UNIVERSITY OF TWENTE.

**Faculty of Electrical Engineering, Mathematics & Computer Science**

# Context Based Personalized Ranking in Academic Search

**Alexandru Constantin Serban**

**M.Sc. Thesis**
**August 2017**

**Supervisors:**
dr. ir. Djoerd Hiemstra
dr. Doina Bucur
drs. Spencer De Groot

Database Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

# Abstract

A criticism of search engines is that queries return the same results for users who send exactly the same query, with distinct information needs. Personalized search is considered a solution as search results are re-evaluated based on user preferences or activity. Instead of relying on the unrealistic assumption that people will precisely specify their intent when searching, the user profile is exploited to re-rank the results.

This thesis focuses on two problems related to academic information retrieval systems. The first part is dedicated to data sets for search engine evaluation. Test collections consists of documents, a set of information needs, also called topics, queries that represent the data structure sent to the information retrieval tool and relevance judgements for the top documents retrieved from the collection.

Relevance judgements are difficult to gather because the process involves manual work. We propose an automatic method to generate queries from the content of a scientific article and evaluate the relevant results. A test collection is generated, but its power to discriminate between relevant and non relevant results is limited.

In the second part of the thesis Scopus performance is improved through personalization. We focus on the academic background of researchers that interact with Scopus since information about their academic profile is already available. Two methods for personalized search are investigated.

At first, the connections between academic entities, expressed as a graph structure, are used to evaluate how relevant a result is to the user. We use SimRank, a similarity measure for entities based on their relationships with other entities. Secondly, the semantic structure of documents is exploited to evaluate how meaningful a document is for the user. A topic model is trained to reflect the user's interests in research areas and how relevant the search results are.

In the end both methods are merged with the initial Scopus rank. The results of a user study show a constant performance increase for the first $10$ results.

# Contents

**Abstract** iii

**List of Figures** vii

**List of Tables** x

**List of acronyms** xiii

**1 Introduction** 1
- 1.1 Problem identification and motivation . . . 1
- 1.2 Research methodology and overview . . . 2
- 1.3 Design science research goals and design problems . . . 3
- 1.4 Thesis outline . . . 4
- 1.5 Implementation Considerations . . . 5

**2 Background** 7
- 2.1 Introduction . . . 7
- 2.2 Definitions and notations . . . 7
  - 2.2.1 Graph . . . 8
  - 2.2.2 Context . . . 10
  - 2.2.3 User . . . 10
  - 2.2.4 Entity similarity and relevance . . . 10
  - 2.2.5 Topics . . . 10
- 2.3 Graph Walk models . . . 11
  - 2.3.1 PageRank . . . 11
  - 2.3.2 Personalized PageRank . . . 13
  - 2.3.3 HITS . . . 13
  - 2.3.4 Graph similarity measures . . . 14
- 2.4 Bibliographic graphs . . . 15
  - 2.4.1 Entities - graph nodes . . . 15
  - 2.4.2 Relationships between entities - graph edge types . . . 15
- 2.5 Topic models . . . 18

2.5.1 Probabilistic Latent Semantic Analysis . . . . . 19
2.5.2 Latent Dirichlet Allocation . . . . . 21

**3 Ranking academic search engines with automatic relevance judgements 25**
3.1 Introduction . . . . . 25
3.2 Related work . . . . . 26
3.3 Treatment design . . . . . 28
3.3.1 Characteristics of academic queries . . . . . 28
3.3.2 Performance measures . . . . . 30
3.3.3 Treatment proposal . . . . . 30
3.4 Treatment validation . . . . . 31
3.4.1 Dataset . . . . . 32
3.4.2 Results . . . . . 33
3.4.3 Discussion . . . . . 37

**4 Personalized Ranking in Academic Search 39**
4.1 Introduction . . . . . 39
4.2 Related work . . . . . 39
4.2.1 Personalization using search history . . . . . 41
4.2.2 Collaborative search . . . . . 42
4.2.3 Result clustering . . . . . 43
4.2.4 Hiperlink personalization . . . . . 43
4.2.5 Ranking in academic search . . . . . 44
4.2.6 Personalized academic search . . . . . 45
4.3 Treatment Design . . . . . 46
4.3.1 Graph models . . . . . 47
4.3.2 Topic models . . . . . 49
4.3.3 Rank fusion . . . . . 51
4.3.4 Treatment Proposal . . . . . 54
4.4 Treatment Validation . . . . . 55
4.4.1 Experiment Setup . . . . . 55
4.4.2 Performance measures and relevance judgements . . . . . 56
4.4.3 Results . . . . . 57
4.4.4 Discussion . . . . . 62

**5 Conclusions and future work 65**
5.1 Conclusions . . . . . 65
5.2 Threats to validity . . . . . 66
5.3 Future research . . . . . 67

**References** **69**

**Appendices**

**A Scopus Data** **79**
A.1 Query examples . . . . . 79
A.2 Scopus Screenshots . . . . . 79
A.3 Scopus Subject areas . . . . . 80
A.3.1 Author subject areas . . . . . 80
A.3.2 Article subject areas . . . . . 80

# List of Figures


2.1 An example of graph scheme. . . . . 9
2.2 Simplified relational model for graph entities. . . . . 16
2.3 Bibliographic coupling. . . . . 17
2.4 Co-citation. . . . . 17
2.5 Co-authorship. . . . . 17
2.6 Co-source. . . . . 17
2.7 Co-authorship networks as in [1]. . . . . 18
2.8 Input and Output of a Topic Model. . . . . 20
2.9 Graphical model representation of LDA. . . . . 21

3.1 The number of queries that returned at least one reference (blue) vs. the number of queries that returned no references (red) for test set. . . 33
3.2 The number of queries that returned at least one reference (blue) vs. the number of queries that returned no references (red) for validation set. . . . . 33

4.1 Personalization process where the user model is used during the retrieval process $(a)$, in a distinct, re-ranking, step $(b)$ or during query pre-processing $(c)$. . . . . 40
4.2 Comparative precision for using a graph similarity measure to re-rank the top 100 results retrieved by Scopus. . . . . 49
4.3 Comparative precision results for using an author-topic model to re-rank the top 100 results retrieved by Scopus (1) and the expanded result set by context at 1 (2). . . . . 52
4.4 The impact on initial precision for different ranking fusion methods. . . 53
4.5 The impact of $k$ constant on RRF ranking fusion method. . . . . 54
4.6 Result sets intersection where red represents the Scopus results, blue the results retrieved through re-ranking, yellow the common results for Scopus and re-ranking and dark blue represents the proportion of results added from the result's context. . . . . 58

4.7 Precision at different thresholds for user generated queries and automatic queries formulated through methods (1) and (6) from Section 3.3.3. . . . . 59
4.8 The impact of personalization on the user provided queries against the article references. . . . . 60

A.1 Scopus screenshot showing articles indexed earlier than their publishing year. . . . . 80

# List of Tables

3.1 Precision on test set. . . . . 34
3.2 Precision on validation set. . . . . 35
3.3 Recall on test set . . . . . 36
3.4 Recall on validation set . . . . . 36

4.1 Types of personalized search classified by the type of feedback used to learn the user profile and keep it updated. . . . . 41
4.2 NDCG results for user study. . . . . 60

A.1 Examples of queries. . . . . 79

# List of acronyms

| | |
|---|---|
| **RQ** | Research Question |
| **CS** | Computer Science |
| **IR** | Information Retrieval |
| **HITS** | Hyper linked induced topic search |
| **DL** | Digital Library |
| **NLP** | Natural Language Processing |
| **LDA** | Latent Dirichlet Allocation |
| **pLSA** | Probabilistic Latent Semantic Analysis |
| **EM** | Expectation Maximisation |
| **MCMC** | Markov Chain Monte Carlo Model |
| **TREC** | Text Retrieval Conference |
| **ODP** | Open Directory Project |
| **RRF** | Reciprocal Rank Fusion |
| **DCG** | Discounted Cumulative Gain |
| **nDCG** | Normalised Discounted Cumulative Gain |
| **IDCG** | Ideal Discounted Cumulative Gain |
| **MAD** | Median Absolute Deviation |
| **TF-IDF** | Term frequency-Inverse document frequency |

# Chapter 1

# Introduction

## 1.1 Problem identification and motivation

Search engines have changed the way users find, interact with and access information. Instead of searching through piles of books and printed media, users can quickly search for and discover relevant information by issuing a *query* to a *search engine*. Given a query, a typical search engine returns a long list of *ranked* items displayed over a number of pages.

The increase of searchable information in web environments makes ranking an intrinsic component of search engines and poses a big number of challenges. Briefly, ranking sorts search results according to pre-defined criteria in order to provide the most relevant documents for an information need. It is the quality of such algorithms that can determine preferences for search tools; a weak performance in ranking can force users to scroll through several pages in order to find the desired information, affecting the user's search experience and overall satisfaction.

Another criticism of search engines is that queries return the same result for each user [2]. Different users can use exactly the same query with distinct information needs [3], [4]. For example, issuing the query *ranking*, an information retrieval researcher can show interest in search engine ranking while an educational researcher can look for ranking models of universities.

Moreover, information needs for a user might change with time: one can use the same query, *ranking*, to search for page ranking or personalised ranking. Without placing the query in a context, it gets impossible to judge the correct sense. In academic search, time dependency becomes even tighter as a user can show interest in papers that represent the foundation of a field, while another can search only for recent publications. We find the value of information and the relevance of specific sources not intrinsic properties of documents, therefore, they can only be assessed in relation to the embedding task environment [5].

Personalized search is considered a solution to these problems as search re-

sults are re-evaluated based on user preferences or activity. Instead of relying on the unrealistic assumption that people will precisely specify their intent when searching [6], the user's profile is exploited to re-rank the results. In order to provide personalization services, user preferences are manually collected or automatically extracted through mining techniques. Various personalization techniques for general web search can be found in the literature, however, little to no attention is dedicated to academic search.

The main motivation behind this research is to improve academic search ranking algorithms through personalization. We focus on the academic background of researchers that interact with information retrieval tools since information about their academic profile is already available for search engines.

## 1.2 Research methodology and overview

This research is organised and conducted following the design science methodology explained in [7]. The framework analyses two parts of design science, design and investigation, corresponding to two kinds of research problems: design problems and knowledge problems. Design problems call for a change in the real world and require an analysis of actual or hypothetical stakeholder goals. A solution is a design and there are usually many different solutions. Knowledge questions, however, do not call for a change in the world, but ask for knowledge about the world as it is [7].

In the problem context stakeholders who may affect the project or may be affected by it can be included among with the knowledge context which consists of existing theories from science and engineering. Improving the performance of a search engine through personalization calls for a design problem as it requires designing an artifact. In this social context both academic search engine owners (1) (e.g. Scopus[1], Google Scholar[2], Microsoft Academic Search[3]) and researchers who interact with search tools (2) represent the stakeholders. In the knowledge context, topics from information retrieval, Graph Theory and Natural Language Processing (NLP) will be used.

Another important concept in [7] is the engineering cycle. Defined as a rational problem-solving process, it consists of the following tasks:

- Problem investigation: What phenomena must be improved? Why?
- Treatment design: Design one or more artifacts that could treat the problem.

[1] http://scopus.com
[2] http://scholar.google.com
[3] https://academic.microsoft.com

- Treatment validation: Would these designs treat the problem?
- Treatment implementation: Treat the problem with one of the designed artifacts.
- Implementation evaluation: How successful has the treatment been?

The current thesis focuses on the design cycle; a subset of the engineering cycle which excludes the treatment implementation and implementation evaluation. These steps are the responsibility of the first class of stakeholders based on the knowledge and design of this research.

## 1.3 Design science research goals and design problems

To understand the goals of a design science project, it is useful to distinguish the goals of the researchers from the goals of an external stakeholder. A researcher's goals invariably include *curiosity* and *fun*: curiosity towards what the answer to knowledge questions is and fun in the design and test of new improved artifacts. In this sense, all design science research is *curiosity-driven* and *fun-driven* research [7]. However, some projects are *utility driven* and *budget constrained* while some do *exploratory research*. In this thesis an exploratory approach to improve the performance of search engines in the academic context is taken i.e. improve the performance of some artifact in a context. As classified in [7], this goal falls into the *technical research goals* or *artifact design goal* category.

To better refine our design problem, we take a leap and introduce the two types of personalization mechanisms, namely (1) online and (2) offline personalization. The first method takes advantage of user preferences or activity at query time to assign a probability of relevance score for each document in a collection or to documents retrieved by another algorithm. As it should be called whenever a user sends a query to a search engine, this method is constrained by computational time.

The second method, however, assigns relevance scores offline and has no constrains. Nevertheless, both methods present quality attributes and come with advantages and disadvantages. Yet we find one important discriminant: *flexibility*. Because the former method is called at query time, it has the potential to be more flexible as users can specify which personalization attributes should be used. Such an approach is impossible for the latter since all the computation is carried offline and is not time dependent.

In academic search users are not bounded to a set of research areas or topics, but can often take exploratory paths following new interests. Given this condition,

providing personalization based only on past events might limit the user's ability to perform research on new frontiers.

As flexibility is a factor that can not be neglected, we seek to *improve* the performance of academic search engines by *designing* a personalization algorithm that *satisfies* computational query time requirements *so that* researchers can easily find and access information.

To treat our design problem we further divide it into smaller knowledge areas: (1) performance evaluation and (2) personalization algorithms for search engines. To measure the effectiveness of a retrieval algorithm, a *text collection* composed of documents, a set of query topics and a set of manual relevance judgements that map topics to relevant documents is used. Given the size of nowadays document collections, relevance judgements are difficult to gather and maintain. Therefore, the first part of the thesis investigates the automatic creation of test collections in academic search. The second part of this thesis is concerned with improving academic search engines through personalization.

We formalize the following research questions:

- Research Question (RQ)1: Can the information from scientific articles be used to create test collections with automatic relevance judgements for academic search?
- RQ2: May be Scopus performance improved through personalization using topic models and graph walks?
- RQ3: Will users consider the same documents relevant after a period of time?

## 1.4 Thesis outline

The rest of the thesis is organised as follows:

- In Chapter 2 we provide the necessary background information from the aforementioned knowledge domains.
- In Chapter 3 we provide the problem investigation, treatment design and treatment validation model for RQ1.
- In Chapter 4 we provide the problem investigation, treatment design and treatment validation for RQ2 and RQ3.
- Chapter 5 analyses the threats to validity and draws the conclusions. Future research is also considered here.

## 1.5 Implementation Considerations

All treatment validation methods are tested on the Scopus data corpus. The references for one article can be found in the dataset several times, with different identifiers: at first as a normal article with all details (e.g. abstract, citations) and, secondly, as a reference for an article, where only the title or a description field is available. This means that, in order to get all the details about an article, for some references the title must be parsed from the description field and used to search for the indexed entry with all details.

Since the description field is not consistent in structure, parsing the titles through a regular expression was not always possible. From our estimations, a total of $\approx 20\%$ of the references could not be parsed. A rigorous influence on the re-ranking algorithm presented in this study is impossible because the number of relevant references for re-ranking remains unknown.

A member of the Scopus team confirmed the behaviour for Computer Science (CS) publications and suggested that it might not be the case for other fields of study. An investigation is proposed for future research, but a user study with persons outside the CS field was not possible for this study.

# Chapter 2

# Background

## 2.1 Introduction

The field of Information Retrieval (IR) tackles a broad range of subjects from Mathematics and Computer Science in order to develop efficient search algorithms. This study focuses on two aspects of information representation: at first, a non textual approach to represent entities and connections between them expressed as a graph structure and, secondly, a textual approach which analyses words in large text collections to discover themes that run through them, how they are connect to each other and how they change over time.

Therefore, this chapter provides an introduction to graph models used both for general and personalized search, graph similarity measures and topic models, a statistical method used to infer themes or categories under which documents can be grouped or which documents can represent. We also provide details about the representation of an academic network as a social graph structure and the relations between entities.

## 2.2 Definitions and notations

The meaning of *information retrieval* can be very broad. Searching the address book for a phone number is a form of information retrieval. However, as an academic field of study, *information retrieval* might be defined thus: Information retrieval (IR) is finding material (documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (stored on computers) [8].

The term *unstructured data* reveals data without a clear, semantically overt, structure. In reality, almost no data are truly *unstructured*. Most text has structure such as headings, paragraph or footnotes and is commonly represented in documents by explicit markup. IR is also used to facilitate ”semistructured” search such

as finding a document where the title contains *Java* and the body contains *threading* [8].

The most standard IR task is called *ad hoc retrieval*: a system aims to provide documents from within the collection, relevant to an arbitrary user information need formulated as a *query*. An *information need* represents the *topic* about which the user wants to have more information and is different from a *query*, which is what the user conveys to the computer to communicate the information need. A document becomes *relevant* if the user perceives it as valuable with respect to the information need.

The assessment of a document's relevance by a user is called *relevance judgement*. Briefly, the goal of the ad hoc task is to return a *ranking* of the documents in the collection in order of decreasing probability of relevance.

During this study, the terms *ranking* and *re-ranking* are used as follows: upon receiving a query from a user, an information retrieval system uses a series of *algorithms* and *models* to *rank* and retrieve the top $N$ relevant documents from the collection. A set of $M$ top documents from $N$ is selected for *re-ranking*, a process which uses a set of *algorithms* and *models* to re-assess the probability of relevance for each document and re-order the $M$ set.

IR has changed considerably in the last years with the expansion of the Web (World Wide Web) and the advent of modern and inexpensive graphical user interfaces and mass storage devices [9]. Firstly, the web pages are extremely diverse ranging from weather information to journals about information retrieval. Secondly, search engines on the Web must also contend with inexperienced users and pages engineered to manipulate search engines ranking functions [10].

However, unlike *flat* document collections, the Web is hypertext and provides considerable auxiliary information on top of the text of the web pages, such as link structure and link text. Exploiting the additional information inherent in the hyperlink structure of the web or of documents is called *(hyper) link analysis*. Such techniques represent the entities as *nodes* of a *graph* and the links between them as *edges* of the same graph.

The field of academic search deals with inter-connected entities (e.g. academic articles, journals, authors) that can be easily represented as a *graph* structure. Since this representation is used throughout the thesis, relevant notations in terms of the underlying graph structures and algorithms are provided.

### 2.2.1 Graph

A graph $G =< V, E >$ consists of a set of nodes $V$, and a set of edges $E$. If the direction of the edges is specified, the graph is called *directed*. Nodes are repre-

sented by letters such as $u, v$ and an edge from $u$ to $v$ as $uv$. Every node $u$ has a type marked as $\tau(u)$ under the assumption that there is a fixed set of possible types and every edge can have a label $l$ denoted as $uv(l)$. Several relations can occur between nodes: for example, $uv(l)$ and $uv(l')$, where $l \neq l'$. We choose to represent a bidirectional relation between two nodes of a directed graph as two symmetrical directed edges.

For illustration purposes we present an example in 2.1. In this figure node types are denoted with different shapes. The edges have different types too, denoted by their labels. Suppose that a square represents a node of type *article*, a circle represents an *author* and a diamond stands for a *source*. The different relations between these entities are represented either by direct and inverse relations or by full paths. For example, the *author* on the left is connected to the *source* on the left through an *author* and an *publisher* relationship and to the article in the middle through an *author* relationship.

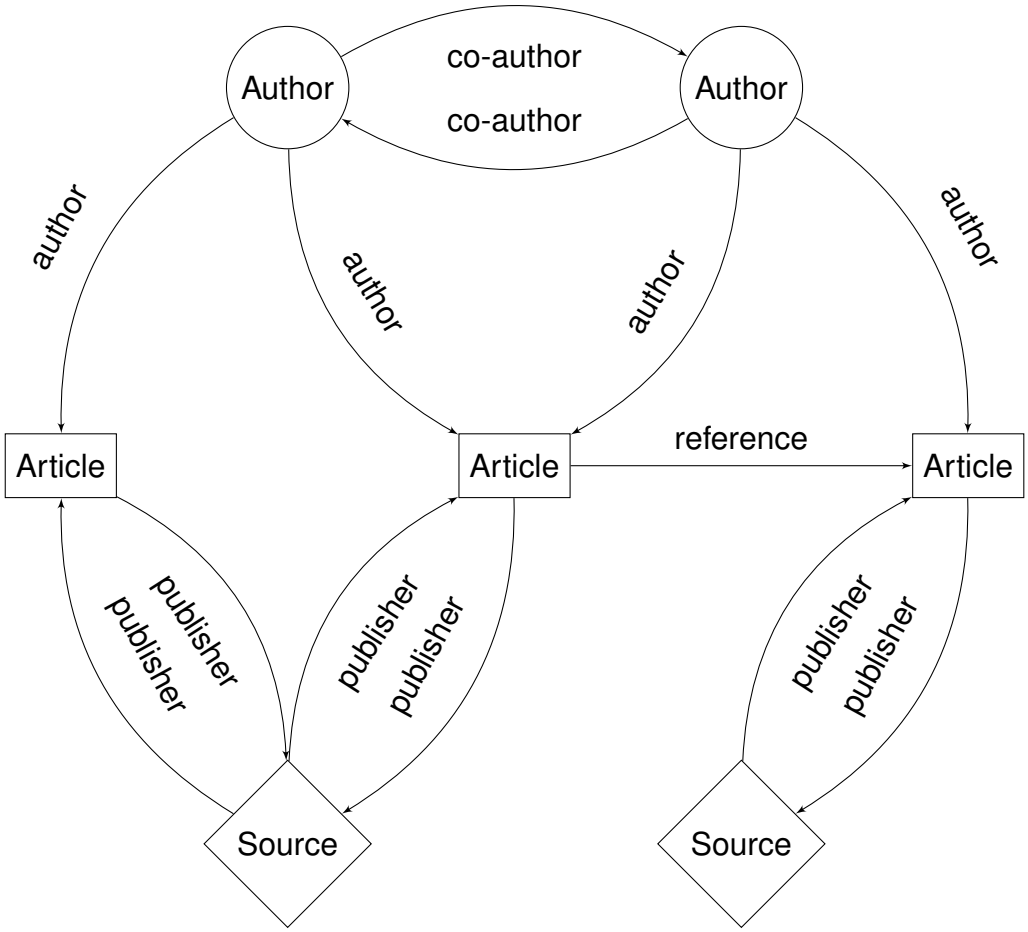


**Figure 2.1:** An example of graph scheme.

### 2.2.2 Context

Given a graph $G =< V, E >$ with a set of vertices $V = \{u, v, .., z\}, V(G)$ and a set of edges $E = \{uv|u, v \in V\}, E(G)$ we define two vertices connected by an edge as *adjacent*. Further on, we note the *neighbourhood* of a node $v$ in an undirected graph as $N(v) = \{u|uv \in E\}$. In a directed graph we distinguish between the incoming neighbours of $v$, $B(v) = \{u|uv \in E\}$ and the outgoing neighbours of $v$, $N(v) = \{u|vu \in E\}$.

We define the context of one vertex $v$, representing one entity in the graph, as the subgraph $H$ with $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ formed by all nodes *adjacent* to $v$, $H = \{u, v|uv \in E(H)\} \cup \{v, u|vu \in E(H)\}$. To extend the context of one entity to a rank *n*, we recursively add to the subgraph *v* the entities *adjacent* to the last neighbours added. So, for rank 1 we add the entities adjacent to $v$: $u$, for rank 2 we add the entities adjacent to $u$: $q$ and so on.

### 2.2.3 User

In this thesis the term *user* is used to describe an academic *author* represented by an entity in the graph $G$ subject to personalized search.

### 2.2.4 Entity similarity and relevance

Given a graph which represents entities and relations between them, we are interested to determine how related (similar) are entities without a direct connection. Since the graph is heterogenous (it contains multiple types of entities and relations), the similarity evaluation method should integrate multiple types of evidence into a similarity score. The similarity score is used to return, given a set of documents relevant to a query, a list of documents ranked by their similarity and relevance to the user's context described in Section 2.2.2.

### 2.2.5 Topics

During this study the term *topic* is used in two ways. In Chapter 3 the term topic is used, as in Text Retrieval Conference (TREC) to describe a natural language statement of information need. In Section 2.5 and Chapter personalised the same term is used to address *themes* or substantively meaningful *categories* under which documents can be grouped or which documents can represent.

Further on, the language of text collections referring to entities such as *words*, *documents* or *corpus* is used. Formally, we define:

- A *word* or *term* represents an unique word type of a fixed length vocabulary indexed by $\{1, ..., W\}$ . Each word is represented as an unit-bias vector of length $W$ that has a single element equal to one and all other elements equal to zero. The *k*-th word in the vocabulary is represented by a vector $w$ such that $w^k = 1$ and $w^i = 0$ for $i \neq k$.

- A *document* of $N$ words is represented as a sequence by $d = (w_1, w_2, ..., w_N)$, where $w_i$ is the *i*-th word in the sequence. Note that this is also a bag of words representation since the word sequence does not need to match the original vector of the document.

- A *corpus* is a collection of $D$ documents denoted by $D = \{d_1, d_2, ..., d_D\}$

Let's consider an example vocabulary, $v = \{be, not, or, to\}$ with indices $\{1, 2, 3, 4\}$. The word $be$ is represented as $w^3 = [1, 0, 0, 0]$. The document $d = "to\ be\ or\ not\ to\ be"$ will be represented as $w_d = (w_1^4, w_2^1, w_3^3, w_4^2, w_5^4, w_6^1)$, using the notation described above.

We will also use $P(t|d)$ to denote a document's distribution over topics, $P(w|t)$ the probability distribution over words $w$ given the topic $t$, $P(w|d)$ the distribution over words within document $D$ and $T$ for the number of topics. For a corpus with $D$ documents and $W$ words, a topic model learns a relation between words and topics and a relation between topics and documents as:

- a $T$ x $W$ matrix, $\Phi$, with elements $\phi ij$ denoting the probability $P(w_j|t = i)$, and

- a $D$ x $T$ matrix, $\Theta$, with elements $\theta_{ij}$ denoting the probability $P(t = j|d_i)$.

## 2.3 Graph Walk models

Beside a textual information representation of documents, IR algorithms use relationships between documents to improve search accuracy. In this section we introduce the most important algorithms that make use of hyperlink data in graph structures for general and personalized search. In Section 2.3.4 we also discuss methods to evaluate how close related are two documents (or nodes of a graph) given the relationships between them.

### 2.3.1 PageRank

The PageRank algorithm [10] was introduced together with the Google web search engine [1]. The idea behind it is that if a page $u$ has a link to page $v$, the author of

[1] www.google.com

$u$ is conferring some importance to page $v$. Intuitively, *Yahoo!* [2] is an important page, reflected by the fact that many pages point to it. Likewise, pages prominently pointed to from *Yahoo!* are themselves probably important.

How much importance does a page $u$ confer to its out links? Let $N_u$ be the outdegree of page $u$ and let $Rank(p)$ represent the importance (i.e. PageRank) of a page $p$. Then the link $(u, v)$ confers $Rank(u)/N_u$ units of rank to $v$ [11]. This simple idea leads to the following fix point computation that yields the rank vector over all of the pages on the web: if $N$ is the number of pages, assign all pages the initial value $1/N$. Let $B_v$ represent the set of pages pointing to $v$, in each iteration, the rank is propagated as follows:

$$\forall v, Rank_{i+1}(v) = \sum_{u \in B_v} Rank_i(u)/N_u \tag{2.1}$$

The model corresponds to the probability distribution of a random walk on the graph of Web. This can be thought of as modelling the behaviour of a *random surfer*. The random surfer simply keeps clicking on successive links at random.

However, if a real Web surfer ever gets into a small loop of web pages, it is unlikely that the surfer will continue in the loop forever. Instead, the surfer will jump to some other page [10]. The *random surfer* behaviour can be described as: starting from a node $i$, the probability of a user to go to a page (reset) randomly in the network is $\gamma \in (0, 1)$ and the probability to move to a node $j$ that has an outgoing link from $i$ is $(1 - \gamma)$. A random walk process is described as:

$$P_{d+1} = \gamma[\frac{1}{N}]_{1XN} + (1 - \gamma)MP_d \tag{2.2}$$

where $N$ represents the number of pages or nodes in the network and $M$ is the transition matrix. Through $M$ the probability to reach a node in the network is distributed uniformly as:

$$M_{ij} = \begin{cases} \frac{1}{|N(i)|} \text{ if there is an edge from i to j} \\ \text{0 otherwise} \end{cases} \tag{2.3}$$

where $N(i)$ represents a set of nodes that have an outgoing link from $i$.

$\gamma$, the damping factor prevents the chain from getting stuck in small loops [12].

For each node, the PageRank score can be computed by recursively applying:

$$R_j = \gamma\frac{1}{N} + (1 - \gamma) \sum_{i \in B(j)} \frac{R_i}{|N(i)|} \tag{2.4}$$

where $B(j)$ is the set of nodes that link to $j$. Equation 2.4 is applied until convergence.

[2] www.yahoo.com

### 2.3.2 Personalized PageRank

The PageRank score reflects a *democratic* importance that has no preference for particular documents. In reality, a user may have a set $P$ of preferred documents (such as his bookmarks or his referenced articles) which he considers most interesting. We can account for preferred documents in the random surfer model by introducing a *teleportation* probability $\gamma$: at each step, a surfer jumps to a random page in $P$ with probability $\gamma$, and with probability $1 - \gamma$ continues forth along a link. The limit distribution of surfers in this model would favour pages in $P$, pages linked-to by $P$, pages linked-to in turn, etc [13].

The idea to bias the PageRank score for personalization was introduced in [10]. A similar approach was suggested by Haveliwala [11] in which the PageRank score is biased towards pages classified before as relevant to a given topic.

The *Personalized PageRank* graph walk is defined as:

$$R_{d+1} = \gamma R_0 + (1 - \gamma) M R_d \tag{2.5}$$

where $R_0$ represents the distribution of interest over the graph nodes. This formula leads Equation 2.2, if $R_0$ is uniform.

### 2.3.3 HITS

Hyper linked induced topic search (HITS) [14] strives to find authoritative documents by analysing the link structure of Web pages. At first, it identifies two kinds of Web pages:

1. *authorities* - pages representing authoritative sources of information for a given query, and

2. *hubs* - resource lists containing pointers on the topic.

This relation between Web pages is mutually enforcing: good hubs point to good authorities and vice versa. The HITS algorithm formalises the relation into a two phases iterative computation:

A *sampling* phase uses the query terms to collect a *root set* of pages from a text-based search engine. HITS expands the root set into a *base set* by adding all the pages that are linked to and from pages in the root set. The idea is to ensure that the base set will contain the best pages for the query even though the root set does not.

A *weight propagation* phase works with the subgraph induced by the base set. The algorithm assigns a nonnegative authority weight $x^{(p)}$ and a nonnegative hub weight $y^{(p)}$ for each page $p$ in the base set. An *update* rule specifies that:

1. $x^{(p)}$ is the sum of the hub weights for the pages that point to $p$ (2.6), and

2. $y^{(p)}$ is the sum of the authority weights for the pages that $p$ points to (2.7).

$$x^{(p)} \leftarrow \sum_{B(p)} y^{(p)} \tag{2.6}$$

$$y^{(p)} \leftarrow \sum_{N(p)} x^{(p)} \tag{2.7}$$

### 2.3.4 Graph similarity measures

Given a graph $G$ we would like to assess the similarity of two vertices $u, v$. One simple measure is the length of the shortest path between the vertices, i.e. the number of edges between the vertices or, if the edges are weighted, the sum of the edge weights for the shortest path. Another way to measure the similarity uses the *maximal network flow* [15], a measure defined as the number of units that can be simultaneously delivered from $u$ to $v$ given a limited capacity for each edge (proportional to its weight).

Faloutsos et al. [16] prove that such measures are not suitable for social network graphs because, at first, a relation between two entities can take several paths and, secondly, the maximum flow does not take into account path lengths.

Other measures of similarity use node neighbourhood ($N(u)$). The most basic measure is computed as the overlap between node neighbours: $|N(u) \cap N(v)|$. For example, the *Jaccard coefficient* [8] measures the probability that two nodes, $u$ and $v$, have a common neighbour, when a node is randomly selected from union $|N(u) \cup N(v)|$ and the *Adamic-Adar* measure [17] considers the frequency of the common neighbours.

Further on we introduce two similarity measures associated with random graph walks:

1. Hitting time: by initiating a random walk from a node $u$ and moving towards a neighbour of $u$, $v$, selected uniformly random, the hitting time $H(u, v)$ represents the expected number of steps required to reach the node $v$.

2. Simrank [18] states that objects are similar if they are related to similar objects (i.e. the nodes $u$ and $v$ are similar if they are connected to $x$ and $z$ and $x$ and $z$ are themselves similar). The SimRank score can be computed as follows:

$$Sim(u, v) = \frac{\gamma}{B(u)B(v)} \sum_{a \in B(u)} \sum_{b \in B(v)} Sim(a, b) \tag{2.8}$$

where $\gamma \in [0, 1]$ and $a, b$ are individual in-neighbours of $u$ and, respectively, $v$. The score is propagated iteratively along the direction of the edges until it converges. In this study, $\gamma$ has, as suggested in [19], a value of $0.6$.

## 2.4 Bibliographic graphs

Social network analysis attracted considerable interest in recent years and plays an important role in many disciplines [20]–[23]. It is based on the premise that the relationships between social actors can be represented as a graph where the graph's nodes are actors and the graph's edges connect pairs of nodes, thus representing interactions.

This representation allows the application of graph theory to the analysis of what would otherwise be considered an inherently elusive and poorly understood problem: the tangled web of social interactions [23]. We assume such a graph representation for our field of study: academic networks.

The relations between nodes in a Digital Library (DL) graph are described in the field of *bibliometrics*, a term defined by Pritchard as the application of mathematics and statistical methods to books and other media of communication [24]. Specifically, we are interested in the subfield of *citation analysis*, the representation of a decision made by an author who wants to show the relation between the document he is writing and the work of another [25].

### 2.4.1 Entities - graph nodes

Actors in academic networks are represented as entities or nodes in a graph structure under the assumption that a research *article* is written by one or more *authors* and published in a *journal* (*source*).

Since later experiments are carried on the Scopus data corpus, we incorporate the *affiliation* node to represent institutions that authors are affiliated with [26]. A simplistic version of this relational model is represented in Figure 2.2.

### 2.4.2 Relationships between entities - graph edge types

The relationships between entities in academic graphs are divided based on their neighbourhood degree as follows: *first order relations* (1) represent edges connecting adjacent nodes and *second order relations* (2) which represent relations derived conceptually from (1).

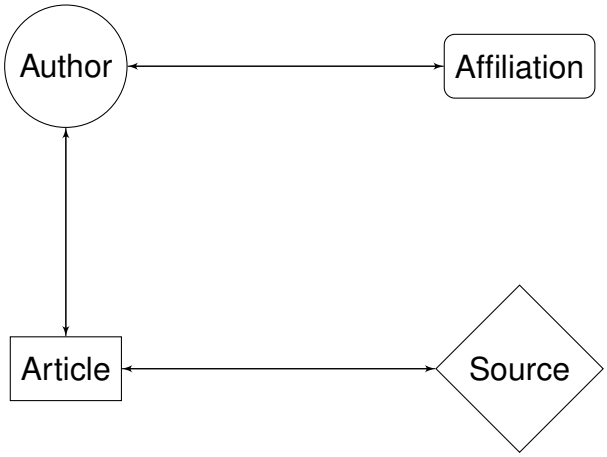


**Figure 2.2:** Simplified relational model for graph entities.

## First order relationships

In general, a scientific paper does not stand alone, it is embedded in the literature of the subject through references and citations relations [27]. Narin et al. [28] define the terms reference and citation stating that a reference is the confirmation that one document *gives* to another.

The acknowledgement that one document *receives* from another is called a citation. For example, when document (a) appears in the list of references of document (b), it means that document (a) has been cited by document (b) as a source of information in support for an idea or a fact. In this case, not only document (a) is a reference of document (b) but also, it has received a citation from document (b). In other words, according to bibliometric terminology, document (b) is a citing document and document (a) is a cited document [29].

In academic graphs a reference is represented as a direct edge of type $l_r$ from node (B) - citing document to node (A) - cited. The number of in $l_r$ edges for node (A) will represent the number of citations it receives and the number of out $l_r$ edges for node (B), the number of references. Moreover, a direct edge of type $l_a$ from a node of type *author* to a node of type *article* represents an *authorship* relationship and a direct edge of type $l_s$ from a *source* node to an *article* represents the *publisher* relation.

## Second order relationships

As defined earlier, second order relationships are derived conceptually from first order relationships and identify nodes likely to be closely related. The field of bibliometrics defines two terms for citations analysis:

1. bibliographic coupling - as defined by Kessler, bibliographic coupling happens

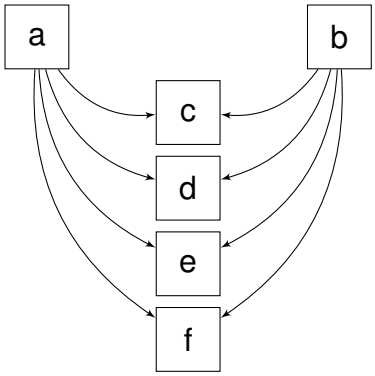


**Figure 2.3:** Bibliographic coupling.

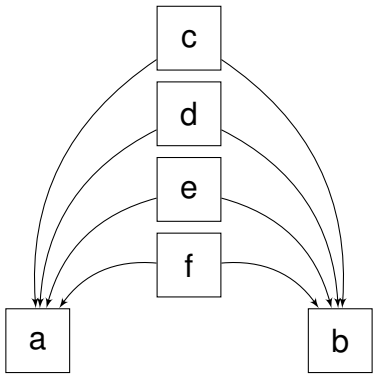


**Figure 2.4:** Co-citation.

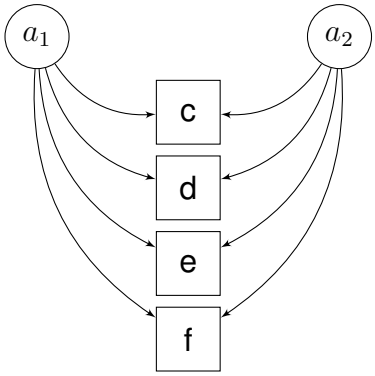


**Figure 2.5:** Co-authorship.

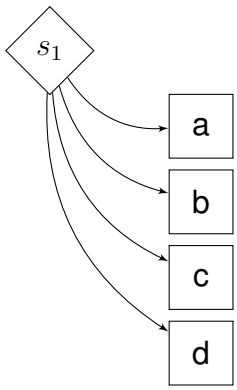


**Figure 2.6:** Co-source.

when a reference is used by two papers as a unity of coupling between the two [30]. The strength of bibliographic coupling depends on the number of references the two papers have in common [31]. This relation is illustrated in Figure 2.3 where nodes (a) and (b) are citing documents and nodes (c), (d), (e), (f) are cited ones.

2. co-citation coupling - co-citation indicates the frequency with which two documents are cited together. Small [32] goes one level deeper and describes the similarity measure of co-citation similar to the similarity measures of the co-occurrence of words. This relation is illustrated in Figure 2.4 where nodes (a) and (b) are cited documents and nodes (c), (d), (e), (f) are citing ones.

Beside the relationships introduced by co-citation analysis there are two more relationships that can identify nodes likely to be closely related:

1. co-authorship - co-authorship happens when one article is authored by one or several authors. The strength of co-authorship depends on the number of

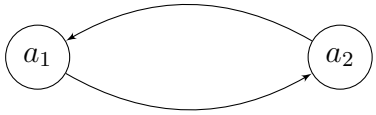


**Figure 2.7:** Co-authorship networks as in [1].

.

articles the authors have worked on together. Figure 2.5 illustrates this relation as follows: nodes ($a_1$) and ($a_2$) representing authors have written together documents (c), (d), (e), (f).

2. co-source - this relationship is derived from the *publisher* relationship and states that two or more articles are likely to be related if they were published in the same journal (source). In Figure 2.6 nodes (a), (b), (c) and (d) representing articles are likely to be similar since they were published in the same journal, ($s_1$).

Alternatively, the co-authorship relationship can be represented as in co-authorship networks [1] by two symmetrical directed edges between two or more author nodes (Figure 2.7).

## 2.5 Topic models

Topic models are statistical methods that analyse words in large text collections to discover the themes that run through them, how they are connect to each other and how they change over time. Given a document collection, topic models learn a set of latent variables called *topics*. Topics are probability distributions over a vocabulary of words where frequently co-occurring words in the collection are associated with high probability. In addition, each document is represented as a distribution over topics.

In this thesis the term *topic models* is used to refer to *probabilistic topic models* [33], [34]. The main assumption of topic models is that documents are generated by a mixture of topics while topics are probability distributions over words. The input of a topic model is a set of documents and its output is a set of topics together with topic assignments to documents.

Figure 2.5 shows an overview of a topic modelling pipeline including input and output. On the input, each document is represented as a bag-of-words. Each document is often tokenised into words and normalised while word order is ignored.

The only information relevant to the model is the number of times a word appears in each document. The input section in Figure 2.8 shows documents represented as bag-of-words. Document 1 contains information about Python programming language while document 2 contains information about python snakes.

The output of a topic model is a set of topics and a set of topic assignments for each document in the training collection. Each topic is a probability distribution over all the unique words in the collection. Topics are often represented by the words with the highest probability in the topic (output in Figure 2.8). Words assigned high probability in some topics frequently appear together in documents and are likely to represent a coherent subject or theme. A document is represented as a probability distribution over topics with only a few topics assigned with high probability. In the output, the first topic (1) is assigned with high probability to document 2. On the other hand, topic 2 which represents information about programming is assigned to document 1.

Topic models have appealing characteristics for organising document collections. They can be used for clustering documents under themes enhancing browsing and improving information access. In addition, topic models soft cluster terms into topics dealing with polysemy (e.g. topic 1 and 2 represent different senses of the word *python* and therefore, users can retrieve different sets of documents relevant to a given search germ (e.g. *python*).

Topic modelling is widely used in NLP and has been applied to a range of tasks including word sense disambiguation [35], multi-document summarisation [36], information retrieval [11], [37] or image labelling [38]. In this thesis a Latent Dirichlet Allocation (LDA) [34] model is used to determine if documents are relevant to authors interests. We also provide an introduction to Probabilistic Latent Semantic Analysis (pLSA) to outline it's disadvantages and the evolution to LDA.

### 2.5.1 Probabilistic Latent Semantic Analysis

The probabilistic latent semantic analysis models each word in a document as a sample of a mixture model. The mixture model is a set of *topics* in the form of multinomial random variables.

Given $T$ topics, the aim is to find the probability distribution of words in a topic and the probability distribution of topics in a document. Words are the observed variables while topics are the latent ones. The generative process is the following:

1. For each document $d \in D$ with probability $P(\theta_d)$:
   (a) select a latent topic $t$ with probability $P(t|d)$,
   (b) generate a word $w$ with probability $P(w|t)$.

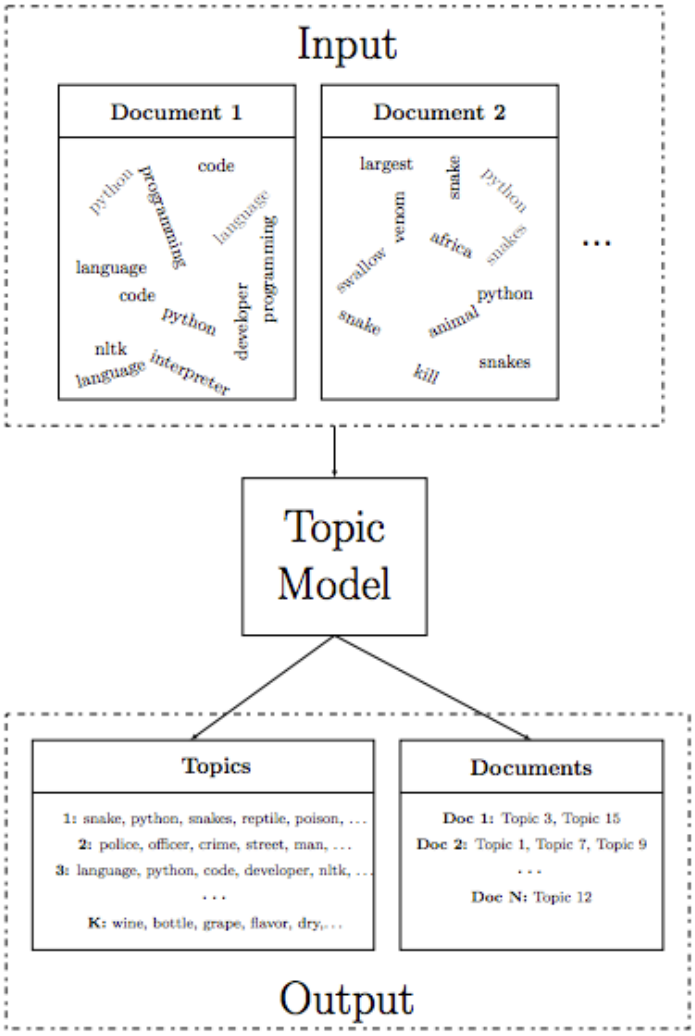


**Figure 2.8:** Input and Output of a Topic Model.

The process is defined by the following expression as a joint probability between a word and a document:

$$P(\theta_d, w) = P(\theta_d)P(w|\theta_d), \tag{2.9}$$

$$P(w|\theta_d) = \sum_{t \in T} P(w|t)P(t|\theta_d) \tag{2.10}$$

$$= P(\theta_d) \sum_{t \in T} P(w|t)P(t|\theta_d). \tag{2.11}$$

The pLSA model satisfies the main assumption of topic models, namely that a document consists of multiple topics. The probability $P(t|\theta_d)$ contains the weight of each topic $t \in T$ given a document $d$. However, representing each document as a list of topic weights without a generative probabilistic model for them leads to two

main problems [34]: (1) the number of parameters grows linearly with the number of documents in the corpus causing overfitting problems, and (2) it is not possible to assign topic probabilities to any unseen documents (i.e. not in the training corpus).

### 2.5.2 Latent Dirichlet Allocation

LDA [34] is an extension of pLSA which introduces symmetric Dirichlet priors on the distribution over topics for a particular document, $\theta$, and the distribution over words for a particular topic, $\phi$. This addresses the problem with pLSA mentioned above by treating the topic weights in each document as a hidden random variable of size $T$, where $T$ is the number of topics. A graphical representation of LDA is shown in Figure 2.9.

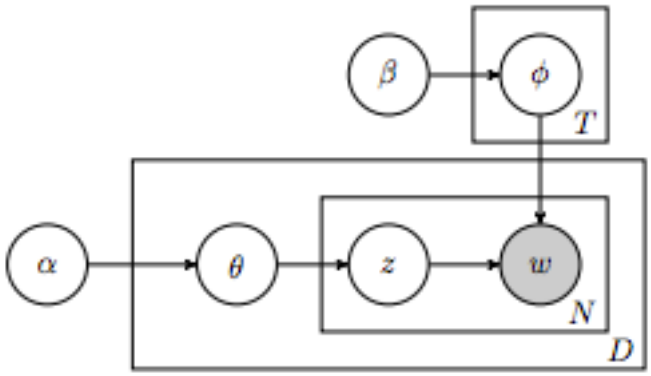


**Figure 2.9:** Graphical model representation of LDA.

The generative process for the topics is the following:

1. For each topic
   (a) Choose a distribution over words $\phi \sim Dir(\beta)$.
2. Choose N number of words
3. Choose $\theta \sim Dir(\alpha)$
4. For each of the $N$ words $w_n$:
   (a) Choose a topic $t_n \sim Multinomial(\theta)$
   (b) Choose a word $w_n$ from $p(w_n|t_n, \beta)$, a multinomial probability conditions on the topic $t_n$.

where:

- $N$ is the number of words in a document.
- $t_n$ is the $n$ topic for the word $w_n$.
- $\Theta$ is the topic distribution for a document.
- $\alpha$ is the parameter of the Dirichlet prior on the per-document topic distributions.
- $\beta$ is the parameter of the Dirichlet prior on the per-topic word distribution.

The joint probability of the corpus $D$ given the hyperparamenters $\alpha$ and $\beta$ is given by the following equation:

$$P(D|\alpha,\beta) = \prod_{t=1}^{T}\prod_{d=1}^{D}\prod_{n=1}^{N} P(\theta_t|\beta)P(\theta_d|\alpha)P(t_{dn}|\theta_d)P(w_{dn}|\phi_{tdn}) \tag{2.12}$$

The main variables that need to be estimated in the model are the per-topic word distribution $\phi$ and the per-document topic distribution $\theta$. Inferring direct estimates from Equation 2.12 is intractable.

Hofmann [33] used the Expectation Maximisation (EM) algorithm to estimate $\phi$ and $\theta$ directly. However, the EM algorithm might get stuck in local maxima. Approximation methods such as Bayesian variation inference [34] or Gibbs sampling [39] have been used to avoid this problem.

Gibbs sampling is a specific type of Markov Chain Monte Carlo Model (MCMC) for obtaining sample vales from complex multivariate distributions. It starts by assigning every word $w$ with a random topic $t \in \{1, ..., T\}$, for every document in the corpus $D$. Then, for each word, it estimates the probability of assigning the current word in each topic, given the assignments of all the other words. Griffiths and Steyvers [39] computed this probability as:

$$P(t_i = j|t_{-i}, w_i, d_i, \cdot)\alpha\frac{C^{WT}_{w_i j} + \beta C^{DT}_{d_i j} + \alpha}{\sum_{w=1}^{W} C^{WT}_{wj} + W\beta\sum_{t=1}^{T} C^{DT}_{d_i t} + T\alpha} \tag{2.13}$$

where $t_i = j$ represents the topic assignment of word $w_i$ in topic $j$, $t_{-i}$ are the topic assignments of all of the other words, and $\cdot$ represents all the other information from word, documents and hyper-parameters $\alpha$ and $\beta$. In addition, the element $C^{VT}_{wj}$ of the matrix $C^{VT}$, $V$ x $T$, contains the number of times word $w$ was assigned to topic $j$. The matrix $C^{DT}$, $D$ x $T$, contains number of time each topic is assigned to words of each document and $C^{DT}_{dj}$ represents the number of times topic $j$ is assigned to words in document $d$.

The Gibbs sampling algorithm estimates the probability of each topic for every word. The elements of matrices $\phi$ and $\theta$, containing the per-topic word distributions

and the per-document topic distributions can be obtained by:

$$\phi_{i,j} = \frac{C_{ij}^{WT} + \beta}{\sum_{k=1}^{W} C_{kj}^{WT} + W\beta} \tag{2.14}$$

$$\theta_{jd} = \frac{C_{dj}^{DT} + \alpha}{\sum_{k=1}^{T} c_{dk}^{DT} + T\alpha} \tag{2.15}$$

where $\phi_{ij}$ is the probability of word $w^i$ in topic $j$ and $\theta_{jd}$ is the probability of topic $j$ in document $d_d$.

# Chapter 3

# Ranking academic search engines with automatic relevance judgements

## 3.1 Introduction

In this chapter we propose several methods to build an IR test collection (similar to TREC collections) with automatic relevance judgements.

A test collection consists of *documents*, a set of *information needs*, also called *topics*, *queries* that represent the *data structure* sent to the information retrieval tool and *relevance judgements* for the top documents retrieved from the collection. TREC collections are organised by *tasks*, which represent the reason that caused the user to submit a query. Since relevance judgements are difficult to gather, we investigate an automatic way to assess the value of documents in a collection.

The search task models researchers that use Scopus to find references for their new paper and the work in progress represents the topic. We use all the documents available in Scopus and several methods to automatically generate queries from the content of a document (also called an article). After issuing these queries to Scopus, only the documents referenced in the article (from which the queries were generated) are considered relevant.

The experiment selects articles already indexed in Scopus, generates queries from their content and sends them to Scopus. In order to represent the task, the search is limited to retrieve documents indexed, at most, in the year before the article (from which the queries were generated) was indexed. From the retrieved documents, only the ones referenced in the article are considered relevant. For example, an article is selected and the title is used as a query. After submitting it to Scopus, the number of references retrieved in the result set are counted and the performance of Scopus is evaluated using this measure.

The same terminology introduced in Section 1.2 is used as follows: *treatment design* (Section 3.3.3) describes the system developed in order to create a test

collection for academic search and *treatment validation* (Section 3.4) judges the quality of the system and evaluates the resulting collection.

## 3.2 Related work

The evaluation of information retrieval systems focuses on what is called the *processing level* [40]: the performance of specific algorithms and techniques are compared in isolation from the larger systems of which they are part of and from the users who employ them. These experiments measure the effectiveness of a retrieval algorithm using a test collection, composed of a set of documents, a set of query topics and a set of relevance judgements which map topics to the documents relevant to them. Test collections are the basic tools for conducting repeatable experiments on retrieval algorithms [41].

Relevance judgements for a test collection are difficult to gather for several reasons. At first, for modern collections containing millions of documents it is not feasible to assess every document with respect to each topic. One solution is to use pooling: a method through which a fraction of the collection is selected for assessment. If the subset examined contains a representative sample of the relevant documents, the pooling method closely approximates the results of assessing the entire collection.

Since retrieval systems attempt to rank documents according to their degree of probability of relevance, the highest ranking documents produced by effective retrieval systems should be a good candidate for inclusion in the assessment pool, even though some relevant documents may be missed [42]. This assumption is the basis of how relevance judgements are created for TREC collections [43].

In TREC, each participating system reports the 1000 top-ranked document for each topic. Of these, the top 100 are collected into a pool for assessment. In his analysis of the TREC results, Zobel concluded that the results obtained given a pool of 100 are reliable even though many relevant documents may be missed. Moreover, the relative performance among systems changed little by limiting the pool depth to 10, although actual precision scores did change for some systems [44].

A second challenge in creating relevance judgements is that people can disagree or assess differently the same document. Even though, as Saracevic states, *intuitively, we understand quite well what relevance means*, many factors influence a person's assessment of relevance [45]. Moreover, even a single individual may be inconsistent in judging relevance. This issue has received attention from the research community. Harter found that not a lot of experimental studies looked at assessor disagreement. He also noticed that many studies assume that assessor disagreement has little influence on the relative effectiveness of the system [46].

Analysing three independent sets of relevance judgements from TREC-4 and TREC-6, Voorhees found that, despite a low average overlap between assessment sets and a wide variation in the overlap among particular topics, the relative ranking of systems remained largely unchanged across the different sets of relevance judgements [47]. Furthermore, *hybrid* sets of judgements were created for the TREC-4 analysis choosing each possible combination of judgements from the three assessors over all 49 topics. On average, the Kendall's $\tau$ [1] correlation between the rankings of systems obtain with these assessments and the actual TREC ranking was 0.938.

To overcome the difficulty of creating relevance judgements, researchers proposed various automatic methods to compare the retrieval effectiveness of IR systems [41], [48]. Chowdhury [48] mined search engine logs to generate query and use documents from the Open Directory Project (ODP) [2] to form query-document pairs. Afterwards, the queries were issued to several search engines and a rank at which an engine returns the document was computed. The score for a search engine was represented as the mean reciprocal rank for all query and document pairs. Moreover, the query-document pairs need to be reasonable and unbiased.

Soboroff asked the following question: *how much would system ranking change if relevant documents are chosen randomly from the pool?* [41] So, if relevant documents occur in a pool of retrieved documents according to some distribution and if human assessors can disagree widely without affecting relative system performance, can the occurrence of relevant documents be modelled? He concluded that the difference among human judges is not randomly distributed with respect to their impact on precision scores, but instead is concentrated on *fringe* cases which don't affect many cases.

Furthermore, disagreement about the number of relevant documents does not seem to have a net impact on system rankings at all, probably because having more relevant documents benefits most systems uniformly. Lastly, with simple sampling models and minimal-effort, the method provides a good first-approximation of the results.

Because academic search was recently included in TREC OpenSearch track, there are not many open collections for evaluating the performance of such systems. One attempt to create an academic test collection can be found in [49] where Harpale et al. gather a set of articles from CiteSeer [3] and CiteULike [4] together with manual personalized queries and relevance judgements. In this chapter we assess

[1] Kendall's $\tau$ correlation is a function of the minimum number of pairwise swaps required to turn one ranking into another

[2] http://dmoztools.net

[3] http://citeseer.ist.psu.edu

[4] http://www.citeulike.org

if automatic methods for query generation and relevance judgement can be used to automatically evaluate academic search engines.

## 3.3 Treatment design

Treatment design represents the process used to build a test collection with automatic relevance judgements, the methods used to generate queries from the content of an article and the evaluation methods. Our search task models researchers that use Scopus to find references for their new article.

The treatment selects articles already indexed in Scopus, generates queries from their content and sends them to Scopus. The search is limited to the year before the selected article was published. Later, all documents retrieved by Scopus that are referenced in the articles (from which the queries were generated) are considered relevant. In order to build the test collection, an understanding of academic search engines and queries is needed.

The following set of objectives is addressed:

1. Explore the characteristics of academic search queries.
2. Find measures to test the dataset's relevance and the underlying assumptions.

### 3.3.1 Characteristics of academic queries

In recent years many studies [50]–[52] reveal the information seeking behaviour of researchers through surveys or user studies on a relatively small sample of researchers. However, despite the widespread usage of academic search engines, little is known about the actual behaviour of users based on large-scale analysis.

In their recent work, X. Li et al. [53] analyse the user behaviour of ScienceDirect [5] to provide a general descriptive analysis of academic queries and highlight the differences between academic search and web search. A marked difference between the two is the query length, where academic queries are on average 1.4 words longer than general web queries. Academic queries can be categorised as in [53] to:

- *Navigational* queries: queries that guide the user to a certain publication (identified by special operators such as DOI, ISBN or a title),
- *Transactional* queries: these are queries that directly aim to retrieve academic information resources (e.g. a PDF file),

[5] http://www.sciencedirect.com

- *Informational* queries: queries that seek, refine or explore research topics,

- *Entity* queries: queries that contain named entities (e.g. Authors, Journals) and

- *Boolean operator* queries: queries used by advanced users who like to use the boolean operators for precise matchings.

Our choice is to build a test collection for informational queries, therefore, the search task models researchers that send informational queries to Scopus in order to find references for their new publication.

An important aspect of user behaviour during search sessions that can reveal valuable informations about the way users formulate queries is *query reformulation*. Analysing how a user improves past approaches to information retrieval (e.g. by adding new keywords to a past query or completely change it) or how her interests evolve over time (e.g. topic shifts) can help to automatically generate queries. In recent years several studies tried to uncover patterns and models of query reformulation [54]–[57] and their applications. Their results show that query reformulation is an important subject in understanding user behaviour, which benefits retrieval tasks such as query auto completion or improve retrieval performance.

To uncover the reformulation preferences for the academic search, Xinyi Li [58] examined the logs from a popular academic search engine following queries behaviour for users spanning up to 14 days. Between timespans of 3 days, the tendency to submit new informational queries is weak; in this timeline users use past queries or search directly for a publication. In a longer timespan (7 or 14 days) users prefer the use of new informational queries. This suggests that submitting new queries tends to happen not immediately, but within a longer gap.

Other reformulation mechanisms such as terms add, drop or substitution showed a negative correlation with the timespan, meaning that these reformulations are less frequently. Moreover, Li compares the correlation between reformulation behaviour and topic shifts. Since a researcher can be interested in different topics at the same time, their preferences for queries is important. Following the same query reformulation methods, the correlation study shows that a user's preference for some methods does not correlate to their topic shift tendency.

Overall, academic searche engine users tend to use the same queries or formulate completely new ones instead of improving on past queries. This means that academic users use different keywords in solitude and not in refined aggregations.

### 3.3.2 Performance measures

The choice for a performance measurement is based on its power to discriminate between results, how meaningful its average values are and how easy to interpret it is. Many of the most frequently used metrics are derived from *precision* and *recall*. Precision is the proportion of retrieved, relevant, documents and recall is the proportion of relevant documents that are retrieved from the total number of relevant documents. In this chapter, precision at maximum 50 documents retrieved (p(50)) and recall at maximum 1000 documents retrieved (r(1000)) are used.

Precision at 50 counts the number of relevant documents in the top 50 results. This measure is extremely easy to interpret. Recall measures the number of relevant documents retrieved over the total number of relevant documents. The choice for these thresholds is given by the nature of re-ranking algorithms. While a user might evaluate the first few pages of a search engine (in Scopus, the first two pages contain 40 results), it will never reach the results in the interval $[500; 1000]$. However, a re-ranking algorithm, which re-evaluates and re-orders the results retrieved by an initial algorithm, can evaluate and boost any relevant document in the given interval.

### 3.3.3 Treatment proposal

In this section we introduce the system that models researchers whom use Scopus to find references for a work in progress. Its goal is to generate queries from an article, submit them to Scopus and evaluate the results. Following this assumption, an article is selected and a series of queries are inferred from its attributes. Afterwards, they are sent to Scopus and the top $n$ results are checked for references to evaluate the search tool's performance.

To automatically infer queries from an article, an understanding of its structure is mandatory. Since all the experiments for treatment validation are carried out on the Scopus data corpus, we introduce the attributes an article in Scopus:

- title,
- abstract,
- author keywords - keywords chosen by the authors and
- index keywords - keywords chosen by Scopus indexers.

Both author and index keywords can miss from an article, the only attribute that is always present being the title. Throughout the experiments no article without abstract was found, however, such an event should not be judged as impossible.

Before query formulation, the Scopus search engine was examined using the user interface to determine its search model and adapt the query formulation methods. Whenever a user fills a query in the search box, it is translated to a boolean *AND* query between all the query words. However, if a sequence of query terms is placed under quotes, Scopus will search for the sequence *as it is* and not generate a boolean query between the terms. Scopus also provides a syntactic way to issue boolean queries using the capital case AND / OR words and placing the other attributes under quotes.

Following this behaviour and the available attributes of an article, several methods to infer queries are introduced:

1. formulate a query from the paper's title by removing the stop words and submit it as a boolean query,
2. submit all keywords as a long OR boolean query,
3. generate all possible combinations of two keyword terms and submit them as individual boolean queries,
4. generate all possible combinations of two keyword terms as a big OR query. An important note here is that Scopus limits the character count for a query to 256,
5. apply the same method in (2) for index terms,
6. apply the same method in (3) for index terms,
7. apply the same method in (4) for index terms,
8. use a language model to generate a query from the abstract and title.

To satisfy point 8 we train a pLSA model as presented by Steinberger to generate a summary from the article's abstract and title [59] (Section 2.5.1). Even though, as discussed in Section 3.3.1, academic users tend to submit new queries instead of reformulation (points 2 and 5), we propose to also combine the power of key and index words as for points 3,4,6 and 7.

Several examples for each query type are provided in Appendix A, Section A.1.

## 3.4 Treatment validation

The treatment validation method provides the answer to the first research question (RQ1) formulated in Section 1.3 and follows the scenario described in Section 3.3.3.

Using the afore-mentioned query formulation methods, a dataset is created and evaluated. This section is divided into smaller sub-sections to individually treat all experimental aspects. In the first sub-section (*dataset*) the methodology for generating a dataset is provided together with valuable insights upon it. Later, the results are showcased and, lastly, a conclusion is given together with the answer for RQ1.

### 3.4.1 Dataset

In order to make the experiment consistent, the initial dataset must be generated through a deterministic methodology. To validate the scenario from Section 3.3.3 only a set of scientific articles is needed. We use a past experiment in expert finding, where all the results are public, as follows:

1. select the top 10 research areas from Aminer [6] as in [60],
2. select the top 5 experts for each area according to [61] that exist in Scopus,
3. for each expert, select the most cited paper and the most recent paper with more than 10 citations available in Scopus. If the papers overlap, the next most recent paper is selected. If there is no paper with more than 10 citations, the paper with the number of citations closest to 10 is selected.

Choosing articles from experts in one field can bias the results. In order to avoid the bias we propose to generate another collection of documents in a stochastic manner and use it to validate the result's on the expert collection. We further address the second set of documents as the *validation* set. To generate it we use the top 10 research area names as in [60] as queries for Scopus and get the first 5 (most recent) results.

Further on, due to the implementation considerations (Section 1.5) the articles which have at least one reference title with less than 5 characters (after parsing) are removed. The articles with more than 50 references are also removed because they are usually books, cover a broader range of topics and have a high number of references. After this procedure is executed, the first collection (test collection) consists of 63 topics and the validation set (collection 2) of 50.

Considering the textual approach for query generation, it is important to illustrate some statistics about the test dataset:

- the average number of references is 26,
- the average title length is 6 words,

[6] https://aminer.org

- the average number of keywords is 2,
- the average number of index terms is 4,
- the average abstract length is 97 words,
- all documents indexed in Scopus one year before the article from which queries are generated are used for search.

### 3.4.2 Results

The performance is evaluated based on the precision and recall measures introduced in Section 3.3.2, at different thresholds. All queries that failed to retrieve any reference were removed before evaluation, as they are not useful for an evaluation collection with automatic relevance judgements.

A comparison between the number of queries that returned at least one reference and queries that returned none, for each query type, is provided in Figure 3.1 for the test collection and in Figure 3.2 for the validation collection. The blue bars illustrate the amount of queries that returned at least one reference and the red bars represent the amount of queries that returned no reference. A method to interpret the graphics is that if the number of queries that returned at least one reference (blue bars) is bigger than the number of queries that returned no reference (red bars), the query generation type performed better.

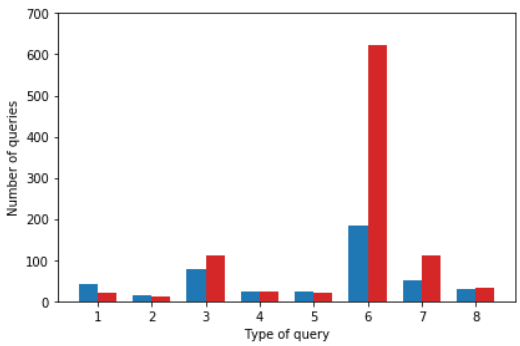


**Figure 3.1:** The number of queries that returned at least one reference (blue) vs. the number of queries that returned no references (red) for test set.

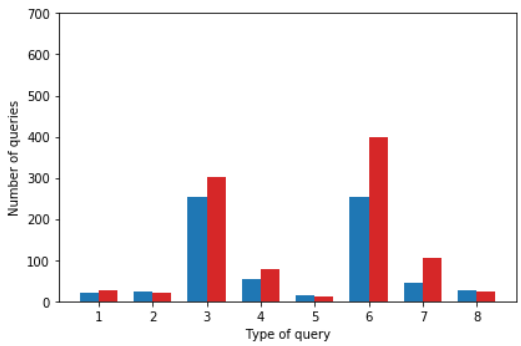


**Figure 3.2:** The number of queries that returned at least one reference (blue) vs. the number of queries that returned no references (red) for validation set.

In total, from a number of $1402$ queries in the test set, $645$ returned at least one reference and $757$ returned none, leading to a percentage of approximately $46\%$

successful queries. Regarding the validation set, from a total of $1662$ queries, $694$ returned at least one reference while $968$ returned none or empty result sets, leading to a percentage of approximately $41\%$ successful queries. One spike might attract attention in both Figure 3.1 and 3.2; the number of queries generated by combining two keywords or index terms together. Since combining all the elements in a set is of exponential order, the spikes for query types (3) and (6) are expected.

Further on, in Table 3.1 and 3.2 the average precisions for the first 3, 5, 10, 20 and 50 results are showcased. The first column, *type of query*, represents the query generation method from Section 3.3.3 while the headers *p@treshold* describe the precision results at different thresholds. In the last column, the percentage of queries that returned at least one reference is displayed.

By removing the queries that returned no references, the average number of references drops to $25$ for collection 1 and $31$ for collection 2. Given the implementation considerations presented in Section 1.5, a total of $20\%$ of the references could not be automatically parsed, leading to an average of $\approx 20$ references and, respectively, $\approx 24$ for the validation test. Thus the average precision at $20$ can be at most $1$ for both the test collection and the validation set. For the last threshold, $50$, the average precision can not exceed the approximate score of $0.4$ for the test collection and $\approx 0.48$ for the validation collection.

The first 3 thresholds introduced by Table 3.1 and 3.2 are smaller because of their implicit importance. The number of times a user checks the results ranging from position 20 to 50 (pages 2 to 3 in Scopus) is relatively small. Nevertheless, the first two pages (up to 40 results in Scopus) have a high probability to be seen.

| Type of query | p@3 | p@5 | p@10 | @p20 | p@50 | Queries that returned at least one reference |
|---|---|---|---|---|---|---|
| (1) | 0.33 | 0.2 | 0.17 | 0.1 | 0.049 | 65 % |
| (2) | 0.0625 | 0.05 | 0.043 | 0.034 | 0.032 | 55 % |
| (3) | 0.17 | 0.11 | 0.085 | 0.06 | 0.038 | 41 % |
| (4) | 0.20 | 0.14 | 0.11 | 0.094 | 0.051 | 50 % |
| (5) | 0.025 | 0.030 | 0.015 | 0.022 | 0.013 | 53 % |
| (6) | 0.12 | 0.095 | 0.071 | 0.052 | 0.025 | 22 % |
| (7) | 0,13 | 0.092 | 0.070 | 0.061 | 0.032 | 31 % |
| (8) | 0.56 | 0.30 | 0.23 | 0.17 | 0.081 | 47 % |

**Table 3.1:** Precision on test set.

Using the title (1) or the abstract's summary (8) leads to good results on the test set. At a threshold of $10$, an average of $0.17$ and, respectively, $0.23$ for the first 10 results means that $\approx 2$ ($\approx$ 10%) references can be found on the first page.

| Type of query | p@3 | p@5 | p@10 | @p20 | p@50 | Queries that returned at least one reference |
|---|---|---|---|---|---|---|
| (1) | 0.27 | 0.162 | 0.081 | 0.07 | 0.040 | 44 % |
| (2) | 0.038 | 0.049 | 0.042 | 0.038 | 0.030 | 52 % |
| (3) | 0.11 | 0.066 | 0.04 | 0.051 | 0.030 | 45 % |
| (4) | 0.24 | 0.144 | 0.029 | 0.091 | 0.072 | 40 % |
| (5) | 0.037 | 0.022 | 0.066 | 0.055 | 0.028 | 58% |
| (6) | 0.14 | 0.084 | 0.071 | 0.047 | 0.025 | 38 % |
| (7) | 0.16 | 0.096 | 0.085 | 0.07 | 0.034 | 30 % |
| (8) | 0.29 | 0.17 | 0.089 | 0.073 | 0.042 | 52 % |

**Table 3.2:** Precision on validation set.

Further on, the number of references increases slowly with the number of results: at a threshold of $50$ only $\approx 2$, respectively, $\approx 4$ references can be found. Out of an average of $20$ maximum references, the last method provides a precision of $\approx 30\%$ compared to $\approx 20\%$ returned when using just the title. Given the heterogenous nature of a scientific article; a publication spans various topics and cites other papers as related work, for various methods, theories or future research, a precision of $\approx 20\%$ can be interpreted as a good result.

On the contrary, the sparse nature of only one keyword (2) or index term (5) leads to poor results. It is interesting to observe that, even though index words are chosen by Scopus, their performance proves weaker than that of keywords chosen by the paper's authors. Yet, when combining several keywords or index terms (4) and (6) the precision increases considerably. Using a combination of two terms (3), (4), (6), (7) leads to an increase of more than $50\%$ over using only one term (2), (5), at the threshold of $10$. However, these methods do not outperform the title (1) or the abstract's summary (8).

The results on the validation set are consistent with the ones obtained from the test set. However, on this data set, the precision for combinations of index terms (6) is bigger than (3) (using combinations of keywords). It can be that the data set contains more keywords per article and more articles with keywords (Figure 3.2). Nevertheless, the precision when using just keywords (2) is bigger than (5) for all thresholds, validating the statement that keywords chosen by the authors perform better than index terms attributed by Scopus. In this case, the author's experience in a field of study has no influence.

In Table 3.3 and 3.4 the recall at higher thresholds ($20, 100, 200, 500$ and $1000$) is illustrated. The first threshold is smaller because the average number of references is around 20. The choice for the higher thresholds is given by the nature of re-ranking algorithms. While a user might evaluate the first few pages of a search engine, it will

never reach the results in the interval $[500; 1000]$. However, a re-ranking algorithm, which re-evaluates and re-orders the results retrieved by an initial algorithm, can evaluate and boost any relevant document in the given interval.

The layout of the tables resembles the structure of Table 3.1 and 3.2, where the type of query is the one introduced in Section 3.3.3, the headers *r@treshold* presents the recall at the aforementioned thresholds and the last column displays the percentage of queries that returned at least one reference. The recall can take any value in the interval $[0, 1]$ since it depends only on the number of relevant results retrieved and the total number of relevant results (references) for an article. It can be interpreted as the fraction of relevant documents that are successfully retrieved. Therefore, the recall measure can, at least, remain constant while the number of results increases.

| Type of query | r@20 | r@100 | r@200 | r@500 | r@1000 | Queries that returned at least one reference |
|---|---|---|---|---|---|---|
| (1) | 0.1 | 0.17 | 0.22 | 0.25 | 0.27 | 65 % |
| (2) | 0.034 | 0.11 | 0.16 | 0.18 | 0.19 | 55 % |
| (3) | 0.06 | 0.13 | 0.14 | 0.16 | 0.19 | 41 % |
| (4) | 0.094 | 0.17 | 0.20 | 0.22 | 0.22 | 50 % |
| (5) | 0.022 | 0.065 | 0.11 | 0.12 | 0.12 | 53 % |
| (6) | 0.052 | 0.080 | 0.009 | 0.011 | 0.015 | 22 % |
| (7) | 0.061 | 0.10 | 0.12 | 0.13 | 0.14 | 31 % |
| (8) | 0.17 | 0.27 | 0.32 | 0.35 | 0.36 | 47 % |

**Table 3.3:** Recall on test set

.

| Type of query | r@20 | r@100 | r@200 | r@500 | r@1000 | Queries that returned at least one reference |
|---|---|---|---|---|---|---|
| (1) | 0.058 | 0.10 | 0.15 | 0.21 | 0.21 | 44 % |
| (2) | 0.032 | 0.074 | 0.096 | 0.18 | 0.27 | 52 % |
| (3) | 0,043 | 0.11 | 0.14 | 0.13 | 0.13 | 45% |
| (4) | 0.076 | 0.11 | 0.13 | 0.18 | 0.19 | 40 % |
| (5) | 0.046 | 0.064 | 0.11 | 0.15 | 0.16 | 58 % |
| (6) | 0.041 | 0.050 | 0.06 | 0.10 | 0.11 | 38 % |
| (7) | 0.058 | 0.083 | 0.09 | 0.11 | 0.12 | 29 % |
| (8) | 0.059 | 0.067 | 0.10 | 0.11 | 0.15 | 52 % |

**Table 3.4:** Recall on validation set

.

For thresholds $20$, $100$ and $200$ the results are coherent with the precision results introduced earlier. As the number of results increases, the recall also increases, but slowly. From a threshold of $200$ up to $500$ the recall increases by a factor of $\approx 0.1$ up to $\approx 0.6$ leading to an average of $2$ relevant results from a total of $300$. Only some query types generate more relevant results as we move from $500$ to $1000$ results. Here, an increase of $0.1/0.2$ in recall generates a maximum number of $1$ relevant result from a total of $500$ articles.

Once again, the second data set presents consistent results. However, since the total number of relevant results (references) is bigger, the overall results on this data set are smaller, proving consistency over the total number of relevant results returned by Scopus.

### 3.4.3 Discussion

The treatment validation is carried out on two different datasets; the first one contains scientific articles written by experts in their field of study, while the second one consists mostly of articles published in the same year the experiment ran, or later (most of them are indexed in Scopus before the official publishing year). For each article in the dataset, a number of queries is generated through the procedure described in Section 3.3.3.

Overall, only a small percentage over $40\%$ from the total number of queries retrieve at least one reference. From the set of queries that retrieved no references, the biggest number of queries are generated through the types (3), (4), (6) and (7), where combinations of keywords or index words are used. This is because the number of queries is significantly bigger, but also because two key or index words can not be found together in the textual information used by Scopus. Usually, such keywords describe general topics tackled in an article. For example, the index term *information retrieval* describes a topic from computer science and does not reveal useful information about the subject studied. Such an article can tackle problems ranging from ranking algorithms to evaluation or relevance assessment.

Out of all queries that retrieve at least one reference, the first and the last query generation methods, namely using the article's title or a summary extracted from the article's abstract and title have the highest precision up to $50$ results. When keywords chosen by authors are used, the precision is higher than searching with index words chosen by Scopus. However, since both keywords and index terms have a general character, the results for using them individually proves weak.

The datasets show consistent results in both performance and the number of relevant queries. The importance of one author and her expertise in a field of study does not bias the query generation methods.

The first research question (RQ1) from Section 1.3 asks if *the information from scientific articles can be used to create test collections with automatic relevance judgements?* In search for an answer we chose to model the following search task: researchers use Scopus in search for references for their new article using informational queries. After issuing a series of queries to Scopus some of the results are referenced in the new article.

In order to simulate the working context a number of articles indexed in Scopus was selected and a set of queries was generated from their content. The search was limited to retrieve documents indexed at most in the year before the article was published. After sending the queries to Scopus, only the results that were referenced in the selected article are considered relevant. For this search task a test collection with automatic queries and relevance judgements was built. However, the queries could only retrieve $\approx 10\%$ of the references. If any result retrieved by Scopus is relevant to the query, but not cited in the article from which the query was inferred, its relevance can not be appreciated.

We suggest to only use such a collection during algorithm training phases and ask for explicit relevance judgements from users or use a manually annotated collection for final validations and discussions. For example, in Chapter 4 the test collection from this chapter is used to try different personalization methods in academic search, but the final conclusion is drawn from a user study.

# Chapter 4

# Personalized Ranking in Academic Search

## 4.1 Introduction

In this chapter we introduce several approaches for online, personalized, ranking in academic search. We focus on the academic background of researchers that interact with Scopus since information about their academic profile is already available. At first, a graph similarity measure is chosen to evaluate how similar to the user's context are the top $200$ results retrieved by Scopus. Later, a topic model is used to evaluate the result's content against the user preferences for research topics. Towards the end both methods are integrated with the initial Scopus rank.

The rest of the chapter is organised as follows: in Section 4.2 we discuss other approaches to personalization for general web and academic search. In Section 4.3 we experiment with two personalization approaches on the dataset from Chapter 3 and present our proposal. The final solution is validated through a user study. Both the experiment setup and the results are discussed in Section 4.4

We use the same terminology introduced in Section 1.2 as follows: *treatment design* (Section 4.3) describes the system developed in order to achieve personalization in academic search and *treatment validation* (Section 4.4) judges the quality of the system and evaluates the results.

## 4.2 Related work

Personalized search aims to build systems that can retrieve custom documents from a collection, based on a model that represents the user's preferences and the context of their activity. The choice for which attributes and methods to use is driven by the different stages of the retrieval process where personalization can be applied.

To implement personalization as part of the initial retrieval process requires the search tool to embed user attributes in the retrieval algorithm. Other approaches are to *re-rank* the first $n$ results retrieved by a neutral algorithm or to expand or modify the query in order to provide a broader representation of the information need. An illustration of these techniques is provided in Figure 4.1.

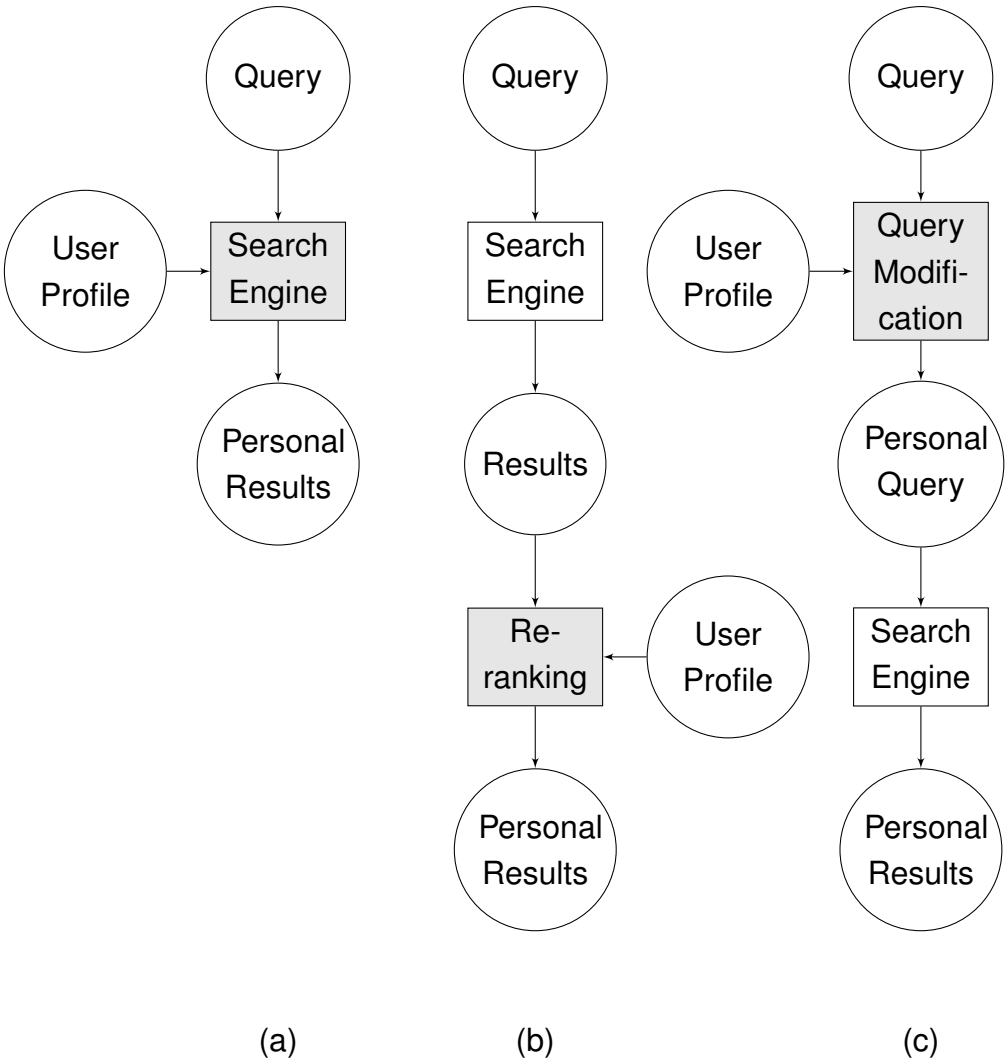


**Figure 4.1:** Personalization process where the user model is used during the retrieval process $(a)$, in a distinct, re-ranking, step $(b)$ or during query pre-processing $(c)$.

Embedding the user profile in the initial ranking algorithm improves the response time since no additional computation is required. However, a re-ranking step allows the user to selectively choose personalization attributes. The latter can also be implemented on the client-side as a software component can retrieve the initial results from a search engine and re-rank them locally. Nevertheless, when implemented on the server side, a more complex model of the user can be exploited. The user model

can also be used to modify the query before sending it to a retrieval algorithm. For example, an initial query can be expanded with keywords or augmented with topics.

Moreover, we classify personalization algorithms based on their processing time as *online* and *offline* algorithms. Online approaches acquire and use data as soon as it is available. Even though this approach provides real-time updates, offline methods can implement more complex algorithms because, as opposed to the online approach, they have no time constrains.

The data acquisition techniques for user modelling fall into two categories: *explicit* and *implicit* feedback. The former requires users to provide explicit relevance feedback on a subset of documents or manually select topics of interest. However, end users are usually unwilling to spend extra effort to explicitly specify their needs or refine them by means of feedback [62]. Since users might not understand how the matching process works, the information provided is likely to miss important attributes. Contrary, implicit feedback techniques track and monitor a user's behaviour without explicit involvement.

Based on the level of personalization, data can be collected either on server-side (e.g. server logs for queries, browsing histories or click-through data) or on the client-side (e.g. cookies). Given the type of feedback required, the attributes used to build the user model and the stage of the retrieval process, a classification of personalization approaches is presented in Table 4.1

| Personalization based on: | Implicit / Explicit Feedback | Retrieval stage | Input Data |
|---|---|---|---|
| Search History | implicit | initial ranking, re-ranking | past queries, selected results |
| Collaborative | both | re-ranking | past queries, selected results |
| Result Clustering | explicit | re-ranking | selected results |
| Hyperlinks | both | initial ranking, Re-ranking | queries, selected results, recent documents |

**Table 4.1:** Types of personalized search classified by the type of feedback used to learn the user profile and keep it updated.

### 4.2.1 Personalization using search history

Techniques based on past queries and search history fall in the same categories introduced earlier: offline and online processing. Sperreta and Gauch [63] proposed

an online ranking tool which provides personalization by creating user profiles from past queries. The profile is later used to re-rank the results returned by an external information retrieval tool. For each user, two types of information is collected: the queries submitted, for which at least one results was visited and related text snippets such as titles or abstracts from the selected results.

A classifier trained on the ODP's hierarchy will choose the relevant concepts from the selected information snippets and assign weights to them. When a query is submitted to the wrapper, it is forwarded to an external search engine and the top results are classified using the same hierarchy. Finally, an algorithm evaluates the similarity between the concepts associated with the results and the user's profile and re-orders the results.

Liu and Yu [64] took a similar, online, approach to personalization. The user profile is built by analyzing the user's search history and comparing the past selected results with the first 3 levels of the ODP hierarchy. For each query, the most appropriate categories are inferred and used along with the query as the search context. Because, in general, the queries are short, they are often ambiguous, therefore likely to match multiple categories in the ODP. Their system uses the top matching categories for query expansion, or asks the user to explicitly choose one of the three top-ranked categories.

One of the state of the art offline algorithms was introduced by Sun et al. [65] in the form of *CubeSVD*. This approach follows the normal search scenario where after submitting a query to a search engine, a user clicks on the interesting pages. During usage, the system stores triplets of click-through data represented as $< user,\ query,\ visited\ page >$ that reflect the user's interests. The algorithm models the click-through data as a 3-order tensor on which a 3-mode analysis is performed using *Singular Value Decomposition* technique, generalised from *Higher Order SVD*. The tensor measures the preferences of a $< user,\ query >$ pair given a document. Although the whole computation is time-consuming, it is carried offline. However, the algorithm has runs periodically in order to use new click-through data.

### 4.2.2 Collaborative search

In order to develop collaborative search engines, a similarity measure between users must be developed. Two queries should be considered similar in spite of the fact that they have no terms in common, just by looking at the relationship between the documents retrieved. For example, in the I-SPY collaborative search engine [66] all queries are treated as sets of unique terms on which the Jaccard measure (Section 2.3.4) is applied to measure the similarity.

Kritikopoulos et al. [67] propose to use web communities for collaborative search.

In a pre-processing step web communities are identified using the hyperlink structure (as in HITS algorithm [14]). Later, if the user frequently visited documents from a community $C$, whenever he submits a query about community $C$, all the results from this community are considered important and their rank is boosted.

### 4.2.3 Result clustering

Clustering groups results into folders by classifying them based on topics of interest. User attributes are used to identify the user's topic interest and show the relevant folders first. This step is usually performed after the initial results are retrieved from an information retrieval tool. In order to respect time constraints, the algorithms only use snippets from documents to represent the result's content.

Zeng et al. [68] formulate the clustering problem as a supervised salient phrase ranking problem and propose an algorithm that learns how to cluster results using different regression models. Ferragina et al. [69] asked users to select a subset of clusters that are likely to satisfy their needs, then expand the query with new keywords from the selected cluster.

### 4.2.4 Hiperlink personalization

One of the first personalization algorithms which makes use of hyper-textual data is the work of Page and Brin [10] who propose to bias the general PageRank score (Equation 2.2) for documents that a random surfer periodically jumps to, by introducing a teleportation factor towards these (Equation 2.5).

However, due to the sheer size of both the indexed documents and the search engine users, the offline computation is time and space consuming. Glen Jeh and Widom Jennifer [13] proposed an enhanced version of Personalized Page Rank using *partial* vectors computed offline and composed at query time, making personalized search easier to scale.

Using this approach, Chirita et al. [70] introduced a platform which provides personalized ranking of web pages based on bookmarks and frequently visited websites. In this system all pages considered relevant by a user during a search session are saved in a module which collects hub pages related to user topics. A second algorithm combines the PageRank scores with personalised hub scores (computed by running a customised version of HITS algorithm) and forwards the result to a Personalized PageRank algorithm (Section 2.3.2) which performs a final re-ranking and returns the results to the user.

Taher Haveliwala [11] proposed to bias the general PageRank algorithm with a *topical* importance score such that a set of pages can be considered important for

a domain, but excluded from others. The algorithm computes, for each document, a set of 16 topic-sensitive PageRank values and uses them each time a user submits a query. The query is classified to a set of topics and a linear combination of the Topic-Sensitive Rank is used, using the weights from the classification.

### 4.2.5 Ranking in academic search

Due to their proprietary character, academic search algorithms are not publicly disclosed. In an experimental attempt to reverse engineer Google Scholar [1], Joran Beel and Bela Gipp compared how publishing date, citation count or terms frequency in a document affect ranking [71]. Their results show that citations count weights more over search term frequency (which proves weak) and title search.

Beside ranking, bibliometric attributes are used to evaluate the scientific quality of documents. Such attributes help to assess the prestige of an author [72], journals [73] or the quality of the documents [74]. Moreover, the quality of bibliometric attributes reciprocally increases the quality of related entities: first rate scientific articles reference similar ones and are published by prestigious journals.

Given the nature of the entity they represent, bibliographic networks fall into two categories:

1. homogenous networks: in which only one type of entity is represented and
2. heterogenous networks: where several, different, entities are linked.

For example, algorithms such as HITS (Section 2.3.3) or PageRank (Section 2.3.1) can be applied on homogenous networks to discover authoritative sources for a field of study. Liu et al. [64] developed a revisited PageRank algorithm for a co-authorship network where edges are weighted by co-authorship frequency to evaluate an author's prestige. The measure can be used to evaluate research impact, journals or conferences.

Works in heterogenous networks propose to either rank one type of entity in relation to others or jointly rank several types of entities. A document can be evaluated based on the social importance of its authors in relation to a research topic.

Jabeur et al. [75] integrated author and documents tags and annotations into a bibliographic graph in order to compute a social relevance score for each document. This measure is then used, similar to PageRank, to rank documents for search. The validation experiments are carried out on a test collection from CiteSeer[2] search engine. Their results show an improvement between $15 - 55\%$ compared to a standard

[1] http://scholar.google.com
[2] http://citeseerx.ist.psu.edu/index

Term frequency-Inverse document frequency (TF-IDF) model. However, information retrieval tools use algorithms more complex than plain TF-IDF.

Another technique to exploit bibliographic networks is to use link analysis or topical indicators for multi-entity ranking and co-ranking. Assuming that important authors publish important documents and important entities cite them, bibliometric indicators are used to compute ranking scores based on graph relationships or text similarities.

Zhou et al. introduced [76] a framework for co-ranking entities of different kinds in a heterogenous network. The network consists of a social graph connecting authors, a citation network connecting documents and a bipartite authorship network that ties the previous two together. Tang et al. [77] investigated the problem of modelling heterogeneous academic networks using a unified probabilistic model. Three topic models for simultaneously modelling papers, authors and publication venues based on LDA (Section 2.5.2) are introduced together with a path to integrate them into a random walk framework.

The work is similar to Topic-sensitive PageRank applied to academic networks given that, since such networks comprise of heterogenous entities, specific topic models should be developed for each entity. However, none of these approaches was suggested for personalization in academic search.

### 4.2.6 Personalized academic search

Personalization in digital libraries was first classified by Neuhold et. al [78] into *services* and *content personalization*. The services side implements recommendation systems while content personalization considers personalized search. Collaborative filtering is one of the most successful techniques for building recommender systems and is being extensively used in personalized search engines.

In academic contexts, recommending suitable research articles can facilitate knowledge discovery or exchange and ultimately improve research productivity. This problem is tackled from two dimensions, namely, the social relations between entities and the related semantic information. However, using just one of the methods can lead to low accuracy since the former can't take into consideration textual information about and the latter judges a probability distributions for words, but can not position a work in a social context.

Xu et. al [79] proposed a framework which integrates both network and semantic aspects in a layered architecture: the first layer represents researchers and connections between them through bibliometrics attributes and the second layer represents semantic concepts and topical terms used to represent the researcher's expertise and interests.

Sun et al. [80] investigated the implicit user feedback from access logs in the CiteSeer academic search engine in order to build a re-ranking algorithm for personalized search. Their model forms pair of any two documents $D_1, D_2$ presented to a user for the same query. One document is preferred if the user requests additional information about it. Such pair of documents represent an instance of implicit feedback from the user. For a given set of document pairs representing the implicit feedback, an optimal preference vector is chosen to correctly rank all pairs of documents.

Based on the user preference model, the personalized ranking algorithm will reorder an initial set of results by the similarity score between documents and user preference vectors. Overall, the personalized ranking method proves accurate on predicting users interests when they are stable over time. However, if a researcher is interested in different topics which are not consistent with her past, the ranking method proves inaccurate.

## 4.3 Treatment Design

We use the term treatment design to define the process of building a personalization algorithm. As introduced in Section 4.2, the design of an algorithm is driven by time constrains. In offline personalization the initial retrieval algorithm embeds user attributes and has no time constraints while an online re-ranking method has to satisfy the normal response time for web search. Offline methods can use complex algorithms, but limit the user's choice to selectively employ personalization attributes, while online methods allow the user to specify which attributes should be used in personalization and provides more flexibility.

We believe that an online algorithm can add versatility to personalization as users can selectively chose the attributes used in re-ranking and can cope with interest shifts over time. To avoid the problem described by Anick [62] where users might not be willing to invest extra effort for personalization, the algorithm should provide a *default* mode, where a set of standard attributes is used.

The choice for which attributes to use in personalization remains an open question. As bibliometric attributes play an important role in academic entity ranking, they are also a good candidate for personalization.

However, using just the relationships between academic entities can have limited impact since no semantic information is included. Xu et al. [79] proved that using both network and semantic elements outperforms systems with only network or only semantic features for personalized research article recommendation. We believe this behaviour preserves for personalized academic search.

One approach to integrate semantic knowledge in the bibliographic graph is to embed topic sensitive scores, similar to the work of Haveliwala [11] presented in Section 4.2.4. Another graph walk approach was suggested by Tang et al. [77] where a bibliographic graph is augmented with topics and query nodes.

Before merging semantic and network elements we analyse the individual use of random walks and topic models for personalization (Sections 4.3.1 and 4.3.2). Later, in Section 4.3.3 we introduce a method to merge these initial results and propose a final algorithm in Section 4.3.4.

### 4.3.1 Graph models

The entities and relationships from academic search are introduced in Section 2.4 as a bibliographic graph structure. In this section we use the same structure for personalized search.

In order to satisfy time constraints for online algorithms and allow users to manually select attributes for personalization, the number of elements in a graph should be limited. To this extent we introduced the notion of *context* at $n$ (Section 2.2.2) which describes the degree of a node in a graph, given the in and out neighbours, at a maximum distance of $n$ edges. Its purpose is to extract attributes for personalization from the structure of the graph.

For example, given an author, the context at rank $1$ consists of the node representing the author and the first neighbours whom have a direct in or out connection; the authored articles and the co-authors (if this relation is defined as in Figure 2.7). Further on, for a scientific article, the context at $1$ represents the article's node together with its references, citations, authors and publication source (journal). This measure allows users to selectively chose attributes for personalization.

One might consider a set of papers or an author important for his current search context, therefore, he can select one of the entities for personalization. If no attributes are selected, we define the *default* mode where the user node in the graph is automatically chosen. The context at $n$ is used in the following way: considering that information about the user's academic background is available in Scopus, the context at $2$ is extracted for the user node from the social graph. The subgraph consists of the papers authored by the user together with their references, citations, journals and all co-authors.

After submitting a query to Scopus, the context at $1$ for the first $n$ results is used to augment the afore mentioned graph. The information consists of references, citations, authors and journals for each result. A similar approach, where the initial result set is augmented using the hyper-link structure, is taken in HITS [14] algorithm (Section 2.3.3) for general web search. After retrieving the first result set (called root)

from an information retrieval tool, the set is expanded by adding all the pages that are linked to and from the root set. Afterwards, the HITS algorithm computes two scores used to re-rank the pages from the new data set. The algorithm ensures that pages relevant to a query which were not initially retrieved are included in the new set.

We investigate the use of a graph similarity measure to evaluate how close a result is to an attribute selected for personalization and use it to reorder the results. Moreover, since the resulting graph contains the result's references and citations, we also evaluate their similarity, as in HITS.

A discussion about graph similarity measures is provided in Section 2.3.4. A simple approach is to measure the shortest path between two nodes. However, it is desired to evaluate several paths between two nodes and propagate the similarity measure accordingly. For example, if two authors are connected through an authorship and a co-citation relation, but also through a co-reference relation for different articles, the similarity measure should take into account both paths. Other measures introduced in Section 2.3.4 use node neighbourhood or the overlap of nodes neighbours to compute similarity scores.

The last measure, SimRank, assumes that two objects are similar if they are related to similar objects. This measure is able to propagate the similarity score through different edges of the graph, therefore, we select it to compute a similarity score between the user's node and the result's context. The score is later used to re-rank the documents. A formal definition of SimRank is given in Equation 2.8.

In order to test if this method will increase Scopus performance, the dataset from Section 3.4.2 is used as follows: we select the queries that returned at least one reference and were formulated through methods (1) and (8) (a total of $54$ queries). Analysing the recall from Table 3.3 and 3.4 for query generation types $1$ and $8$ shows an insignificant increase after the first $200$ results. Therefore, after sending each query to Scopus, only the first $200$ results are considered for re-ranking.

We assume that the first author of an article from which the queries were generated initiates the search and, for each query, build a graph consisting of the context at $2$ for the author and the context at $1$ for each result. When building the graph, the co-author relationship can be represented either as two symmetrical edges between two author nodes (Figure 2.7) or it can be inferred from two author edges as in Figure 2.5. If the first method is used, the context at $2$ for an author should also include all the articles published by the user's co-authors.

The similarity between a user and each result is computed using SimRank and, using this score, the initial results are re-ordered. Moreover, we measure the similarity between the user and the new articles added by expanding the dataset with each result's references and citations. Both ways to represent the co-author rela-

tionship are analysed and the impact on precision is illustrated in Figure 4.2. Similar to Chapter 3, one result is judged as relevant if it is referenced in the paper from which the query was generated. Precision reflects the number of relevant results, in our case the number of references, from an initial set of results.

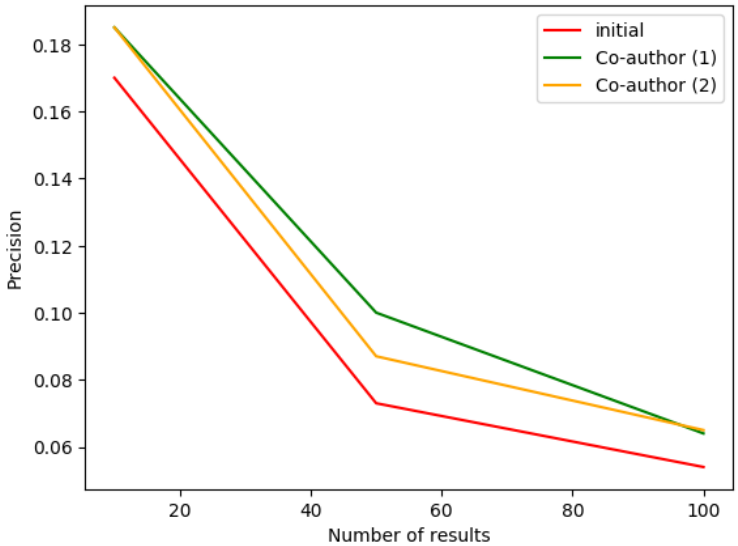


**Figure 4.2:** Comparative precision for using a graph similarity measure to re-rank the top 100 results retrieved by Scopus.

Using the similarity between an author and the result's context shows a small increase in precision for both representions of the co-author relationships. The first method performs slightly better for the first $20 - 100$ results, however, it introduces a lot of complexity.

Given the time constraints and because the probability to access results in the interval $[20, 100]$ is smaller, we further use the second method to represent the co-author relationship; as a second order relation derived from author relationships.

Even though the precision difference between the similarity ranking and the initial results is small, the testing method can not evaluate documents that are not referenced in an article from which a query was formulated. The recall scores from Table 3.3 and 3.4 show that for the top $100$ results only maximum $3$ or $4$ references are returned, in average, by Scopus. Therefore, it is not yet practicable to judge the confidence of this increase. Later, in Section 4.4.3 we present the results for a significance test.

### 4.3.2 Topic models

The use of SimRank, as introduced in Section 4.3.1, can not assess if a result is semantically relevant to a query. Recent approaches to model the documents content

assume that a probability distribution of words in documents can be expressed as a mixture of topics.

For example, in LDA (Section 2.5.2) the generation of a collection of documents is modelled as a three step process. At first, for each document a distribution over topics is sampled from a Dirichlet distribution. Afterwards, for each word in the document a single topic is chosen according to this distribution and, finally, each word is sampled from a multinomial distribution over words specific to the topic selected.

The same approach was suggested in [81] to model the interests of authors for documents. In the author-topic model, a group of authors, $a_d$, write document $d$, and, for each word in the document an author is chosen uniformly at random and a topic is inferred from a distribution over topics specific to that author. By estimating the parameters $\phi$ and $\theta$ (Section 2.5.2) we obtain information about which topics the authors write and the content of a document in relation to these topics.

More complex models include topic distribution of authors and journals are suggested in [60]. Here, the topic model is able to simultaneously represent the topic distribution of papers, authors and conferences. There are several strategies to integrate topic models into ranking: at first, the topics that an author prefers can be added as nodes of a graph and edges can be used to represent the connections between authors, documents, queries and topics. Another approach, suggested in [77] for academic search and [11] for general search is to integrate topic relevance scores in the graph walk process.

In Scopus, for each author and article a set of *research* or *subject areas* is automaticaly assigned. There are, in total, $27$ subject areas [82] for authors. However, the article research areas do not have a consistent structure. We believe they are smaller topics within the author's subject areas. In Appendix A, Section A.3.1 and A.3.2 some examples of research areas are given.

Since the information is already available, one approach is to train a topic model for all author research areas, match a query to one or several topics and integrate the author and query topics in the graph presented in Section 4.3.1. The SimRank measure will automatically reflect any correlations between topics and documents.

However, after a manual inspection of the author subject areas we discovered the research areas are not accurate. For example, for Yann Lecun, a computer scientist interested in machine learning and computational neuroscience, the list of subject areas from Scopus is *Computer Science, Engineering, Mathematics, Neuroscience, Social Sciences, Physics and Astronomy, Materials Science, Arts and Humanities, Medicine, Biochemistry, Genetics and Molecular Biology, Decision Sciences, Energy, Agricultural and Biological Sciences, Business, Management and Accounting, Multidisciplinary*. While the first 4 topics match his interests, the latter are far from his fields of study. *Biochemistry*, *Materials Science* or *Agricultural and Biological*

*Sciences* suggest that subject areas from Scopus are not accurate.

The same behaviour preserves for Andrew Ng, a co-author of the Latent Dirichlet allocation: within his research topics we find *Arts and Humanities*, *Business, Management and Accounting* or *Social Sciences*. Therefore, a choice is made to train, for each author, an author topic model and use it in re-ranking.

Because the training process is computational expensive, we use only the first author from an article to train the author topic model. This model corresponds to an LDA topic model (Section 2.5.2) for an author. A number of $3$ topics is chosen for each author by analysing the number of relevant topics assigned by Scopus on a small dataset. We use Equation 2.15 to choose the most relevant author topic given a query and, later, to evaluate the result's probability scores for the selected topic.

The same dataset from Section 4.3.1 is used for experiments. Since each query was generated from an article, the author topic model is limited to use only information available before the selected article was indexed. For each $54$ queries we train the author-topic model for the first author using the titles and abstracts from their papers indexed in Scopus. After a query is generated from the content of an article, we select the topic with the highest probability from the article's author topics. The query is sent to Scopus and the first $200$ results are retrieved.

A probability of relevance score using Equation 2.15 is computed for all results and their context at $1$ and used to order the documents. Two cases are analysed: the impact on precision without augmenting the graph with the context at $1$ for each result and the impact on precision after expanding the root result set. We use precision to measure the retrieval performance under the assumption that a result is relevant if it was cited in the article from which the query was generated. Figure 4.3 illustrates the results. The (1) indicates the precision results before expanding the result set and (2) indicates the precision after adding the result's context.

The second measurement, author topic model (2) leads to a decrease of precision. This is expected since the number of articles added through expansion is high. A small increase in precision can be observed for the first author topic model up to $\approx 40$ results. Once again, the testing method can not evaluate documents that are not referenced in an article from which a query was formulated. Therefore, it is not yet practicable to judge the confidence of this increase. Later, in Section 4.4.3 we present the results for a significance test.

### 4.3.3 Rank fusion

Instead of biasing the similarity measure from Section 4.3.1 with topic scores from Section 4.3.2, we propose to merge the results with the initial Scopus rank. Since both SimRank and the author model return results in the $[0, 1]$ interval, a very sim-

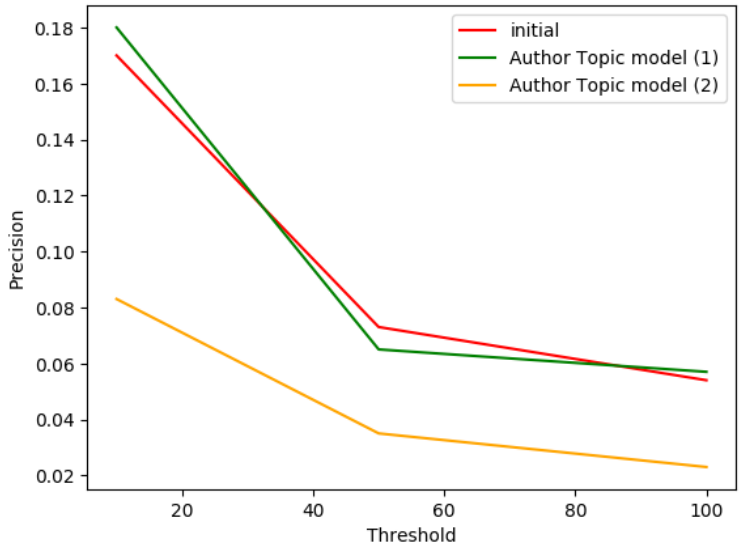


**Figure 4.3:** Comparative precision results for using an author-topic model to re-rank the top 100 results retrieved by Scopus (1) and the expanded result set by context at 1 (2).

ple approach would be to sum the scores together with a normalized Scopus rank. Another simplistic approach is to use a weighted, multivariable linear regression model. However, a phenomena with $N$ observed variables has an $N$ dimensional representation. Collapsing the $N$ variables into a single variable comes with loss of information, therefore, with a loss in accuracy. If the variables are correlated, the loss is maximised. In the current experiment the SimRank and author topic scores are not correlated, therefore, the loss of information and accuracy can be minimized.

An alternative is to use a fusion algorithm which can combine different document ranking functions. To this extent we introduce the Reciprocal Rank Fusion (RRF) algorithm [83], a simple, yet efficient rank fusion method described by:

$$RRFscore_{d \in D} = \sum_{r \in R} \frac{1}{k + r(d)} \tag{4.1}$$

where $R$ is a set of rankings generated through several methods and $D$ is a collection of documents to be ranked. $k$ is a constant that mitigates the impact of high rankings by outliers systems and was set to $60$ during the initial experiments.

In order to normalize the Scopus index, the following formula was used:

$$N(d) = \frac{d - min(D)}{max(D) - min(D)} \tag{4.2}$$

where $d$ is the rank of the document and $D$ is the full list of Scopus rankings. For documents without a Scopus rank, an equal rank of $N(max(D) + 1)$ was assigned.

After retrieving the top $200$ results from Scopus for each query introduced in Section 4.3.1, the scores from Section 4.3.1 and 4.3.2 are merged with the initial, normalized Scopus rank through an uniform linear regression model and the RRF approach. The results are illustrated in Figure 4.4.

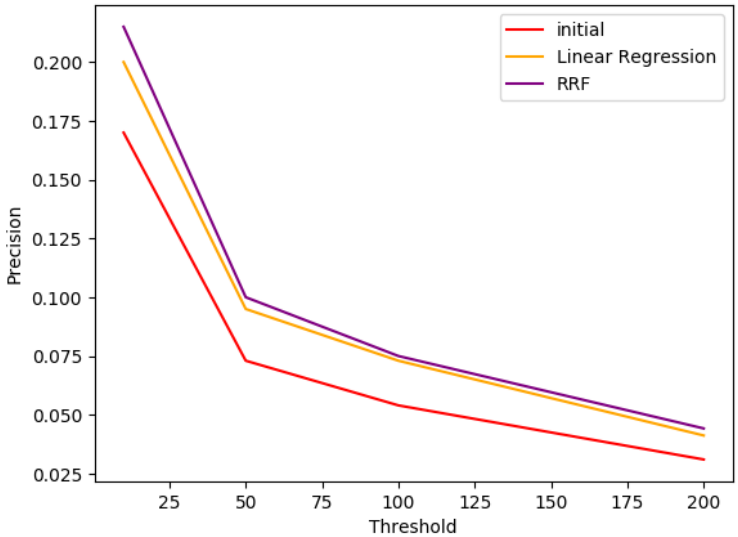


**Figure 4.4:** The impact on initial precision for different ranking fusion methods.

Overall, the RRF method for rank fusion performs better than linear regression. However, after randomly sampling $10$ queries and manually inspected the results, we discovered that both methods boost different relevant results to top $200$. This is mainly because the Scopus ranking has different scores for both methods. In the uniform, multivariable, linear regression method the Scopus ranking is normalized with values in the interval $[0.0049, 1]$ while in the RRF $[0.0016, 0.00038]$, making the Scopus initial rank smoother.

Before drawing conclusions, an impact of the $k$ constant over RRF ranking is needed, as for a variation from $60$ to $100$, the initial Scopus rank interval changes to $[0.0033, 0.0099]$. Using a different $k$ constant for all initial ranking functions might impact the overall sum. In Figure 4.5 we present this impact at thresholds of $60, 100$ and $200$.

With an increase from $60$ to $100$ the initial RRF precision increases with an average order of $0.01$, however, for an increase up to $200$, the ranking improves only with an average of $0.0073$. Therefore, we empirically select $k = 100$. A similar approach can be taken in the multivariable linear regression model by weighting the different variables.

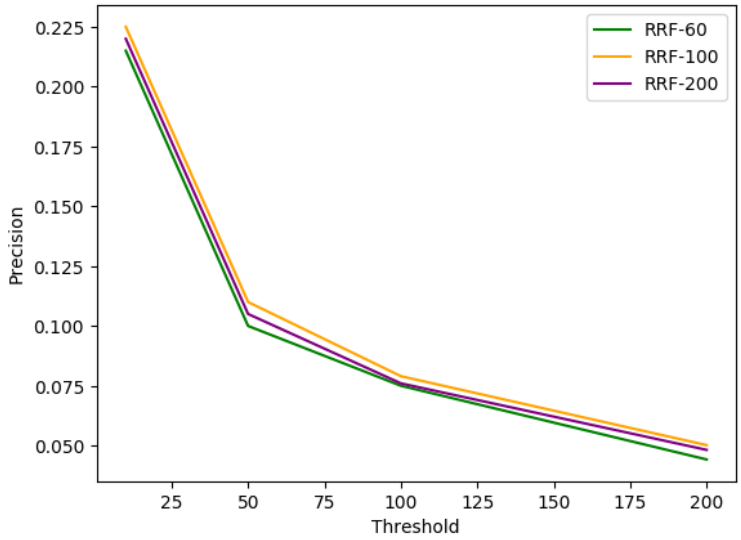


**Figure 4.5:** The impact of $k$ constant on RRF ranking fusion method.

### 4.3.4 Treatment Proposal

The final solution assumes that all author models are built and stored offline. In *default* mode, personalization is provided for one user (as described in Section 2.2.3). At query time, the re-ranking algorithm performs the following steps:

1. identify the relevant author topic from a query,
2. retrieve the initial 200 results from Scopus,
3. generate a uniform, direct, graph from the author's context at $2$ and the result's context at $1$ with the co-authorship relation as in Figure 2.5,
4. compute SimRank between the author and the following nodes: each result, each result's citation and each result reference,
5. estimate the documents topic distribution,
6. independently re-rank documents by SimRank and author model,
7. compute RRF and re-order results

Steps (1) and (2), (4) and (5) are done in parallel, while steps (2) and (3) can be merged.

If another attribute is selected for personalization (e.g. another author / paper), SimRank will measure the similarity between this node all other articles in the augmented graph, while, in case a paper is selected, the topic model will be retrieved from the first author. Different variations of these scenarios must be tested on a data collection with manual relevance judgements.

## 4.4 Treatment Validation

In order to study the potential performance improvement of our proposal, we conduct a user driven experiment and gather manual relevance judgements for a set of queries and results.

### 4.4.1 Experiment Setup

The user's study main goal is to evaluate the context based re-ranking algorithm proposed in Section 4.3 and validate the assumption from Section 3.3.3 which states that researchers looking for articles close to their work in progress will issue a query to an information retrieval tool and later cite documents considered relevant from the result set.

In order to achieve the proposed goals, the study will simulate a past working context such that the reference sets are available. Two key points which correspond with RQ2 and RQ3 are followed: (1) may be Scopus performance improved through personalization using topic models and graph walks? and (2) will users consider the same documents relevant after a period of time?

We asked $5$ participants from different areas of Computer Science to formulate $5$ queries for $5$ articles authored, each query corresponding to a different article. For each query, the information need is work related to the article for which the query was formulated. In order to not bias the results, the word *reference* was never mentioned in the experiment description.

The performance of a re-ranking algorithm must be measured against a result set called *baseline*. Since all the experiments are conducted on the Scopus data corpus, the results obtained by issuing a query to Scopus are the best candidate.

Each query formulated by a participant was sent to Scopus and the first $200$ results were selected for re-ranking (based on the recall results from Section 3.4.2). Later, relevance judgements were requested for the first $10$ results returned by Scopus and the first $10$ results returned through re-ranking. The personalization algorithm performed in *default* mode, which means it used the context at $2$ for each author together with the context at $1$ for each result (see Section 4.3.4 for more details). Both result sets were merged and shuffled before displaying them to the users.

We formalise the following steps for the user study:

1. select participants for user study,

2. select $5$ articles for each participant that are published in Scopus,

3. ask participants to formulate, for each article, a query they would use to search for related work at the time of writing the paper,
4. send each query to Scopus and retrieve the first $200$ results,
5. perform re-ranking as discussed in Section 4.3.4, using the context at $2$ for the authors and context at $1$ for each results,
6. take the first $10$ results from Scopus and the first $10$ results from re-ranking, merge and shuffle the data set,
7. present the results to each participant and ask for relevance judgements,
8. collect and evaluate the results using nDCG measure introduced in Section 4.4.2.

Based on the performance evaluation metrics, the relevance judgement for an article can vary from boolean values (relevant / non-relevant) to a graded relevance scale (e.g. slightly relevant, relevant, highly relevant, etc.). The choice for the current user study is motivated in the next section.

### 4.4.2 Performance measures and relevance judgements

Since documents are not of equal relevance to the users, highly relevant documents should be identified and ranked first for presentation. This is often desirable from the user point of view [84].

The practice of binary judgements gives equal credit for highly and marginally relevant documents. Therefore, differences between average and excellent retrieval techniques may not be revealed in the evaluation. In order to identify such differences, a discounting function, that progressively reduces the document score as its rank increases is needed. A simple way of discounting is to divide the document score by the log of its rank [84].

The Discounted Cumulative Gain (DCG) measurement at a particular rank position $p$ is formalised as:

$$DCG_p = \sum_{i=1}^{p} \frac{rel_i}{log_2(i+1)} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{log_w(i+1)} \tag{4.3}$$

An alternative formulation of DCG emphasises even more relevant documents [85]:

$$DCG = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log_2(i+1)} \tag{4.4}$$

However, in order to compare the performance of a search engine for multiple queries, the cumulative gain at each position should be normalized across queries by sorting all *relevant* documents in a corpus and producing the maximum possible DCG for $p$, also called Ideal Discounted Cumulative Gain (IDCG). For a query, the Normalised Discounted Cumulative Gain (nDCG) can be computed as follows:

$$nDCG_p = \frac{DCG}{IDCG} \tag{4.5}$$

where:

$$IDCG = \sum_{i=1}^{|REL|} \frac{2^{rel_i} - 1}{log_2(i+1)} \tag{4.6}$$

$|REL|$ represents the list of relevant documents in the corpus p to position $p$ and $DCG$ is computer using Equation 4.4.

In order to discriminate between highly relevant documents and marginally relevant ones, for the user study introduced in Section 4.4.1 a graded relevance scale was chosen as follows:

0. *Not Relevant* describes documents that are out of the (selected) paper's scope or research area and no information can be used as relative work,

1. *Slightly Relevant* documents are close to the paper's or the author's research area and have promising content; the user will click on it, but not use it as related work (in this case the paper can be used as a hub towards relevant articles or as general knowledge for the reader),

2. *Relevant* describes a highly relevant result that should or should have been used as relative work for the paper in question.

The choice for the relevance scale allows two types of relevant documents: (1) documents that are tangent to the working context, but can lead to relevant content or provide general knowledge close to the reader's interests and (2) documents that are highly relevant both to the query and the working context.

### 4.4.3 Results

One of the first observation provided by the participants was that one query can not encompass the whole context of an article. Some participants chose to provide multiple queries for the same article. A dataset of $37$ queries together with relevance judgements was created after completing the user study.

The result set for each query contains the top $10$ results returned by Scopus and the top $10$ results obtained from re-ranking the initial $200$ results. In some cases

the re-ranking result set contains documents present in the Scopus result set and documents that were never retrieved by Scopus.

On average, $54\%$ of documents are commonly retrieved by both algorithms and only $4\%$ from the remaining $46\%$ were selected by augmenting the result set with the corresponding context at $1$. An intersection of the result sets is showcased in Figure 4.6.

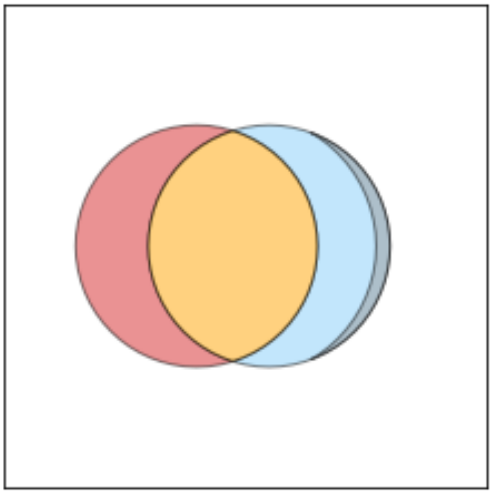

**Figure 4.6:** Result sets intersection where red represents the Scopus results, blue the results retrieved through re-ranking, yellow the common results for Scopus and re-ranking and dark blue represents the proportion of results added from the result's context.

From the common result set only a proportion of $6\%$ document ranks preserved after re-ranking. This shows that personalization can also impact the top results retrieved by an information retrieval tool. On average, from $94\%$ of documents which changed positions, $78\%$ decreased in rank and only $22\%$ increased in rank from the initial Scopus score.

Before presenting the users relevance judgements, the same methodology to generate automatic queries as in Section 3.3.3 was used for query types (1) and (8) (the ones that performed best). In this experiment a result is relevant if it is referenced in the chosen article. Figure 4.7 illustrates the precision for thresholds up to $200$ for both automatic query generation methods and the user formulated queries.

Interestingly, the methods provided in Section 3.3.3 perform better than the user formulated queries for thresholds up to $50$, when the precision for user formulated queries outperforms the automatic queries. For thresholds up to $10$ the precision for method (6) is, on average, $1.6$ times higher. Nevertheless, the increase in precision for thresholds in the interval $[50, 200]$ is substantially higher for the user formulated

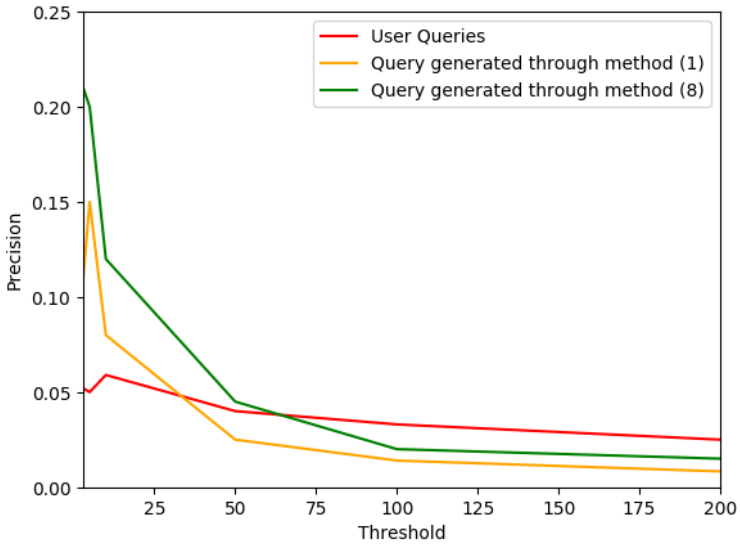


**Figure 4.7:** Precision at different thresholds for user generated queries and automatic queries formulated through methods (1) and (6) from Section 3.3.3.

queries. This shows that more relevant results are retrieved close to the end of the result set.

When applying the re-ranking approach introduced in Section 4.3.4 on the results retrieved through user formulated queries, the precision for thresholds up to $50$ outperforms Scopus. For thresholds up to $10$, the precision obtained through personalization is up to $4$ times higher. The results are illustrated in Figure 4.8.

Once again, the evaluation method misses relevant documents that were not cited by the article in question. In order to solve this problem we present the manual relevance judgements obtained from the user study.

Until now, the results were only judged using a boolean relevance scale. As suggested in Section 4.4.2, a boolean relevance scale can not discriminate between highly and marginally relevant documents. To this extend we proposed the use of a graded relevance scale that can enforce highly relevant documents in relation to their ranking position.

After completing the user study, the results were evaluated using the normalized discounted cumulative gain measurements (Equation 4.5) where the discounted cumulative gain and the ideal discounted cumulative gain were computed using Equation 4.4.

Before evaluation, the data set was filtered such that only the first query provided by a user was selected for an article. This reduced the dataset to $25$ queries. The results for the first $3$, $5$ and $10$ results are showcased for both Scopus and the personalization approach in Table 4.2. The table's header presents the threshold

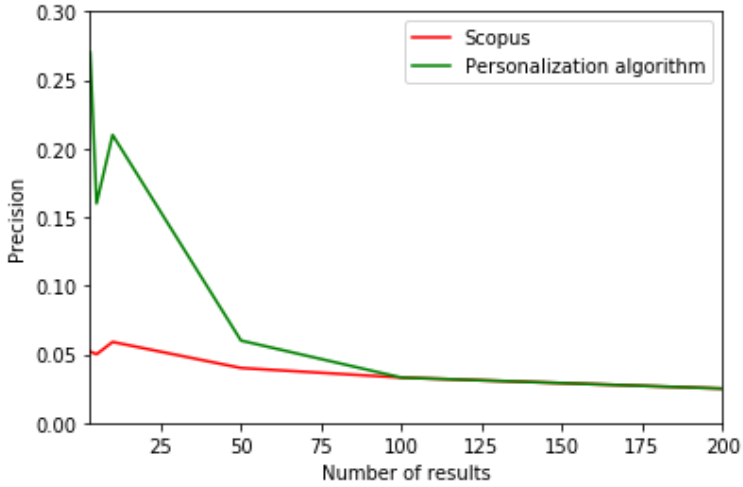


**Figure 4.8:** The impact of personalization on the user provided queries against the article references.

at which the nDCG measurement was taken. nDCG can take values in the interval $[0, 1]$ and results close to the upper limit are considered better.

| Retrieval Method | ndcg@3 | ndcg@5 | ndcg@10 |
|---|---|---|---|
| Scopus | 0.53 | 0.56 | 0.70 |
| Personalization | 0.72 | 0.71 | 0.91 |

**Table 4.2:** NDCG results for user study.

All articles present in the dataset that were also referenced for the paper in question were considered relevant. The results show a constant improvement through re-ranking, with at least $15\%$ for all thresholds. When using personalization only $1$ out of $10$ results was considered, on average, not relevant, compared to $\approx 3$ results for Scopus.

Moreover, out of all the results brought in the first $10$ through personalization, only $\approx 8\%$ were considered *slightly relevant* or *not relevant*. This percentage confirms that using a random walk similarity measure and a topic model can boost results close to the user's interest and working context.

Given how SimRank works, documents closer to the user's context will receive a higher similarity score, therefore, a higher rank. In such cases, if the results set contains documents authored by the participant, it is highly probable that they receive a higher rank both using the graph walk and the author topic model. This was, indeed, the case in the user study. $9$ out of the initial $37$ query set contained a result authored by the participant in question, out of which $5$ were used as related work (cited in the selected articles). From this set $4$ were judged as *relevant* and only one as $1$ as *slightly relevant*.

This shows that, in a majority of cases, users will consider relevant their own work, even though the content is already known. Thus the similarity measure can be used without any change. If any work authored by the user is retrieved in the first $200$ results from Scopus, it is very likely to be relevant for the query. The question of either a user will click to see the document or not is still open.

In order to validate the experiment results and check if personalization helped uniformly for the set of queries, or just for some, a statistical measure that quantifies the amount of variation or dispersion is needed. A simple approach would be to measure the *standard deviation* on the set of $nDCG$ results. However, the standard deviation is not considered robust because it uses the mean as a measure of centrality [86]. An alternative is to use the median, a measure of centrality that is not sensitive to the presence of outliers. To this extend we introduce the Median Absolute Deviation (MAD), a measure that involves finding the median of absolute deviations from the median, also described as:

$$MAD = bM_i(|x_i - M_j(x_j)|) \tag{4.7}$$

where $x_j$ is the $n$ original observation and $M_i$ is the median of the series. Usually, $b = 1.4826$, a constant linked to the assumption of normality of the data, disregarding the abnormality induced by outliers [86].

The measure can be interpreted as how far are the results from the median. After applying it on the $nDCG@10$ results for Scopus and the re-rank approach we obtained a score of $1.3$ for Scopus and $0.97$ through personalization. It means that the nDCG obtained through personalization varies slightly from the median, confirming that the algorithm performed uniformly for all users.

The same measurement was used to identify how personalization varies when it was used for the same topic. A number of $12$ queries from the initial set of $37$ were used to search for related work in the same topic (same selected articles). When applying the $MAD$ measurement on the $nDCG@10$ results, a score of only $0.89$ was obtained. The score shows that personalization affects in the same way the search process on different, or on the same topics.

Lastly, in order to validate the performance increase through personalization, a statistical significance test was used. Information retrieval researchers commonly use three tests for statistical significance: the Student's paired t-test, the Wilcoxon signed rank test, and the sign test [87].

A statistical significance test runs against a *null hypothesis*. When the significance level is low, the null hypothesis can be rejected. If the null hypothesis cannot be rejected, the differences between two systems may be the result of the inherent noise in evaluation [87]. When comparing two information retrieval systems, each system produces a score for a query and on a per-query basis a matched pairs of

scores is obtained. All the tests mentioned evaluate significance in light of the paired design, which is common to batch-style information retrieval experiments.

Smucker et al. [87] recommend the Fisher's randomization test under the following null hypothesis: two systems, A and B are identical and thus system $A$ has no effect compared to system $B$ on the average score for the given topics and relevance judgements. If system $A$ and system $B$ are identical, there is some system $N$ that produced the results for $A$ and $B$. Under the null hypothesis, system $N$ produced both results and we can merely label results as being produced by $A$ or $B$.

The score used in our case was the $nDCG$ for each query, computed on the first $10$ results of each system. After running the significance test, a $p$ value of $0.0967$ was obtained. In order to declare the test valid and reject the null hypothesis, the $p$ score returned by the significance test must be smaller than $0.1$ or $0.05$. While the test rejects the null hypothesis, the result is close to the upper limit. We believe this is due to the large set of common results, making it harder to distinguish which system produced the results.

### 4.4.4 Discussion

The user study followed two key points: (1) may be Scopus performance improved through personalization using topic models and graph walks? and (2) will users consider the same documents relevant after a period of time?

Results show that all participants considered the referenced documents relevant and prove consistency in the user's relevance judgements. Regarding the first point, the performance of Scopus was improved up to $20\%$ for the first $10$ results. Even though the result sets contain common documents, their rank changed in a proportion of $94\%$ through personalization. The user study is validated by running the Fisher's significance test on the $nDCG$ for the first $10$ results. The score obtained is very close to the upper limit.

The personalization algorithm introduced in Section 4.3.4 can either re-rank documents or add new ones from the result's sub-graph. Only a proportion of $4\%$ new results were added, out of which $91\%$ were considered relevant. The small percentage of new results is given by the re-ranking approach through $RRF$ (Section 4.3.4) where the Scopus initial rank is also considered. Since the documents outside of Scopus results received the smallest rank, they had to perform significantly better through graph similarity and for the topic model. Nevertheless, the result is interesting and deserves further investigation.

Another important point brought to light through the user study is the use of personalization when using different queries for the same information need. A first observation from the participants was that a single query can not encompass the

full content of a scientific article. Some users chose to provide several query for an information need. Measuring the median absolute deviation on these queries proved that personalization can be successfully used in one context. However, the number of queries is still small to draw a consistent conclusion.

The same procedure was applied for the main query set where subjects from $4$ different areas of Computer Science took part in the study. The $MAD$ results show little dispersion within $nDCG$ values and prove that personalization can be successfully used for different queries in the same topic. However, the subjects covered are not outside Computer Science. It is interestingly to observe the variance for queries that tackle non related fields such as Computer Science and Biology. Unfortunately, such an attempt was not possible for the current study.

The second research question (RQ2) from Section 1.3 asks *may be Scopus performance improved through personalization using topic models and graph walks?* The initial experiments from Section 4.3.4 show that using both a graph similarity measure or a topic model can help in personalization. However, as enforced by Xu et al. [79] and by the later experiments in Section 4.3.4, combining both methods leads to better results.

We propose to integrate the initial Scopus ranking together with a graph similarity measure and a topic model through reciprocal rank fusion. The results show that a performance increase up to $\approx 20\%$ in $nDCG$ on the first $10$ results.

Lastly, the third research question (RQ3) asks *will users consider the same documents relevant after a period of time?*. The user study shows that all participants considered the referenced documents relevant and prove consistency in the user's relevance judgements.

# Chapter 5

# Conclusions and future work

## 5.1 Conclusions

This study investigates two problems related to academic search formalised through the research questions in Section 1.3. At first, since test collections with manual relevance judgements are difficult to build and no TREC collection is available for our use case, various methods to construct test collections with automatic relevance judgements are analysed.

The first research question (RQ1) from Section 1.3 asks if *the information from scientific articles can be used to create test collections with automatic relevance judgements?* In search for an answer, we chose to model the following search task: researchers use Scopus to search for references for their new article using informational queries. After issuing a series of queries to Scopus, some of the results are referenced in the new article.

In order to simulate the working context a number of articles from Scopus was selected and a set of queries was generated from their content. The search was limited to retrieve documents indexed at most in the year before the article was published. After sending the queries to Scopus, only the results that were referenced in the selected article were considered relevant.

For this search task a test collection with automatic queries and relevance judgements was built. However, the queries could only retrieve $\approx 10\%$ of the references. If any result retrieved by Scopus is relevant to the query, but not cited in the article from which the query was inferred, its relevance can not be appreciated. We suggest to only use such a collection during algorithm training phases and ask for explicit relevance judgements from users or use a manually annotated collection for final validations and discussions.

In the second part the study concentrates on improving Scopus performance through personalization. We focus on the academic background of a user, in this case a researcher, as information about her past interests and publications is al-

ready available.

The second research question (RQ2) from Section 1.3 asks *may be Scopus performance improved through personalization using topic models and graph walks?* The initial experiments from Section 4.3.1 and 4.3.2 show that using both a graph similarity measure or a topic model can help in personalization. However, as enforced by Xu et al. [79] and by the later experiments in Section 4.3.3, combining both methods leads to better results. We propose to integrate the initial Scopus ranking together with a graph similarity measure and a topic model through reciprocal rank fusion. The results show a performance increase up to $\approx 20\%$ in $nDCG$ on the first $10$ results.

The third research question (RQ3) asks *will users consider the same documents relevant after a period of time?* The user study introduced in 4.4 shows that all participants considered the referenced documents relevant and proves consistency in the user's relevance judgements.

For this study personalization had a beneficial effect on the Scopus performance, however, such services might limit the ability of a user to find new information and might lead to the so called *filter bubble* effect in which a user is trapped in an information context. We recommend a careful implementation in production ready systems.

## 5.2 Threats to validity

Manual relevance judgements are difficult to gather for several reasons: at first, running a user study is expensive in terms of resources because participants need to invest a lot of time in the feedback process. Secondly, using pooling on a large test collection must ensure that the subset examined contains a representative sample of the relevant documents.

Since no test collection for personalization in academic search could be found, the validity of this study is proved by running a user study in which the participants selected the queries. Two things might threat the result's validity: (1) the number of participants and queries involved is relatively small and (2) all participants perform research in the field of Computer Science.

The first concern must be proved by including more subjects in the user study or using log data from Scopus, while the second point was tackled by including participants from different areas of Computer Science (e.g. information retrieval, cybersecurity and safety or service architectures) and measuring the result's dispersion. This proved that personalization can be applied in different areas of interest. However, a deeper analysis for non tangent fields is needed.

## 5.3 Future research

A first step in improving the conclusions is to validate the results on a bigger test collection. One approach is to use log data from Scopus to build a test collection and assess the results with click through data. Nevertheless, the study opens perspectives for future research. For example, the topic model used only takes into account information about one author.

More complex models, as suggested in Section 4.3.4, can take into account a distribution of authors interests or a topic model for journals. Using more complex topic models can have a significant impact on personalization. Moreover, the similarity measure was used on an uniform graph, without edge weights. As different graph entities (e.g. journals) have a sparse nature, adding edge weights to the graph can influence the output of personalization. A more complex attempt can automatically learn the graph weights.

Further on, when judging the relevance of personalized ranking systems, we have to consider that personalization can be ineffective for some queries, therefore, such techniques must be carefully used and various personalization strategies might have different effects on different queries. A learning approach can be applied to detect when personalization leads to relevant results and enforce personalization just in specific use cases.

# Bibliography


[1] X. Liu, J. Bollen, M. L. Nelson, and H. Van de Sompel, “Co-authorship networks in the digital library research community,” *Information processing & management*, vol. 41, no. 6, pp. 1462–1480, 2005.

[2] Z. Dou, R. Song, and J.-R. Wen, “A large-scale evaluation and analysis of personalized search strategies,” in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 581–590.

[3] B. J. Jansen, A. Spink, and T. Saracevic, “Real life, real users, and real needs: a study and analysis of user queries on the web,” *Information processing & management*, vol. 36, no. 2, pp. 207–227, 2000.

[4] X. Shen, B. Tan, and C. Zhai, “Implicit user modeling for personalized search,” in *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 2005, pp. 824–831.

[5] P. Pirolli and S. Card, “Information foraging in information access environments,” in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co., 1995, pp. 51–58.

[6] J. Teevan, S. T. Dumais, and E. Horvitz, “Personalizing search via automated analysis of interests and activities,” in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2005, pp. 449–456.

[7] W. R, *Design Science Methodology for Information Systems and Software Engineering*. Springer, 2014.

[8] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1, no. 1.

[9] R. Baeza-Yates, B. Ribeiro-Neto *et al.*, *Modern information retrieval*. ACM press New York, 1999, vol. 463.

[10] L. Page, S. Brin, R. Motwani, and T. Winograd, “The pagerank citation ranking: Bringing order to the web.” Stanford InfoLab, Technical Report 1999-66, November 1999, previous number = SIDL-WP-1999-0120. [Online]. Available: http://ilpubs.stanford.edu:8090/422/

[11] T. H. Haveliwala, “Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search,” *IEEE transactions on knowledge and data engineering*, vol. 15, no. 4, pp. 784–796, 2003.

[12] S. Brin and L. Page, “Reprint of: The anatomy of a large-scale hypertextual web search engine,” *Computer networks*, vol. 56, no. 18, pp. 3825–3833, 2012.

[13] G. Jeh and J. Widom, “Scaling personalized web search,” in *Proceedings of the 12th international conference on World Wide Web*. ACM, 2003, pp. 271–279.

[14] J. M. Kleinberg, “Authoritative sources in a hyperlinked environment,” *Journal of the ACM (JACM)*, vol. 46, no. 5, pp. 604–632, 1999.

[15] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press Cambridge, 2001, vol. 6.

[16] C. Faloutsos, K. S. McCurley, and A. Tomkins, “Fast discovery of connection subgraphs,” in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 118–127.

[17] L. A. Adamic and E. Adar, “Friends and neighbors on the web,” *Social networks*, vol. 25, no. 3, pp. 211–230, 2003.

[18] G. Jeh and J. Widom, “Simrank: a measure of structural-context similarity,” in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2002, pp. 538–543.

[19] D. Lizorkin, P. Velikhov, M. Grinev, and D. Turdakov, “Accuracy estimate and optimization techniques for simrank computation,” *Proceedings of the VLDB Endowment*, vol. 1, no. 1, pp. 422–433, 2008.

[20] A.-L. Barabási, “Linked: The new science of networks,” 2003.

[21] J. Scott, *Social network analysis*. Sage, 2012.

[22] E. Otte and R. Rousseau, “Social network analysis: a powerful strategy, also for the information sciences,” *Journal of information Science*, vol. 28, no. 6, pp. 441–453, 2002.

[23] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*. Cambridge university press, 1994, vol. 8.

[24] A. Pritchard, "Statistical bibliography or bibliometrics," *Journal of documentation*, vol. 25, p. 348, 1969.

[25] A. Sandison, "Documentation note: thinking about citation analysis," *Journal of Documentation*, vol. 45, no. 1, pp. 59–64, 1989.

[26] "Elsevier docs," https://dev.elsevier.com/tecdoc_ir_cris_vivo.html, [Online; accessed 16-04-2017].

[27] J. M. Ziman, *Public knowledge: An essay concerning the social dimension of science*. CUP Archive, 1968, vol. 519.

[28] F. Narin *et al.*, *Evaluative bibliometrics: The use of publication and citation analysis in the evaluation of scientific activity*. Computer Horizons Washington, D. C, 1976.

[29] V. P. Diodato and P. Gellatly, *Dictionary of bibliometrics*. Routledge, 2013.

[30] M. M. Kessler, "Bibliographic coupling between scientific papers," *Journal of the Association for Information Science and Technology*, vol. 14, no. 1, pp. 10–25, 1963.

[31] L. Egghe and R. Rousseau, *Introduction to informetrics: Quantitative methods in library, documentation and information science*. Elsevier Science Publishers, 1990.

[32] H. Small, "Co-citation in the scientific literature: A new measure of the relationship between two documents," *Journal of the Association for Information Science and Technology*, vol. 24, no. 4, pp. 265–269, 1973.

[33] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 50–57.

[34] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[35] J. L. Boyd-Graber, D. M. Blei, and X. Zhu, "A topic model for word sense disambiguation." in *EMNLP-CoNLL*, 2007, pp. 1024–1033.

[36] A. Haghighi and L. Vanderwende, “Exploring content models for multi-document summarization,” in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2009, pp. 362–370.

[37] X. Wei and W. B. 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*. ACM, 2006, pp. 178–185.

[38] Y. Feng and M. Lapata, “How many words is a picture worth? automatic caption generation for news images,” in *Proceedings of the 48th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 1239–1249.

[39] T. L. Griffiths and M. Steyvers, “Finding scientific topics,” *Proceedings of the National academy of Sciences*, vol. 101, no. suppl 1, pp. 5228–5235, 2004.

[40] T. Saracevic, “Evaluation of evaluation in information retrieval,” in *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1995, pp. 138–146.

[41] I. Soboroff, C. Nicholas, and P. Cahan, “Ranking retrieval systems without relevance judgments,” in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2001, pp. 66–73.

[42] G. V. Cormack, C. R. Palmer, and C. L. Clarke, “Efficient construction of large test collections,” in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1998, pp. 282–289.

[43] E. M. Voorhees, “Overview of trec 2003.” in *TREC*, 2003, pp. 1–13.

[44] J. Zobel, “How reliable are the results of large-scale information retrieval experiments?” in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1998, pp. 307–314.

[45] T. Saracevic, “Relevance: A review of and a framework for the thinking on the notion in information science,” *Journal of the American Society for information science*, vol. 26, no. 6, pp. 321–343, 1975.

[46] S. P. Harter, “Variations in relevance assessments and the measurement of retrieval effectiveness,” *JASIS*, vol. 47, no. 1, pp. 37–49, 1996.

[47] E. M. Voorhees, “Variations in relevance judgments and the measurement of retrieval effectiveness,” *Information processing & management*, vol. 36, no. 5, pp. 697–716, 2000.

[48] A. Chowdhury and I. Soboroff, “Automatic evaluation of world wide web search services,” in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2002, pp. 421–422.

[49] A. Harpale, Y. Yang, S. Gopal, D. He, and Z. Yue, “Citedata: a new multi-faceted dataset for evaluating personalized search performance,” in *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, 2010, pp. 549–558.

[50] B. M. Hemminger, D. Lu, K. Vaughan, and S. J. Adams, “Information seeking behavior of academic scientists,” *Journal of the American society for information science and technology*, vol. 58, no. 14, pp. 2205–2225, 2007.

[51] X. Niu, B. M. Hemminger, C. Lown, S. Adams, C. Brown, A. Level, M. McLure, A. Powers, M. R. Tennant, and T. Cataldo, “National study of information seeking behavior of academic researchers in the united states,” *Journal of the American Society for Information Science and Technology*, vol. 61, no. 5, pp. 869–890, 2010.

[52] D. Ellis, “Modeling the information-seeking patterns of academic researchers: A grounded theory approach,” *The Library Quarterly*, vol. 63, no. 4, pp. 469–486, 1993.

[53] X. Li, B. J. Schijvenaars, and M. de Rijke, “Investigating queries and search failures in academic search,” *Information Processing & Management*, vol. 53, no. 3, pp. 666–683, 2017.

[54] P. Boldi, F. Bonchi, C. Castillo, and S. Vigna, “Query reformulation mining: models, patterns, and applications,” *Information retrieval*, vol. 14, no. 3, pp. 257–289, 2011.

[55] P. D. Bruza and S. Dennis, “Query reformulation on the internet: Empirical data and the hyperindex search engine,” in *Computer-Assisted Information Searching on Internet*. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D’INFORMATIQUE DOCUMENTAIRE, 1997, pp. 488–499.

[56] B. J. Jansen, D. L. Booth, and A. Spink, “Patterns of query reformulation during web searching,” *Journal of the Association for Information Science and Technology*, vol. 60, no. 7, pp. 1358–1371, 2009.

[57] S. Jones, S. J. Cunningham, R. McNab, and S. Boddie, “A transaction log analysis of a digital library,” *International Journal on Digital Libraries*, vol. 3, no. 2, pp. 152–169, 2000.

[58] X. Li and M. de Rijke, “Do topic shift and query reformulation patterns correlate in academic search?”

[59] J. Steinberger and K. Jezek, “Using latent semantic analysis in text summarization and summary evaluation,” in *Proc. ISIM 04*, 2004, pp. 93–100.

[60] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, “Arnetminer: extraction and mining of academic social networks,” in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 990–998.

[61] J. Tang, J. Zhang, R. Jin, Z. Yang, K. Cai, L. Zhang, and Z. Su, “Topic level expertise search over heterogeneous networks,” *Machine Learning*, vol. 82, no. 2, pp. 211–237, 2011.

[62] P. Anick, “Using terminological feedback for web search refinement: a log-based study,” in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 2003, pp. 88–95.

[63] M. Speretta and S. Gauch, “Personalized search based on user search histories,” in *Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on*. IEEE, 2005, pp. 622–628.

[64] F. Liu, C. Yu, and W. Meng, “Personalized web search for improving retrieval effectiveness,” *IEEE Transactions on knowledge and data engineering*, vol. 16, no. 1, pp. 28–40, 2004.

[65] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen, “Cubesvd: a novel approach to personalized web search,” in *Proceedings of the 14th international conference on World Wide Web*. ACM, 2005, pp. 382–390.

[66] B. Smyth, E. Balfe, J. Freyne, P. Briggs, M. Coyle, and O. Boydell, “Exploiting query repetition and regularity in an adaptive community-based web search engine,” *User Modeling and User-Adapted Interaction*, vol. 14, no. 5, pp. 383–423, 2004.

[67] A. Kritikopoulos and M. Sideri, "The compass filter: Search engine result personalization using web communities," *Intelligent Techniques for Web Personalization*, pp. 229–240, 2005.

[68] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma, "Learning to cluster web search results," in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2004, pp. 210–217.

[69] P. Ferragina and A. Gulli, "A personalized search engine based on web-snippet hierarchical clustering," *Software: Practice and Experience*, vol. 38, no. 2, pp. 189–225, 2008.

[70] P.-A. Chirita, D. Olmedilla, and W. Nejdl, "Pros: A personalized ranking platform for web search," in *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*. Springer, 2004, pp. 34–43.

[71] J. Beel and B. Gipp, "Google scholar's ranking algorithm: an introductory overview," in *Proceedings of the 12th International Conference on Scientometrics and Informetrics (ISSI'09)*, vol. 1. Rio de Janeiro (Brazil), 2009, pp. 230–241.

[72] J. E. Hirsch, "An index to quantify an individual's scientific research output," *Proceedings of the National academy of Sciences of the United States of America*, pp. 16 569–16 572, 2005.

[73] E. Garfield, "The history and meaning of the journal impact factor," *Jama*, vol. 295, no. 1, pp. 90–93, 2006.

[74] D. Walker, H. Xie, K.-K. Yan, and S. Maslov, "Ranking scientific publications using a model of network traffic," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2007, no. 06, p. P06010, 2007.

[75] L. B. Jabeur, L. Tamine, and M. Boughanem, "A social model for literature access: towards a weighted social network of authors," in *Adaptivity, Personalization and Fusion of Heterogeneous Information*. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, 2010, pp. 32–39.

[76] D. Zhou, S. A. Orshanskiy, H. Zha, and C. L. Giles, "Co-ranking authors and documents in a heterogeneous network," in *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*. IEEE, 2007, pp. 739–744.

[77] J. Tang, R. Jin, and J. Zhang, "A topic modeling approach and its integration into the random walk framework for academic search," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 2008, pp. 1055–1060.

[78] E. Neuhold, C. Niederée, and A. Stewart, "Personalization in digital libraries–an extended view," in *International Conference on Asian Digital Libraries*. Springer, 2003, pp. 1–16.

[79] Y. Xu, X. Guo, J. Hao, J. Ma, R. Y. Lau, and W. Xu, "Combining social network and semantic concept analysis for personalized academic researcher recommendation," *Decision Support Systems*, vol. 54, no. 1, pp. 564–573, 2012.

[80] Y. Sun, H. Li, I. G. Councill, J. Huang, W.-C. Lee, and C. L. Giles, "Personalized ranking for digital libraries based on log analysis," in *Proceedings of the 10th ACM workshop on Web information and data management*. ACM, 2008, pp. 133–140.

[81] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth, "The author-topic model for authors and documents," in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*. AUAI Press, 2004, pp. 487–494.

[82] "Elsevier docs," http://api.elsevier.com/documentation/search/AUTHORSearchTips.html, [Online; accessed 16-04-2017].

[83] G. V. Cormack, C. L. Clarke, and S. Buettcher, "Reciprocal rank fusion outperforms condorcet and individual rank learning methods," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2009, pp. 758–759.

[84] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 422–446, 2002.

[85] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005, pp. 89–96.

[86] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, vol. 49, no. 4, pp. 764–766, 2013.

[87] M. D. Smucker, J. Allan, and B. Carterette, “A comparison of statistical significance tests for information retrieval evaluation,” in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, 2007, pp. 623–632.

# Appendix A

# Scopus Data

## A.1 Query examples

| Type of query | Examples |
|---|---|
| (1) | semantic gateway service architecture iot interoperability |
| (2) | ”internet of things”OR”mqtt”<br>OR”semantic gateway service”<br>OR”semantic sensor network” |
| (3) | ”internet of things”AND”mqtt”,<br>”internet of things”AND”semantic gateway service”,<br>”internet of things”AND”semantic sensor network” |
| (4) | ’”internet of things”AND”mqtt”’ OR<br>’”internet of things”AND”semantic gateway service”’ OR<br>’”internet of things”AND”semantic sensor network”’ |
| (5) | ”interoperability”OR”gateways”OR<br>”mobile telecommunication systems” ... |
| (6) | ”interoperability” AND ”gateways”,<br>”interoperability” AND ”mobile telecommunication systems”, ... |
| (7) | ’”interoperability” AND ”gateways”’ OR<br>’”interoperability” AND ”mobile telecommunication systems”’ |
| (8) | iot intelligent architecture semantic gateway interoperability |

**Table A.1:** Examples of queries.

## A.2 Scopus Screenshots

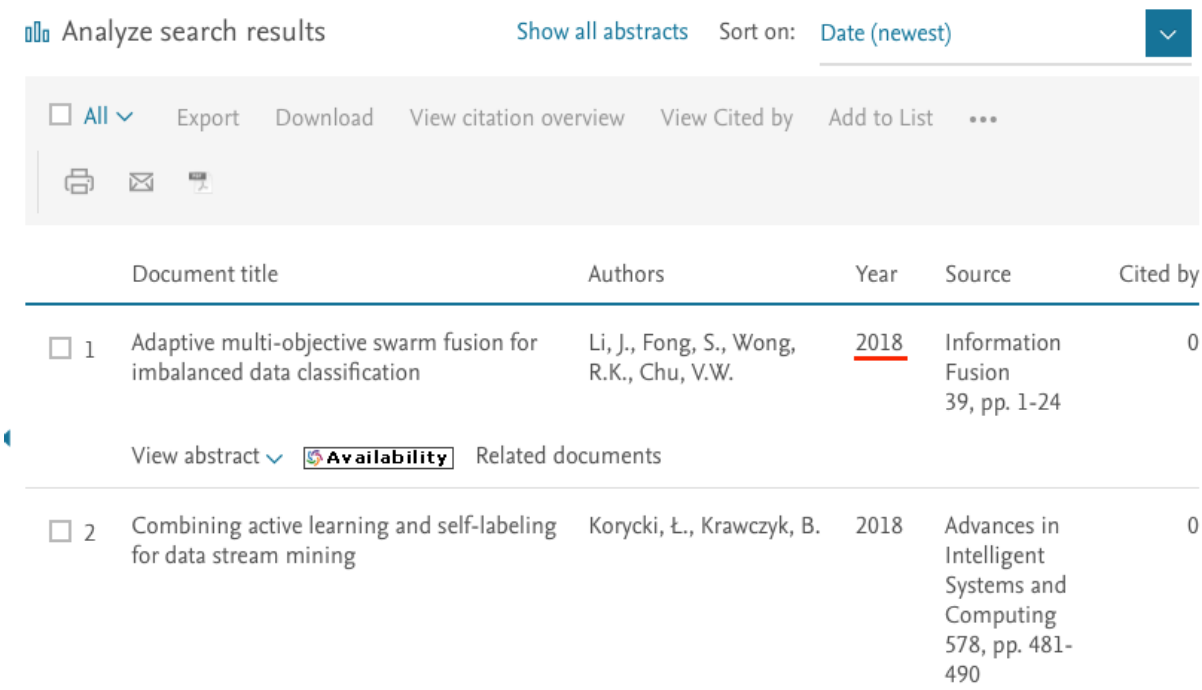


**Figure A.1:** Scopus screenshot showing articles indexed earlier than their publishing year.

## A.3 Scopus Subject areas

### A.3.1 Author subject areas

- Arts and Humanities
- Business, Management and Accounting
- Computer Science
- Engineering
- Energy
- Mathematics

### A.3.2 Article subject areas

- Applied Mathematics
- Computer Science Applications
- Electrical and Electronic Engineering

- Information Systems
- Management Science and Operations Research
- Media technology