5 SUMMARY

The rise of digital content in libraries has a deep impact on the foundations of library management and forces us to revisit and rethink the established library tools and processes. In this paper, we revisit library classification systems and analyze their application portfolio in the digital era. What we find is that classification systems are no longer urgently required to retrieve relevant documents, but are rather supposed to complement the abilities of a query-based search engine. Classification systems are a valuable tool for digital libraries if they provide structured access to any given library subset, if they complement the set with further relevant documents, and if they embed the relevant documents into a broader context. To this end, we present

a dynamic taxonomy composition framework, which acquires its classes directly from keyqueries against a library's search engine. This close connection to the search engine brings a variety of advantages compared to existing approaches, including cost-efficient maintenance and seamless integration into search interfaces. Our experiments demonstrate the feasibility of our approach despite the complexity of the query space.

For future work, we propose a qualitative assessment of our framework under different retrieval models. Most promising in this regard seems the use of explicit topic models in combination with lexical knowledge bases, as our study on the CORE dataset substantiates. A retrieval model of this kind is able to improve the quality of the class labels compared to standard bag of word models. This is especially true for macroscopic concepts that appear only rarely as words in the documents' text.

# 8

## Exploratory Search Pipes with Scoped Facets

This chapter presents faceted search technology tailored to the peculiarities of exploratory search tasks. In the context of DH projects, the technology can be used to explore the annotations made during the corpus analysis activity. In contrast to traditional faceted search systems, facets in our system are arranged as a pipe, i.e., are applied sequentially one after the other. Moreover, the facets in a pipe are not applied throughout the whole sequence, but are limited to a user-definable scope, such that the search query can be broadened by adding a facet to the pipe. By this modification, the user interaction with our exploratory search engine resembles rather an open walk through the facet space spanned by the connections between documents and facets, than a progressive filtering of the document collection. We argue that this shift in the interaction paradigm is a much better fit to exploratory search scenarios. A user study with a prototype implementation attests an improved usability of our system compared to traditional faceted search systems for complex exploratory search tasks.

### 8.1 INTRODUCTION

The objective of this chapter is to introduce scoped facets as a concept to improve the exploratory search support of faceted retrieval systems. In line with the introduction of exploratory search in Chapter 2, we consider the search task of a user to be exploratory, if the user's goal is to learn about the contents of a document (sub-) collection rather than finding a specific piece of information. In other words, exploratory searchers ask for the in-

formation needs that could be answered on the basis of a specific document collection more than they ask for the documents that could answer a specific information need.

For example, a sub class of exploratory search is serendipity search (Toms, 2000), where the users' goal is to find interesting topics to search for in subsequent focused search tasks. More elaborated exploratory search tasks arise in the present context of digital humanities research, where corpora are explored with the goal of answering and discovering relevant research questions (cf. distant reading (Moretti, 2013)), but also in computational social science or investigative journalism. Even further, exploratory search tasks arise from vague memories (Blanc-Brude and Scapin, 2007), where users have to retrieve forgotten keywords which are necessary to formulate a focused search query, or because users want to attest the lack of relevant documents in a collection, for which all reasonable search queries have to be found and tested (cf. negative search (Garfield, 1970)).

Besides query suggestions, which allow users to get a glimpse into the information needs supported by a search engine, the use of facets as a paradigm to facilitate learning about and browsing through the contents of a document collection has been proposed to support exploratory search tasks (see Section 8.2). By presenting a set of relevant facets next to the search results for a query, so the intuition, exploratory searchers can start their search with a broad query, and then proceed with selecting facet terms to progressively narrow down into smaller result sets. While this search scenario covers an essential activity of exploratory searchers, we like to argue in line with White and Roth (2009) that supporting the opposite scenario, starting out from a small result set, then broadening the query to refocus on related but different document subsets, is at least equally important.

To improve the exploratory search experience of users in this regard, we introduce the concept of scoped facets in conjunction with exploratory search pipes in Chapter 8.3. In contrast to traditional faceted search systems known e.g. from e-commerce sites or digital libraries, our proposal is to arrange facets selected by the user sequentially in a *pipe*. At each position of the pipe, the user can choose which prior facets should be evaluated to determine the relevant documents at this position. If this *facet scope* does not comprise all prior facets, the resulting search query is broadened, potentially leading to documents not present in previous exploration steps. With the concept of facet scopes, the user can conveniently switch between narrowing down and broadening the current search query. And since even broadening the query means progressing in the pipe, the full exploration

path is retained over the whole search session, allowing the user to reflect on the search directions taken.

The usability of our proposal is evaluated by the means of a user study, where we ask participants to solve exploratory search tasks with a prototypical implementation of our concept and a classical faceted search system as baseline. The design and results of this study are presented in detail in Section 8.4. While for simple search tasks the overall usability of both systems is assessed equally as being of average quality, the usability differs significantly for complex exploratory search tasks. Here, the usability score for our system increases from average to good, while the usability score for the baseline system decreases from average to bad.

## 8.2 RELATED WORK

As already noted, exploratory search had been coined as an information retrieval topic around 2005. From a historical perspective, exploratory search picked up on information retrieval problems such as subject search (Bates, 1986) and search result clustering (Cutting et al., 1992), which address broad or vague information needs of users by providing topical overviews over retrieved document sets. Exploratory search departures from these problems in the respect that no longer the focus is on supporting users finding relevant documents, but is generalized to the task of supporting users learning about and investigating the contents of a specific document collection (including “which subtopics are addressed in my search results?” as one example) (Marchionini, 2006).

Over the years, a variety of exploratory search systems, like the Flamenco browser (Yee et al., 2003), the mSpace Explorer (schraefel et al., 2006), the Relation Browser (Capra and Marchionini, 2008), Querium (Golovchinsky et al., 2012), or SearchLens (Chang et al., 2019) have been presented. A commonality of all these exploratory search systems is that they employ facet-like structures as their primary exploratory search feature. Concerning our proposal of exploratory search pipes with scoped facets, the mSpace Explorer represents the most relevant prior work, as it also implements the idea of a sequential arrangements of facets. However, without the concept of scoped facets, adding facets to the sequence may only result in more specific queries. Hence, the added value over traditional faceted search systems concerning exploratory search support is limited in the mSpace explorer in our view.

A further stream of relevant research originates from the semantic web community, where faceted search systems have meanwhile been proposed

for the exploration of ontologies (see Tzitzikas et al. (2017) for a recent survey). The systems most related to our work from this stream of research are gFacet (Heim et al., 2010), and the proprietary semspect system<sup>1</sup>. In these two systems, the facets selected by a user are laid out not as a sequence but as a graph, visualized on a two-dimensional plane. Though the representation of the exploratory search path as a graph is more powerful in the sense that it allows branching at any position of the path, the two-dimensional layout makes displaying and interacting with (especially long) facet terms challenging, which is why we opted for a linear sequence, instead. More importantly, as holds for the mSpace Explorer, also these two systems do not consider the broadening of a search query by reducing the facet scope, but always apply all facets that share a common path.

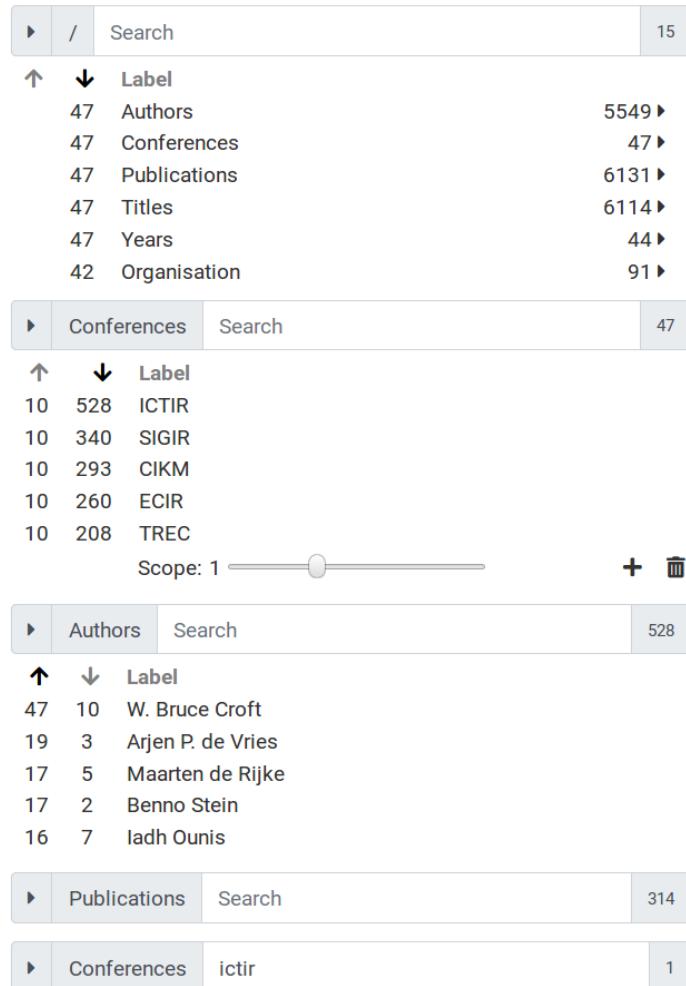
### 8.3 APPROACH

In this section, we formalize the concept of exploratory search pipes with scoped facets, and present how this concept is implemented in the prototype system *podascope* that we use in our evaluation. We roughly follow the notation of Sacco and Tzitzikas (2009), and define a faceted search system as the quadruple  $S = ((T, \leq), D, I, Q)$ , where  $(T, \leq)$  is a taxonomy which itself consists of a terminology  $T$  with a root term, as well as a partial order  $\leq$  over  $T$  called subsumptions, which defines the parent-child relationships between the terms in  $T$ . Every term in  $T$  is represented by a term id, a label, as well as an optional description. Terms that are subsumed only by the root node are called facets, and belong to the facet set  $F \subset T$ . Further on,  $D$  denotes the document collection that can be explored by our system. Each document in  $D$  consists of fields, i.e., key-values pairs with keys such as “authors”, “title”, “entities”, etc.. For all keys which refer to a term in  $T$ , we take all unique values and add them as leaf terms to the taxonomy. From these values, we also infer the relations between the documents in  $D$  and the terms in  $T$ . These term-document relations are denoted by  $I$ , a function  $I : T \rightarrow 2^D$  commonly referred to as interpretation. Ultimately,  $Q$  is the set of all queries that can be formulated over  $T$  using standard boolean operators.

**Faceted Search** The starting point for our formalization is a user who wants to explore the subset of documents  $D_q$  that are relevant for a user specified search query  $q$ . To allow the user to issue keyword based search

---

<sup>1</sup><http://semanticspect.de/>



**FIGURE 8.1:** Screenshot of our exploratory search prototype highlighting the benefits of facet scopes. By adding the first facet (shown collapsed at the bottom) with a reduced scope to the pipe again, similar terms with respect to the intermediate facets can be determined.

queries, our prototype initially provides a search box for keyword based searches within the fields of  $D$ . To facilitate faceted search, the subset of facets  $F$  which are associated with  $D_q$  via  $\leq$  and  $I$  are displayed next to the search results. A screenshot of our prototype, where the remaining facets available for a search query are shown in the top most element, is provided in Figure 8.1. Users can browse through the facets by clicking on the navigation element next to the child count of a facet. The current browsing path is displayed left hand to a term search box, which can also be used to navigate back up in the taxonomy. By selecting a facet term, the user can, as usual,

add the term to the current search query.

**Exploratory Search Pipes** With the concept described so far, the user can search and browse through the relevant facets, but cannot select terms from different facets and explore the relations between them. To this end, we introduce the concept of a facet sequence which we refer to as a *pipe*  $p = (e_0, \dots, e_n)$ . In fact, the initial facet element shown in response to search query represents already the first element of a pipe. To add an element to  $p$ , in our prototype, a set of controls including a plus-button appears below an element when hovering over it. Clicking on the plus-button adds a new pipe element above, which then shows the relevant facets at this position of the pipe. To illustrate the concept, Figure 8.1 shows a pipe with five elements.

As a special feature of our prototype system, we decided to compute as term scores for a facet term  $t$  not the number of documents in  $D_q$  that are associated with  $t$  by default, but rather the number of distinct terms from the facets above and below that are connected with  $t$  through  $D_q$ . Though users are more familiar with document counts as term scores (Wilson and m.c. schraefel, 2006), we choose the concept of up- and down-scores in our prototype as it is more expressive: By re-arranging the facets in the pipe, users can explore the relationships between arbitrary facet pairs. Document counts can always be obtained by inserting a facet with a 1:1 relationship to the document collection  $D$  as a neighboring element (such as the collapsed “Publications” facet in Figure 8.1).

Leaving the introduction of scoped facets to the next subsection, the term scores for a pair of neighboring pipe elements  $(e_{i-1}, e_i)$  are computed as follows: With all currently selected terms in the pipe elements  $\{e_k \mid k = [0, i]\}$ , a query is formulated and run against our system to retrieve the set of relevant documents  $D_i \subseteq D$  at pipe position  $i$ . Following common practice, all selected terms of one pipe element are joined to an expression with disjunctions (*or*) in the query, while these expressions themselves are joined with conjunctions (*and*):

$$D_i = \{d \mid d \in \cap_{k=0}^i (\cup_{t \in e_k} I(t))\}.$$

I.e.,  $D_i$  contains those documents that are associated with at least one selected term in every pipe element  $e_k$ . To compute the term scores for each term in  $e_i$ , it is counted how many distinct terms of  $e_{i-1}$  are related to the term through  $D_i$ :

$$score(t) = |\{t_j \mid t_j \in e_{i-1} \wedge \exists d : d \in D_i, I(t), I(t_j)\}|$$

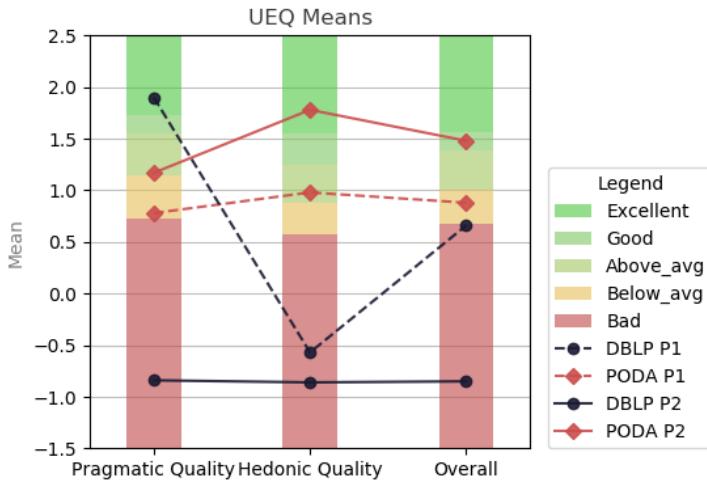
The resulting counts are referred to as the *up-scores* of the terms in  $e_i$ . Canonically, the *down-scores* of the terms in  $e_{i-1}$  are computed by counting the number of distinct relations to terms in  $e_i$ . As shown in Figure 8.1, the up- and down-scores are displayed in the two leftmost columns of the child term table. Since the first and the last pipe-elements have only one neighbor, only one of the score types is defined, respectively.

**Scoped Facets** With the concept described so far, the documents that can be explored at pipe element  $e_i$  are limited to the retrieved documents  $D_{i-1}$ , i.e., users can always only build subsets of the currently retrieved documents by adding a new element to the pipe. We argue that this limitation is undesirable, as users may want to begin an exploratory search with a very specific query, and then explore the document collection from there.

One possibility to overcome this limitation would be to only consider, for the computation of term scores at position  $i$ , the terms of  $e_{i-1}$  and  $e_i$  to formulate the respective search query. This solution, however, would trade the possibility of broadening a search by inserting a pipe element with the possibility to progressively narrow down the search results with more than one facet (e.g. in order to find the papers an author published at a specific conference).

Instead, we propose to give the user the possibility to specify the pipe elements which should be considered for the term scoring at each position of the pipe by introducing the concept of *scopes*. The scope  $s_i$  of a pipe element  $e_i$  is an integer in the range  $[0, i]$ . The scope determines which of the pipe elements below  $e_i$ , i.e.,  $\{e_k \mid k = [i - s, i]\}$ , shall be taken into account for the computation of the term scores of  $e_i$ . By default, the scope is set to  $s = i$ , which means that all prior pipe elements are considered. In the special case of  $s = 0$ , the pipe element  $e_i$  is considered a new starting point, and no term relations are computed for  $(e_{i-1}, e_i)$ . To set the scope of a facet in our prototype, the controls of our user interface feature a range slider (cf. second pipe element from top in Figure 8.1).

Though a simple concept, with the introduction of facet scopes, our system gains a powerful exploratory search feature which sets it apart from prior work. A special usage pattern of facet scopes that we observe is that users apply a reduced scope to discover facet terms which are similar with respect to intermediate facets. For example, in Figure 8.1, the user discovers conferences similar to ICTIR in terms of author overlap.



**FIGURE 8.2: Results of the user study.** The usability of both systems are assessed as average for simple search tasks (dashed lines). For complex search tasks, the usability of our prototype *podascope* (red) increases to good. For the baseline system (black) it decreases to bad.

## 8.4 EVALUATION

In order to evaluate our concept, we conducted a user study with 14 participants. As a baseline to compare our prototype system *podascope* with, we choose the digital library DBLP<sup>2</sup>, which indexes a large dataset of currently around 4.5 million computer science research paper records. DBLP was chosen as our baseline system for two reasons: (1) The dataset indexed by DBLP is freely available, such that we could index the same data with our research prototype. (2) The domain of DBLP matches the interests of our 14 participants (all researchers in various computer science disciplines), which is according to Kules and Capra (2008) a desirable circumstance for the evaluation of exploratory search systems.

In our user study, we were interested in two main questions: (1) Does our proposal of exploratory search pipes with scoped facets harm the search experience of users with simple search tasks that do not require query broadening? (2) Does our proposal increase the user experience for search tasks which are complex (do require query broadening)?

To answer these questions, the user study was divided into two parts. In the first part of the study, the participants were asked to solve five simple search tasks with both systems. During the study, an adviser was physically present and assisted the participants if needed. One half of the participants always started solving the task with the baseline system, whereas the other

<sup>2</sup><https://dblp.uni-trier.de/>

half started with our prototype. The five simple search tasks were:

- Find all papers with “exploratory search” in the title.
- How many authors published on this topic?
- Which author published the most papers on this topic?
- At which conferences did he/she publish these papers?
- What other papers did he/she publish?

After solving the simple search tasks, the participants were asked to evaluate the systems by answering the user experience questionnaire UEQ-S<sup>3</sup>, which asks for four pragmatic and four hedonic quality dimensions. The results of this first evaluation is presented in Figure 8.2 (dashed lines). With an overall mean quality score of 0.88 (podascope) and 0.66 (DBLP), the user experience of both systems is assessed comparably well by the annotators, leading us to the conclusion that our concept did not harm the search experience for simple search tasks.

In the second part of the study, the participants were asked to solve the following three complex search tasks:

- Which five conferences are most similar to “ICTIR” in terms of author overlap?
- In how many years did each of these conferences take place?
- Who published most papers at these conferences?

To solve these tasks, the facet scope has to be reduced in our prototype system (cf. Figure 8.1). For the baseline system, solving the tasks requires to first select all relevant facet terms of the intermediate query (e.g. all ICTIR authors), and then de-selecting the facets terms of the first query. Since DBLP does not allow to select multiple terms of a facet to formulate a disjunctive query, the first complex task could be solved only roughly with this system by selecting one of the authors. After working on the tasks, the participants were asked to answer the UEQ-S again (see solid lines in Figure 8.2): Whereas the baseline system is assessed worse than before (mean UEQ-S score of -0.85), the user experience score of our system increases to 1.48, leading us to the conclusion that our concept improves the exploratory search experience for the case of complex search tasks.

## 8.5 SUMMARY

Our concept of exploratory search pipes with scoped facets contributes a novel and powerful exploratory search paradigm that facilitates the intuitive, continuous, and dynamic exploration of annotated corpora. The user

---

<sup>3</sup><https://www.ueq-online.org/>

study conducted on the basis of a prototype of our concept in an exploratory search system revealed that for simple search tasks, our paradigm does not harm the overall search experience of users. For complex search tasks requiring the broadening of current search queries, the usability of our system is perceived as superior to a traditional exploratory search system.

# 9

## Topical Sequence Profiling

This chapter introduces the problem of topical sequence profiling as a further technology to explore corpora which have been annotated with cardinal facets, i.e., which allow to divide a corpus into sequences of sub-collections. Given a sequence of text collections, such as the annual proceedings of a conference, the topical sequence profile is the most diverse explicit topic embedding for that sequence that is both representative and minimal. Topic embeddings represent a text collection sequence as numerical topic vectors by storing the relevance of each text collection for each topic. Topic embeddings are called explicit, if human readable labels are provided for the topics. A topic embedding is representative for a sequence, if for each text collection the percentage of documents that address at least one of the topics exceeds a predefined threshold. If no topic can be removed from the embedding without loosing representativeness, the embedding is minimal. From the set of all minimal representative embeddings, the one with the highest mean topic variance is sought and termed as the topical sequence profile. Topical sequence profiling can be used to highlight significant topical developments, such as raise, decline, or oscillation. The computation of topical sequence profiles is made up of two steps, topic acquisition and topic selection. In the first step, the sequence's text collections are mined for representative candidate topics. As a source for semantically meaningful topic labels, we propose the use of Wikipedia article titles, whereas the respective articles are used to build a classifier for the assignment of topics to documents. Within the second step, the subset of candidate topics that constitutes the topical sequence profile is determined, for which we present an efficient greedy selection strategy. We demonstrate the potential of top-

**TABLE 9.1:** Concept for the visualization of topical sequence profiles. The sequence flows from left to right, the topics from top to bottom in descending order of their diversity. The height of each of the bars in the cells is proportional to the topic coverage values  $T_{ij}$ . Colors encode raising (green), declining (red), and oscillating (gray) topics.

Topic	$D_1$	$D_2$	$D_3$	$D_4$
Topic 1	5%	6%	15%	20%
Topic 2	14%	6%	15%	5%
Topic 3	15%	12%	11%	8%
<b>Coverage</b>	15%	15%	15%	15%

ical sequence profiling as an effective DH technology with a case study on a sequence of conference proceedings.

## 9.1 INTRODUCTION

The capability to mine and visualize insights from data has become the basis for competition and growth of companies, and it causes an increasing demand for data science experts and technology (Manyika et al., 2011). This is also true for the digital humanities, where the distant reading (Moretti, 2013) of annotated corpora by the means of corpus visualization is becoming an established research practice. In this chapter, we reveal data science technology for the analysis of sequences of text collections. Interesting sequences of this kind may be daily business news feeds, the social media mentions of a company over time frames, or the collected annual proceedings of a research field. Our working hypothesis is that in cases such as those mentioned, statistical insights into the topic distribution of both the individual text collections and the topic development over the sequence is of a high value for the respective stakeholders; however, the amount of potentially relevant topics often prohibits a comprehensive examination. To provide stakeholders with a selection of topics that is informative on the one hand and small enough to be surveyed quickly on the other, we introduce the problem of topical sequence profiling. Taking the annual proceedings of a research field as illustrative example, the goal of topical sequence

profiling is to showcase research topics that peak as “hot topic” in distinct years, but show a significant decline throughout the remaining years. We argue that, in contrast to topics that never peak or that constantly belong to the “usual suspects”, especially from these topics valuable insights can be expected.

### 9.1.1 Problem Definition

The problem of topical sequence profiling can be stated as follows. Given a sequence of text collections  $\mathcal{D}$ ,  $\mathcal{D} = (D_1, D_2, \dots, D_n)$ , where each  $D \in \mathcal{D}$  is a set of documents, find the most diverse, explicit topic embedding  $\mathbf{T}^*$  for the sequence that is both minimal and representative.  $\mathbf{T}^*$  is called the topical sequence profile of  $\mathcal{D}$ .

A topic embedding  $\mathbf{T}$  can be considered as a matrix that represents each  $D \in \mathcal{D}$  as a column of  $k$  topics,

$$\mathbf{T} = \begin{matrix} & D_1 & \cdots & D_n \\ t_1 & \left[ \begin{matrix} \mathbf{T}_{11} & & \mathbf{T}_{1n} \\ & \ddots & \\ \mathbf{T}_{k1} & & \mathbf{T}_{kn} \end{matrix} \right] \\ \vdots \\ t_k & \end{matrix}.$$

The matrix entries  $\mathbf{T}_{ij}$  are denoted as *topic coverage* and correspond to the percentage of documents in  $D_j$  that are relevant for topic  $t_i$ . To compose a topic embedding  $\mathbf{T}$ , first a set of topics has to be acquired and assigned to the documents in  $\mathcal{D}$ . Subsequent to the presentation of related work in Section 9.2, an algorithm for this topic acquisition step that utilizes Wikipedia articles as a topic resource and employs text classification to label documents with these topics is introduced in Section 9.3.1. Once a topic set has been acquired and assigned, topic embeddings can be composed by selecting specific topic subsets.

Table 9.1 illustrates our concept for the visualization of topic embeddings, which depicts the rows  $T_{i:}$  of an embedding in the form of bar charts. In order to become interpretable for users, the topics have to be made *explicit*, i.e., a semantically meaningful label must accompany each bar chart. While our approach utilizes Wikipedia titles as explicit topic labels, latent topic models such as LDA (Blei et al., 2003) or doc2vec (Le and Mikolov, 2014) could be employed in combination with topic labeling.

A topic embedding should reveal insights into the topic distribution of each text collection. We call a topic embedding *representative* for a sequence, if for every  $D \in \mathcal{D}$ , the percentage of documents covered by at least one of the embedded topics exceeds a predefined threshold  $c \in [0, 100]$ . Since one

document may be relevant for multiple topics, the collection coverage for  $D_j$  is in general not the sum over column  $T_{:j}$  but less or equal. We state the collection coverage for every  $D \in \mathcal{D}$  in the last row of our visualization.

To address the requirement of a topic embedding to be of a manageable size, the notion of *minimality* is introduced: A topic embedding is minimal if no topic can be removed without losing representativeness. From all representative minimal topic embeddings we are interested in the instance  $\mathbf{T}^*$  that on average contains the *most diverse* topics. Section 9.3.2 introduces an effective greedy strategy to the optimization problem of finding  $\mathbf{T}^*$  based on a representative non-minimal topic embedding  $\mathbf{T}$ . As a measure of topic diversity, we propose the variance of the topic distributions. Our choice is motivated by the fact that in order to achieve high topic variance, the topic coverage for the individual text collections must deviate significantly from the mean. This happens if the topic coverage peaks for some text collections and is low elsewhere.

To conveniently spot raising, declining, or oscillation topics in our visualization, we apply linear regression to the topic coverage values of each topic and color-code the slope of the resulting regression curve (cf. Table 9.1). A positive slope (raise) is encoded by green bars, a negative slope (decline) by red bars. The lighter the color, the steeper the slope. Zero slope is encoded by gray bars and, in case the topic's diversity is high, represents an oscillating topic.

The potential of topical sequence profiling as a DH tool is highlighted with a case study on the basis of conference proceedings in Section 9.4, and we close with a discussion of our contributions in Section 9.5.

## 9.2 RELATED WORK

Although, to the best of our knowledge, we are the first to study the problem of topical sequence profiling, a close relation exists to the task of labeling a clustering of documents, for which we point out the state of the art in this section. The problem of cluster labeling can be framed as follows. Given a clustering  $\mathcal{C} = \{C_1, \dots, C_n\}$ , where each  $C \in \mathcal{C}$  is a set of documents, find a set of explicit topics (the cluster labels) that characterize each of the clusters. Obviously, the above sequence  $\mathcal{D}$  can be interpreted as a clustering, and a topic embedding  $\mathbf{T}$  may be used to represent the (binary) assignment of labels to clusters. Moreover, both problems include a topic acquisition step that facilitates the composition of  $\mathbf{T}$ . The most obvious way to acquire explicit topics is to extract (key-) words (Geraci et al., 2006; Stein and Meyer zu Eißen, 2004), phrases (Erbs et al., 2013; Treeratpituk and Callan, 2006),

or queries (Gollub et al., 2013a) from the documents in the text collections that are relevant with respect to a retrieval model. The main disadvantage of these approaches is that the label of a relevant topic may not appear in a document, or at least not in a statistically significant way (Carmel et al., 2009). To overcome this problem, the use of external knowledge resources as a source for explicit topics can be considered state of the art. Proposed resources are thesauri such as WordNet (Wei et al., 2015b), linked open databases such as Dbpedia (Hulpus et al., 2013), or encyclopedias such as Wikipedia (Carmel et al., 2009; Scaiella et al., 2012; Syed et al., 2008; Wei et al., 2015a). Depending on the resource, different classification strategies are proposed, which decide whether or not an external topic is relevant for a document. For example, Carmel et al. formulate search queries from document keyphrases against Wikipedia, and classify the top articles as relevant topics. By contrast, the Wikipedia-based approach that we apply for topic acquisition is adopted from the ESA retrieval model, which relies on the cosine similarity between a document and an article for relevance assessments (Gabrilovich and Markovitch, 2007).

What distinguishes topical sequence profiling from cluster labeling are the properties topics should satisfy. While we strive for a set of topics that is minimal, representative, and diverse, the desired properties of cluster labels are different. Stein and Meyer zu Eißen (2004) have compiled a set of commonly accepted properties, which are listed in Table 9.2; a formal specification of the respective semantics is detailed in their paper. Though half of the properties coincide with our definition of minimality and representativeness, the properties unique, discriminating, and contiguous collide with our definition of diversity.

### 9.3 APPROACH

The approach for computing the topical sequence profile for a sequence  $\mathcal{D}$  comprises two steps. In the first step, the topic acquisition step, a comprehensive set of explicit topics is determined, and the topic coverage of each text collection is assessed for each of the topics. The result of the topic acquisition step is a topic embedding  $\mathbf{T}$  that is representative but not minimal. In the second step, topics are removed from  $\mathbf{T}$  with the objective to find the most diverse minimal topic embedding  $\mathbf{T}^*$ .

**TABLE 9.2: Comparison of topical sequence profiling and cluster labeling with respect to desired topic label properties.**

Property	Cluster Labeling	Sequence Profiling
Unique	✓	✗
Summarizing	✓	✓
Expressive	✓	✓
Discriminating	✓	✗
Contiguous	✓	✗
Irredundant	✓	✓
Minimal	✓	✓
Representative	✓	✓
Diverse	✗	✓

### 9.3.1 Topic Acquisition

As pointed out in Section 9.2, there are several ways for obtaining a set of explicit topics that are tailored to a collection of documents. In line with the state of the art, our approach of choice is to consider the titles of Wikipedia articles as explicit topics. To decide whether a document is relevant for a Wikipedia topic or not, the cosine similarity between the vector space representations of the Wikipedia article and the document is computed under the BM25 model (Robertson and Zaragoza, 2009). To make the binary decision upon relevance, which is needed to compute the topic coverage values  $T_{ij}$ , we apply again our unsupervised sparsification criterion (Gollub and Stein, 2010). The main idea of this approach is to compute for every topic an expected similarity score based on the aggregated vector representation of the whole sequence. Only if the similarity score of a document exceeds the expected value, the document is classified as being relevant for the topic.

With more than five million English articles, the pairwise computation of similarities between Wikipedia articles and sequence documents is inefficient for large sequences, and an efficient strategy is desired that determines a subset of articles that contain the topics of  $T^*$  with full recall and acceptable precision. To this end, we reuse the aggregated vector representation of the sequence and determine its most similar Wikipedia articles. The rationale is that if the text collections of the sequence have a common general topic domain (such as a common research field), the most similar articles of the aggregated vector should reveal this. Using these articles as seeds, we can traverse the Wikipedia link graph until a representative topic embedding  $T$  is obtained that is tailored to the sequence. To optimize the

quality of the traversal, we consider only links that have been clicked at least ten times according to a Wikipedia clickstream dataset (Wulczyn and Taraborelli, 2015).

### 9.3.2 Topic Selection

Given the representative topic embedding  $\mathbf{T}$  of the topic acquisition step, the optimization problem of the topic selection step is to determine the subset of topics in  $\mathbf{T}$  that maximizes the average topic diversity and satisfies the minimality property:

$$\begin{aligned} \text{maximize} \quad & \frac{1}{k} \sum_{i=1}^k \text{Var}(\mathbf{T}_{i:}) \\ \text{subject to} \quad & \mathbf{T} \text{ is minimal} \end{aligned}$$

Note that the above optimization problem is an instance of the set cover problem, and hence it cannot be solved efficiently for large sequences. Here we present, in form of Pseudocode 6, an efficient greedy strategy to find an approximate solution. First, the rows (topics) of the given topic embedding  $\mathbf{T}$  are sorted by diversity in ascending order, and  $\mathbf{T}^*$  is initialized with  $\mathbf{T}$ . Then, for each of the  $k$  topics in  $\mathbf{T}$ , starting with the least diverse topic, it is checked whether the removal of the topic still yields a representative topic embedding. If so, the topic is removed from  $\mathbf{T}^*$ . After applying this procedure,  $\mathbf{T}^*$  is minimal, and since the topics are removed in ascending order of their diversity, the algorithm strives for maximizing the average topic diversity.

---

#### Pseudocode 6 Greedy Topic Selection

---

```

Input: Topic Embedding  $\mathbf{T}$ 
Output: Topic Embedding  $\mathbf{T}^*$ 
1: sortAscending( $\mathbf{T}$ )
2:  $\mathbf{T}^* \leftarrow \mathbf{T}$ 
3: for  $i = 1; i \leq k; i = i + 1$  do
4:   if representative( $\mathbf{T}^* \setminus \mathbf{T}_{i:}^*$ ) then  $\mathbf{T}^* \leftarrow \mathbf{T}^* \setminus \mathbf{T}_{i:}^*$ 
5: return  $\mathbf{T}^*$ 

```

---

## 9.4 CASE STUDY

Due to the complexity of the task, a thorough evaluation of the usefulness of topical sequence profiling in practical scenarios would require an extensive user study. In this work, however, we resort to a case study on the basis of a sequence of SIGIR conference proceedings to gather first empirical insights into the performance of our approach. We choose SIGIR proceedings since we, and likely the reader, are familiar with the information retrieval research domain. What we wish to obtain is a topical sequence profile that (1) consists of reasonable information retrieval research topics that are objectively representative, and that (2) show an interesting development over the years. We consider proceedings from 2007 to 2015 which results in the following sequence.

$\mathcal{D}$	Year	# Papers	$\mathcal{D}$	Year	# Papers
$D_1$	2007	198	$D_6$	2012	216
$D_2$	2008	193	$D_7$	2013	205
$D_3$	2009	193	$D_8$	2014	226
$D_4$	2010	214	$D_9$	2015	193
$D_5$	2011	232			

*Topic Acquisition.* First we obtain a small set of seed Wikipedia articles that match the general topic domain of the sequence to facilitate an efficient acquisition of a representative topic embedding  $\mathbf{T}$ . For this purpose, the BM25 vector space representations for all papers in  $\mathcal{D}$  are aggregated. In terms of the cosine similarity with this aggregated vector, the ten most similar Wikipedia articles obtained are:

1. Concept Search (0.678)
2. Information Retrieval (0.593)
3. Human-Computer Information Retrieval (0.588)
4. Web Query Classification (0.582)
5. Enterprise Search (0.549)
6. Search engine technology (0.540)
7. Document retrieval (0.539)
8. Cognitive models of information retrieval (0.524)

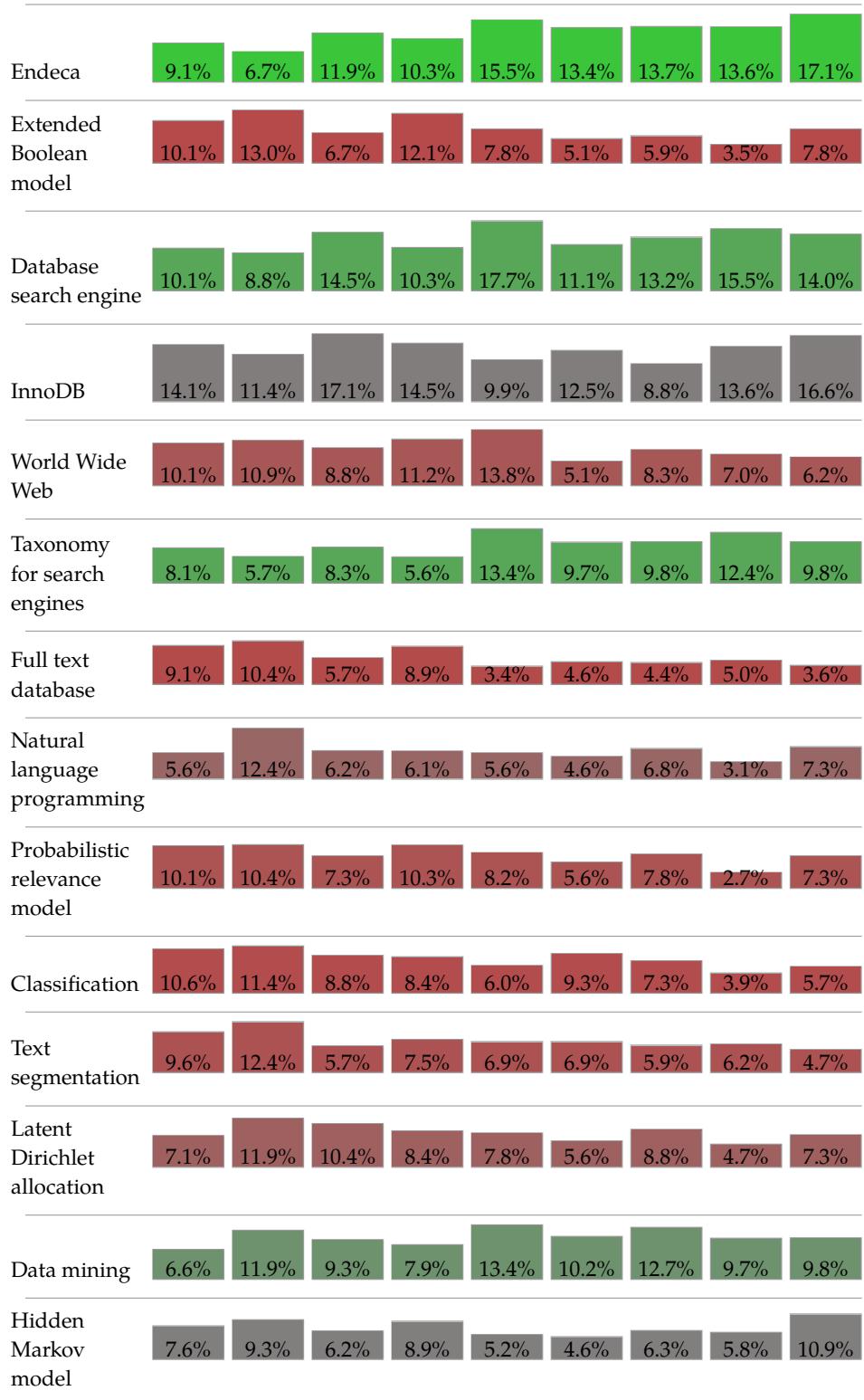
9. Federated search (0.524)
10. Web search query (0.518)

Starting from these ten articles, the Wikipedia link graph is traversed in a breadth first manner and every sequence document is classified against the visited articles. With a collection coverage threshold  $c$  of 80%, the traversal stopped with a representative topic embedding  $\mathbf{T}$  after visiting 1261 Wikipedia articles, which indicates that link graph traversal based on seed articles leads to significant efficiency improvements over the classification against the whole Wikipedia.

*Topic Selection.* The topical sequence profile  $\mathbf{T}^*$  obtained after applying Greedy Topic Selection to  $\mathbf{T}$  with  $c$  set to 60% is illustrated in Table 9.3. Since each of the proceedings contain about 200 documents, a topic coverage of one percent roughly corresponds to two papers that have been assigned to a topic in a specific year. The topical sequence profile consists of 19 topics. I.e., altogether, these topics are representative for each of the years and no topic can be removed without losing representativeness. Though some of the topics may be interpretable only after looking up the respective Wikipedia article, a reasonable selection from information retrieval research can be observed. “Library Classification” is declining and the most diverse topic, followed by “Query” and the raising topics “Search engine results page” and “Endeca”. The topics “InnoDB” and “Hidden Markov model” show oscillating behavior. The collection coverage values at the bottom reveal that the document coverage for 2014 prevents the removal of further topics (reaches  $c$ ).

**TABLE 9.3:** Topical sequence profile for the proceedings of the SIGIR conference from 2007 to 2015. The sequence flows from left to right, the topics, in descending order of their diversity, from top to bottom. The height of each cell is proportional to the coverage of the topic in the respective year. Colors encode raising (green), declining (red), and oscillating (gray) topics.

Topic	2007	2008	2009	2010	2011	2012	2013	2014	2015
Library classification	14.6%	12.4%	11.9%	10.7%	6.0%	11.1%	5.9%	2.7%	5.7%
Query	14.1%	12.4%	15.0%	17.8%	12.1%	11.6%	9.3%	6.2%	6.2%
Search engine results page	10.6%	8.3%	14.0%	12.1%	19.0%	13.4%	13.2%	16.7%	16.1%



Cosine similarity	7.1%	6.7%	7.3%	6.1%	5.6%	2.3%	3.9%	4.3%	9.3%
Gensim	7.1%	13.0%	9.8%	7.5%	8.2%	9.3%	7.8%	8.5%	6.2%
<b>Coverage</b>	75%	76%	78%	74%	72%	67%	67%	60%	71%

## 9.5 SUMMARY

With topical sequence profiling, we contribute a new research problem for the analysis and visualization of sequential text collections. In contrast to cluster labeling, sequence profiles aim at revealing representative topics that are subject to significant changes in terms of their coverage throughout a sequence of text collections. The presented case study revealed that the computation of topical sequence profiles is efficient and produces promising results. For practical applications, we observe that through interactive topical sequence profiles, which update after users explicitly remove or pin topics from the profile, the perceived quality can be further increased. Due to the efficiency of the greedy topic selection algorithm, the updating of a profile in response to user interaction can be achieved instantaneously. Further, the post-acquisition of topics based on a given topical sequence profile seems worth considering. Looking again at the SIGIR profile in Table 9.3, it appears that the post-acquisition of a raising topic that contributes to the collection coverage of 2014 would help balancing both the collection coverage values as well as the ratio of raising, declining, and oscillating topics.



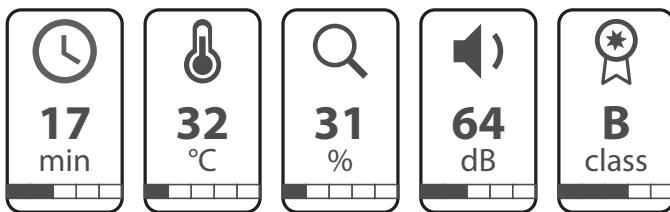
# 10

## Conclusion

The digital humanities strive to answer humanities research questions through the acquisition and analysis of a digital corpus. The thesis models this process by the means of five research activities, namely research question development and modeling, corpus acquisition and analysis, as well as result dissemination. From Chapters 4 to 9, information retrieval technologies were presented which are tailored to the IR tasks that arise during the first four of the five research activities. The presented technologies approach the IR tasks related work search, recall-oriented search, search in meta-data, dynamic taxonomy composition, exploratory search, and search result profiling. In the course, most of the technologies make use of the keyquery paradigm, which proposes an additional search operation that returns relevant search queries for a given set of input documents. In this final chapter, I like to conclude by focusing on the remaining research activity: the dissemination of research results.

### 10.1 A HOLISTIC VIEW ON THE DH RESEARCH PROCESS

From the point on where, during the corpus analysis activity, a research corpus has been annotated according to a research model, the answer to the modeled research question can be obtained by querying a faceted search engine that indexes the annotated corpus. E.g., the exploratory search engine presented in Chapter 8 could be used to index the annotations of organism names and experiment descriptions within a corpus of 19th century life science publications. To obtain an answer to the question of organism use in 19th century life science experiments, a faceted search pipe with an organ-



**FIGURE 10.1:** Visual representation of the information nutrition label. Time, temperature, transparency, volume, and credibility are taken as quantities to describe the five nutrition fact categories (1) effort, (2) kairos, (3) logos, (4) pathos, and (5) ethos.

isms facet, which is filtered for annotations within experiment descriptions, could be build. The result in the organism facet would display, by default, as a list of organisms names with respective frequency scores, which could then be published as a research result. Hence, from a holistic viewpoint, solving the specific IR tasks that arise during the individual research activities contributes to solving a higher order IR task: the development of a faceted search engine that ultimately returns a direct answer to the research questions posed. In other words, *what IR4DH advocates is the development of search technology that is eventually capable of directly answering specific humanities research questions based on the evidence in a corpus*. As a benefit of consequently applying IR4DH in digital humanities projects, such search technology could be disseminated as part of a research publication to give readers the ability to explore a published research result down to individual annotations, a feature that would greatly contribute to the transparency and credibility of DH research.

## 10.2 (RE)SEARCH RESULT PRESENTATION

Lists as a visual paradigm are in general well suited for the presentation of facet values. For the sake of result dissemination, however, more sophisticated facet representations may be achievable in individual cases. This pertains especially to facets with a cardinal scale, such as dates, physical quantities, or the relative location of components in complex objects. In the following, two of such facet representations from my research are presented.

### 10.2.1 The Information Nutrition Label

The first facet representation is a visualization of the information nutrition label, which was proposed by Fuhr et al. (2017). Like its food counterpart,

the information nutrition label is supposed to help people making more informed decisions upon which news items to consume. As an attempt to convey information nutrition facts by an intuitive, unambiguous, and intelligible label representation, we propose in Gollub et al. (2018b) a label design that first groups the 13 original information nutrition dimensions into five categories, and then associates each category with a well-known quantity from physics and finance. An illustration of the design is given in Figure 10.1. Each of the five categories is visualized by a rounded rectangle featuring a category symbol and an article-specific category value. For the latter, both the absolute value as well as its relative position in the value range are depicted. The semantics of each category are as follows.

Category (1), effort, groups all dimensions that affect the time a reader has to allot to comprehend an article. Besides verbosity, the dimensions readability and technicality fall into this category. To express effort, we consider time in minutes as an intuitive choice. The effort category allows readers to check whether they have enough time to read an article and to identify articles of a specific depth.

Category (2), kairos, groups all dimensions that pertain to the trendiness, momentum, or hotness of an article or a topic, i.e., topicality and virality. As a quantity to express kairos, we consider the temperature in the range of 0 – 100°C as intuitive. The kairos category can bring articles to readers' attentions which would be "out of their bubble" otherwise.

Category (3), logos, groups all dimensions that capture how well an author supports her claims with evidence, i.e., factuality and verifiability. As a quantity to express logos, we consider transparency in the range from 0 - 100% as intuitive. The logos category can help readers to assess the journalistic quality of an article up front.

Category (4), pathos, groups all dimensions that are related to subjectivity and discrepancies, i.e., emotion, opinion, and controversy. As a quantity to express pathos, we consider volume, measured as sound pressure, as intuitive. The pathos category can help readers creating awareness that communities sharing alternative arguments or opinions likely exist.

Category (5), ethos, finally groups all dimensions related to the credibility of an author or publisher, i.e., authority, credibility, and trust. As a quantity to express ethos, we consider credit ratings in the range from A+ to D, as used in finance,<sup>1</sup> as an adequate choice. The ethos category can help readers assessing the risk of becoming misinformed or, alternatively, the potential of learning about non-mainstream viewpoints.

---

<sup>1</sup>[https://en.wikipedia.org/wiki/Bond\\_credit\\_rating#Credit\\_rating\\_tiers](https://en.wikipedia.org/wiki/Bond_credit_rating#Credit_rating_tiers)

We see three advantages when using the proposed categories as attributes for the information nutrition label instead of the original dimensions. First, the reduced number of attributes makes the label both easier to present and easier to digest in practical settings. Second, by resorting to well-known quantities for the categories, readers can intuitively interpret the label without the need of detailed instructions. Third, the chosen quantities allow for the design of a non-textual visualization of the nutrition label.

### 10.2.2 Philosophical Bodies

The second sophisticated facet representation is the result of a DH pilot study conducted in a collaboration with media scientists and product designers (Schmidgen et al., 2021). The primary research question of the study asked for the conceptions of the human body that have been developed in the philosophical discourse of modernity. A secondary research question asked whether the physicalization of facet representations by the means of 3D printing provides added value for the analysis and dissemination of research results.

As corpus for the study the collected writings of seven German philosophers were acquired, as well as the German Wikipedia which served as a reference sub corpus. In order to model the primary research question, we assumed that what we call the conception of the human body can be determined by the use of terms referring to individual body parts in the corpus. Drawing on lexical resources such as Dornseiff's "Der deutsche Wortschatz nach Sachgruppen" (in English: "The German vocabulary in subject groups"), but also physiological and psychological dictionaries of the 19th and early 20th centuries, we created a linguistic model of the human body which assigns body-related terms to one of five regions: face/head, arm/hand, torso/abdomen, leg/foot, and a general region for non or weakly localized elements such as skin, muscle, or blood. By annotating and indexing the corpus according to the linguistic body model, the frequencies with which terms from individual body regions occur in the collective writings of a philosopher can be obtained.

For the physicalization of the results, a 3D model of the famous Penfield homunculus was created using the computer-aided design environment Rhinoceros 3D. This initial 3D model served as a reference body, which was set as corresponding to the frequency distribution of body regions in the Wikipedia reference corpus. To represent the sub-corpora of the philosophers, two boundary bodies were defined that, taking into account material,



**FIGURE 10.2:** Photographs of the digitally fabricated 3D-printed philosophical bodies. From left to right the bodies represent (1) the Wikipedia reference, (2) Kant, (3) Freud, (4) Heidegger, (5) Hegel, and (6) Luhmann.

morphologic, and production conditions, represent the maximum and minimum extent of each body region, respectively. For proportional conversion of relative term frequencies into geometric extents, the maximum extent of a body region was associated with the case where all body-related terms in a corpus refer to this one body region (100% relative frequency). The complete absence of terms from a body region (0% relative frequency) was associated with the minimum boundary body of that body region. Intermediate values were linearly interpolated, taking into account the relative frequency of the Wikipedia reference body as a fixed point.

Photographs of the 3D-printed “philosophical bodies” are shown in Figure 10.2. Looking at the bodies, it becomes apparent that each sculpture has distinctive features that differ from the Wikipedia reference body (the sculpture on the far left). For example, the comparatively small head of the Kant sculpture, the less pronounced hands in the Hegel figure, and the oversized torso in Freud’s and short legs in Heidegger’s are striking. The most remarkable sculpture, however, is the one created with the collected writings of Niklas Luhmann (the sculpture on the far right): In this corpus, more than half of all body-related terms refer to arms and hands. Looking at the figures in chronological order, an overarching development - such as a consistent shift from the face to the hands - cannot be discerned, though.

In comparison with a representation of the research results as a list, the three-dimensional representation as bodies has three features that make it particularly suitable for our purposes.

(1) The human body is one of those ontological entities with which we are all familiar. Miniaturized models of the human body therefore do not require detailed explanation. Interpretation of the results („What is shown?”, „What is characteristic?”) becomes an almost effortless and intuitive process. (2) Furthermore, the space for projecting the data is precisely the space the data is about, i.e., the space of the human body. Consequently,

no explicit labels of the dimensions and no legends are required to define the projection space - an advantage over other two- and three-dimensional projections like diagrams and maps for which legends are required to communicate their specific meaning to the viewer. (3) Also, since geometric modeling is used to encode the research results (rather than, for example, color), the bodies easily capture the viewer's attention, especially when transformed into a sculpture via 3D printing. It is the Penfield homunculus' striking yet arresting form that has contributed in a crucial way to its success as an icon of brain research. This impressive form is now reused in the field of digital humanities.

The validity especially of this third feature was reinforced by two exhibitions of the sculptures. The uncommon exhibition format as such already ensured that the project could be shown in each case in an exposed exhibition location that could not have been occupied by a conventional poster presentation. Remarkably, the sculptures were not only frequently viewed, but also in many cases photographed using smartphones and thus re-contextualized into the digital world. Overall, the pilot study encouraged us to apply physical 3D data visualization to other corpora and facets in the future, thus opening up further digital humanities contexts.

## References

- Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases, VLDB '94*, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc. ISBN 1-55860-153-8. URL <http://dl.acm.org/citation.cfm?id=645920.672836>.
- Anne Aula and Daniel M. Russell. Complex and exploratory web search. In *Proceedings of the Information Seeking Support Systems Workshop (ISSS 2008)*, 2008.
- Leif Azzopardi and Vishwa Vinay. Retrievability: an evaluation measure for higher order information access tasks. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008*, pages 561–570, 2008.
- Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. Query recommendation using query logs in search engines. In *Proceedings of the 2004 International Conference on Current Trends in Database Technology, EDBT'04*, pages 588–596, Berlin, Heidelberg, 2004. Springer-Verlag. ISBN 3-540-23305-9, 978-3-540-23305-3. doi: 10.1007/978-3-540-30192-9\_58. URL [http://dx.doi.org/10.1007/978-3-540-30192-9\\_58](http://dx.doi.org/10.1007/978-3-540-30192-9_58).
- Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval - the concepts and technology behind search, Second edition*. Pearson Education Ltd., Harlow, England, 2011. ISBN 978-0-321-41691-9.
- Ken Barker and Nadia Cornacchia. Using noun phrase heads to extract document keyphrases. In *Advances in Artificial Intelligence, 13th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2000, Montréal, Quebec, Canada, May 14-17, 2000, Proceedings*, pages 40–52, 2000.
- H. Bast, Björn Buchhold, and Elmar Haussmann. Semantic search on text and knowledge bases. *Found. Trends Inf. Retr.*, 10:119–271, 2016.
- Hannah Bast and Marjan Celikik. Efficient index-based snippet generation. *ACM Trans. Inf. Syst.*, 32(2):6:1–6:24, April 2014. ISSN 1046-8188.

Marcia J. Bates. Subject access in online catalogs: A design model. *Journal of the American Society for Information Science*, 37(6):357–376, 1986. doi: 10.1002/(SICI)1097-4571(198611)37:6<357::AID-ASI1>3.0.CO;2-H.

Marcia J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13(5):407–424, 1989. doi: 10.1108/eb024320.

Joeran Beel, Stefan Langer, Marcel Genzmehr, Bela Gipp, Corinna Breitinger, and Andreas Nürnberg. Research paper recommender system evaluation: a quantitative literature survey. In *Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation*, pages 15–22. ACM, 2013. doi: 10.1145/2532508.2532512. URL <http://doi.acm.org/10.1145/2532508.2532512>.

Nicholas J Belkin. Anomalous states of knowledge as a basis for information retrieval. *Canadian journal of information science*, 5(1):133–143, 1980.

Adam Berger and John Lafferty. Information retrieval as statistical translation. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’99, pages 222–229, NY, USA, 1999. ACM. ISBN 1-58113-096-1.

David M Berry. The computational turn: Thinking about the digital humanities. *Culture Machine*, 12, June 2012.

David M Berry and Anders Fagerjord. *Digital humanities: knowledge and critique in a digital age*. Polity Press, Cambridge, June 2017.

Steven Bethard and Dan Jurafsky. Who should I cite: learning literature search models from citation behavior. In *CIKM*, pages 609–618, 2010.

Tristan Blanc-Brude and Dominique L. Scapin. What do people recall about their documents?: implications for desktop search tools. In *Proceedings of the 12th International Conference on Intelligent User Interfaces, IUI 2007, Honolulu, Hawaii, USA, January 28-31, 2007*, pages 102–111, 2007. doi: 10.1145/1216295.1216319. URL <https://doi.org/10.1145/1216295.1216319>.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, March 2003. ISSN 1532-4435.

Francesco Bonchi, Carlos Castillo, Debora Donato, and Aristides Gionis. Topical query decomposition. In *Proceedings of the 14th ACM SIGKDD*

*international conference on Knowledge discovery and data mining*, KDD '08, pages 52–60, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-193-4. doi: 10.1145/1401890.1401902. URL <http://doi.acm.org/10.1145/1401890.1401902>.

Andrei Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, September 2002. ISSN 0163-5840. doi: 10.1145/792550.792552. URL <http://doi.acm.org/10.1145/792550.792552>.

Katriina Byström and Preben Hansen. Conceptual framework for tasks in information studies: Book reviews. *J. Am. Soc. Inf. Sci. Technol.*, 56(10):1050–1061, August 2005. ISSN 1532-2882. doi: 10.1002/asi.v56:10. URL <http://dx.doi.org/10.1002/asi.v56:10>.

Robert G. Capra and Gary Marchionini. The relation browser tool for faceted exploratory search. In *Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '08, pages 420–420, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-998-2. doi: 10.1145/1378889.1378967. URL <http://doi.acm.org/10.1145/1378889.1378967>.

Cornelia Caragea, Adrian Silvescu, Prasenjit Mitra, and C. Lee Giles. Can't see the forest for the trees? A citation recommendation system. In *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 111–114, 2013.

David Carmel, Haggai Roitman, and Naama Zwerdling. Enhancing Cluster Labeling Using Wikipedia. In *Proceedings of the 32nd international ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 09)*, pages 139–146, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-483-6. doi: <http://doi.acm.org/10.1145/1571941.1571967>.

Claudio Carpineto and Giovanni Romano. AMBIENT corpus. <http://credo.fub.it/ambient>, 2008. Accessed in February, 2012.

Claudio Carpineto, Stanisław Osiński, Giovanni Romano, and Dawid Weiss. A Survey of Web Clustering Engines. *ACM Comput. Surv.*, 41:1–38, July 2009. doi: <http://doi.acm.org/10.1145/1541880.1541884>.

Patrick L. Carr. Serendipity in the stacks: Libraries, information architecture, and the problems of accidental discovery. *College & Research Libraries*, 76(6):831–842, 2015. ISSN 2150-6701. doi: 10.5860/crl.76.6.831. URL <https://crl.acrl.org/index.php/crl/article/view/16465>.

Joseph Chee Chang, Nathan Hahn, Adam Perer, and Aniket Kittur. Search-lens: composing and capturing complex user interests for exploratory-search. In *Proceedings of the 24th International Conference on Intelligent User Interfaces, IUI 2019, Marina del Ray, CA, USA, March 17-20, 2019*, pages 498–509, 2019. doi: 10.1145/3301275.3302321. URL <https://doi.org/10.1145/3301275.3302321>.

Shui-Lung Chuang and Lee-Feng Chien. Towards automatic generation of query taxonomy: A hierarchical query clustering approach. In *Proceedings of the 2002 IEEE International Conference on Data Mining, ICDM '02*, pages 75–82, Washington, DC, USA, 2002. IEEE Computer Society. ISBN 0-7695-1754-4. URL <http://dl.acm.org/citation.cfm?id=844380.844775>.

Charles L.A. Clarke, Nick Craswell, and Ian Soboroff. Overview of the TREC 2009 Web Track. In *Proceedings of TREC 2009*, 2009.

Charles L.A. Clarke, Nick Craswell, Ian Soboroff, and Gord V. Cormack. Overview of the TREC 2010 Web Track. In *Proceedings of TREC 2010*, 2010.

Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '92*, pages 318–329, New York, NY, USA, 1992. ACM. ISBN 0-89791-523-2. doi: 10.1145/133160.133214. URL <http://doi.acm.org/10.1145/133160.133214>.

Jeffrey Dalton, Laura Dietz, and James Allan. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR '14*, pages 365–374, NY, USA, 2014. ACM. ISBN 978-1-4503-2257-7.

Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, 51(1):107–113, January 2008. ISSN 0001-0782.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

Abdigani Diriye, Max L. Wilson, Ann Blandford, and Anastasios Tombros. Revisiting exploratory search from the hci perspective. In *Proceedings of*

*the Workshop on Human-Computer Interaction and Information Retrieval 2010*, page 18, 2010.

Miles Efron, Peter Organisciak, and Katrina Fenlon. Improving retrieval of short texts through document expansion. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12*, pages 911–920, NY, USA, 2012. ACM. ISBN 978-1-4503-1472-5.

Ofer Egozi, Shaul Markovitch, and Evgeniy Gabrilovich. Concept-based information retrieval using explicit semantic analysis. *ACM Trans. Inf. Syst.*, 29(2):8:1–8:34, April 2011. ISSN 1046-8188.

Michael D. Ekstrand, Praveen Kannan, James A. Stemper, John T. Butler, Joseph A. Konstan, and John Riedl. Automatically building research reading lists. In *RecSys*, pages 159–166, 2010.

Khalid El-Arini and Carlos Guestrin. Beyond keyword search: discovering relevant scientific literature. In *KDD*, pages 439–447, 2011.

Samhaa R. El-Beltagy and Ahmed A. Rafea. KP-Miner: a keyphrase extraction system for English and Arabic documents. *Inf. Syst.*, 34(1):132–144, 2009.

D. Ellis. A behavioral approach to information retrieval system design. *J. Doc.*, 45(3):171–212, October 1989. ISSN 0022-0418. doi: 10.1108/eb026843. URL <http://dx.doi.org/10.1108/eb026843>.

Faezeh Ensan and Ebrahim Bagheri. Document retrieval model through semantic linking. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM '17*, pages 181–190, NY, USA, 2017. ACM. ISBN 978-1-4503-4675-7.

Nicolai Erbs, Iryna Gurevych, and Marc Rittberger. Bringing order to digital libraries: From keyphrase extraction to index term assignment. *D-Lib Magazine*, 19(9/10), 2013.

Paolo Ferragina and Antonio Gullì. The anatomy of SnakeT: A hierarchical clustering engine for web-page snippets. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, *Knowledge Discovery in Databases: PKDD '04*, volume 3202 of *Lecture Notes in Computer Science*, pages 506–508. Springer, 2004. ISBN 978-3-540-23108-0. doi: 10.1007/978-3-540-30116-5\_48. URL [http://dx.doi.org/10.1007/978-3-540-30116-5\\_48](http://dx.doi.org/10.1007/978-3-540-30116-5_48).

- J. Friedrich, C. Lindemann, and M. Petrifke. Utilizing query facets for search result navigation. In *2015 26th International Workshop on Database and Expert Systems Applications (DEXA)*, pages 271–275, Sept 2015. doi: 10.1109/DEXA.2015.66.
- Norbert Fuhr. A probability ranking principle for interactive information retrieval. *Information Retrieval*, 11(3):251–265, June 2008. ISSN 1386-4564. doi: 10.1007/s10791-008-9045-0. URL <http://dx.doi.org/10.1007/s10791-008-9045-0>.
- Norbert Fuhr. Some common mistakes in ir evaluation, and how they can be avoided. *SIGIR Forum*, 51(3):32–41, feb 2018. ISSN 0163-5840. doi: 10.1145/3190580.3190586. URL <https://doi.org/10.1145/3190580.3190586>.
- Norbert Fuhr, Marc Lechtenfeld, Benno Stein, and Tim Gollub. The Optimum Clustering Framework: Implementing the Cluster Hypothesis. *Information Retrieval*, 15(2):93–115, July 2012. ISSN 1386-4564. doi: <http://dx.doi.org/10.1007/s10791-011-9173-9>.
- Norbert Fuhr, Anastasia Giachanou, Gregory Grefenstette, Iryna Gurevych, Andreas Hanselowski, Kalervo Jarvelin, Rosie Jones, Yiqun Liu, Josiane Mothe, Wolfgang Nejdl, Isabella Peters, and Benno Stein. An Information Nutritional Label for Online Documents. *SIGIR Forum*, 51(3):44–66, December 2017. ISSN 0163-5840. doi: 10.1145/3190580.3190588. URL <https://sigir.org/forum/issues/december-2017/>.
- Evgeniy Gabrilovich and Shaul Markovitch. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 1606–1611, 2007.
- Debasis Ganguly, Johannes Leveling, and Gareth J.F. Jones. An lda-smoothed relevance model for document expansion: A case study for spoken document retrieval. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '13*, pages 1057–1060, NY, USA, 2013. ACM. ISBN 978-1-4503-2034-4.
- E. Garfield. When is a negative search result positive? *Essays of an Information Scientist 1*, pages 117–118, 1970.
- Filippo Geraci, Marco Pellegrini, Marco Maggini, and Fabrizio Sebastiani. Cluster Generation and Cluster Labelling for Web Snippets: A Fast and

Accurate Hierarchical Solution. In *Proceedings of the 13th Symposium on String Processing and Information Retrieval (SPIRE 06)*, pages 25–36, 2006.

Fosca Giannotti, Mirco Nanni, Dino Pedreschi, and F. Samaritani. WebCat: Automatic Categorization of Web Search Results. In *Proceedings of the Eleventh Italian Symposium on Advanced Database System (SEBD 03)*, pages 507–518. Rubettino Editore, 2003. ISBN 88-498-0629-9.

Tim Gollub and Benno Stein. Unsupervised Sparsification of Similarity Graphs. In Hermann Locarek-Junge and Claus Weihs, editors, *Classification as a Tool for Research. Selected papers from the 11th IFCS Biennial Conference and 33rd Annual Conference of the German Classification Society (GFKL)*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 71–79, Berlin Heidelberg New York, 2010. Springer. ISBN 978-3-642-10744-3. doi: [http://dx.doi.org/10.1007/978-3-642-10745-0{\\\_}7](http://dx.doi.org/10.1007/978-3-642-10745-0{\_}7).

Tim Gollub, Benno Stein, and Steven Burrows. Ousting Ivory Tower Research: Towards a Web Framework for Providing Experiments as a Service. In Bill Hersh, Jamie Callan, Yoelle Maarek, and Mark Sanderson, editors, *35th International ACM Conference on Research and Development in Information Retrieval (SIGIR 12)*, pages 1125–1126. ACM, August 2012a. ISBN 978-1-4503-1472-5. doi: <http://dx.doi.org/10.1145/2348283.2348501>.

Tim Gollub, Benno Stein, Steven Burrows, and Dennis Hoppe. TIRA: Configuring, Executing, and Disseminating Information Retrieval Experiments. In A Min Tjoa, Stephen Liddle, Klaus-Dieter Schewe, and Xiaofang Zhou, editors, *9th International Workshop on Text-based Information Retrieval (TIR 12) at DEXA*, pages 151–155, Los Alamitos, California, September 2012b. IEEE. ISBN 978-1-4673-2621-6. doi: <http://doi.ieeecomputersociety.org/10.1109/DEXA.2012.55>.

Tim Gollub, Matthias Hagen, Maximilian Michel, and Benno Stein. From Keywords to Keyqueries: Content Descriptors for the Web. In Cathal Gurrin, Gareth J.F. Jones, Diane Kelly, Udo Kruschwitz, Maarten de Rijke, Tetsuya Sakai, and Páraic Sheridan, editors, *36th International ACM Conference on Research and Development in Information Retrieval (SIGIR 13)*, pages 981–984. ACM, July 2013a. doi: <http://dx.doi.org/10.1145/2484028.2484181>. URL <http://dl.acm.org/citation.cfm?id=2484181>.

Tim Gollub, Martin Potthast, Anna Beyer, Matthias Busse, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. Recent Trends in

Digital Text Forensics and its Evaluation. In Pamela Forner, Henning Müller, Roberto Paredes, Paolo Rosso, and Benno Stein, editors, *Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 4th International Conference of the CLEF Initiative (CLEF 13)*, pages 282–302, Berlin Heidelberg New York, September 2013b. Springer. ISBN 978-3-642-40801-4. doi: <http://dx.doi.org/10.1007/978-3-642-40802-1\}28>.

Tim Gollub, Michael Völske, Matthias Hagen, and Benno Stein. Dynamic Taxonomy Composition via Keyqueries. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 14)*, pages 39–48. ACM/IEEE, September 2014. ISBN 978-1-4799-5569-5. URL <http://dl.acm.org/citation.cfm?id=2740769.2740777>.

Tim Gollub, Matthias Busse, Benno Stein, and Matthias Hagen. Keyqueries for Clustering and Labeling. In *Proceedings of the 12th Asia Information Retrieval Societies Conference (AIRS 16)*, November 2016a.

Tim Gollub, Nedim Lipka, Eunyee Koh, Erdan Genc, and Benno Stein. Topical Sequence Profiling. In A Min Tjoa, Zita Vale, and Roland Wagner, editors, *13th International Workshop on Text-based Information Retrieval (TIR 16) at DEXA*, pages 207–211. IEEE, September 2016b. ISBN 978-1-5090-3643-9. doi: <http://dx.doi.org/10.1109/DEXA.2016.39>.

Tim Gollub, Erdan Genc, Nedim Lipka, and Benno Stein. Pseudo Descriptions for Meta-Data Retrieval. In *8th International Conference on the Theory of Information Retrieval (ICTIR 2018)*, pages 139–146. ACM, September 2018a. doi: 10.1145/3234944.3234957. URL <http://doi.acm.org/10.1145/3234944.3234957>.

Tim Gollub, Martin Potthast, and Benno Stein. Shaping the Information Nutrition Label. In Dyaa Albakour, David Corney, Julio Gonzalo, Miguel Martinez, Barbara Poblete, and Andreas Valochas, editors, *Second International Workshop on Recent Trends in News Information Retrieval (NewsIR 2018) co-located with 40th European Conference on Information Retrieval (ECIR 2018)*, volume 2079 of *CEUR Workshop Proceedings*, pages 9–11. CEUR-WS.org, March 2018b. URL <http://ceur-ws.org/Vol-2079/>.

Tim Gollub, Leon Hutans, Tanveer Al Jami, and Benno Stein. Exploratory Search Pipes with Scoped Facets. In *The 2019 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '19)*. ACM, October 2019. doi: 10.1145/3341981.3344247. URL <http://doi.acm.org/10.1145/3341981.3344247>.

- Gene Golovchinsky, Abdigani Diriye, and Tony Dunnigan. The future is in the past: Designing for exploratory search. In *Proceedings of the 4th Information Interaction in Context Symposium, IIIX '12*, pages 52–61, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1282-0. doi: 10.1145/2362724.2362738. URL <http://doi.acm.org/10.1145/2362724.2362738>.
- Behzad Golshan, Theodoros Lappas, and Evmaria Terzi. SOFIA SEARCH: a tool for automating related-work search. In *SIGMOD Conference*, pages 621–624, 2012.
- Matthias Hagen and Christiane Glimm. Supporting More-Like-This Information Needs: Finding Similar Web Content in Different Scenarios. In *Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 5th International Conference of the CLEF Initiative (CLEF 14)*, pages 50–61, 2014. doi: [http://dx.doi.org/10.1007/978-3-319-11382-1\\_6](http://dx.doi.org/10.1007/978-3-319-11382-1_6). URL [http://dx.doi.org/10.1007/978-3-319-11382-1\\_6](http://dx.doi.org/10.1007/978-3-319-11382-1_6).
- Matthias Hagen and Benno Stein. Capacity-Constrained Query Formulation. In Mounia Lalmas, Joemon M. Jose, Andreas Rauber, Fabrizio Sebastiani, and Ingo Frommholz, editors, *Research and Advanced Technology for Digital Libraries. 14th European Conference on Digital Libraries (ECDL 10)*, volume 6273 of *Lecture Notes in Computer Science*, pages 384–388, Berlin Heidelberg New York, September 2010. Springer. ISBN 978-3-642-15463-8. doi: [http://dx.doi.org/10.1007/978-3-642-15464-5\\_38](http://dx.doi.org/10.1007/978-3-642-15464-5_38).
- Matthias Hagen and Benno Stein. Candidate Document Retrieval for Web-Scale Text Reuse Detection. In *18th International Symposium on String Processing and Information Retrieval (SPIRE 11)*, volume 7024 of *Lecture Notes in Computer Science*, pages 356–367, Berlin Heidelberg New York, October 2011. Springer. doi: [http://dx.doi.org/10.1007/978-3-642-24583-1\\_35](http://dx.doi.org/10.1007/978-3-642-24583-1_35).
- Matthias Hagen, Anna Beyer, Tim Gollub, Kristof Komlossy, and Benno Stein. Supporting Scholarly Search with Keyqueries. In Nicola Ferro, Fabio Crestani, Marie-Francine Moens, Josiane Mothe, Fabrizio Silvestri, Giorgio Maria Di Nunzio, Claudia Hauff, and Gianmaria Silvello, editors, *Advances in Information Retrieval. 38th European Conference on IR Research (ECIR 16)*, volume 9626 of *Lecture Notes in Computer Science*, pages 507–520, Berlin Heidelberg New York, March 2016. Springer. doi: [http://dx.doi.org/10.1007/978-3-319-30671-1\\\_\\\_37](http://dx.doi.org/10.1007/978-3-319-30671-1\_\_37).
- Matthias Hagen, Martin Potthast, Payam Adineh, Ehsan Fatehifar, and Benno Stein. Source Retrieval for Web-Scale Text Reuse Detection. In *Pro-*

*ceedings of the 26th ACM International Conference on Information and Knowledge Management (CIKM 17)*, pages 2091–2094. ACM, November 2017.

Qi He, Jian Pei, Daniel Kifer, Prasenjit Mitra, and C. Lee Giles. Context-aware citation recommendation. In *WWW*, pages 421–430, 2010.

Qi He, Daniel Kifer, Jian Pei, Prasenjit Mitra, and C. Lee Giles. Citation recommendation without author supervision. In *WSDM*, pages 755–764, 2011.

Philipp Heim, Thomas Ertl, and Jürgen Ziegler. Facet graphs: Complex semantic querying made easy. In Lora Aroyo, Grigoris Antoniou, Eero Hyvönen, Annette ten Teije, Heiner Stuckenschmidt, Liliana Cabral, and Tania Tudorache, editors, *The Semantic Web: Research and Applications*, pages 288–302, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-13486-9.

Geneva Henry. *Core Infrastructure Considerations for Large Digital Libraries*. Council on Library and Information Resources, Digital Library Federation, 2012.

Frank Hopfgartner, Allan Hanbury, Henning Müller, Ivan Eggel, Krisztian Balog, Torben Brodt, Gordon V. Cormack, Jimmy Lin, Jayashree Kalpathy-Cramer, Noriko Kando, Makoto P. Kato, Anastasia Krithara, Tim Gollub, Martin Potthast, Evelyn Viegas, and Simon Mercer. Evaluation-as-a-Service for the Computational Sciences: Overview and Outlook. *Journal of Data and Information Quality (JDIQ)*, 10(4):15:1–15:32, October 2018. doi: 10.1145/3239570.

Hao Hu, Mingxi Zhang, Zhenying He, Peng Wang, and Wei Wang. Diversifying query suggestions by using topics from wikipedia. In *Web Intelligence*, pages 139–146, 2013.

Wenyi Huang, Saurabh Kataria, Cornelia Caragea, Prasenjit Mitra, C. Lee Giles, and Lior Rokach. Recommending citations: translating papers into references. In *CIKM*, pages 1910–1914, 2012.

Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 465–474, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1869-3. doi: 10.1145/2433396.2433454. URL <http://doi.acm.org/10.1145/2433396.2433454>.

- Peter Ingwersen. Cognitive perspectives of information retrieval interaction: Elements of a cognitive ir theory. *Journal of Documentation*, 52(1):3–50, 1996. doi: 10.1108/eb026960. URL <https://doi.org/10.1108/eb026960>.
- iProspect. iProspect Blended Search Results Study, 2008. URL <http://www.iprospect.com>.
- Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.
- Jimmy, G. Zuccon, and B. Koopman. Payoffs and pitfalls in using knowledge-bases for consumer health search. *Information Retrieval Journal*, 22:350–394, 2018.
- Rosie Jones and Kristina Lisa Klinkner. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008*, pages 699–708, 2008.
- Nattiya Kanhabua, Roi Blanco, and Kjetil Nørvåg. Temporal information retrieval. *Foundations and Trends® in Information Retrieval*, 9(2):91–208, 2015. ISSN 1554-0669.
- Maryam Karimzadehgan and ChengXiang Zhai. Estimation of statistical translation models based on mutual information for ad hoc information retrieval. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '10*, pages 323–330, NY, USA, 2010. ACM. ISBN 978-1-4503-0153-4.
- Maryam Karimzadehgan and ChengXiang Zhai. Axiomatic analysis of translation language model for information retrieval. In *Proceedings of the 34th European Conference on Advances in Information Retrieval, ECIR'12*, pages 268–280, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-28996-5.
- Saurabh Kataria, Prasenjit Mitra, and Sumit Bhatia. Utilizing context in generative bayesian models for linked corpus. In *AAAI*, 2010.
- Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval '10*, pages 21–26, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

- tion for Computational Linguistics. URL <http://dl.acm.org/citation.cfm?id=1859664.1859668>.
- W. Kintsch. *Comprehension: A Paradigm for Cognition*. Cambridge University Press, 1998. ISBN 9780521629867. URL <https://books.google.de/books?id=LuycnLrY3k8C>.
- Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 170–178, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. ISBN 1-55860-486-3. URL <http://dl.acm.org/citation.cfm?id=645526.657130>.
- Weize Kong and James Allan. Extending faceted search to the general web. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 839–848, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2598-1. doi: 10.1145/2661829.2661964. URL <http://doi.acm.org/10.1145/2661829.2661964>.
- Austin C. Kozlowski, Matt Taddy, and James A. Evans. The geometry of culture: Analyzing meaning through word embeddings. *CoRR*, abs/1803.09288, 2018. URL <http://arxiv.org/abs/1803.09288>.
- Onur Küçüktunç, Erik Saule, Kamer Kaya, and Ümit V. Catalyürek. Theadvisor: a webservice for academic recommendation. In *JCDL*, pages 433–434, 2013.
- C.C. Kuhlthau. *Seeking Meaning: A Process Approach to Library and Information Services*. Information management, policy, and services. Ablex Publishing Corporation, 1993. ISBN 9781567500196. URL <https://books.google.de/books?id=R64mAQAAMAAJ>.
- Bill Kules and Robert Capra. Creating exploratory tasks for a faceted search interface. In *Proceedings of the Second Workshop on Human-Computer Interaction (HCIR*, 2008.
- Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning (ICML2014)*, pages 1188–1196, 2014.
- Cheng Li, Paul Resnick, and Qiaozhu Mei. Multiple queries as bandit arms. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, pages 1089–1098, New York, NY, USA,

2016. ACM. ISBN 978-1-4503-4073-1. doi: 10.1145/2983323.2983816. URL <http://doi.acm.org/10.1145/2983323.2983816>.

Ruirui Li, Ben Kao, Bin Bi, Reynold Cheng, and Eric Lo. Dqr: A probabilistic approach to diversified query recommendation. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 16–25, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1156-4. doi: 10.1145/2396761.2396768. URL <http://doi.acm.org/10.1145/2396761.2396768>.

Nedim Lipka, Tim Gollub, and Eunyee Koh. Organizing electronically stored files using an automatically generated storage hierarchy, October 13 2020. US Patent 10,803,037.

Xiaoyong Liu and W. Bruce Croft. Cluster-based retrieval using language models. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 186–193, NY, USA, 2004. ACM. ISBN 1-58113-881-4.

Xueqing Liu, Yangqiu Song, Shixia Liu, and Haixun Wang. Automatic taxonomy construction from keywords. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1433–1441, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1462-6. doi: 10.1145/2339530.2339754. URL <http://doi.acm.org/10.1145/2339530.2339754>.

Avishay Livne, Vivek Gokuladas, Jaime Teevan, Susan Dumais, and Eytan Adar. CiteSight. In *SIGIR*, 2014.

Yang Lu, Jing He, Dongdong Shan, and Hongfei Yan. Recommending citations with translation model. In *CIKM*, pages 2017–2020, 2011.

Hao Ma, Michael R. Lyu, and Irwin King. Diversifying query suggestion results. In *AAAI*, 2010.

Shulei Ma, Zhi-Hong Deng, and Yunlun Yang. An unsupervised multi-document summarization framework based on neural document model. In *COLING*, 2016.

Kaustubh Mani, Ishan Verma, and Lipika Dey. Multi-document summarization using distributed bag-of-words model. *CoRR*, abs/1710.02745, 2017.

- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008. ISBN 0521865719, 9780521865715.
- L. Manovich. *Cultural Analytics*. Knowledge Unlatched Select 2017. MIT Press, 2020. ISBN 9780262037105.
- James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela Hung Byers. Big data: The next frontier for innovation, competition, and productivity. Technical report, McKinsey Global Institute, 2011.
- Gary Marchionini. Exploratory search: From finding to understanding. *Commun. ACM*, 49(4):41–46, April 2006. ISSN 0001-0782. doi: 10.1145/1121949.1121979. URL <http://doi.acm.org/10.1145/1121949.1121979>.
- Lokman I. Meho and Helen R. Tibbo. Modeling the information-seeking behavior of social scientists: Ellis's study revisited. *J. Am. Soc. Inf. Sci. Technol.*, 54(6):570–587, April 2003. ISSN 1532-2882. doi: 10.1002/asi.10244. URL <http://dx.doi.org/10.1002/asi.10244>.
- Sven Meyer zu Eißen, Benno Stein, and Martin Potthast. The Suffix Tree Document Model Revisited. In Klaus Tochtermann and Hermann Maurer, editors, *5th International Conference on Knowledge Management (I-KNOW 05)*, Journal of Universal Computer Science, pages 596–603, Graz, Austria, July 2005. Know-Center.
- Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, et al. Quantitative analysis of culture using millions of digitized books. *science*, 331(6014):176–182, 2011.
- R. Mihalcea and P. Tarau. TextRank: Bringing order into texts. In *Proceedings of EMNLP-04 and the 2004 Conference on Empirical Methods in Natural Language Processing*, July 2004.
- F. Moretti. *Distant Reading*. Verso, 2013. ISBN 9781781680841.
- Ramesh Nallapati, Amr Ahmed, Eric P. Xing, and William W. Cohen. Joint latent topic models for text and citations. In *KDD*, pages 542–550, 2008.
- Cristiano Nascimento, Alberto H. F. Laender, Altigran Soares da Silva, and Marcos André Gonçalves. A source independent framework for research paper recommendation. In *JCDL*, pages 297–306, 2011.

Roberto Navigli, Paola Velardi, and Stefano Faralli. A graph-based algorithm for inducing lexical taxonomies from scratch. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three, IJCAI'11*, pages 1872–1877. AAAI Press, 2011. ISBN 978-1-57735-515-1. doi: 10.5591/978-1-57735-516-8/IJCAI11-313. URL <http://dx.doi.org/10.5591/978-1-57735-516-8/IJCAI11-313>.

Stanislaw Osiński, Jerzy Stefanowski, and Dawid Weiss. Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition. In *Proceedings of the International Conference on Intelligent Information Processing and Web Mining (IIPWM 04), Advances in Soft Computing*, pages 359–368. Springer, 2004. ISBN 3-540-21331-7.

Jeremy Pickens, Matthew Cooper, and Gene Golovchinsky. Reverted indexing for feedback and expansion. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 1049–1058, 2010.

Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *SIGIR'98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, New York, NY, USA, 1998. ACM Press. ISBN 1-58113-015-5. doi: <http://doi.acm.org/10.1145/290941.291008>.

Hoifung Poon and Pedro Domingos. Unsupervised ontology induction from text. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pages 296–305, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL <http://dl.acm.org/citation.cfm?id=1858681.1858712>.

Martin Potthast, Benno Stein, and Maik Anderka. A Wikipedia-Based Multilingual Retrieval Model. In Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven, and Ryen W. White, editors, *Advances in Information Retrieval. 30th European Conference on IR Research (ECIR 08)*, volume 4956 of *Lecture Notes in Computer Science*, pages 522–530, Berlin Heidelberg New York, 2008. Springer. ISBN 978-3-540-78645-0. doi: [http://dx.doi.org/10.1007/978-3-540-78646-7\\_51](http://dx.doi.org/10.1007/978-3-540-78646-7_51).

Martin Potthast, Martin Trenkmann, and Benno Stein. Netspeak: Assisting Writers in Choosing Words. In Cathal Gurrin, Yulan He, Gabriella Kazai, Udo Kruschwitz, Suzanne Little, Thomas Roelleke, Stefan M.

Rüger, and Keith van Rijsbergen, editors, *Advances in Information Retrieval. 32nd European Conference on Information Retrieval (ECIR 2010)*, volume 5993 of *Lecture Notes in Computer Science*, page 672, Berlin Heidelberg New York, March 2010. Springer. ISBN 978-3-642-12274-3. doi: 10.1007/978-3-642-12275-0\\_75.

Martin Potthast, Tim Gollub, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. Improving the Reproducibility of PAN's Shared Tasks: Plagiarism Detection, Author Identification, and Author Profiling. In Evangelos Kanoulas, Mihai Lupu, Paul Clough, Mark Sanderson, Mark Hall, Allan Hanbury, and Elaine Toms, editors, *Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 5th International Conference of the CLEF Initiative (CLEF 2014)*, pages 268–299, Berlin Heidelberg New York, September 2014. Springer. ISBN 978-3-319-11381-4. doi: 10.1007/978-3-319-11382-1\\_22.

Martin Potthast, Tim Gollub, Matthias Hagen, and Benno Stein. The Clickbait Challenge 2017: Towards a Regression Model for Clickbait Strength. *CoRR*, abs/1812.10847, December 2018a. URL <https://arxiv.org/abs/1812.10847>.

Martin Potthast, Tim Gollub, Kristof Komlossy, Sebastian Schuster, Matti Wiegmann, Erika Patricia Garces Fernandez, Matthias Hagen, and Benno Stein. Crowdsourcing a Large Corpus of Clickbait on Twitter. In Emily M. Bender, Leon Derczynski, and Pierre Isabelle, editors, *27th International Conference on Computational Linguistics (COLING 2018)*, pages 1498–1507. The COLING 2018 Organizing Committee, August 2018b. URL <https://www.aclweb.org/anthology/C18-1127>.

Martin Potthast, Tim Gollub, Matti Wiegmann, and Benno Stein. TIRA Integrated Research Architecture. In Nicola Ferro and Carol Peters, editors, *Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF*. Springer, 2019.

Jiani Qu, Anny Marleen Hißbach, Tim Gollub, and Martin Potthast. Towards Crowdsourcing Clickbait Labels for YouTube Videos. In Yiling Chen and Gabrielle Kazai, editors, *6th AAAI Conference on Human Computation and Crowdsourcing (HCOMP 2018)*, July 2018.

Yan Qu and George W. Furnas. Model-driven formative evaluation of exploratory search: A study under a sensemaking framework. *Inf. Process. Manage.*, 44(2):534–555, March 2008. ISSN 0306-4573. doi: 10.1016/j.ipm.2007.09.006. URL <http://dx.doi.org/10.1016/j.ipm.2007.09.006>.

- Urbano Reviglio. Serendipity as an emerging design principle of the info-sphere: challenges and opportunities. *Ethics and Information Technology*, 21:151–166, 2018.
- Cornelis Joost van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 1979. ISBN 0408709294.
- S. E. Robertson. *The probability ranking principle in IR*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997. ISBN 1-55860-454-5.
- S. E. Robertson and Karen Sparck-Jones. Relevance Weighting of Search Terms. *American Society for Information Science*, 27(3):129–146, 1976.
- S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 232–241, New York, NY, USA, 1994. Springer-Verlag New York, Inc. ISBN 0-387-19889-X.
- Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, April 2009. ISSN 1554-0669. doi: 10.1561/1500000019. URL <http://dx.doi.org/10.1561/1500000019>.
- Stephen E. Robertson, Hugo Zaragoza, and Michael J. Taylor. Simple BM25 extension to multiple weighted fields. In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, CIKM '04, pages 42–49, New York, NY, USA, 2004. ACM. ISBN 1-58113-874-1. doi: 10.1145/1031171.1031181. URL <http://doi.acm.org/10.1145/1031171.1031181>.
- J. J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. 1971.
- Daniel E. Rose and Danny Levinson. Understanding user goals in web search. In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, pages 13–19, New York, NY, USA, 2004. ACM. ISBN 1-58113-844-X. doi: 10.1145/988672.988675. URL <http://doi.acm.org/10.1145/988672.988675>.
- Tuukka Ruotsalo, Kumaripaba Athukorala, Dorota Glowacka, Ksenia Konyushkova, Antti Oulasvirta, Samuli Kaipiainen, Samuel Kaski, and Giulio Jacucci. Supporting exploratory search tasks with interactive user

- modeling. In *Proceedings of the 76th ASIS&T Annual Meeting: Beyond the Cloud: Rethinking Information Boundaries*, ASIST '13, pages 39:1–39:10, Silver Springs, MD, USA, 2013. American Society for Information Science. ISBN 0-87715-545-3. URL <http://dl.acm.org/citation.cfm?id=2655780.2655819>.
- D. M. Russell, D. Tang, M. Kellar, and R. Jeffries. Task behaviors during web search: The difficulty of assigning labels. In *Proceedings of the 42Nd Hawaii International Conference on System Sciences*, HICSS '09, pages 1–5, Washington, DC, USA, 2009. IEEE Computer Society. ISBN 978-0-7695-3450-3. doi: 10.1109/HICSS.2009.417. URL <http://dx.doi.org/10.1109/HICSS.2009.417>.
- G.M. Sacco and Y. Tzitzikas. *Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience*. The Information Retrieval Series. Springer, 2009. ISBN 9783642023590.
- G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.
- Gerard Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. *Commun. ACM*, 18(11):613–620, 1975.
- Mark Sanderson, Monica Lestari Paramita, Paul Clough, and Evangelos Kanoulas. Do user preferences and evaluation measures line up? In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 555–562, NY, USA, 2010. ACM. ISBN 978-1-4503-0153-4.
- Ugo Scaiella, Paolo Ferragina, Andrea Marino, and Massimiliano Ciaramita. Topical clustering of search results. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 223–232, New York, NY, USA, 2012.
- Henning Schmidgen, Benno Stein, Tim Gollub, Michael Braun, and Jan Willmann. Philosophische Körper. Von digitalem Text zu greifbarem Material. *Zeitschrift für digitale Geisteswissenschaften (ZfdG)*, 6(6), May 2021. ISSN 2510-1358. doi: 10.17175/2021\_001. URL [https://zfdg.de/2021\\_001](https://zfdg.de/2021_001).
- m.c. schraefel, Max Wilson, Alistair Russell, and Daniel A. Smith. mspace: Improving information access to multimedia domains with multimodal

exploratory search. *Commun. ACM*, 49(4):47–49, April 2006. ISSN 0001-0782. doi: 10.1145/1121949.1121980. URL <http://doi.acm.org/10.1145/1121949.1121980>.

Garrick Sherman and Miles Efron. Document expansion using external collections. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17*, pages 1045–1048, NY, USA, 2017. ACM. ISBN 978-1-4503-5022-8.

Amit Singhal and Fernando Pereira. Document expansion for speech retrieval. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '99*, pages 34–41, NY, USA, 1999. ACM. ISBN 1-58113-096-1.

Yang Song, Dengyong Zhou, and Li-wei He. Post-ranking query suggestion by diversifying search results. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '11*, pages 815–824, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0757-4. doi: 10.1145/2009916.2010025. URL <http://doi.acm.org/10.1145/2009916.2010025>.

Benno Stein and Sven Meyer zu Eißen. Topic Identification: Framework and Application. In Klaus Tochtermann and Hermann Maurer, editors, *4th International Conference on Knowledge Management (I-KNOW 04)*, Journal of Universal Computer Science, pages 353–360, Graz, Austria, July 2004. Know-Center.

Benno Stein, Tim Gollub, and Dennis Hoppe. Beyond Precision@10: Clustering the Long Tail of Web Search Results. In Bettina Berendt, Arjen de Vries, Wenfei Fan, Craig Macdonald, Iadh Ounis, and Ian Ruthven, editors, *20th ACM International Conference on Information and Knowledge Management (CIKM 11)*, pages 2141–2144. ACM, October 2011. ISBN 978-1-4503-0717-8. doi: <http://doi.acm.org/10.1145/2063576.2063910>.

Benno Stein, Tim Gollub, and Dennis Hoppe. Search Result Presentation Based on Faceted Clustering. In Xuewen Chen, Guy Lebanon, Haixun Wang, and Mohammed J. Zaki, editors, *21st ACM International Conference on Information and Knowledge Management (CIKM 12)*, pages 1940–1944. ACM, October 2012. ISBN 978-1-4503-1156-4. doi: <http://doi.acm.org/10.1145/2396761.2398548>.

Benno Stein, Tim Gollub, and Maik Anderka. *Encyclopedia of Social Network Analysis and Mining (ESNAM)*, chapter Retrieval Models, pages 1–7.

- Springer, Berlin Heidelberg New York, August 2017. ISBN 978-1-4614-7163-9. doi: 10.1007/978-1-4614-7163-9\_117-1.
- Kazunari Sugiyama and Min-Yen Kan. Exploiting potential citation papers in scholarly paper recommendation. In *JCDL*, pages 153–162, 2013.
- Zareen Saba Syed, Tim Finin, and Anupam Joshi. Wikipedia as an ontology for describing documents. In *ICWSM*, 2008.
- Jie Tang and Jing Zhang. A discriminative approach to topic-based citation recommendation. In *PAKDD*, pages 572–579, 2009.
- Xuewei Tang, Xiaojun Wan, and Xun Zhang. Cross-language context-aware citation recommendation in scientific articles. In *SIGIR*, pages 817–826, 2014.
- Tao Tao, Xuanhui Wang, Qiaozhu Mei, and ChengXiang Zhai. Language model information retrieval with document expansion. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, HLT-NAACL '06*, pages 407–414, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.
- A.G. Taylor and D.N. Joudrey. *The Organization of Information*. Library and Information Science Text Series. Libraries Unlimited, Incorporated, 2009. ISBN 9781591585862.
- Jaime Teevan, Kevyn Collins-Thompson, Ryen W. White, Susan T. Dumais, and Yubin Kim. Slow search: Information retrieval without time constraints. In *Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval, HCIR '13*, pages 1:1–1:10, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2570-7. doi: 10.1145/2528394.2528395. URL <http://doi.acm.org/10.1145/2528394.2528395>.
- Elaine Toms. Serendipitous information retrieval. In *Proceedings of the First DELOS Network of Excellence Workshop on Information Seeking, Searching and Querying in Digital Libraries*, pages 11–12, 2000.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology -*

*Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1073445.1073478. URL <http://dx.doi.org/10.3115/1073445.1073478>.

Pucktada Treeratpituk and Jamie Callan. An experimental study on automatically labeling hierarchical clusters using statistical features. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 707–708, New York, NY, USA, 2006. ACM. ISBN 1-59593-369-7. doi: 10.1145/1148170.1148328. URL <http://doi.acm.org/10.1145/1148170.1148328>.

George Tsatsaronis and Vicky Panagiotopoulou. A generalized vector space model for text retrieval based on semantic relatedness. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, EACL '09, pages 70–78, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

Suppawong Tuarob, Line C. Pouchard, and C. Lee Giles. Automatic tag recommendation for metadata annotation using probabilistic topic modeling. In *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '13, pages 239–248, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2077-1. doi: 10.1145/2467696.2467706. URL <http://doi.acm.org/10.1145/2467696.2467706>.

Daniel Tunkelang. *Faceted Search*, volume 1 of *Synthesis Lectures on Information Concepts, Retrieval, and Services*. Morgan & Claypool Publishers, 2009. doi: <http://dx.doi.org/10.2200/S00190ED1V01Y200904ICR005>.

Andrew Turpin, Yohannes Tsegay, David Hawking, and Hugh E. Williams. Fast generation of result snippets in web search. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 127–134, NY, USA, 2007. ACM. ISBN 978-1-59593-597-7.

Yannis Tzitzikas, Nikos Manolis, and Panagiotis Papadakos. Faceted exploration of RDF/S datasets: a survey. *Journal of Intelligent Information Systems*, 48(2):329–364, Apr 2017. ISSN 1573-7675. doi: 10.1007/s10844-016-0413-8. URL <https://doi.org/10.1007/s10844-016-0413-8>.

Ted Underwood. A genealogy of distant reading. *Digital Humanities Quarterly*, 11(2), 2017. URL <http://www.digitalhumanities.org/dhq/vol/11/2/000317/000317.html>.

Pertti Vakkari. Exploratory searching as conceptual exploration. In *Proceedings of 4th workshop on Human-computer interaction and information retrieval*, pages 24–27, 2010.

Vijay V. Vazirani. *Approximation Algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001. ISBN 3-540-65367-8.

Michael Völske, Tim Gollub, Matthias Hagen, and Benno Stein. A Keyquery-Based Classification System for CORE. In Laurence Lanham, editor, *3rd International Workshop on Mining Scientific Publications (WOSP 2014)*, volume 20. Corporation for National Research Initiatives (CNRI), September 2014. doi: <http://dx.doi.org/10.1045/november14-voelske>. URL <http://www.dlib.org/dlib/november14/voelske/11voelske.html>.

Michael Völske, Janek Bevendorff, Johannes Kiesel, Benno Stein, Maik Fröbe, Matthias Hagen, and Martin Potthast. Web Archive Analytics. In Ralf H. Reussner, Anne Kozolek, and Robert Heinrich, editors, *50. Jahrestagung der Gesellschaft für Informatik, INFORMATIK 2020*, volume P-307 of *Lecture Notes in Informatics, LNI*, pages 61–72. Gesellschaft für Informatik, GI, January 2021. ISBN 978-3-88579-701-2. doi: 10.18420/inf2020\\_05. URL <https://dl.gi.de/handle/20.500.12116/34759>.

Jörg Waitelonis, Claudia Exeler, and Harald Sack. Enabled generalized vector space model to improve document retrieval. In *NLP-DBPEDIA@ISWC*, 2015.

Chong Wang and David M. Blei. Collaborative topic modeling for recommending scientific articles. In *KDD*, pages 448–456, 2011.

Bifan Wei, Jun Liu, Qinghua Zheng, Wei Zhang, Chencheng Wang, and Bei Wu. Df-miner: Domain-specific facet mining by leveraging the hyperlink structure of wikipedia. *Knowledge-Based Systems*, 77:80 – 91, 2015a. ISSN 0950-7051. doi: <http://dx.doi.org/10.1016/j.knosys.2015.01.001>. URL <http://www.sciencedirect.com/science/article/pii/S0950705115000088>.

Tingting Wei, Yonghe Lu, Huiyou Chang, Qiang Zhou, and Xianyu Bao. A semantic approach for text clustering using wordnet and lexical chains. *Expert Systems with Applications*, 42(4):2264 – 2275, 2015b. ISSN 0957-4174. doi: <http://dx.doi.org/10.1016/j.eswa.2014.10.023>. URL <http://www.sciencedirect.com/science/article/pii/S0957417414006472>.

- Xing Wei and W. Bruce Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06*, pages 178–185, NY, USA, 2006. ACM. ISBN 1-59593-369-7.
- D. Weinberger. *Everything Is Miscellaneous: The Power of the New Digital Disorder*. Henry Holt and Company, 2007. ISBN 9781429927956.
- Dawid Weiss. *Descriptive Clustering as a Method for Exploring Text Collections*. PhD thesis, Politechnika Poznańska and Instytut Informatyki, Poznań University of Technology, Poland, 2006.
- Ryen W. White and Resa A. Roth. *Exploratory Search: Beyond the Query-Response Paradigm*. Synthesis Lectures on Information Concepts, Retrieval, and Services Series. Morgan & Claypool, 2009. ISBN 9781598297836.
- Barbara M. Wildemuth and Luanne Freund. Assigning search tasks designed to elicit exploratory search behaviors. In *Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval, HCIR '12*, pages 4:1–4:10, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1796-2. doi: 10.1145/2391224.2391228. URL <http://doi.acm.org/10.1145/2391224.2391228>.
- Max L. Wilson and m.c. schraefel. mspace: What do numbers and totals mean in a flexible semantic browser. 2006. URL <https://eprints.soton.ac.uk/262666/>.
- Max L. Wilson, Bill Kules, m. c. schraefel, and Ben Shneiderman. From keyword search to exploration: Designing future search interfaces for the web. *Found. Trends Web Sci.*, 2(1):1–97, January 2010. ISSN 1555-077X. doi: 10.1561/1800000003. URL <http://dx.doi.org/10.1561/1800000003>.
- S. K. M. Wong, Wojciech Ziarko, and Patrick C. N. Wong. Generalized vector spaces model in information retrieval. In *Proceedings of the 8th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '85*, pages 18–25, NY, USA, 1985. ACM. ISBN 0-89791-159-8.
- Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 14(1):1–37, December

2007. ISSN 0219-1377. doi: 10.1007/s10115-007-0114-2. URL <http://dx.doi.org/10.1007/s10115-007-0114-2>.

Ellery Wulczyn and Dario Taraborelli. Wikipedia clickstream. *figshare*, 2015. doi: 10.6084/m9.figshare.1305770.

G.H. Yang, M. Sloan, and J. Wang. *Dynamic Information Retrieval Modeling*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2016. ISBN 9781627055260. URL <https://books.google.de/books?id=xRV8DAAAQBAJ>.

Ka-Ping Yee, Kirsten Swearingen, Kevin Li, and Marti Hearst. Faceted metadata for image search and browsing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 401–408, New York, NY, USA, 2003. ACM. ISBN 1-58113-630-7. doi: 10.1145/642611.642681. URL <http://doi.acm.org/10.1145/642611.642681>.

Xing Yi and James Allan. A comparative study of utilizing topic models for information retrieval. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, pages 29–41, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-00957-0.

Oren Zamir and Oren Etzioni. Web document clustering: A feasibility demonstration. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 46–54, New York, NY, USA, 1998. ACM. ISBN 1-58113-015-5. doi: 10.1145/290941.290956. URL <http://doi.acm.org/10.1145/290941.290956>.

Hugo Zaragoza, B. Barla Cambazoglu, and Ricardo Baeza-Yates. Web search solved?: All result rankings the same? In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 529–538, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0099-5. doi: 10.1145/1871437.1871507. URL <http://doi.acm.org/10.1145/1871437.1871507>.

Cheng X. Zhai, William W. Cohen, and John Lafferty. Beyond Independent Relevance: Methods and Evaluation Metrics for Subtopic Retrieval. In *Proceedings of the 26th annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 03)*, pages 10–17, New York, NY, USA, 2003. ACM. ISBN 1-58113-646-3. doi: <http://doi.acm.org/10.1145/860435.860440>.

Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '01*, pages 334–342, NY, USA, 2001. ACM. ISBN 1-58113-331-6.

Guido Zuccon, Bevan Koopman, Peter Bruza, and Leif Azzopardi. Integrating and evaluating neural word embeddings in information retrieval. In *Proceedings of the 20th Australasian Document Computing Symposium, ADCS '15*, pages 12:1–12:8, NY, USA, 2015. ACM. ISBN 978-1-4503-4040-3.