République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A.MIRA-BEJAIA

Faculté des Sciences Exactes
Département d'Informatique

THÈSE

Présentée par

Messaoud CHAA

Pour l'obtention du grade de

DOCTEUR EN SCIENCES

Filière : Informatique

Option : Réseaux et Systèmes Distribués

Thème

Social Information Retrieval : A Hybrid Approach Based on Analysis and Information Extraction

Soutenue le : 23/10/2021 Devant le Jury composé de :

Nom et Prénom	Grade		
Mr Abdelkamel TARI	Professeur	ESTIN, Bejaia, Algérie	Président
Mr Omar NOUALI	Dir. de Recherche	CERIST, Alger, Algérie	Rapporteur
Mr Patrice BELLOT	Professeur	Univ. Aix-Marseille, France	Co-Rapporteur
Mme Houda EL BOUHISSI	MCA	Univ. de Bejaia, Algérie	Examinatrice
Mr Abderrazak SEBAA	MCA	Univ. de Bejaia, Algérie	Examinateur

Année Universitaire : 2020/2021

Acknowledgement

Above all, I would like to express my heartfelt thanks to Almighty God, who gave me great will and sufficient knowledge to complete this research work successfully.

I extend my sincere gratitude and appreciation to many people who made this PhD thesis possible.

First, I would like to thank my supervisors, Prof Omar Nouali and Prof Patrice Bellot for their excellent supervision, sympathy, orientations, and advice during the years of my PhD study.

More thanks go to my thesis evaluation panel consisting of Prof Abdelkamel Tari for agreeing to chair the evaluation jury, Dr Abderrazak Sebaa and Dr Houda EL Bouhissi, who accepted to be part of the examination board.

This research is dedicated to the soul of my late mother in heaven, whose prayers and wishes made this research possible. I would also thank my father, brothers, and sisters for their support.

My thanks are also extended to my faithful wife for her unconditional support throughout this journey. Thank you to Mehdi, Imene, Asma and Amani, my precious children, for bringing immeasurable happiness to my life.

My special thanks also go to all my friends and all my colleagues of the research center on scientific and technical information CERIST.

Lastly, I would never forget to express all my gratitude to all the members of computer science department of University of Bejaia.

The author publications

Journal paper

1. Messaoud Chaa, Omar Nouali, and Patrice Bellot. "Leveraging app features to improve mobile app retrieval" International Journal of Intelligent Information and Database Systems 14, no. 2 (2021): 177-197.

Conference papers

2. Messaoud Chaa, Omar Nouali, and Patrice Bellot. "New technique to deal with verbose queries in social book search." In Proceedings of the International Conference on Web Intelligence, pp. 799-806. 2017.

3. Messaoud Chaa, Omar Nouali, and Patrice Bellot. "Combining tags and reviews to improve social book search performance." In International Conference of the Cross-Language Evaluation Forum for European Languages, pp. 64-75. Springer, Cham, 2018.

Workshop papers

4. Messaoud Chaa, Omar Nouali, and Patrice Bellot. "Verbose Query Reduction by Learning to Rank for Social Book Search Track." In CLEF (Working Notes), pp. 1072-1078. 2016.

5. Messaoud Chaa, Omar Nouali. "CERIST at INEX 2015: Social Book Search Track." In CLEF (Working Notes). 2015.

Table of Contents

List of Figures

List of Tables

Acronyms

app Mobile Application.

BOW Bag of Words.

IR Information Retrieval.

IRS Information Retrieval System.

L2R Learning to Rank.

LDA Latent Dirichlet Allocation.

LR Log-Likelihood ratio.

LT Library Thing.

MAP Mean Average Precision.

MRR Mean Reciprocal Rank.

NDCG Normalized Discounted Cumulative Gain.

Q/A Question Answering.

RBM Review Based Model.

SBS Social Book Search.

SIR Social Information Retrieval.

SIS Social Information Seeking.

TBM Tag Based Model.

TF Term Frequency.

TREC Text REtrieval Conference.

UGC User Generated Content.

General Introduction

Nowadays, information retrieval (IR) has become common in our daily life, after being restricted only to professional researchers for many years. The widespread internet use and the wide availability and ease of use of mobile devices have directly contributed to the wide use of information retrieval. In this digital age, most of us connect to internet using our smart-phones and tablets at any time and from anywhere to seek information for various needs (ex: looking for a document about a particular topic, Images, videos, solution of a problem,...).

The field of information retrieval is a research area whose primary goal is retrieving from a collection a subset of information that satisfy a user's information need. Its history can be traced back to the early 60s when the IR was restricted to laboratories with deployed information retrieval systems (IRS) applied to information stored in structured repositories of textual documents. These systems usually used manual indexing and had only minimal capabilities. Users wishing to find documents about some topics keying a few terms to a search system that returns the documents that contain the searched terms based on the presence of those terms in the documents.

The emergence of internet in the 1990s, has led to the appearance of search engine like Google, Bing and Yahoo, which used the web as a source of information. All the main search engines use a web crawler to collect and index information and offer a user interface to help the users to search through the indexed information. Several ranking models are used in order to rank the search results based on their relevance to the user input. PageRank [104] uses links between web pages to meseare the importance of pages and determine their ordering in search results. BM25, tf-idf, language models [109, 112, 117] are based on two basic variables, namely, term frequency usually normalized by document length and inverse document frequency. The web pages and the digitized textual documents were the primary target of retrieval, but over time the scope was broadened to include other kinds

of stored material that are not text, such as images, videos, sounds or their combination.

With the appearance of the social web, many social media have emerged and have changed the habits of Internet users. Through these social media, users become able, not to only consume, but also to produce information and contribute to the content of documents. Two kinds of social information are generated by social users: textual information like tags, reviews or non textual information like rating, social interactions (share, like, etc.) or users' relation, which represents a valuable source of relevant information. To exploit the experience and opinions of others users, social information has been used in several application domains such as marketing [11], commerce [20], etc. The field of information retrieval is no exception to this. A few years ago, a new area of research, namely social information retrieval (SIR) has emerged and gained popularity. SIR systems are distinguished from traditional IR systems by the incorporation of social information in the search process in order to improve the IR performance.

In the same context, several models have been proposed to exploit social information to enhance information retrieval results. The implemented state-of-the-art models can be categorized into two groups according on how they use social information for information retrieval. The first one consists in including the information produced by social users in the IR process as a complementary source of information to enrich documents and refine the results, whereas the second category corresponds to models that use social information for query reformulation to expand the original query with additional information that reflecting the user preferences and provide enhanced personalized search results.

The web 2.0 is more and more spread in a diversity of fields, for instance education, research, marketing and leisure. Users can tag, comment and rate hotels they have stayed in (e.g. www.tripadvisor.fr), share and annotate books they have read (e.g. www.librarything.com or www.goodreads.com), rate and write reviews about application they have installed and used or give their opinions to any product they have used before. Hence, social media allow users to interact with any entity in the web we can imagine. By entity, we mean any identifiable object or thing that users can talk about(e.g. person, car, place, hotel, program, etc.), characterized by its types, attributes, and relationships to other entities. Users share their experiences and give their opinions on person, hotels, cars, apps, etc. In addition, Web 2.0 allows users, through social forums, to ask questions to other forum users for looking for information. Furthermore, users discuss their complex

information needs and request recommendations and suggestions using natural language which is considered as the most user-friendly and expressive way. Thus, the information research behaviour has changed both for the sought information (from documents to any entity) and for the way the users express their needs (from keyword queries to natural language queries). This created new challenges for information retrieval systems, which must be overcome. Consequently, the main objective of this thesis is to answer the following research questions:

- How can we deal with natural language queries "Verbose queries" of forums' users in order to optimize their search intent and help them to get what they want to retrieve?
- How to adopt and adapt the different classical models to such different types of social information such as tags and reviews and which model is the most appropriate for each type?
- How do we represent the content of entities based on social information (tags, reviews, etc.)? Using a simplified representation like Bag-Of-Words (BoW) or using a feature representation.

Our goal is to come up with solutions to support social media users in a complex search to find their needs. In this thesis, we propose three main contributions to overcome the aforementioned drawbacks related to social information retrieval. The proposed solutions are built on two existing dataset, Social Book Search (SBS) [75, 76, 78] and App retrieval [106]. In the first one, the sought entity is a book represented by Amazon user reviews, extended with social metadata from LibraryThing. Requests and suggestions are extracted from the LibraryThing (LT) discussion forums as a way to model book search queries. The second one, use application (app) as a sought entity. Each app is represented by a description given by the app's developer and a set of reviews given by online users in Google Play Store [1]. As for queries, they are collected from android forum[2].

In this dissertation, we start with the current part (general introduction) which introduces the context of our study, the challenges, as well as the thesis outline. The five following chapters will present the details of our work.

[1] http://play.google.com
[2] http://forums.androidcentral.com/

Chapter 1 will be dedicated to present the basic concepts of information retrieval. Therein, we explain in detail, the main components of the information retrieval system, document representation and indexing, the different developed models and the various measures used for evaluation.

Chapter 2 will start with the description of the working principles of social media tools and their use in different domains as well as the different types of generated social information. The chapter continues with the review and analysis of some of the most important works in the field of social information retrieval, which are categorised based on the type of, and how they use social information in the retrieval process. We conclude the chapter by summarizing the datasets used for evaluation and their sources.

In Chapter 3, we present our first contribution termed "Approaches to deal with natural language queries in social information retrieval". To address the issues of natural language queries submitted by users in social forums, we propose two techniques in order to reduce the queries and keep only the important terms that match well with user's needs.

Chapter 4 is dedicated to the presentation of our second contribution termed "Combining tags and reviews to improve social search performance". Since documents can be represented by different social information such as tags and reviews, we propose to analyse the sensitivity of the retrieval model with respect to social information used to represent documents as well as to query representation.

In Chapter 5, we present our third contribution termed "Leveraging features extracted from social information to improve the retrieval performance". In this chapter, we propose a new feature-based representation for both sought entity (a mobile app in our case) and query. Social user reviews and the description of app's developer are used to extract features that characterize the app. We then extract the request features from the user query. Finally, we match between the two representations in order to return the most relevant apps for a given user query.

This thesis ends with a general conclusion, which briefly summarizes our different contributions, as well as possible future research directions.

Part I

Information retrieval

Chapter 1

Information retrieval

1.1 Introduction

Information retrieval is historically linked to information science whose activity is to retrieve and deliver information to users based on their information needs. This activity is carried out with information retrieval systems (IRS). These systems use models and techniques to match the user needs (query, usually described by a set of keywords) with the set of documents (collection) through a comparison function (scoring function) and return a ranked list in which the most likely documents to be relevant to the query are ranked higher than less relevant ones. In this chapter we present the general structure of an information retrieval system and its different components which are : documents preprocessing and indexing module, query representation module and maching/ranking module. We will detail the different matching and ranking models that have been proposed in the information retrieval literature such as exact match models, vector space models and probabilistic models. in the last section we will describe the fundamental evaluation measures used to evaluate the models' performance.

1.2 Definition of information retrieval

Information retrieval (IR) is one of the earliest areas of research in computer science. Its computer-based search system can be traced to the end of the 1940s [141]. The idea of information retrieval appeared in 1945 with Vannevar Bush in his paper [24] in which he introduced the idea of large document repositories and automated search tools [33]. At first the information retrieval was considered as a subfield of information science,

but with the advent computer science, it has became close to numerous areas including database systems, human computer interaction, language technology, image recognition or a subfield of many other areas like compression and algorithmics and artificial intelligence [141]. One of the first definitions of IR is given by Salton in 1968 :

Définition 1.1 *"Information retrieval is a field concerned with the structure, analysis, organization, storage, searching, and retrieval of information." [116]*

In 1980 another definition given by *Rijsbergen* :

Définition 1.2 *"Information retrieval systems address the representation, organization of, and access to large amounts of heterogeneous information encoded in digital format." [124]*

Other definitions have been proposed in the 2000s, which are almost similar but they differ from the above-mentioned in that they introduce the concept of *"information need"* and this because user's information need became the central purpose of any information retrieval system. Among these definition :

Définition 1.3 *"Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers)" [36]*

Définition 1.4 *"It is the means by which users of an information system or service can find the documents, records, graphic images, or sound recordings that meet their needs or interests." [22]*

1.3 Information retrieval systems

An information retrieval system (IRS) consists of a software program that use a set of methods and techniques to store and manage information in order to help and assist users in finding the information they need. As an example, the web search engines like Google, Bing and Yahoo are the most popular IR systems that search information in the web. The IRS stores a set of documents (corpus) in an electronic format that is easier to be accessed when it is needed. Users specify their needs in the form of queries and the IR systems attempts, following a set of processes, to retrieve the documents that contain information sought by the user in his query. The list of documents that best match the query will

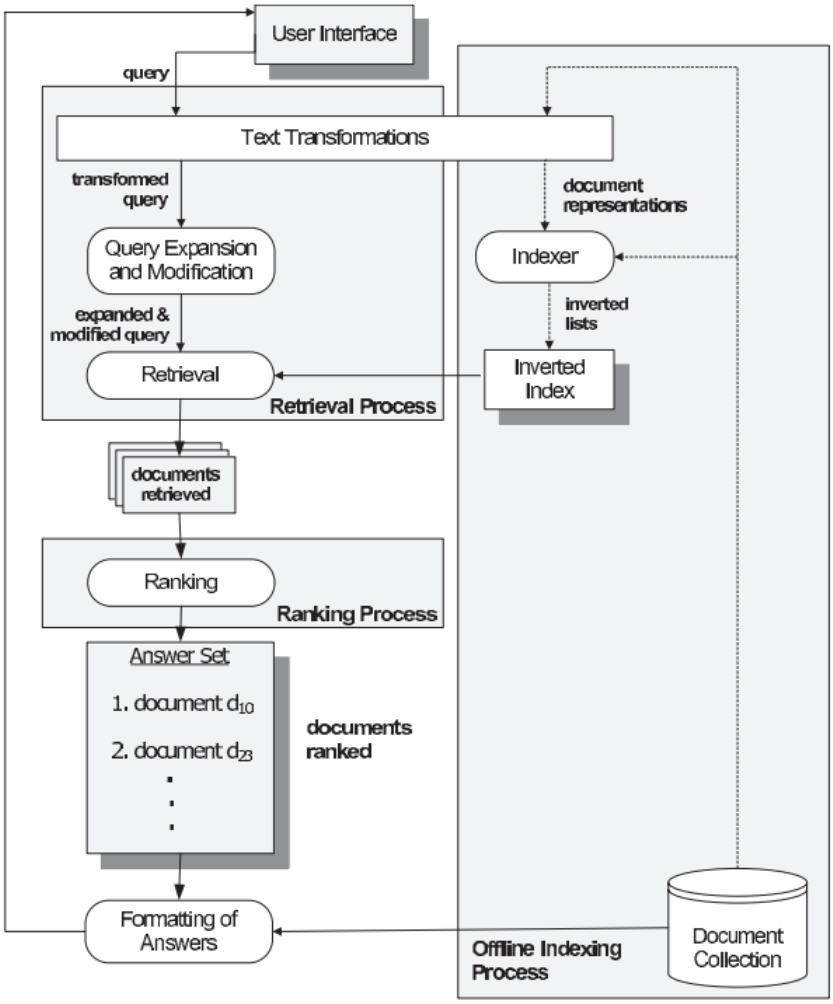

Figure 1.1: Information retrieval processes [4]

be returned by the system and they are called relevant documents. The Performance of the IR system is evaluated according to the user's satisfaction with respect to returned documents. A perfect retrieval system would rank the most relevant documents on the top of the list. Any information retrieval system is made up of two components: the indexing system and the retrieval and ranking system. The first of these is an offline process. It is in charge of analyzing the documents of the collection and creating the index that then allow search queries to be made; while the second is an online process and is in charge of analyzing the query and retrieving and ranking the documents that match the query and it is the part with which users interact. A typical architecture of the information retrieval system given by Baeza et al. [4] is illustrated in Figure 1.1

1.3.1 Indexing system

Indexing is an important process in Information Retrieval systems, it is an offline process that consists of representing and organizing the corpus of documents using indexing structure. The main role of this process is identifying the important terms that represent the documents and store them into the index. Therefore, for each document of the corpus, several text operations will be applied as follows :
- *Tokenization* : Split the sentences of the document into individual words based on a certain delimiter.
- *Stop word removal* : Eliminate words that does not add much meaning to the sentences such as articles and prepositions (e.g. the, which, it, at, in, and, etc.). By default, a standard list of stopwords is used but in some cases a domain specific list is built.
- *Stemming* : Truncate the words of the document into the root word that will reduce vocabulary. For instance, the three words *proposes*, *proposing* and *proposed* will be stemmed to *propos*.
- *Term weighting* : Give each word a weight that reflects its importance in the document, usually based on term frequency (tf) and inverse document frequency (idf).

The resulted set of terms extracted from each document, which are significantly reduced compared to the number of terms contained in the document itself, is then used to compose the index. The inverted index is the most popular one among many different index structures. After completing the indexing process, the retrieval process can be start.

1.3.2 Retrieval and ranking system

The second component of the information retrieval system is the retrieval and ranking process with which users interact through an interface. First, the user specifies and sent to a system a set of keywords (query) that reflects their information need. This query is then processed by text operations (tokenization, stopword removal and stemming) that are almost the same as applied to the documents in the indexing process. The query can be expanded and reformulated using some techniques such as frequent terms in the pseudo-feedback, synonyms of query terms or users interactions with documents in user logs. Next the system use the expanded query and the index to retrieve the set of documents that contain the query terms. Once the documents are found, they will be ranked according to a likelihood of relevance to the query, from the more relevant to the less relevant one. The top ranked documents are then formatted for presentation to the user. This step is the essential step of the retrieval system who should rank the more relevant documents at

the top of the output list because users will pay more attention to the top results. Various information retrieval models are proposed to estimate the relevance of a document to a user query. Among these models, Boolean model, vector space model, and probabilistic model. In the following, we give a detail of the main state of the art models.

1.4 Information retrieval models

Various information retrieval models are proposed to estimate the relevance of a document to a user query. Among these models, Boolean model, vector space model and probabilistic model. Boolean model does not take into account the weights of terms in documents unlike vector space model and probabilistic model. [56].

1.4.1 Basic concepts

According to [4] an information retrieval model is a quadruple [$\boldsymbol{D}$, $\boldsymbol{Q}$, $\boldsymbol{F}$, $\boldsymbol{R}(q_i, d_j)$] where :

- $\boldsymbol{D}$ is a set of representations of documents in a collection that created in the indexing step.
- $\boldsymbol{Q}$ is a set of representations of queries.
- $\boldsymbol{F}$ is a framework for modeling representations of document and query and their relationship.
- $\boldsymbol{R(q_i,d_j)}$ is a ranking function that give a score (real number) to a document representation d_j with respect to a query representation q_i.

Therefore, to build the model, the two representations of any document(d_j) from D and any query (q_j) from Q are given and then apply a ranking function R that associate a relevance score of the document (d_j) with regard to a query (q_j). The representation of documents and queries is differ from model to model.

1.4.2 Boolean model

The Boolean model is the first model that have been proposed in information retrieval. In these model a document is represented as a set of keywords, while queries are boolean expressions of keywords, connected by the three basic operators AND, OR, and NOT.

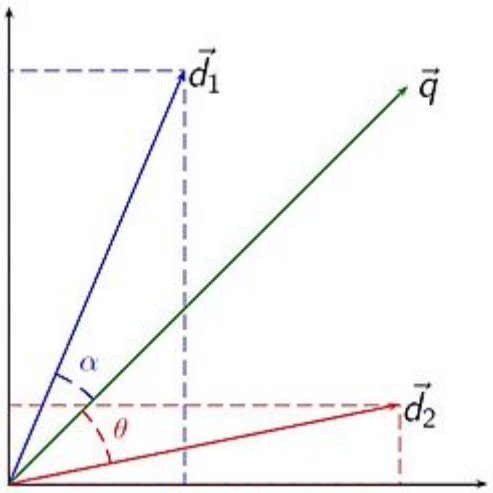

Figure 1.2: Vector Space Model (Wikipedia)

The model considers which terms are present or absent in a document . For instance, the query *"Social **AND** Information"* will produce the set of documents that are indexed both with the term *social* and the term *information*, i.e. the intersection of the set of documents that contains the term *social* and the set of documents that contains the term *information*. An advantage of this model is that it gives users a sense of control over the system. They can change or modify the operators depending on the results returned by the system. However, the main disadvantage of this model is that it does not provide a ranking list of relevant documents because it uses the notion of *"exact matching"* and there is no concept of a *"partial match"* between documents and queries. The model either retrieves a document or not, and consider all the retrieved documents as relevant.

1.4.3 Vector space model

The idea of vector space model proposed by [117] is based on Luhn's similarity criterion that has a stronger theoretical motivation [91]. Thus, the documents and queries are represented as vectors of weights embedded in a high dimensional Euclidean space, Figure 1.2. A document d and a query q vectors are then :

$$\vec{d} = \{d_1, d_2, ..., d_T\} \tag{1.1}$$

$$\vec{q} = \{q_1, q_2, ..., q_T\} \tag{1.2}$$

where d_i and q_i are the weights of index term i in the document d and query q respectively. T is the number of unique terms in the collection (vocabulary dimension). Each weight represents the importance of the index term in the document and the query which can be calculated by several formulas such as binary, tf or tf-idf weights. For the binary weight the weights of terms are calculated as follows :

$$di = \begin{cases} 1 & if \quad index \quad term \quad i \quad occurs \quad in \quad d \\ 0 & otherwise \end{cases} \tag{1.3}$$

Another weight of terms in documents is the Term Frequency (tf) which is a measurement of how frequently a term occurs within a document. and can be calculated according to one of the following formulations :

$$d_i = tf_i \tag{1.4}$$

or

$$d_i = 1 + log(tf_i) \tag{1.5}$$

or

$$d_i = \frac{tf_i}{\sum_j tf_j} \tag{1.6}$$

Where tf_i is the number of occurrences of index term i in the document and $\sum_j tf_j$ is the sum of all term occurrences in the document (i.e. length of the document).

The term frequency is not sufficient to distinguish the important terms from less important ones. Thus, it is often used in conjunction with the Inverse Document Frequency (idf) that calculated as follows :

$$idf_i = log(\frac{N}{df_i}) \tag{1.7}$$

Where N is the total number of document in the collection and df_i is the number of documents containing the index term i. The final weight for each individual term will therefore be :

$$d_i = tf_i.idf_i \tag{1.8}$$

As for the query vector, the weight of an index term is often, mostly for short queries, equal 1 if the term appears in the query, and 0 otherwise. Once the document and the

query are represented by vectors that contain weights of terms calculated using methods mentioned above. The score of the document d with respect to a query q is measured by the cosine of the angle between the two vectors (Figure 1.2).

$$score(\vec{d}, \vec{q}) = \frac{\sum_{i=1}^{T} d_i.q_i}{\sqrt{\sum_{i=1}^{T}(d_i^2)}.\sqrt{\sum_{i=1}^{T}(q_i)^2}} \tag{1.9}$$

1.4.4 Probabilistic models

The probabilistic models are based on the theory of probabilities and is based in the following fundamental assumption given by Robertson [4] [110]:

Définition 1.5 *Assumption (Probabilistic Principle): Given a user query q and a document dj in the collection, the probabilistic model tries to estimate the probability that the user will find the document dj interesting (i.e., relevant). The model assumes that this probability of relevance depends on the query and the document representations only. Further, the model assumes that there is a subset of all documents which the user prefers as the answer set for the query q. Such ideal answer set is labeled R and should maximize the overall probability of relevance to the user. Documents in the set R are predicted to be relevant to the query. Documents not in this set are predicted to be non-relevant.*

Given a query q as a subset of index terms and a document d represented by a victor of binary weights (w=1 if term occurs in the document, w=0 otherwise). Let R be the set of documents known to be relevant and $\bar{R}$ be the complement of R (i.e., the set of non-relevant documents). The relevance score $RSV(d, q)$ of the document d to the query q is defined as the ratio:

$$RSV(d, q) = \frac{P(R|d, q))}{P(\bar{R}|d, q)} \tag{1.10}$$

Where $P(R|d, q)$ be the probability that the document d is relevant to the query q and $P(\bar{R}, d, q)$ be the probability that d is non-relevant to q. After using the Bayes' rule, the equation 1.10 yields :

$$RSV(d, q) = \frac{P(d|R, q).P(R, q)}{P(d|\bar{R}, q).P(\bar{R}, q)} \tag{1.11}$$

$P(d|R, q)$ and $P(d|\bar{R}, q)$ represent the probabilities of randomly selecting the document d from the set R of relevant documents and the set $\bar{R}$ of non-relevant documents respectively. $P(R)$ and $P(\bar{R})$ stand respectively for the probabilities that a document randomly selected from the entire collection is relevant and non-relevant which are constant

Case	Relevant	non-relevant	Total
Docs that contain k_i	r_i	$n_i - r_i$	n_i
Docs that do not contain k_i	$R - r_i$	$N - n_i - (R - r_i)$	$N - n_i$
All documents	R	$N - R$	N

Table 1.1: Contingency table [4].

and the same for all the documents of the collection. Therefore,

$$RSV(d,q) \sim \frac{P(d|R,q)}{P(d|\bar{R},q)} \tag{1.12}$$

Because the document d is represented by a vector of index term weights and assume that terms are statistically independent, we can write

$$RSV(d,q) \sim \frac{(\prod_{k_i|w_{i=1}} P(k_i|R,q))(\prod_{k_i|w_{i=0}} P(\bar{k_i}|R,q))}{(\prod_{k_i|w_{i=1}} P(k_i|\bar{R},q))(\prod_{k_i|w_{i=0}} P(\bar{k_i}|\bar{R},q))} \tag{1.13}$$

After adopting the notation $P_{iR} = P(k_i|R,q)$ and $q_{iR} = P(k_i|\bar{R},q)$ and simplifying the equation by taking the logarithm , the equation becomes :

$$RSV(d,q) \sim log(\frac{P_{iR}}{1-P_{iR}}) + log(\frac{q_{iR}}{1-q_{iR}}) \tag{1.14}$$

Using the contingency table 1.1, we could write

$$p_{iR} = \frac{r_i}{R} \quad and \quad q_{iR} = \frac{n_i - r_i}{N - R} \tag{1.15}$$

Then the equation 1.14 can be written as

$$RSV(d,q) \sim \sum_{k_i \in q \wedge k_i \in d} log(\frac{N - n_i + 0.5}{ni + 0.5}) \tag{1.16}$$

We notice that the score is based only on the IDF of query terms and does not take into account neither the frequency of the term in the document nor its length. These limitations are corrected in BM25 and language model as we will discuss below.

1.4.4.1 BM25 model

BM25 model [112] is an extension of the classic probabilistic model. The main objective was to extend the classic formula (equation 1.16) with information on term frequency and

document length normalization that seems to improve the ranking results in the same way in vector space model. As to improve the results, the first idea is to multiply the equation 1.16 with term frequency factor. This function can be written as follows:

$$RSV(d,q) \sim \sum_{k_i \in q \wedge k_i \in d} log(\frac{N - n_i + 0.5}{ni + 0.5}) \frac{(K_1 + 1).tf_i}{K_1 + tf_i} \tag{1.17}$$

Where tf_i is the frequency of term k_i within the document d, K_1 is a constant that controls term frequency scaling and can be tuned experimentally for a given collection. Notice that if $K_1 = 0$ the factor will be equal to 1 and nothing is going to be changed in the equation.

The second idea is to introduce the document length normalization into the score equation that became :

$$RSV(d,q) \sim \sum_{k_i \in q \wedge k_i \in d} log(\frac{N - n_i + 0.5}{ni + 0.5}) \frac{(K_1 + 1).tf_i}{K_1 \left[(1-b) + b\frac{len(d)}{avg\_dl}\right] + tf_i} \tag{1.18}$$

Where $len(d)$ is the length of document d and avg_dl is the average document length of the whole collection. b is a constant in the interval $[0, 1]$ that controls document length normalization.

The third idea is to take into acount the frequency terms in the query (the weight of the term in the query). Then the general ranking function of BM25 model can then be written as

$$RSV(d,q) \sim \sum_{k_i \in q \wedge k_i \in d} log(\frac{N - n_i + 0.5}{ni + 0.5}) \frac{(K_1 + 1).tf_i}{K_1 \left[(1-b) + b\frac{len(d)}{avg\_dl}\right] + tf_i} \frac{(k3 + 1)qtf_i}{k3.qtf_i} \tag{1.19}$$

Where K_3 is a constant that controls the weight of terms in the query. The best parameters of the model are $K_1 \in [1.2, 2]$, $b = 0.75$ and $K_3 = 1000$ [4]. This model gave a better results than the VSM in several collection. Thus, he became the baseline for evaluating a new proposed model.

1.4.4.2 Language model

Language model for information retrieval is another probabilistic model that was proposed by Ponte and Croft [109]. This model is based on Language Model (LM), where assigning a score for each document by estimating the probability that the query was generated from the document model. Given a query $q = q1q2...qn$ and document $d = d1d2...dm$. In order to rank documents, we have to estimate the probability $p(d|q)$, which after applying

Method	$p_s(w\|d)$	α_d	Parameter
Jelinek-Mercer	$(1-\lambda)p_{pl}(w,d)+\lambda p(w\|C)$	λ	λ
Dirichlet	$\frac{c(w;d)+\mu p(w,C)}{\|d\|+\mu}$	$\frac{\mu}{\|d\|+\mu}$	μ
Absolute discount	$\frac{max(c(w;d)-\delta,0)}{\|d\|}+\frac{\delta\|d\|_\mu}{\|d\|}p(w\|C)$	$\frac{\delta\|d\|_\mu}{\|d\|}$	δ

Table 1.2: Summary of the Three Primary Smoothing Methods Compared in this Article [136]

Bayes' rule is given by

$$p(d|q) \sim \frac{p(q|d).p(d)}{p(q)} \tag{1.20}$$

$P(d)$ and $p(q)$ are assumed to be uniform the same for all documents; so does not change the ranks of documents. Thus, the retrieval model reduces to the calculation of $p(q|d)$.

$$p(q|d) = \prod_{i=1}^{n} p(q_i|d) \tag{1.21}$$

To avoid the Zero-frequency problem, several smoothing methods have done, they used the two distributions models, $p_s(w|d)$ used for *seen* words that occur in the document, and $p_u(w|d)$ for *unseen* words that do not. The probability of a query q can be written in terms of these models as follows

$$log\ p(q|d) = \sum_{i:c(q_i;d)>0} log\frac{p_s(q_i|d)}{\alpha_d p(q_i|d)} + n\ log\ \alpha_d + \sum_{i=1}^{n} log\ p(q_i|C). \tag{1.22}$$

Where $c(q_i;d)$ denotes the count of word q_i in d and n is the length of the query, α_d is a document dependent constant and $p(q_i|C)$ is the collection language model.

The three popular smoothing methods are : Jelinek–Mercer Method, Bayesian Smoothing using Dirichlet Priors and Absolute Discounting. The three methods are summarized in Table 1.2

1.5 Machine learning to rank documents

We saw in the previous section that in order to tackle the problem of ranking in information retrieval many heuristic ranking models have been proposed in the literature.

Recently, given the abundance of training data (documents, queries and relevance judgments), it has become possible to harness machine-learning (ML) algorithms to learn efficient ranking models. Machine learning was applied for information retrieval in two ways. The first is about the Learning to Rank (L2R) methods which use supervised machine learning techniques to combine features related to documents and queries and relevance judgement associated with them to build a learned model. The set of features is prepared manually and provided as an input to the learning algorithm to enable it to learn how to rank documents for unseen queries. The second way is about the more recently proposed neural models for IR which do not need any features preparation, the model learn document and query representation by itself. However, unlike learning to rank models, the neural models are data-hungry, requiring enormous volumes of data before they can be deployed.

1.5.1 Learning to Rank

Since learning to rank is a supervised machine learning algorithms, and therefore, a labelled dataset is required to build the model. From the Figure 1.3, we can see that in the training phase, each query-document pair (item to be ranked) is represented by a set of features such as the number of query terms, document length, the sum of term frequencies for the terms in the query, the sum of their inverse document frequencies, etc. Each query-document pair is associated by a label that indicates the relevance judgment between the document and the query. Then, a ranking model is automatically learned using the training data, such that the model can accurately rank documents for a new query. In the test phase, the learnt model is applied to rank documents to the unseen queries.

Depending on their input representation and loss functions, the L2R approaches have been categorized into three groups [88, 97].

- In the pointwise approach, every single document is considered as a learning instance, and a regression model is typically trained on the data to predict for each instance the numerical label associated. If the label is binary (0,1), the classification model is applied.
- In the pairwise approach, the model regards document pairs as learning instances, and formulates ranking as a pairwise classification problem.
- In the listwise approach, each instance is the entire set of documents associated

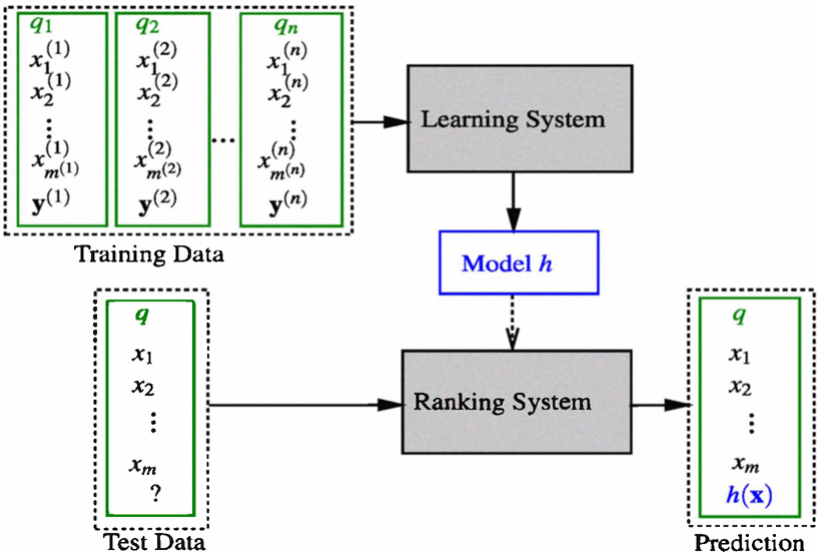

Figure 1.3: Learning to Rank framework [87]

with one query, and the loss function is generally related to evaluation measures for information retrieval such as MAP or NDCG.

1.5.2 Neural network for ranking

Recently, neural networks (deep learning) have been successfully applied to information retrieval applications. By exploiting deep architectures, deep learning techniques can be used to perform the three primary steps of IR (generate a document representation, generate a query representation, and match them to estimate the relevance of the document to the query). Three categories of models were distinguished [97]. In the models of the first category, documents and queries are represented by features, extracted manually, as in L2R and neural networks are used only to learn the relevance score of documents for different queries [50, 98]. In contrast, in the second category, the models are used to learn a low-dimensional vector representation of documents and queries, and then simple similarity metrics (eg. cosine similarity) is used to compute the distance between the document and query embedding [59, 99]. Finally, for the last category, the neural networks are used to learn low-dimensional vector representation of words [95, 107] first, then using this representation to select the most similar words to the query terms and expand the query [41, 115].

1.6 Evaluation of information retrieval systems

Evaluating a system is an important step in its development process to measure its success in achieving its initial goals. The objective of the IR system is to retrieve relevant documents to a user query from a large document collection. Thus, the success of an IRS rests on how well the system meets the users' needs. Baeza-Yates and Ribeiro-Neto [4] give a more complete definition as follows :

Définition 1.6 *Retrieval evaluation is a process of systematically associating a quantitative metric to the results produced by an IR system in response to a set of user queries. This metric should be directly associated with the relevance of the results to the user. A common approach to compute such a metric is to compare the results produced by the system with results suggested by humans for that same set of queries. Notice that Retrieval evaluation here means evaluating the quality of the results, not the performance of the system (in term of how fast it processes queries).*

There are many retrieval models in literature so in order to figure out which of them is effective, there is a need to evaluate them. For this purpose various metrics have been proposed for evaluating the retrieval quality of IR systems, i.e. the quality of the results returned by the system depending on the user's request. These metrics require a reference collection, which contain a set of m document $D = d1, d2, ..., dm$, a set of n query (information need) and a set of relevance judgments associated with each pair $[q_i, d_j]$ that describe the relevance of d_j to q_i, the value of relevance judgment can be either 0 (non-relevant) or 1 (relevant) in the simplest case or more nuanced in other cases [0,1,2,3,4] for [Bad, Fair, Good, Excellent, Perfect]. These relevance judgments are produced by human specialists, more details on how to construct a test collection can be found in [4]. The widely used metrics for evaluating IR systems are recall, precision and ndcg.

1.6.1 Recall and precision

Given a query q from a reference collection. Let R be the set of relevant documents to q and $|R|$ the number of documents in this set. Assume that a given system to be evaluated returns a set of documents A that contains $|A|$ documents and $|R \cap A|$ is the number of documents in the intersection of the two sets R and A. The precision and recall measures are defined as follows :

Precision is the fraction of retrieved documents that are relevant

$$Precision = p = \frac{|R \cap A|}{|A|} \tag{1.23}$$

Recall is the fraction of relevant documents that are retrieved

$$Recall = r = \frac{|R \cap A|}{|R|} \tag{1.24}$$

1.6.2 F-measure

The F-score is the harmonic mean of the precision and recall, well known as F-measure in the context of information retrieval. It is one of the methods used to combine recall and precision in a single value. The generalized F-score in position j is computed as follows:

$$F_\beta Score(j) = (1 + \beta^2)\frac{p(j) * r(j)}{\beta^2 * p(j) + r(j)} \tag{1.25}$$

Where $r(j)$ and $p(j)$ are recall and precision at position j respectively. β is a factor indicating the importance of precision over recall, or vice-versa. The standard F-score is when $\beta = 1$ and it is called $F1$ metric.

1.6.3 MAP : Mean average precision

The idea of mean average precision is to generate a single value summary of the ranking by averaging the precision figures obtained after each new relevant documents is observed [4]. We keep the same notations given in the section 1.6.1. Let $R[k]$ be the reference to the k-th document in $|R|$ and $P(R[k])$ is the precision when the this document is observed in the ranking of the query q. The mean average precision (MAP) of the query q is defined as

$$MAP = \frac{1}{|R|}\sum_{k=1}^{|R|} P(R[k]) \tag{1.26}$$

To calculate the MAP over all queries, we firs calculate the MAP_i for each query q_i and average the all as follows:

$$MAP = \frac{1}{|Q|}\sum_{i=1}^{|Q|} MAP_i \tag{1.27}$$

Where $|Q|$ is the number of queries in the test collection.

1.6.4 MRR : Mean reciprocal rank

Generally users interest to the first results (documents) returned by the retrieval systems. Reciprocal Rank is a metric that evaluate the system on its ability to rank relevant documents as high as possible in the ranked list. Let $|A_i|$ be the ranked list rturned by the system in response to a query q and $rank_i$ refers to the rank position of the first relevant document for the i-th query. Given a threshold ranking position S_h, the reciprocal rank RR_i of the query q_i is then defined as:

$$RR_i = \begin{cases} \frac{1}{rank_i} & if \quad rank_i \leq S_h \\ 0 & Otherwise \end{cases} \tag{1.28}$$

For a set Q of $|Q|$ queries, the mean reciprocal rank across all queries is averaged as follows

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} RR_i \tag{1.29}$$

1.6.5 NDCG : Normalized discounted cumulative gain

The metrics described above are typically used to evaluate information retrieval systems. They are based on binary judgments and which are unable to differentiate between highly relevant documents and less relevant ones when documents are associated with multiple levels of relevance. To address this problem, Researchers have formulated a new evaluation measure called DCG (discounted cumulative gain) that allows for graded relevance judgments. The formula of DCG accumulated at a particular rank position p is defined as

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{log2(i+1))} \tag{1.30}$$

Where rel_i is the graded relevance of the document at position i in the ranked list returned by the system.

The DCG will be compared to the ideal discounted cumulative (IDCG) calculated by the function above after sorting all relevant documents in the corpus by their relative relevance judgments. Then, the normalized discounted cumulative gain, or nDCG, is computed as

$$nDCG_p = \frac{DCG_p}{IDCG_p} \tag{1.31}$$

1.7 Conclusion

We addressed in this chapter the basics of information retrieval. We talked about information retrieval systems and their main components such as document and query representation for indexing and matching. Afterwards, we presented the different relevance models proposed in the literature matching queries and documents, as well as the metrics proposed for evaluation. In the next chapter, we present the recent research area called social information retrieval that use the social information extracted from social media to enhance the retrieval performance. We will show also how these models can be adapted to take into account the new social dimension for documents,users and queries.

Chapter 2

Social information retrieval

2.1 Introduction

The explosion of web 2.0 and social media networks has created many kinds of information either it is textual (Tags and reviews) or numerical (number of like, share, etc.). This social information has been exploited in several application domains such as marketing [71], commerce [122], etc. The field of information retrieval is no exception to this. A few years ago, a new area of research namely social information retrieval (SIR) emerged and gained popularity. SIR systems are distinguished from traditional information retrieval systems by exploiting social information generated by users in social media into the information retrieval process. in this chapter, we intend to provide a clear understanding of social information retrieval area by :

1. Describing the working principle of social media tools and their use in different domains as well as the different types of social information that can be used in the retrieval process.

2. Reviewing some of the most important works in the field of SIR and categorizing them based on the type of, and how they use, social information in the retrieval process.

2.2 Social media and knowledge sharing

The rise of social media has broadly impacted the evolution of the web. The web has evolved from a static web, which only delivers information, to a dynamic web that en-

gages users in the production of information. Users' behavior has also changed, from simple users who just consume information to users that can interact or participate in content production. Social media refers to any on-line service through which users and/or organizations can create and share a variety of content and interact with other users. Thus, social media can be defined as any website or application that are developed to allow users to create content and share their thoughts, ideas and information with other users. There are many different types of social media and we can categorize them into three categories, according their working principles. The first category concerns the social networks in which users connect and exchange thoughts, ideas, and content with their friends such as Facebook and Twitter who are the most used public social networks. There are also other social networks like Linkedin or Researchgate which are professional and academic-oriented than public. Linkedin is used by professionals to get connected with professional of their domain for work-related purposes, whereas Researchgate is used by researchers from all across the globe to get connected with researchers of their field of research. The second category is dedicated to social media that users create and share some type of content, video sharing (i.e. Youtube), photo sharing (i.e. Flicker), book sharing (i.e. LibraryThing or Goodreads) or app sharing (i.e. Play Store, or App Store). Other users can tag, review, rate, etc. Social media of this category do not require an account to access and consult the content, However, an account is needed for users that want to upload, comment or rate content. The third one is the discussions' forums like Reddit, Quora or Yahoo! Q/R which are favorite destination for users to submit posts and generate in-depth discussion among users on different topics of everyday life such as, politics, sports or religion. Users can ask for suggestions and recommendations, and other users can leave detailed responses by writing comments or responding directly to those comments, allowing conversations to grow automatically. Usually users who do not find their needs through the search engine, use these forums to ask other users for their needs. To summarize, whatever the type of the social media, the main purpose is to share experiences and knowledge about multiple topics among users. The content created by users and the evaluative information in the form of comments and social signals (shares, dislike/like) are referred as social information. We can also find social information in commercial websites like Amazon and news websites that allow users interact with their products and articles although they are not social media. In the next section we will talk about the social information that can be generated by users through their interactions in social media.

Type	Examples	Description
Content	Videos, images, blogs, articles, questions, etc.	Every kind of content posted by users on social media including mainly videos, images, blogs, articles, songs, questions, etc.
Metadata associated to content	Tags	Keywords used to annotatate a content
	Comments	Piece of text used to reply to a post
	Reviews	Piece of text used to evaluate a content
	Number of likes	Number of times a content was liked
	Number of shares	Number of times a content was shared
User relations	Friends, Followers or Following	A list of users that are friends, followers or following of a specified user

Table 2.1: Different types of social information generated on social media

2.3 Social information

Several types of information can be generated by users through their interactions in social media. Table 2.1 shows and describes the three different types of social information generated with some example of each. The first type of social information created on social media is the content itself. Users post videos on Youtube, images on Flickr, questions on forums (Quora, StackOverflow, etc.) or post any thought that comes to their mind. As for the second type of social information , it consists of the metadata add by users to the content. Among this information, there are tags which are a keywords that added by users to describe the content. Tags are also known as labels or annotations, and the process of associating tags to content is known as tagging. There are also comments and reviews which are a pieces of text used to reply to a post or evaluate a content respectively. Number of likes and shares are another metadata associated by users to content which are represent the number of times the contend is liked or shared by users with their friends. The third type of social information that can be extracted from social media, it concerns of users relationship. We differ two relationship, explicit and implicit relationship. Explicit relationship consist of explicitly declared relation of friendship (e.i Facebook)or follower (e.i. Twitter), whereas, hidden or implicit relationship consist of similar users based on their similarity as decided by their interaction on content in the social media.

2.4 Social information retrieval

Social information retrieval (SIR) sometimes referred to *social search* or *Social information seeking (SIS)*, is a very wide new domain and could be defined in several ways. At first, the term social search was used in [47, 61] and limited to *"a group of approaches that use past user behavior to assist current users in finding information that satisfies a specific information need"* [23]. however, with increased popularity of social media and social networking in the decade that followed, the term social information retrieval appeared which typically defined as concerned with the leveraging of social information from social media in the information retrieval process in order to enhance the retrieval performance and satisfy an user's information need. As per our knowledge, the first appearance of this concept in the area of information retrieval was in 2004 by [126], which refers to information filtering in recommendation systems regarding the personal social context of users. Several definitions can be found in the literature, many of which differ from one another. Here are some definition given in the literature :

In 2006 a definition of of SIR is given in [73] :

Définition 2.1 *"Social information retrieval systems are distinguished from other types of information retrieval systems by the incorporation of information about social networks and relationships into the information retrieval process."*

After the emerging of the area [35] consider that :

Définition 2.2 *"Social information retrieval is an emerging area for the design and implementation of a new generation of information retrieval systems."*

Bouadjenek et al. [20] provide a definition by specifying the social information used in the retrieval process :

Définition 2.3 *"Social Information Retrieval is the process of leveraging social information, (both social relationships and the social content), to perform an IR task with the objective of better satisfying the users' information needs."*

According to the definitions given above, Social IR systems are distinguished from other IR systems by integrating user generated content (UGC) and users' relationships from social media in the information retrieval process. Thus, SIR systems consider social information as a new important source of evidence from the collective contributions of users "*wisdom of the crowd*" that can be used in the retrieval process. However, these

definitions talk about the incorporation of social information in retrieval process but do not specify when and how this information will be used. In Social search panel session of the 2012 Microsoft Research Faculty Summit [1], the researchers had given a more complete definition using the term social search :

Définition 2.4 *Social search is an emerging research area that explores how social interactions and social data can enhance existing information-seeking experiences, as well as enable new information retrieval scenarios. This session will showcase different models of social search, including 1) the use of social data to augment search, 2) social data as new information to be searched, and 3) social interaction and collaboration as part of the search process.*

According to the last definition, there are three different directions in which social information can be leveraged in the retrieval process:

- Social data as a complementary source of information to augment search
- Social data as new information to be searched
- Social interaction and collaboration as part of the search process

We give a brief explanation to the second and the last category then we give a more details in section 2.4.3 for the first category as it is fully consistent with research objectives of our thesis.

2.4.1 Social data as the information to be searched

This group consider that the social information is the content to be searched (i.e., social contents generated in social media such as wikis, blogs, forums, and social network sites like Facebook and Twitter). In such a context, the huge quantity of information created by users in social media, which represents a useful information for different purposes. Hence, users use social media to gather recent information about a particular subject or to consult users opinions about event, product, hotel or restaurant. Therefore, social content search systems come as a mean to index content explicitly created by users on social media and provide a real-time search support [JCC10]. Figure 2.1 shows the social-search [2] interface that users can choose the social media sources in which the research

[1] https://www.microsoft.com/en-us/research/video/social-search-panel/
[2] https://www.social-searcher.com/

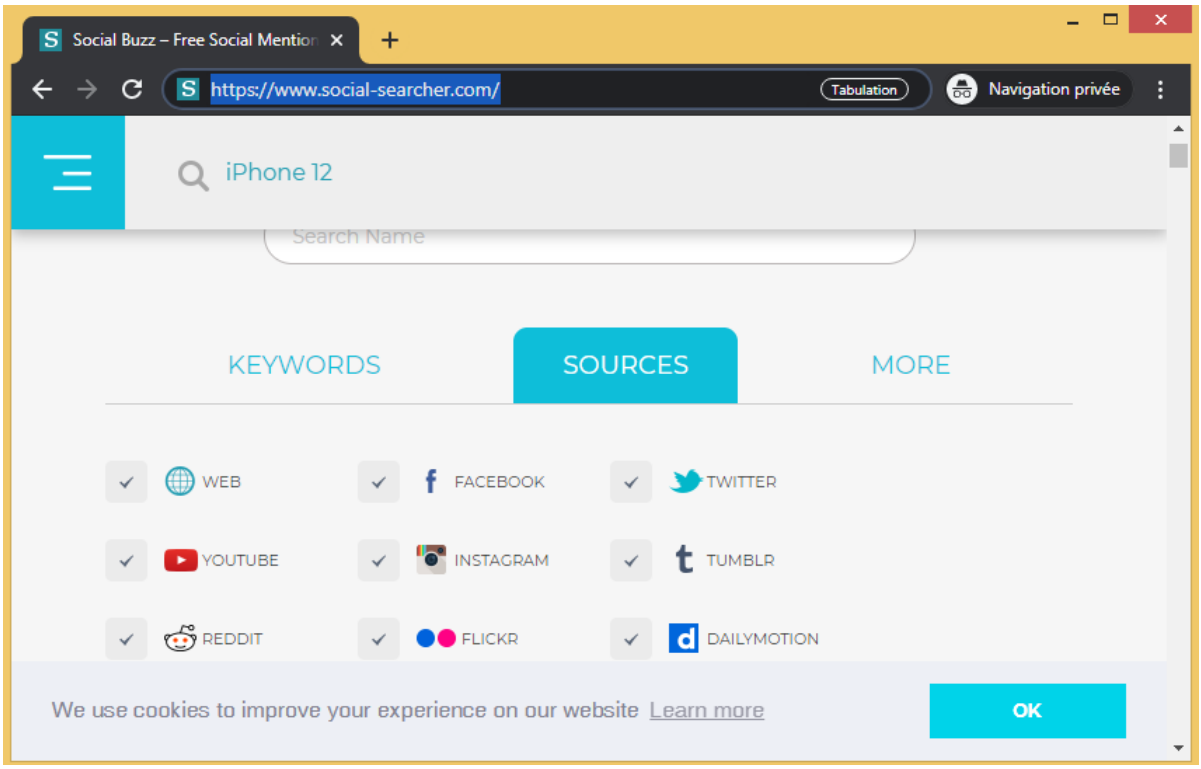

Figure 2.1: Social-search interface

will be carried out, and figure 2.2 shows the results for the query "iPhone 12" with search restricted to Facbook, Twitter and Instagram sources.

2.4.2 Social interaction and collaboration

Another form of social search is asking questions of other people using online services. Question-answering (Q&A) is an example of social online services where people ask questions on a broad range of topics and receive answers, suggestions and recommendations from other users that can also comment and evaluate the quality of answers by giving ratings and votes. Among these social services, we can mention Quora [3] and Yahoo! Search [4] which are intended for researching information for different topics like technology, sports, politic, etc. Others are intended for researching information for a very specific area such as Librarything forum[5], where users that looking for books, discuss their needs and receive suggestions and recommendations from other forum's users. These services become more

[3] www.quora.com
[4] search.yahoo.com
[5] www.librarything.com/talk

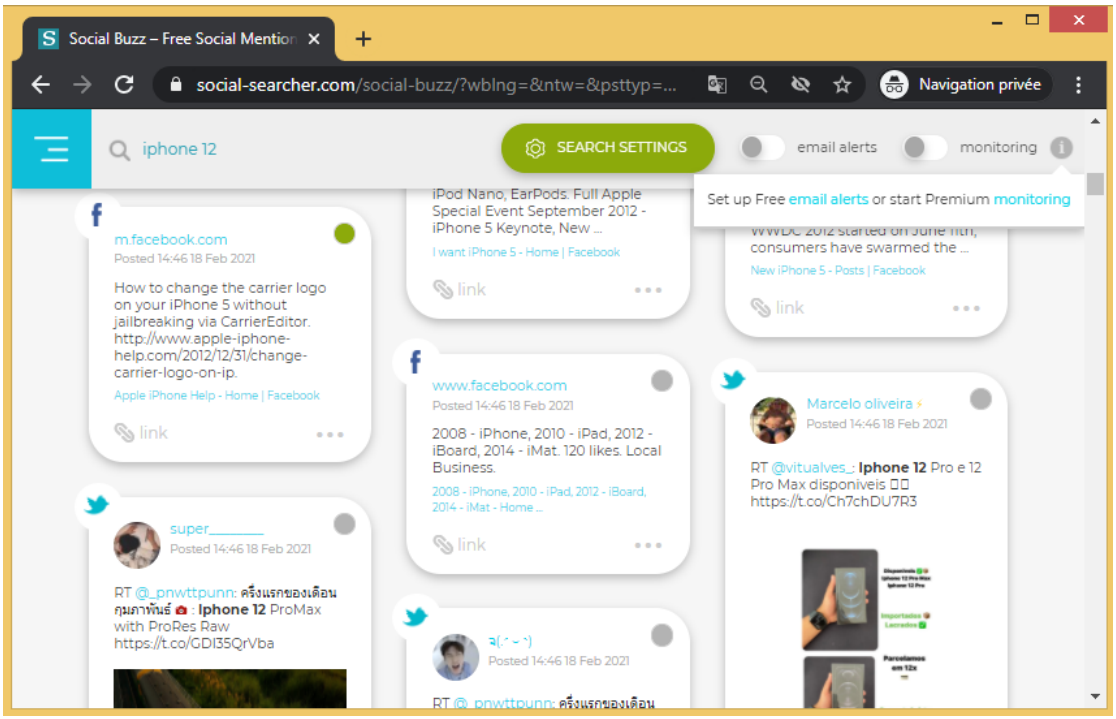

Figure 2.2: Results for the query "iPhone 12" with search restricted to Facbook, Twitter and Instagram sources

and more widely used, because often users, with complex search needs that are difficult to process by a search engine, turn to online forums to get recommendations from others users. Figure 2.3 and 2.4 are two examples of Q&A services, the first one shows a question posted by a user asking Librarything forum' users for books recommendation on *"introduction to Lisp"* and the second shows a question on quora where the writer ask about on how to keep safe from COVID-19. However, a drawback of such systems is that the questions may take a long time to get a response. Thus, many automated question answering systems have been developed to automaticaly help users finding their needs. Social Book Search (SBS)[6] investigates book search in a social environment to help users of LibraryThing forum to find their books requests. More details will be in Section 2.5.1.

2.4.3 Exploiting social information to enhance search

The last group of social information retrieval concern the works that aim to leverage social information to enhance a search task. In the last few years, many contributions in this group have been proposed. Each of these contributions focuses on a specific application

[6]http://social-book-search.humanities.uva.nl/

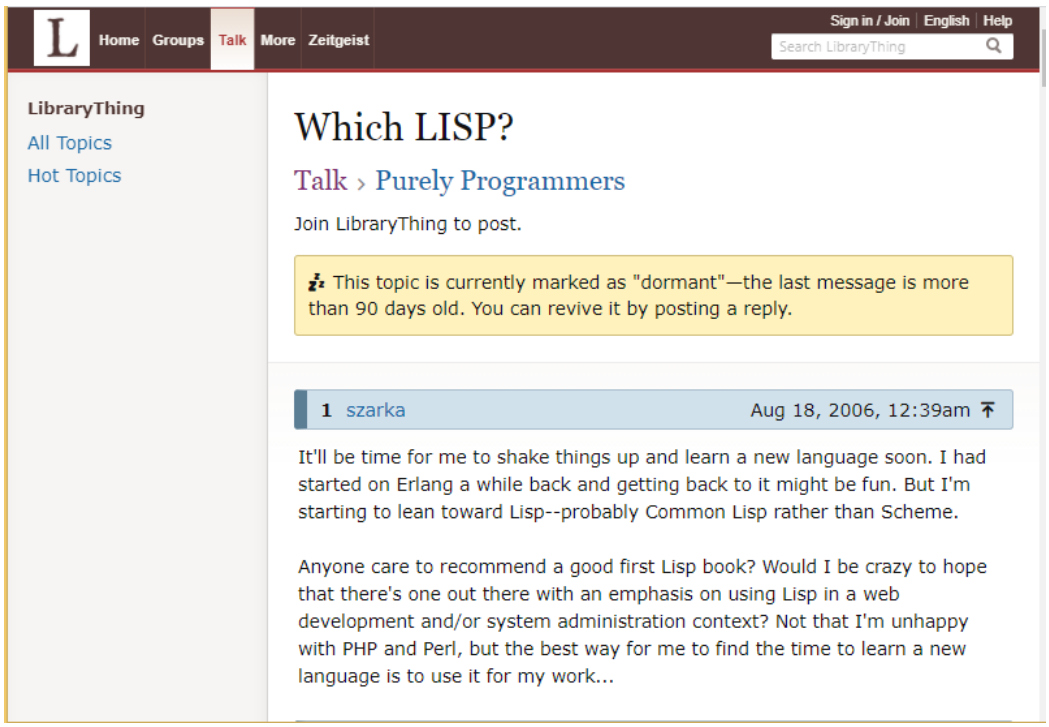

Figure 2.3: A topic thread in Librarything forum

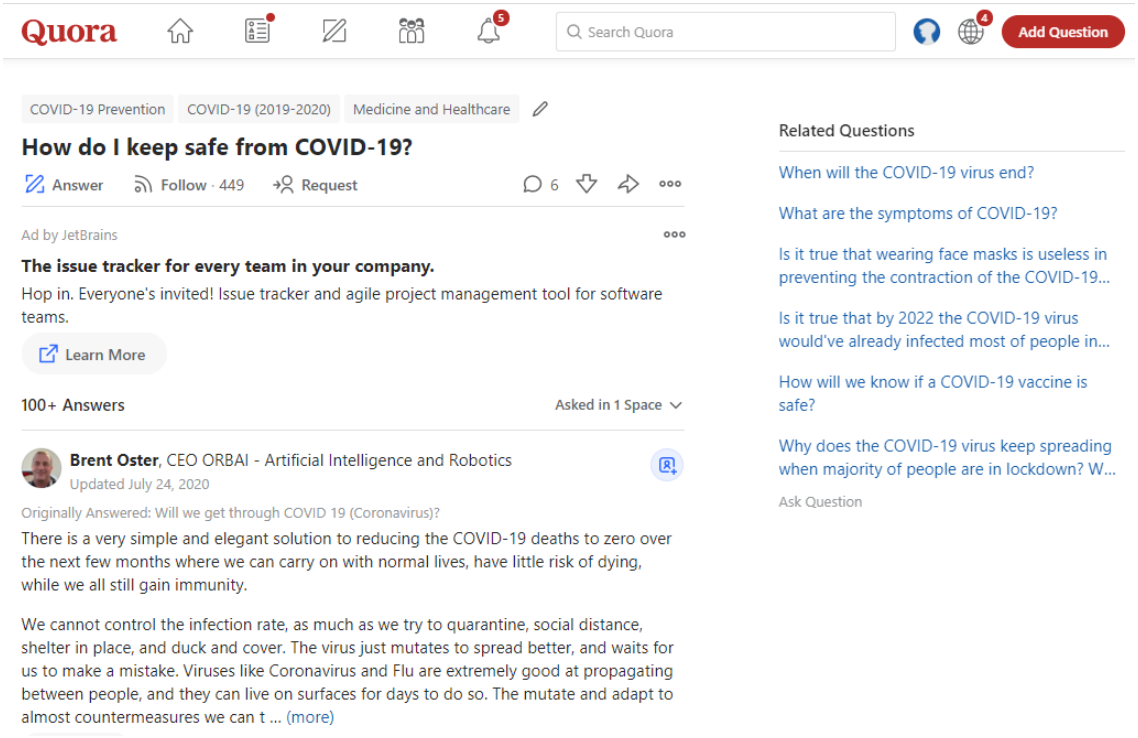

Figure 2.4: Example of question on Quora

domain and considers a particular social media, and a particular social information to develop an information retrieval system differently. Social information can be used in the retrieval process at different levels. The work of [20] categorizes them depending on the level of integration in the following way: (i) enriching document representation with social information in which tags and reviews are used to represent the documents (ii) query reformulation to enrich or re-weight the original query terms based on the user profile, (iii) result re-ranking based on user interest or other social relevance factors such as popularity and reputation. By analyzing the approaches proposed in the literature, we noticed that the approaches of the third category can be classified in the first or in the second since they are based on query and document representation to re-rank results. Hence, we categorize them into two categories, document representation and query reformulation.

2.4.3.1 Documents representation

Several research works reported that using social information to index documents enhances the search quality [8, 19, 55, 101, 133]. This may be due to two reasons [8]. Firstly, social information can be very useful for short documents having few repeating terms because simple indexing is not expected to provide good retrieval performances and most IR models rely on term frequency in a document. Secondly, social information can be useful for solving the vocabulary mismatch problem, especially in case there is a probability that the terms of relevant documents do not match the query terms. That is because the vocabulary gap between social information given by users and users' queries is smaller than the vocabulary gap between the content of documents and users' queries. The works of this category could be classified into three groups :

The first group of works studies to exploit a particular form of social information, social annotations, into the ranking function. Among these works, we can mention the work of Dmitriev et al [42] that explore the use of user annotations, to improve the quality of intranet search in enterprise environments using $tf.idf$ term weighting score to weight annotations. Bao et al [8] investigate the capability of social annotations in improving the quality of web page's search. For that, two social scores were defined : (1) SocialSimRank to estemate the similarity between query and annotations using the term matching method, (2) SocialPageRank is the popularity score of each page web calculated based on PageRank algorithm. The proposed approach was evaluated on a Delicious [7] corpus which consists of 1,736,268 web pages with 269,566 different social

[7] del.icio.us/

annotations and the experimental results show that both the proposed scores enhance the quality of a web search. SoPRa, a social personalized ranking function proposed by Bouadjenek et al. [19] that represent document and queries in vector space model using both content and social context (annotations) of document. They used the content score and the social score between the document and the query and they boost them with the interest score of a user to document based on his profile.

The second kind of studies illustrated the use of another form of social information, social reviews, to index documents and enhance the ranking process. Yee et al. [133] study the potential impact of comments on search accuracy. In addition to title and description of Youtube's videos, they used users' comments to improve search accuracy. Their results show that comments indeed improve the quality of search compared with just using titles and descriptions to describe videos. The exploitation of social reviews for the improvement of application (app) search is the objective of the work of Park et al. [106] who presented an approach named AppLDA using latent dirichlet allocation (LDA) on both representations, app description and social user reviews and identified topics in reviews which were also mentioned in the app description. AppLDA achieved a significant improvement in retrieval performance compared to traditional information retrieval models and showed the positive impact of social reviews on the retrieval performance. Social book search [75,76] is another domain that also used social reviews extracted from Amazon with annotations extracted from LibraryThing to index book. Several works have shown that despite using social information alone without book content, the retrieval systems can return relevant books to user queries.

The third kind of social information used to represent document is a non-textual information sometimes referred to social signals such us the number of Like, share and +1, tags and comments that the resource receives in social media. Abel et al. [1] reported that there is a strong interdependency between the popularity of users, tags and resources. Motivating by the observation Bao et al. [8] proposed an algorithm SocialPageRank that estimate the popularity score of each page web to optimize social search. Furthermore, Badache et al. [2,3] proposed to use language model incorporating temporal characteristic of social signals (number of like, shares and comments) to estimate the relevance score of the resources in order to re-rank the search results. Moreover, Chelaru et al. [32] proposed to explore the impact of social features on the video retrieval, the explicit and implicit user interaction with the system (such as views, likes, dislikes, favorites, comments, etc) are used to enhance video retrieval performance.

2.4.3.2 Query reformulation

Query reformulation is a process that modifies the original query submitted by the user, before used as an input to the retrieval system, in order to better understand the underlying user intent. A definition is given by [20] as follows:

Définition 2.5 *(Query reformulation). Query reformulation is the process which consists of transforming an initial query Q to another query Q'. This transformation may be either a reduction or an expansion. Query Reduction (QR) [81] reduces the query such that superfluous information is removed, while Query Expansion QE) [45] enhance the query with additional information likely to occur in relevant documents*

From the definition there are two techniques used for query reformulation, query expansion and query reduction. As of our knowledge, all contributions in query reformulation in social information retrieval focuse on query expansion [20]. Many methods have been proposed in this area. Based on the assumption that social tags that annotated the same resource have semantic relevance, Jin et al. [66] propose a method in which they used tag co-occurrence method for similar tags selection. Hence, a set of highly semantically similar tags was selected and used to expand the original query which had a potential impact on the efficiency. By the same way, Lin et al. [86] proposed a term ranking approach based on social annotations collected from the social annotation service (Delicious). Their proposed approach consists of two phases: (1) A set of most likely expansion tags are selected; (2) A learning to rank algorithm (ListNet) [25] was used to weight the selected tags using a set features to describe those tags and a MAP measure to label each tag. Experiments are conducted on three TREC collections and show that using social annotations for query expansion is significantly better than using feedback documents. The two above approach used query expansion identically, thereby they do not take into consideration the interests of the query user, so they are often not really suitable nor efficient as relevance of documents is relative for each user [20]. To overcome this problem, researchers have attempted to personalize the query expansion. Zhou et al. [139] proposed a method that builds upon individual user profiles in which terms are mined from both the tags a user has used and the resources the user has annotated. They associated a weight to each term in the user profile based on its similarity with other terms in the profile and terms extracted from the top-ranked documents. After calculation, the terms with the highest scores will be selected to expand the original query. The popular social tagging site delicious was used to construct a real-world data for evaluation. The proposed personalized expansion

technique performed well on the social data, delivering statistically significant improvements over non-personalized expansion. Wu et al. [127] propose to use word embedding to assign each term in user profile a low-dimensional vector in order to calculate the semantic similarity between them. Then, a graph is built based on the features derived from tags' vectors which will be used to rank the terms. Finally, the top ranked terms are used to expand the query. A recent work [140] proposes a novel model that bases on the use of word embeddings with topic models to enrich users profiles with the help of an external corpus, especially for users with limited previous activity with the system. The two techniques proposed are based on topical weights-enhanced word embeddings, and the topical relevance between the query and the user profile terms. The proposed methods performed well on two real-world social tagging datasets delicious and Bibsonomy [8], delivering statistically significant improvements over non-personalized methods.

2.5 Datasets for evaluation

As we saw in the previous chapter, evaluating information retrieval systems requires a reference collection that contains three parts: a set of documents, a list of query topics, and a set of relevance judgments. Social information retrieval is no exception to this rule. Thus, many collections was created for this purpose, however, in most of them, a fourth part add to the collection namely the user profile. The main challenge in this task is how can we estimate the relevance judgment of a document with respect to a given query submitted by a given user. Several social online services are used to create the datasets for social information retrieval; delicious[9], Flickr[10] and CiteULike[11] in the work [18]. IMDB, an online database of information related to (films, television programs, home videos, video games, etc.) was used by Badache et al. in [2]. LibraryThing and Playstore are used respectivelly in *Social Book Search* [75, 76] and *App Retrieval* [106]. In the following we will talk about the last two datasets namely, *Social Book Search* and *App Retrieval* as they are used in this thesis.

[8] www.bibsonomy.org

[9] delicious(del.icio.us): a social bookmarking web service for storing, sharing, and discovering web bookmarks.

[10] Flickr (www.flickr.com): an image and video hosting, tagging and sharing website.

[11] CiteULike (https://fr.wikipedia.org/wiki/CiteULike): an online bookmarking service that allows users to bookmark academic papers.

2.5.1 Social book search

Social book search (SBS)[12] was a CLEF lab (2011-2016) that particularly investigated book search in social media. It was interested in the use of user generated content from social media to support users of LibraryThing forum in finding documents(books) that interest them and that are relevant to their request [75, 77]. The organizers of INEX SBS have used Amazon[13] and Librarything(LT) to provide a document collection which consists of 2.8 million books. This collection contains both textual and non-textual social information about books. Textual social information consists of users' reviews and tags that are respectively extracted from Amazon and LibraryThing (see Figure 2.5). As to non-textual information like rating, number of time a book is catalogued, etc. they are extracted from LT whereas ratings, number of reviews, of total votes and helpful votes are extracted from Amazon.
In order to evaluate systems in SBS, a set of topics with relevance assessments are provided. These topics are based on discussion threads from LT forums, where users express their needs and ask suggestions and recommendations about books to other forum users. These topics contain many fields : group, title, narrative and examples (see Figure 2.6). The group field designates the discussion group in which a user posts their thread, the title is the short representation of the topic which often contains a brief summary about the user's need. Narrative is a long representation of the topic in which the user utilizes natural language to explain their needs in details. As to the field examples, it consists of some similar books add by some users in order to indicate the kind of books they want. For evaluation, the set of books suggested by forum users is used as the relevance judgments for evaluation.

2.5.2 App retrieval

App retrieval [106] is a dataset that contains 43,041 mobile application and 56 queries with their relevance judgments. Each app is represented by a description given by the app's developer and a set of reviews (max =50) given by online users. As for queries, each query is a set of keywords generated by domain experts collected from android forum (forums.androidcentral.com). To assign a relevance judgment to each app in the dataset for each query, the top retrieved apps from different retrieval systems [14] were manually

[12] www.social-book-search.humanities.uva.nl
[13] www.amazon.com
[14] To perform the retieval systems, apps are represented by their descriptions only.

```
<book>
<isbn>0194518000</isbn>
<title>New English File: Student's Book Intermediate level</title>
<listprice>$26.61</listprice>
<publisher>Oxford University Press</publisher>
<reviews>
<review>
<date>2006-07-20</date>
<summary>New English File Elementary</summary>
<content>New English File, as its name suggests, is the new and improved version
of English File. Unlike the original English File, which came in four levels (Beginner,
Pre-Intermediate, Intermediate and Upper-Intermediate),...Thank you for your attention.
</content>
<rating>5</rating>
<totalvotes>4</totalvotes>
<helpfulvotes>4</helpfulvotes>
</review>
</reviews>
<tags>
<tag count="1">english</tag>
<tag count="1">hard</tag>
<tag count="1">en</tag>
<tag count="1">@home-a</tag>
</tags>
</book>
```

Figure 2.5: XML file represents an example book in Social Book Search.

<topic id="41306"> <request>Spanish Civil War : For Whom The Bell Tolls :: French Revolution: ?? I don't think there's a right answer, I'm just looking for suggestions for a good fictional book set during the French Revolution. (My interest was piqued after reading the Jim Shepard short story "Sans Farine" in The Best American Short Stories 2007.) <request>
<group>Book talk</group>
<title>Fill in this historical fiction analogy</title>
<examples>
<example>
<booktitle>For Whom the Bell Tolls</booktitle>
<author>Ernest Hemingway</author>
<workid>10084</workid>
</example>
<example>
<booktitle>The Best American Short Stories 2007</booktitle>
<author>Stephen King</author>
<workid>3539369</workid>
</example>
</examples>

Figure 2.6: XML file represents an example topic in Social Book Search.

annotated through the crowdsourcing service,CrowdFlower[15].

2.6 Discussion

Information produced by users in social media has been used in the IR process, by several works, as a complementary source of evidence to refine the research results. These works have shown that social information such as clicks, annotations, social interactions and comments are important factors in the ranking process. We saw that these works was performed using two methods namely, query reformulation and document representation. However, although the important numbers of studies addressing the use of social information to enhance the retrieval process, there are still considerable challenges to be overcome to ensure their effective exploitation. These challenges are as follows:

For query reformulation, we reported that all the existing works focus on query expansion and there are no contributions on query reduction, while users on social forums discuss their complex information needs and request in natural language which require an additional processing before used as an imput to the searche engine. Several studies [10, 31] have shown that search engines generally perform poorly on verbose natural language queries when compared to short keyword queries. They indicate that the longer is the query the less efficient is the search engine and the research results improved when the number of the query terms are reduced.

As for document representation, we also reported that works that exploit the textual user generated content process annotations and comments similarly, and there is no study that has adapted a specific model for each type of content. As it seems logical that integrating tags in the retrieval model should not be in the same way taken to integrate reviews. Thus, the challenge is to analyse the different influences of using tags and reviews on both the settings of retrieval parameters and the retrieval effectiveness.

We noticed also that, taking into account social user reviews in the same way as using the text of documents, is not as logical as it seems. The user reviews contain other information that can be extracted, like user opinions or the features of the entity reviewed. Hence, it requires additional processing to be able to exploit them effectively.

[15]www.crowdflower.com

2.7 Conclusion

In this chapter, we have presented a state of the art of models proposed in the last few years to solve the issue of social information retrieval. Besides, we have established the two categories to distinguish between the different solutions proposed, query reformulation and document representation. Afterwards, we have described the main approaches of each category followed by a discussion in which we identified some challenges. Based on this critical study, The rest of this thesis will be dedicated to describing in detail our three contributions. The first contribution is to deal with verbose natural language queries often used by users in social forums to describe their needs. The second is to analyse the different influences of using tags and reviews on both the settings of retrieval parameters and the retrieval effectiveness. The social book search collection will be used for the evaluation of performance analysis of the approaches proposed. Whereas the last contribution is to leverage features extracted from social user' reviews about applications to improve mobile app retrieval.

Part II

Contributions

Chapter 3

Approaches to deal with natural language queries in social information retrieval

3.1 Introduction

The emergence of social media services such as question answering systems and discussion forums like LibraryThing allow users to utilize natural language to express their information needs in the form of long and verbose queries. On the other hand, and as most of the information retrieval models [17, 109, 111] are based on the term frequency of all query terms, to retrieve and rank relevant documents to the query, the performance of such models, with long queries, is not always satisfactory. Therefore, this type of queries requires pre-processing before being submitted to search engine. We present in this chapter the methods developed, to deal with natural verbose queries in social information retrieval, and the results obtained during our participation in Social Book Search evaluation campaign in 2015 and 2016 [26, 27] in which the queries are collected from the LibraryThing forum. In the first approach, several features, such as statistic features, syntactic features, and other features like whether the term is present in similar books and in the profile of the topic users, are used to describe each query term, and a learning to rank algorithm was applied to automatically weight those terms. The ranking function learned by the learning algorithm in the training phase, using 2014 and 2015 topics, has been used to weight and rank the terms of the 2016 topics. As for the second approach, we combine query reduction based on the statistic measure ($tf.iqf$) and query expansion based on

example books mentioned in the query. We expanded the experiments to include all SBS topics from (2011 to 2016) [28]. During this chapter, we report the details of the proposed solutions as well as the results found.

3.2 Motivations

It is certainly that the most of queries, submitted to search engine, are less than five terms long [40]. However, it is more advantageous, for web users, to use natural language to express their needs in detail instead of using minimal keywords in their queries. Thus, the use of long verbose queries by users has increased over time and the average query length has grown year after year [123]. Moreover, the emergence of social applications such as question answering systems , discussion forum like LibraryThing and voice queries on mobile devices. Such applications allow users to utilize natural language to express their information needs in the form of long and verbose queries. However, most of information retrieval models [17, 109, 111] are based on the term frequency of all query terms, to retrieve and rank relevant documents to the query. The performance of such models, with long queries, is not always satisfactory. This is due that most of retrieval models assume that all terms given by user in verbose natural language query are equally important and tend to retrieve documents that contain all of the query terms, while this is not the case. Some terms are completely extraneous to the context of the request and they have been used only as: an introduction to the request "Hey there! I am looking for suggestions/ recommendations for reading about...", or as a conclusion to the request "Thanks for any help.", "Any suggestions!".

The study of [10] has shown that search engines generally perform poorly on verbose natural language queries when compared to short keyword queries. This comparison was made using the queries of TREC corpora which considers that short queries are those with four terms at the most and long queries are those containing five terms or more. The same study indicates that the longer is the query the less efficient is the search engine. Moreover, a recent empirical study of [31] indicates that 73% of verbose queries, used in TR-based software maintenance are improved when the number of the query terms are reduced. The results increased by 21.8% and 13.4% in terms of MRR and MAP respectively. To tackle the problem of verbosity in natural language queries and improve search engine effectiveness, verbose queries have received more attention in recent years [10, 37, 105]. Most of them are classified in two main categories: Query Reduction and Query term

weighting Approaches. Instead of using all query terms of the original query, the query reduction approaches select only a subset of important terms to form the reduced query and submits them to search engine. The Query term weighting Approaches consist of assigning to each term, an appropriate weight, before submitting the new query to the search engine.

In summary, all this works, whether from query reduction approaches or query term weighting approaches, show an improvement in search engine performance over verbose long queries. Despite the fact that most of these works have focused on Ad-hoc information retrieval, there exist others that tackled specific domains: web image retrieval [49], e-commerce [132], spoken document retrieval [85], and medical document retrieval [7]. And to our knowledge no study has focused on social information retrieval [21].

In this chapter, we want to deal with verbose queries for social information retrieval using social book search collection as test to evaluate our approach. For book representation we only used the annotations extracted from LibraryThing ignoring social reviews from amazon. The topics used in SBS are based on discussion threads from LT forums and contain many fields namely: group, title, narrative and examples (see Fig.3.1). The group field means the discussion group in which the user posts their thread. The title is often a brief summary of the user' information need but sometimes out of the context, while in narrative the user utilizes natural language to explain their needs in details. As of examples field, it consists of a small number of similar books to the request that some LibraryThing users provide in their topics in order to indicate the kind of books they request. The terms of those example books can be used to expand the original query considering that those examples are pseudo-relevant to the query. From the narrative field we realize that the queries in SBS are verbose and sometimes contain much more than just the information need. This extraneous information to user need can lead to topic drift as well as search engine performs poorly. For this reason, we decided to remove this extraneous terms and keep only the appropriate to the context of the request and the users' information need.

3.3 Natural language query processing

As mentioned above, works on processing verbose queries can be categorized in two main categories [51]. In this section we introduce in details some related works for each category.

<topicid>107277</topicid>
<group>FantasyFans</group>
<title>Fantasy books with creative heroines?</title>
<request>Greetings! I'm looking for suggestions of fantasy novels whose heroines are creative in some way and have some sort of talent in art, music, or literature. I've seen my share of "tough gals" who know how to swing a sword or throw a punch but have next to nothing in the way of imagination. I'd like to see a few fantasy-genre Anne Shirleys or Jo Marches. Juliet Marillier is one of my favorite authors because she makes a point of giving most of her heroines creative talents. Even her most "ordinary" heroines have imagination and use it to create. Clodagh from "Heir to Sevenwaters," for example, may see herself as being purely domestic, but she plays the harp and can even compose songs and stories. Creidhe of "Foxmask" can't read, but she can weave stories and make colors. The less ordinary heroines, like Sorcha from "Daughter of the Forest" and Liadan from "Son of the Shadows," are good storytellers. I'm looking for more heroines like these. Any suggestions? </request>
<examples>
<example> <booktitle>Daughter of the Forest</booktitle> <author>Juliet Marillier</author>
<workid>6442</workid>
</example>
<example><booktitle>Foxmask</booktitle> <author>Juliet Marillier</author>
<workid>349475</workid>
</example>
<example> <booktitle>Son of the Shadows</booktitle> <author>Juliet Marillier</author>
<workid>6471</workid>
</example>
<example> <booktitle>Heir to Sevenwaters</booktitle> <author>Juliet Marillier</author>
<workid>5161003</workid>
</example>
</examples>
<catalogue/>

Figure 3.1: XML file represents the topic number 107277 from Social Book Search.

3.3.1 Query term weighting Approaches

Based on the idea that in natural language queries, each term can have different weights in different queries depending on the context of the query. Therefore, assigning a weight to query terms should have a significant impact on the results returned by search engine. The approach proposed in [10] used machine learning technique for identification and weighting query concepts. This method was achieved by representing each concept by different features and significantly improves retrieval effectiveness using a large set of natural language queries derived from TREC topics. Another work of [83] propose a learning to rank framework to weight all term of the query instead of concept. [131] used Hidden markov Model to weight terms in vebose queries. In this work, Part-of-speech (POS) features of terms are used as observations and the weight levels of the query terms as the hidden states. Drawing on an idea from text summarization, [105] propose an unsupervised method to estimate which term are most central to the query. An initial set of more relevant documents to the original query are used to define a recursion on the query word weight vector that converges to a fixed point representing the vector that optimally describes the initial result set. Recently, [37] propose a discriminative query language modeling by estimating the probability that a particular query term is topical and correctly weights the salient aspects of the query. Since the approach is based on language model, the frequency of terms in the query, in collection, across all queries and the length of the query are used as features.

3.3.2 Query reduction approaches

Based on the idea that natural language verbose queries contain many noisy terms, i.e., terms that are extraneous to the context of the request and might cause a topic drift. Therefore, query reduction whom remove inappropriate terms from long queries and keep only the appropriate terms to the topic and user's need should improve the performance of search engine. [82] proposed to generate a set of sub-queries (subset of terms) from the original query and then using RankSVM [67] to rank sub-queries. Finally, the top-ranked sub-query was selected to replace the original query. The work shows an 8% significant average improvement of the mean average precision. In the same context, [130] rank sub-queries using Conditional Random Field model (CRF) however they select the top-ranked sub-queries instead of the best sub-query, to replace the original query, using several types of retrieval models. [129] proposed a reformulation tree framework to organize the sub-queries as a tree structure, where each node is a reformulated query (sub-query).

Considering the relationships between nodes, a weight estimation approach assigns weights to each sub-query and directly optimizing the retrieval performance. [132] build a classifier to predict which term is the most likely to be deleted from a given query using various term-dependent and query-dependent measures as features. The authors validate their approach using a large collection of query sessions logs from an e-commerce site.

3.4 Proposed approaches

We present in this section the two proposed methods to deal with natural language queries in social book search. The first one uses learning to rank algorithm to automatically reduce language queries by weighting their terms. As for the second one, it uses the frequency of the term in the query versus its frequency across all queries in the dataset for query reduction and the terms of similar books for query expansion.

3.4.1 Learning to rank for query reduction

The key idea of our approach is to reduce the long verbose queries by assigning a weight to query terms. This weight should reflect their importance in the query. We then filtering the less important terms and select the top terms with highest weight to replace the original query. In order to do so we must find a function f to weight the original terms that satisfies the following assumption: Given an arbitrary query $Q = q_1, q_2, \ldots; q_n$, let $P(q_i, q_j)$ denotes all possible pairs of the terms of the query. For each existing pair, if q_i is more important than q_j then $f(q_i)$ must be superior to $f(q_j)$. In our approach, we consider the problem of weighting terms and reducing the verbose queries as a learning to rank problem [88]. Instead of ranking documents for each topic, as we usually do (see section 1.5.1 of chapter 1), we rank the terms of the topic. This can be formally described as follows: Given n training queries $q_i(i = 1...n)$, their associated terms represented by features vectors, and the corresponding labels (degree of importance of the terms). Then a learning to rank algorithm is used to learn the ranking function. The function learned is applied to rank terms for the test (unseen) topics.

3.4.1.1 Methodology

In order to learn the ranking function, we have to prepare a training data first. There are two important steps to be considered as mentioned in Figure 3.2

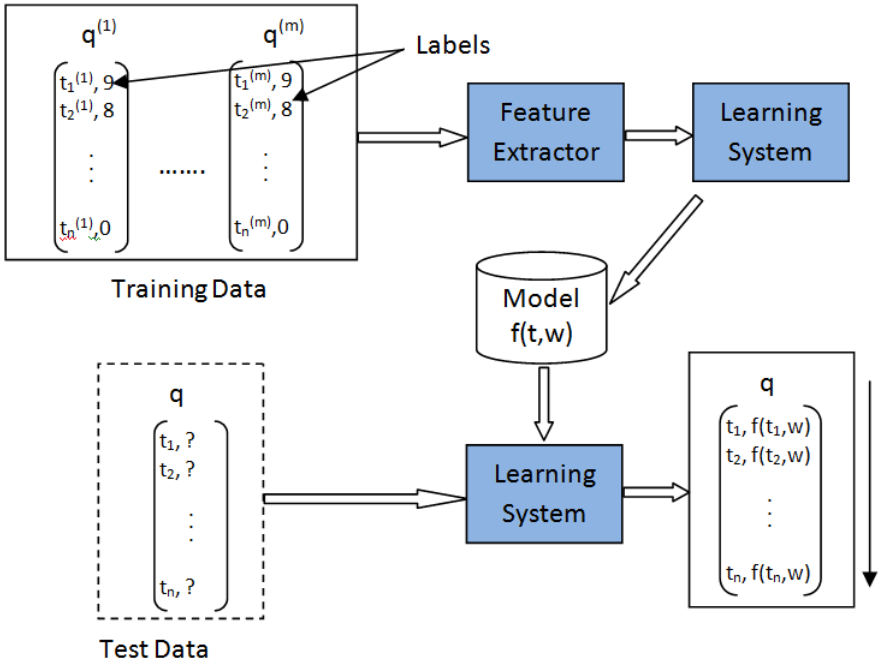

Figure 3.2: Learning to rank query terms

Training phase

- Select queries and their associated terms to be used in the training phase;
- Assign to each term a ground truth label (degree of importance);
- Extract a list of features which represent each term: such features have to be as decisive as possible for term weighting;
- Choose and apply a learning to rank algorithm.

Testing phase

- Apply the ranking function learned in the training phase, to rank terms associated to each unseen query;
- Select the top terms of each query from the ranked list to form the new query;
- Apply an information retrieval model to machining books with this new query.

3.4.1.2 Experimental setup

We used the topics of 2014 and 2015[1] for training. As to the terms associated to each query, we selected all the terms present in the three topic fields title, group, and narrative, as well as the terms of similar books. The natural language processing toolkit MontyLingua[2] is used to analyse the text of queries and keep only the nouns and adjectives while eliminating prepositions, verbs and articles. Regarding the ground truth label for learning and since we have for each topic the relevant books, from Qrels2014 file, but of course not the most relevant terms. Hence, we have to rank, for each topic, the terms of the relevant books by using the $tf.idf$ function. The label of each previously selected term will be assigned the inverse rank if the term is present in the ranked list, otherwise 0. For the features, several different categories have been used, including Statistical, Linguistic, Field, Profile, and similar book features. Table 3.1 describes the features of the five categories we used. After preparing the training data set as described previously, the learning to rank algorithm Coordinate Ascent [94] from RankLib[3] have been used to learn the function of weighting and ranking terms, this efficient linear algorithm have been chosen due to the unbalanced data that we have and in order to avoid the overfitting in the training phase. Finally, the ranking function learned by the learning algorithm in the training phase have been used to weight and rank the terms of the 2016 topics. The top-10 ranked terms of each topic have been selected to calculate the score of books for each query. The BM15 model [112] was used to matching queries and books while for indexation the tags associated to books are used. In order to improve the performance of our system, several combinations of features are experimented to determine the optimal set. Table 3.2 summarizes the different combinations.

According to the number of combinations described in Table 3.2, six models have been learned using 2014 topics and tested using 2016 topics. Table 3.3 reports the evaluation results of the combinations on 2014 and 2016 topics for the training and testing phase respectively. We mention that the result of this approach are submitted for participation in SBS 2016 track[4] and our team was ranked second out of 11 teams[5]. Table 3.3 shows the official evaluation results of our submissions. From the table, we note that combining linguistic features with statistical features improves the results more than using the

[1] 2015 topics are used to extract example books mentioned by a LT user for the 208 topics
[2] http://alumni.media.mit.edu/ hugo/montylingua/
[3] https://sourceforge.net/p/lemur/wiki/RankLib/
[4] http://social-book-search.humanities.uva.nl/
[5] http://social-book-search.humanities.uva.nl/#/suggestion16

Features	categories	Feature Feature description
Statistical Features	*In_Query*	"1" if the term appears in the query and "0" otherwise
	Tf_Iqf(t,q,Q)	Product of Term Frequency and Inverse Query Frequency
	Tf(t,q)	Term Frequency of t in the topic
	Iqf(t,Q)	Inverse Query Frequency of the term among all topics
Linguistic Features	*Is_Proper_Noun*	"1" if the term is a proper noun and "0" otherwise
	Is_Noun	"1" if the term is a noun and to "0" otherwise
	In_Noun_Phrase	"1" if the term appears in the list of noun-phrases extracted from the query and "0" otherwise
	Nb_Noun_Phrases	The number of noun phrases in which the term appears
Field Features	*In_Title_Topic*	"1" if the term appears in the title of the topic and "0" otherwise
	In_Narrative_Topic	"1" if the term appears in the narrative of the topic and "0" otherwise
	In_Group_Topic	"1" if the term appears in the group field of the topic and "0" otherwise
Profile Features	*In_profile*	"1" if the term appears in the list of tags extracted from the profile of the user and "0" otherwise
	nTF(t,u)	The ratio of the use of term t to tag resources to amount of resources tagged by the user u
Example Features	*In_Example_Book*	"1" if the term appears in the example books and "0" otherwise
	Tf_Idf(t,d,D) (in example book)	Product of Term Frequency and Inverse Document Frequency

Table 3.1: List of features categorized in five categories

Features	Combinations Description
Stat_features	Statistical features only
Stat_ling_features	Statistical and linguistic features
Stat_ling_field_features	Statistical, linguistic and Field features
Stat_lin_field_profile_feat	Statistical, linguistic, Field and profile features
Stat_ling_profil_expl_feat	Statistical, linguistic, profile and example features
All_features	All categories of features

Table 3.2: List of different combinations of features

Combinations	2014 topics (training)				2016 topics (Testing)			
	NDCG@10	MRR	MAP	R@1000	NDCG@10	MRR	MAP	R@1000
Stat_feautres	0.1078	0.2124	0.0805	0.5169	0.1082	0.2279	0.0749	0.4326
Stat_ling_features	0.124	0.2546	0.0897	0.5366	0.129	0.297	0.0816	0.456
Stat_ling_field_features	0.1103	0.2424	0.0801	0.5401	0.1084	0.2408	0.0714	0.4557
Stat_lin_field_profile_feat	0.1156	0.2368	0.0843	0.5249	0.1077	0.2635	0.0627	0.4368
Stat_ling_profil_expl_feat	0.1301	0.2673	0.0945	0.5063	0.1438	0.3275	0.0754	0.3993
All_features	0.1277	0.2626	0.0924	0.5133	**0.1567**	**0.3513**	**0.0838**	**0.433**
Baseline					0.0820	0.1865	0.0514	0.4046

Table 3.3: Evaluation results of the 2014 and 2016 topics used as a training and testing sets respectively

statistical features only. In term of ndcg@10 measure, the result increases from 0.1082 to 0.1290. However, when we add the field features and profile features we obtain significantly lower ndcg@10 (0.1084 and 0.1077). We can also clearly mention that combining all features gives the best results in term of ndcg@10, MRR and MAP compared to all other combinations of features. It has 91.09% improvement on ndcg@10, 88.36% on MRR and 63.03% on MAP compared with the baseline. Finally, we can say that our approach has advantage because all the combinations of features perform better than the baseline in term of ndcg@10. For all combinations ndcg@10 is superior than 0.1077 while ndcg@10 of the baseline is 0.0820.

3.4.2 A Combination of reduction and expansion for natural language queries

In this section, we explain our second proposed approach for dealing with natural language queries in social book search. At first, this approach was proposed for our participation in social book search 2015 [6] [26]. Following its success where we won second place and the important impact of statistic and similar-book features described in the previous section. Hence, we further expand the experiments to include all SBS topics from (2011 to 2016). The proposed approach is based on the following key intuitions:

The more frequently a term appears in many queries, the less informative it is whilst the more times this term appears in the same query, the more relevant this term is to the query context.

Terms present in the set of similar books mentioned by users in their queries could be relevant to the query context.

According to the two intuitions mentioned above, our approch includes three techniques: (i) stopword removal technique to reduce the verbose queries, (ii) new statistical measure tf.iqf to weighting terms and (iii) the expansion of this query using similar books mentioned in the topic.

3.4.2.1 Stopword removal for query reduction

The work of [90] shows that constructing a specific stopword list of a given collection can improve the performance results. In their works, all terms of the collection are ranked according to their importance in the collection. Then they choose a threshold and any words that appear above the particular threshold are treated as stop-words and will not be indexed. In the same way, [62] proposed to chose to apply these technique by removing from the query all words which occur on the stopword list. Both works utilize the statistic measure IDF to rank terms and decide which term is a stopword or not. Unlike the approaches discussed above, which take into account the number of documents in which the term appears. Our approach uses the number of queries in which the term appears instead. First, we ranked the terms according to the number of times they appear in the

[6] http://social-book-search.humanities.uva.nl/#/suggestion15

Stem of term	#topics	Stem of term	#topics
The	966	But	534
And	895	About	438
Book	816	With	432
For	803	Thi	420
Read	627	Anyon	417
That	600	Suggest	416
Ani	600	Look	414
Recommend	571	Can	381
Have	567	Like	375

Table 3.4: Top ranked stems of terms with the corresponding number of queries in which they appear

queries. Then we choose a threshold and all the terms that appear above will construct a stop-words list. Finally the terms of the list will be removed from each query in order to form a reduced query (RQ) before querying by an IR model. Table 3.4 shows a sample of the top ranked terms with the corresponding number of queries in which they appear.

3.4.2.2 Query terms weighting

The frequency of terms in the query, in the corpus or in the external resources are commonly used by researchers to weight terms on verbose queries [72, 130]. In our case, and based on the first intuition mentioned in section 3.4.2 and inspired from the commonly used term weighting $tf.idf$ [118], we have introduced a new measure Term Frequency-Inverse Query Frequency (tf.iqf). This measure was used in order to increase the weight of terms which appear the less in the set of queries and decrease the weight of terms which appear the most in the set of queries. The tf.iqf measure is calculated as follow:

$$TFIQF(t) = tf(t,q).iqf(t) \tag{3.1}$$

Where *tf(t,q)* is the frequency of term t in the topic q, and the *iqf(t)* is the inverse query frequency calculated as follow:

$$iqf(t) = \frac{log(|Q| - qf(t) + 0.5)}{(qf(t) + 0.5)} \tag{3.2}$$

Where qf(t) is the number of topics that contain t, and |Q| is the total number of topics in a collection (all topics from INEX 2011 to 2016 are used).

Year	#Topics	Fields
2011	211	Title,Group,Narrative,type,genre,specificity
2012	96	Title,Group,Narrative,type,genre
2013	370	Title,Group,Narrative,Query
2014	672	Title,Group,Narrative,mediated_query
2015	178	Title,Group,Narrative,mediated_query
2016	119	Title,Group,Request

Table 3.5: Details of the six years topics used for the experiment.

3.4.2.3 Query expansion

At first glance,the expansion of verbose queries seems counterintuitive. Nevertheless,query expansion has improved performance in such queries. Various query expansion techniques have been developped like adding a category labels to long queries [5], adding latent concepts extracted from pseudo-relevance feedback [11] or using multiple information sources [12] and interactive query expansion using pseudo-relevance feedback [80] In our case, and assuming that similar books mentioned by users in their topics are relevant, their terms are also important to the query. In order to exploit these similar books, we expand the weighted reduced query (WRQ) by automatically adding terms from these similar books. Rocchio relevance feedback [113] is one of the most popular techniques used for this task. The function used to expand the queries is as follow :

$$\overrightarrow{EWRQ} = \overrightarrow{WRQ} + \beta \sum_{d \in similar\_books} \overrightarrow{d} \tag{3.3}$$

Where $\overrightarrow{EWRQ}$ and $\overrightarrow{WRQ}$ are the expanded wheighted reduced query and the weighted reduced query vector respectively, $\overrightarrow{d}$ denotes the weighted term vector of the similar book d.

3.4.3 Experimental results

As mentioned-above the document collection used in our experiments has a total of 2.8 million of records extracted from Amazon and Librarything Forum. For each book the professional metadata was ignored and only the social information data (tags assigned to books by users in LT Forum) were used to represent books. As regards topics, from 2011 to 2016 the organizers of SBS have used Librarything forum to extract a different set of topics with relevance judgments for each year. Table 3.5 summarizes the set of

topics utilized for each year with some details. The same table shows that the majority of topics contain the three fields (title, group and narrative), while the 2011 and 2012 topics contain (type, genre and specificity) which are ignored in our experiments. In 2013 and 2014, the field entitled query (2013) and mediated query (2014) are just the same and are created manually by a trained annotator. This field is provided by the organizers of SBS to compensate for non-representative thread titles for some of the forum topics. The example field which contains a list of books that are related to the topic is present in all years. The Terrier IR platform [102] was used to index the collection by applying basic stopword filtering and Porter stemming algorithm. The BM25 model was used for querying with the parameters (b=0, k3=1000, k1=2). Using the BM25 model, the relevance score of a book d for query Q is given by:

$$S(d,Q) = \sum_{t \in Q} \frac{(k_1+1)w(t,d)}{k_1 + w(t,d)} . idf(t) . \frac{(k_3+1)w(t,Q)}{k_3 + w(t,Q)} \tag{3.4}$$

Where w(t,d) and w(t,Q) are respectively the weigths of terms in document d and in query Q. idf(t) is the inverse document frequency of term t, given as follow:

$$idf(t) = \frac{log(|D| - tf(t) + 0.5)}{(df(t) + 0.5)} \tag{3.5}$$

Where *df(t)* is the number of documents tagged with *t*, and |D| is the number of documents in the collection.

In the first step of our experiments we build a three set of queries according to the topic field used as a query namely: title only, narrative only and title+narrative. The results are presented in table 3.7. The table shows the modest results obtained from narrative representation of the query were caused by its verbosity. However the user used it to explains their need which means that it contains some informative terms. Thus, we combined the title with the narrative as a representation of the queries and we generated a stop-words list to reduce these queries. To do so, we varied the threshold values (number of queries in which the term appears), in order to find one particular set of stop-words list that would produce a better ndcg@10. The threshold was varied from 5 to 60, with 5 steps. After several experiments on all six years topics, Table 3.6 shows the results obtained. From this table we notice that query reduction by removing stop words from queries performs better,whatever the value of the threshold, than the case of no reduction (column title+narrative in table 3.7). However, the same table indicates that most of

these different sets of topics achieve the optimal results when the threshold is set to 20 or 30. The model achieves the optimal results across all queries in term of ndcg@10 (0.1429) when the threshold is set to 30. Then we chose setting the threshold to 30 for the remaining experiments. This means that each term that appears in more than thirty topics is considered as a stop word and must be removed from queries.

Thirdly, we used the $tf.iqf$ function 3.1 in section 3.4.2.2 to weight terms in the reduced query. The result of this step gave a new reduced query with weighted terms. Finally, the weighted reduced query was expanded by adding new terms from similar books using the function 3.3 in section 3.4.2.3. The rocchio function was used with their default parameter settings β=0.4, and the number of terms selected from each similar book was set to 10. Table 3.8 presents the results of the whole process. Where the columns RQ, WRQ and EWRQ represent the results obtained by the reduced query, weighted reduced query and expanded weighted reduced query respectively. Figure 3.3 shows the different representations as well as the results obtained for the topic 107277 in term of ndcg@10 and map. From the table, we show the improvement of the results

Threshold	5	10	15	20	25	30	35	40	45	50	55	60
2011	0.214	0.2308	0.2376	**0.2421**	0.2385	0.2362	0.233	0.2367	0.2344	0.2291	0.2222	0.2173
2012	0.1734	0.1777	0.183	**0.196**	0.193	0.1921	0.1935	0.194	0.1947	0.1898	0.187	0.1818
2013	0.1102	0.1277	0.1325	0.1299	0.1327	**0.1342**	0.1296	0.1266	0.1247	0.1261	0.126	0.1255
2014	0.106	0.1179	0.1209	0.1212	0.1218	**0.1257**	0.1181	0.1171	0.1133	0.1129	0.1128	0.1108
2015	0.0856	0.0885	0.0944	0.0937	0.0933	**0.0997**	0.0935	0.0934	0.094	0.0891	0.0873	0.0892
2016	0.1198	0.1283	0.1298	**0.1338**	0.1286	0.1264	0.1259	0.125	0.1227	0.12	0.1125	0.1117
All	0.1235	0.1356	0.1399	0.141	0.1408	**,0.1429**	,0.1377	0.1371	0.1347	0.1332	0.1313	0.1296

Table 3.6: Results in term of ndcg@10 obtained for different values of the thresholds for query reduction.

Year	Title only	Narrative only	Title+Narrative
2011	0.2567	0.0715	0.117
2012	0.1804	0.0526	0.1025
2013	0.1158	0.0483	0.0719
2014	0.1167	0.0506	0.0717
2015	0.1164	0.0442	0.0615
2016	0.1145	0.037	0.0595
ALL	0.138	0.0512	0.0774

Table 3.7: The comparison in terms of ndcg@10 of the three representation of the query

3.4.4 Results and analisys

In this section, we present the findings of our experiments across all six years topics. Table 3.7 shows that using only the title field of topics as a query gives better results comparing to when using narrative only or the combination of title and narrative. This reinforces what has been already stated namely that search engines generally perform poorly on verbose queries when compared to short keyword queries. We consider the results obtained by the title field of the topic as a baseline model. We performed the three reformulation of the query (reduced query, reduced weighted query and expanded weighted reduced query). Table 3.8 shows the results obtained by our approach for the three reformulations of the query. For the query reduction we show that this technique outperforms the baseline in only four of the six sets of topics. However, for all queries the ndcg@10 increases from 0.1380 to 0.1429. From the same table we show that when applying both weighing function $tf.iqf$ technique as well as the query expansion technique the results are better across all the sets of queries. In term of ndcg@10 the results increase from 0.1429 to 0.1561 when applying the term weighting technique and from 0.1561 to 0.1790 when using the query expansion technique.

We further compare the performance of our best combination (EWRQ)to the best official runs of the six years in Social Book Search. Table 3.9 represents the comparative results. The results show that the ndcg@10 value of our best results is better than the best runs of the four years (2011, 2012, 2013 and 2014) but lower than the best runs of the two years(2015 and 2016). The table shows also that our approach outperforms the best runs across all topics ndcg@10=0.1790 of our approach against ndcg@10=0.1726 for the best runs.

In order to report more informatively the effect of reduction, weighting and expansion

Year	Baseline model	RQ	WRQ	EWRQ
2011	0.2567	0.2362	0.2675	**0.3291**
2012	0.1804	0.1921	0.2036	**0.2210**
2013	0.1158	0.1342	0.1444	**0.1524**
2014	0.1167	0.1257	0.1336	**0.1565**
2015	0.1164	0.0997	0.1176	**0.1380**
2016	0.1145	0.1264	0.1411	**0.1500**
ALL	0.138	0.1429	0.1561	**0.1790**

Table 3.8: The comparison in terms of ndcg@10 of the combination of the three techniques (weigthing, reducing and expansion)

	Our approch	Best run
2011	**0.3291**	0.3101
2012	**0.2210**	0.1456
2013	**0.1524**	0.1361
2014	**0.1565**	0.142
2015	0.1380	**0.1870**
2016	0.1500	**0.2157**
All	**0.1790**	0.1726

Table 3.9: The comparison in terms of ndcg@10 of our best combination (EWRQ) to the official best runs of each year submitted to Social Book Search

techniques, the statistical significant testing (two-tailed t-test) has been conducted. As shown in table 3.10 the reducing technique (RQ) is not statistically significant compared to the baseline (p-value = 0.23) , this due that in the reduced query we used the title and narrative while in the base line only the title were used. From the same table we show that combining reducing and weighting techniques (WRQ) is statistically significant compared to baseline (p-value=0.0038) but less significant compared to RQ. Combining all techniques (EWRQ) is statistically significant compared to the baseline (p-value = 0.000002) or to reducing technique (p-value=0.003) , but less significant compared to WRQ (p-value = 0.0601), this could be due that similar books does not present in all topics and only the topics that contains similar books have had improvements.

Method1	Method2	p-value
Baseline Model	RQ	0.2364
Baseline Model	WRQ	**0.0038**
Baseline Model	EWRQ	**0.000002**
RQ	WRQ	0.0896
RQ	EWRQ	**0.0003**
WRQ	EWRQ	0.0601

Table 3.10: The p-value obtained using the Statistical Significance (two-tailed T-Test), figure in bold indicate a significance of the difference between two methods at level 0.05

title	fantasi :1 book :1 creativ :1 heroin :1	0.0000	0.0233
Narrative	heroin :1 like :0.6 can :0.6 look :0.4 suggest :0.4 fantasi :0.4 creativ :0.4 wai :0.4 talent :0.4 imagin :0.4 see :0.4 ordinari :0.4 stori :0.4 greet :0.2 novel :0.2 sort :0.2 art :0.2 music :0.2 literatur :0.2 ve :0.2 seen :0.2 share :0.2 tough :0.2 gal :0.2 who :0.2 know :0.2 swing :0.2 sword :0.2 throw :0.2 punch :0.2 next :0.2 genr :0.2 ann :0.2 shirlei :0.2 jo :0.2 march :0.2 juliet :0.2 marilli :0.2 favorit :0.2 author :0.2 make :0.2 point :0.2 give :0.2 creat :0.2 clodagh :0.2 heir :0.2 sevenwat :0.2 exampl :0.2 mai :0.2 be :0.2 pure :0.2 domest :0.2 plai :0.2 harp :0.2 compos :0.2 song :0.2 creidh :0.2 foxmask :0.2 read :0.2 weav :0.2 color :0.2 sorcha :0.2 daughter :0.2 forest :0.2 liadan :0.2 son :0.2 shadow :0.2 storytel :0.2	0.0000	0.0083
title+narrative	heroin :1 fantasi :0.5 creativ :0.5 like :0.5 can :0.5 look :0.33 suggest :0.33 wai :0.33 talent :0.33 imagin :0.33 see :0.33 ordinari :0.33 stori :0.33 book :0.17 greet :0.17 novel :0.17 sort :0.17 art :0.17 music :0.17 literatur :0.17 ve :0.17 seen :0.17 share :0.17 tough :0.17 gal :0.17 who :0.17 know :0.17 swing :0.17 sword :0.17 throw :0.17 punch :0.17 next :0.17 genr :0.17 ann :0.17 shirlei :0.17 jo :0.17 march :0.17 juliet :0.17 marilli :0.17 favorit :0.17 author :0.17 make :0.17 point :0.17 give :0.17 creat :0.17 clodagh :0.17 heir :0.17 sevenwat :0.17 exampl :0.17 mai :0.17 be :0.17 pure :0.17 domest :0.17 plai :0.17 harp :0.17 compos :0.17 song :0.17 creidh :0.17 foxmask :0.17 read :0.17 weav :0.17 color :0.17 sorcha :0.17 daughter :0.17 forest :0.17 liadan :0.17 son :0.17 shadow :0.17 storytel :0.17	0.0000	0.0100
RQ	heroin :1 creativ :0.5 talent :0.33 imagin :0.33 ordinari :0.33 greet :0.17 art :0.17 music :0.17 seen :0.17 share :0.17 tough :0.17 gal :0.17 swing :0.17 sword :0.17 throw :0.17 punch :0.17 ann :0.17 shirlei :0.17 jo :0.17 march :0.17 juliet :0.17 marilli :0.17 creat :0.17 clodagh :0.17 heir :0.17 sevenwat :0.17 pure :0.17 domest :0.17 plai :0.17 harp :0.17 compos :0.17 song :0.17 creidh :0.17 foxmask :0.17 weav :0.17 color :0.17 sorcha :0.17 daughter :0.17 forest :0.17 liadan :0.17 son :0.17 shadow :0.17 storytel :0.17	0.2148	0.0276
WRQ	heroin :1 creativ :0.6 talent :0.43 ordinari :0.42 imagin :0.32 marilli :0.26 creidh :0.26 clodagh :0.26 weav :0.26 domest :0.26 harp :0.26 liadan :0.26 sorcha :0.26 compos :0.26 sevenwat :0.26 foxmask :0.26 storytel :0.24 pure :0.24 swing :0.24 gal :0.24 punch :0.24 march :0.24 juliet :0.24 shirlei :0.23 heir :0.21 throw :0.21 forest :0.21 shadow :0.21 greet :0.21 sword :0.21 song :0.2 plai :0.18 tough :0.18 color :0.18 music :0.17 creat :0.16 son :0.16 ann :0.16 seen :0.16 art :0.15 daughter :0.15 share :0.15	0.4714	0.0575
EWRQ	sevenwat :1.26 heroin :1 fantasi :0.75 creativ :0.6 marilli :0.5 talent :0.43 ordinari :0.41 juliet :0.4 celtic :0.38 ireland :0.38 fairi :0.37 swan :0.36 tale :0.34 imagin :0.32 creidh :0.26 clodagh :0.26 weav :0.26 domest :0.26 harp :0.26 liadan :0.26 sorcha :0.26 compos :0.26 foxmask :0.26 storytel :0.24 pure :0.24 swing :0.24 gal :0.24 punch :0.24 march :0.24 trilogi :0.23 shirlei :0.23 heir :0.21 throw :0.21 forest :0.21 shadow :0.21 greet :0.21 sword :0.21 song :0.2 histor :0.19 plai :0.18 tough :0.18 color :0.18 music :0.17 seri :0.17 creat :0.16 son :0.16 ann :0.16 retel :0.16 seen :0.16 art :0.15 daughter :0.15 share :0.15 mytholog :0.14 read :0.11 magic :0.11 fairytal :0.1 romanc :0.1 retold :0.1 fiction :0.09 folklor :0.07 fae :0.07 love :0.07 palencar :0.07 druid :0.06 faeri :0.06 folk :0.06 irish :0.06 myth :0.06 romant :0.06 femal :0.06 tbr :0.06 strong :0.06 australian :0.05 wild :0.05 specul :0.05 keltisch :0.05 jude :0.05 unread :0.05 adventur :0.05 fi :0.05 sci :0.05	0.5959	0.0907

Figure 3.3: The diffrent representations of the topic 107277 obtained from the deffrent techniques used in our approch as well as the results obtained

3.5 Conclusion

In this chapter, we have proposed two techniques in order to deal with the issue of long natural language queries submitted by users in online social forums. The first approach used several categories of features to represent query terms namely, statistical, linguistic, fields, profile and example features and a learning to rank algorithm has been used to weight and rank terms of the query and then select only the top important. As for the second approach, a combination of reduction and expansion has made to suppress the extraneous terms and keep only the important that match user' needs. We automatically generate a stopword list in order to reducing verbose queries. Moreover, we introduced a new function, called $tf.iqf$, inspired by the $tf.idf$ measure, to weight terms and measure how informative a given term is. In addition, the rocchio technique was used to add new terms from the similar books mentioned by users in their topics. To validate these proposed techniques, we used a complex test collection provided by social book search lab in which the topics are a real-world information needs obtained from the discussion threads of LibraryThing forum. The results of the two approaches were submitted for participation in SBS 2015 and 2016 suggestion track and our team was ranked second. The experimental results demonstrate that the proposed approaches has given satisfactory results with all the techniques that have been used. From the results we conclude that query representation is an important aspect in social book search. This is confirmed by the organizers of SBS: *From these results it seems clear that topic representation is an important aspect in social book search. The longer narrative of the request field as well as the metadata in the user profiles and example books contain important information regarding the information need, but many terms are noisy, so a filtering step is essential to focus on the user's specific needs.* [75]

Chapter 4

Combining tags and reviews to improve social search performance

4.1 Introduction

The emergence of Web 2.0 and social media have provided important amounts of information that led researchers in social information retrieval to exploit it to improve the information retrieval performance. This new information necessitates an extended model for information retrieval, as well as new techniques that make use of it. This information can be in different form, for example, textual like tags or reviews, or non textual like ratings, number of likes, number of shares, etc. As it seems logical that integrating tags in the retrieval model should not be in the same way taken to integrate reviews. Hence, this chapter describes a simple way of adapting the BM25 ranking function to deal with the different types of social information that represent the books. We will analyse the different influences of using tags and reviews to represent documents on both the settings of retrieval parameters and the retrieval effectiveness. Thus, we index each social information type in a separately index and analyse the sensitivity of the model parameters by taking into consideration both the document and query representations. Afterwards, we choose the optimal parameters of the models and combining them by a linear function to build one model. After several experiments, on social book search collection, we concluded that the parameters of the model are sensitive to book representation. Consequently, combining the results obtained from two separate indexes and two models with specific parameters for each of them gives good results compared to when using a single index and a single model. The results obtained in this work have been published in [29].

4.2 Motivations

Since its first edition in 2011, participants of SBS have proposed approaches and submitted their results (runs) to get them evaluated. The majority of these approaches use textual information (tags and reviews) to estimate the initial score of books then non-textual information like (rating, popularity, number of tags, number of reviews, profile of user, etc) is used to re-rank the initial ranking and improve the initial text-based search results.

The authors in [16] used the language modeling with Jelinek-Mercer (JM) smoothing to build the initial content based results. They also experimented different re-ranking approaches using different information sources, such as user ratings, tags, authorship, and Amazon's similar products. The results show that the re-ranking approaches are often successful.

In [46, 137], the probability of the query content produced by the language model is used to rank the documents based on textual information. Ratings, number of reviews, popularity and high frequent books are then used to re-rank the initial ranking. Finally, random forest was used to learn the different combinations of scores and the results are better than the initial ranking.

In [53] the BM25F model was used to optimize the weight of the four book fields (title, summary, tags, reviews). Popularity, reputation, ratings and similarity between users were used to improve the results however, their integration did not give any improvement. [64] employ the textual model BM25 and enhance it by using social signals such as rating. Finally, they applied a random forest learning to improve the results by including non-textual modalities like price and number of pages according to the user's preferences. This approach improves the results and shows good performances.

The authors in [13] used two textual retrieval model Divergence from randomness model(DFR) and Sequential Dependence Model(SDM) before combining the results with the ratings of books. The best results were obtained when using DFR model with textual information only. In [26–28] (works of chapter 3), we used only tags as textual information about books. we investigate the representation of the query by transforming the long verbose queries to a reduced queries before applying the BM15 retrieval model. The result of these approaches show also good performance despite not using Amazon reviews at all.

To sum up, the majority of these works used textual information as a baseline before using non-textual information for reranking. As for indexing, they used users' tags and reviews together to represent books. However, some of them [16, 53, 74, 77] studied the influence of using tags only, reviews only or tags with reviews (in a single index) as books

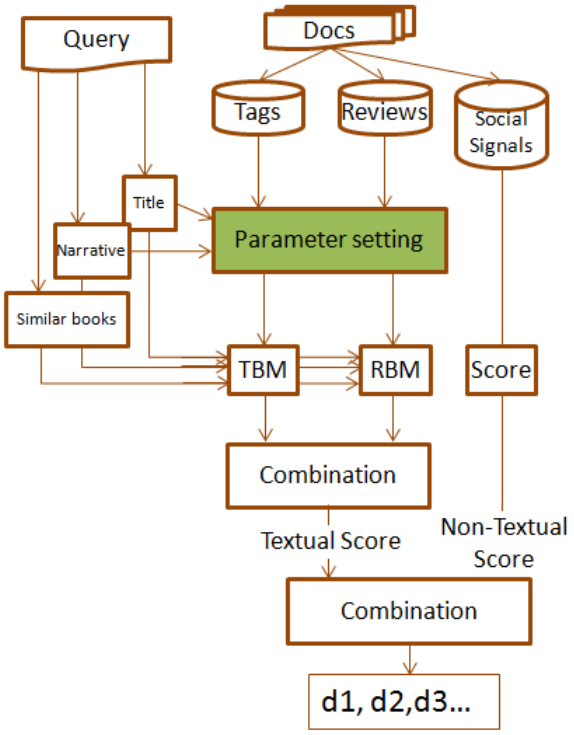

Figure 4.1: Scheme of our approach

representation on the retrieval performance. To our knowledge, there is no study that has adapted a specific model for each document representation. Hence, for works that combining tags and reviews, [16, 53, 74, 77] the authors used the same retrieval model to compute scores and rank documents for each query.

In this chapter, we use both textual and non textual information of books as used in the studies mentioned above. However, instead of using one index for all textual fields and one function to compute scores of documents, we build a separate index for each field. We build two models, Tag Based Model (TBM) and Review Based Model (RBM) which respectively use the index of tags and the index of reviews. Then, we analyse the different influences of using tags and reviews on the settings of retrieval parameters as well as on the retrieval effectiveness. Then, we combine the results of the two models to form the textual score. Finally, we combine textual and non textual score to form the final result. The overview of our approach is presented in Figure 4.1.

Number of books	2,781,400
Number of books that have been reviewed at least once	1,915,336
Number of books that have been tagged at least once	2,306,368
Number of tokens in reviews	1,161,240,462
Number of single terms in reviews	1,135,910
Number of tokens in tags	246,552,598
Number of single terms in tags	194,487

Table 4.1: Statistics on tags and reviews of SBS collection

4.3 The proposed approach

In this approach, we have used the collection of social book search as described in chapter 1 section 2.5.1. For documents, we have considered two kinds of representations : user tags from LibraryThing and users' reviews from Amazon. Table 4.1 summarizes the statistics of the collection. The numbers of tokens and single terms are calculated after stopword removal and Porter stemming. This table shows that the number of books that have been tagged is greater than the number of books that have been reviewed however, the number of tokens (all occurrences) in reviews are greater than those of tags. This because, for reviews the users use natural language to give their opinions and speak freely about books. However for tags the users assign a few keywords or terms to describe books.

As to queries, we have used all the six-year topics (1646 topics) provided by SBS from 2011 to 2016. Because each query is composed of title and narrative, we have considered two types of queries: short and long queries. The short query is constructed from the title of the topic while the long query is the narrative and title+narrative.
The prime target of our study is to investigate the impact of using the different query representations as well as the two representations of documents on the retrieval performance. To achieve this, we built three indexes, the first contains tags assigned by users to books in LT, the second contains reviews extracted from Amazon while the last one merges tags and reviews in the same index. The Terrier IR platform [103] was used to index the collection by applying basic stopword filtering and Porter stemming algorithm. The BM25 model [112] was used for querying. Using the BM25 model, the relevance score of a book d for query Q is given by:

$$S(d,Q) = \sum_{t \in Q} \frac{(k_1+1)tf_{td}}{tf_{td} + k_1(1 - b + b.\frac{|D|}{avgdl})}.idf(t).\frac{(k_3+1)tf_{tq}}{k_3 + tf_{tq}} \tag{4.1}$$

Where $\mathrm{tf_{td}}$ and $\mathrm{tf_{tq}}$ are respectively the frequency of term t in document d and in query

Q. the three free parameters of the function are : k_1 and k_3 that respectively controls term frequency scaling of the document and the query, the parameter b controls the document length normalization. $idf(t)$ is the inverse document frequency of term t, given as follow:

$$idf(t) = log\frac{|D| - df(t) + 0.5}{df(t) + 0.5} \tag{4.2}$$

Where *df(t)* is the number of documents where the term t appears, and $|D|$ is the number of documents in the collection.

4.3.1 Length normalization vs document and query representation

Document length normalization is a technique that attempts to adjust the term frequency or the relevance score in order to normalize the effect of document length on the document ranking. Several works [38, 54, 92] show that this technique has an important impact on the performance of the model.

In order to determine the sensitivity of the model performance as to the length normalization, using the different cases of document representations as well as query representations, we have set the BM25 standard parameters for $k1$ and $k3$ ($k1 = 2$, $k3 = 1000$) and varied the length normalization parameter b from 0 to 0.75 (in steps of 0.05) then we evaluated the results in terms of ndcg@10 on both indexes (tags and reviews). The results obtained for all queries together are shown in Figure 4.2.

From Figure 4.2, we can see that in TBM, the performance of the model is very sensitive to the normalization of the length of the document. ndcg@10 drops from 0.1415 (b=0) to 0.0375 (b=0.75) in the case of title (short queries). It increases from 0.0368 (b=0) to 0.0961 (b=0.05) then drops to 0.0321 (b=0.75) when using narrative (long queries).The same sensitivity of the model was shown When combining title and narrative to represent the query, ndcg@10 increases from 0.0695 (b=0) to 0.1429 (b=0.1) and then drops to 0.0490 (b=0.75).

In the case of RBM, the model performance is not very sensitive to the normalization of the document length compared to TBM. The values of ndcg@10 are very close so there is no big difference between them especially when varying b from 0 to 0.35. ndcg@10 is 0.1096 (b=0) and 0.1003 (b=0.35) in the case of short queries. The same measure increases from 0.0731 (b=0) to 0.1042 (b=0.25) and then decreases to 0.0803 (b=0.75) in the case of long queries. In the same way, the ndcg@10 increases from 0.0957 (b=0) to 0.1370

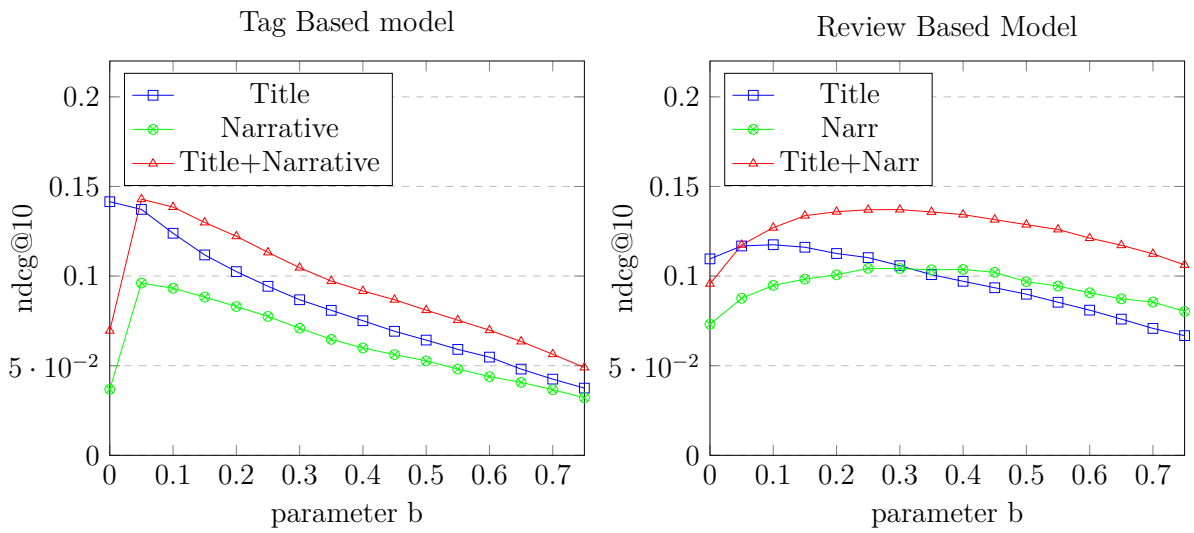

Figure 4.2: Sensitivity of ndcg@10 for length normalization for Tag based model and Review based model.

(b=0.25) and then decreases to 0.1062 (b=0.75) when combining title and narrative.

The same figure clearly shows that in short queries the best performance is obtained when b is very small, b=0 for the tags (no length normalization is required) and b=0.1 for reviews. However, when using long queries we find that the best performance is obtained when b=0.1 for tags and b=0.25 for reviews (length normalization is required). We can conclude that the setting parameters of the model are not the same in the TBM model or in the RBM model. Each of them has its specific parameters.

Because these evaluations are obtained for all queries together and to make our results more meaningful, we have decided to learn the parameter b by selecting the set of topics of each year as the testing set and the remaining sets of topics (of other years) as the training set. The evaluation in term of ndcg@10 of the results, for Single index(b=0.3), TBM model (b=0.05) and RBM Model(b=0.25), obtained after training and testing is shown in Table 4.2.

As shown in Table 4.2, TBM gave the best results, compared to RBM and Single index, except for 2011 when RBM gave the best results. By the way, we were surprised that, in most cases, using tags only (TBM) or reviews only (RBM) to index documents gave better results compared to when using tags and reviews together. The only exceptional case was in 2011 where single index (ndcg@10=0.2810) was better than TBM (ndcg@10=0.2459).

Year	Single Index	TBM	RBM
2011	0.2810	0.2459	**0.3007**
2012	0.1387	**0.2012**	0.1479
2013	0.1143	**0.1193**	0.1191
2014	0.1169	**0.1291**	0.1175
2015	0.0682	**0.1222**	0.0794
2016	0.0803	**0.0951**	0.0906
All	0.1307	**0.1429**	0.1370

Table 4.2: Results in term of ndcg@10 of the three index , using Title+Narrative as a representation of the topic; best results are shown in bold.

4.3.2 Impact of document representation on query expansion

Following the encouraging results of query expansion on the retrieval performance obtained in the previous chapter, in this chapter, we also use this technique to extract the highly relevant terms of similar books and expand the original query for the three indexes. That in order to show the impact of query expansion for the three different models. The Rocchio function [114] used to expand the query is as follows:

$$\overrightarrow{Q}_{new} = \overrightarrow{Q}_{T+N} + \frac{\beta}{|EXP|} \sum_{d \in EXP} \overrightarrow{d} \tag{4.3}$$

Where $\overrightarrow{Q}_{new}$ is the expansion query, $\overrightarrow{Q}_{T+N}$ is the original query represented by Title+Narrative. $\overrightarrow{d}$ denotes the weighted term vector of the example book d using the default term weighting model Bo1(Bose-Einstein 1). EXP is the set of example books and $/EXP/$ is the number of example books mentioned in the topic. The function was used with their default parameter settings β =0.4, and the number of terms selected from each example book was set to 10.

Table 4.3 shows the results after the query expansion for the Single index, TBM model as well as for the RBM model. The same table shows that the query expansion technique has improved the results for the three indexes. We also notice that TBM gave the best results for all cases except for 2011 when RBM gave the best ndcg@10 (0.3418) and for 2013 when Single index has an ndcg@10=0.1496.

4.3.3 Combining the scores of the two models

Once the parameters of TBM and RBM are optimized, a combination of the two scores obtained is necessary to obtain the final textual score of each book with respect to the

Year	Single Index		TBM		RBM	
	Before expansion	After expansion	Before expansion	After expansion	Before expansion	After expansion
2011	0.281	0.3310	0.2459	0.2725	0.3007	**0.3418**
2012	0.1387	0.1667	0.2012	**0.2341**	0.1479	0.1754
2013	0.1143	**0.1496**	0.1193	0.1452	0.1191	0.1478
2014	0.1169	0.1436	0.1291	**0.1525**	0.1175	0.1423
2015	0.0682	0.0887	0.1222	**0.1409**	0.0794	0.0907
2016	0.0803	0.1088	0.0951	**0.1367**	0.0906	0.1205
All	0.1307	0.1619	0.1429	**0.1686**	0.1370	0.1639

Table 4.3: NDCG@10 results obtained after applying a query expansion technique.

query. The linear combination function is as follows:

$$S(d,Q) = \alpha.S_{TBM}(d,Q) + (1-\alpha).S_{RBM}(d,Q) \tag{4.4}$$

Where $S(d,Q)$ is the final score of document d with respect to query Q. $S_{TBM}(d,Q)$ and $S_{RBM}(d,Q)$ are the scores of document d with respect to query Q respectively obtained from TBM and RBM models. The query Q is represented by Title+Narrative and expanded using similar books as explained in the previous section. α [0 1] is a free parameter that controls the weight of the two models.This parameter was tuned using six-fold cross-validation in the same way it was performed to tuned the parameter b as indicated in section 4.1. Thus, a single set of topics of one year is used for testing the model and the remaining five-year topics are used as a training set. After the process is repeated six times (once for each year), the results have been summed up in Table 4.4. This table shows the results in term of ndcg@10 obtained from the combination and compares them to the results obtained from the previous experiments. The best results of the combination are obtained when $\alpha \in$ is set to 0.4.

From the results, we note that for all years the combination of the two scores gave good results, compared to the results of each index. For all queries, the ndcg@10 increased from 0.1619 (single model), 0.1686 (TBM model) and 0.1639 (RBM model) to 0.2091 when combining the two scores so there is an improvement of 29.15%, 24.02% and 27.54% compared to the three models. This results shows that the technique of using two separate indexes and combining the results is an effective technique to get best results.

Year	Single Index	TBM	RBM	Combination
2011	0.3310	0.2725	0.3418	**0.3595**
2012	0.1667	0.2341	0.1754	**0.2425***
2013	0.1496	0.1452	0.1478	**0.1888***
2014	0.1436	0.1525	0.1423	**0.1886***
2015	0.0887	0.1409	0.0907	**0.1526***
2016	0.1088	0.1367	0.1205	**0.1793***
All	0.1619	0.1686	0.1639	**0.2091***

Table 4.4: NDCG@10 obtained by the combination after tuning the parameter α using six-fold cross-validation compared to a single index, RBM model and TBM Model. Asterisks indicate statistically significant differences compared with single index (Student's t-test, $P < 0.05$)

4.3.4 Non textual information to re-rank documents

The non textual information of documents in social media like number of likes, number of rating, number of times the document was catalogued or rated represents an important information and can be used to re-rank the documents to improve the results. Several Re-ranking approaches were proposed by [137] at INEX2014 and [16] in 2012 , which proved to be effective. Thus, we combine the textual score obtained above with the the number of times the book was rated to re-rank the documents. The combination of scores is calculated by the flowing function:

$$S(d,Q) = \lambda.S_{Textual}(d,Q) + (1-\lambda).S_{Non-textual}(d) \tag{4.5}$$

Where $S(d,Q)$ is the final score of document d with respect to query Q. $S_{Textual}(d,Q)$ is the textual score of document d obtained by combining the scores of TBM and RBM as explained in the previous section. λ [0 1] is a free parameter that controls the weight of the two scores. $S_{Non-textual}(d)$ is the normalized non-textual score of document d calculated as follow:

$$S_{Non-textual} = \frac{Nb\_rated - Min\_nb\_rated}{Max\_nb\_rated - Min\_nb\_rated} \tag{4.6}$$

Where nb_rated is the number of times the document is rated, Min_nb_rated and Max_nb_rated is the minimum and the maximum of the number of times that all books of the collection have. Table 4.5 shows the results obtained when using the non-textual information for re-ranking. The best results obtained when $\lambda \in$ was set to 0.9. From this Table we show that when using the Non-Textual information to re-rank documents gave

	Textual score only	Textual and Non-Textual scores
2011	**0.3595**	0.3551
2012	0.2425	**0.2469**
2013	0.1888	**0.1907**
2014	0.1886	**0.1943**
2015	0.1526	**0.1585**
2016	0.1793	**0.1935**
All	0.2091	**0.2133**

Table 4.5: NDCG@10 obtained by the combination of Textual and Non-textual scores

	Our approach	Best non official runs	Best official runs
2011	**0.3551**	0.3423	0.3101
2012	**0.2469**	0.2325	0.1456
2013	**0.1907**	0.1856	0.1361
2014	0.1943	**0.1960**	0.1420
2015	0.1585	**0.2040**	0.1870
2016	0.1935	**0.2157**	**0.2157**

Table 4.6: Comparison between the results of our approach and the best runs submitted to the different years of SBS; best results are shown in bold.

good results of all years of topics, except in 2011 when the ndcg@10(0.3595) obtained by using the textual information only is better than the ndcg@10(0.3551) after re-ranking the documents.

Finally, we compare the performance of our best results, obtained by our approach, to the best official and non official runs. The official runs are those that have been submitted by participants to SBS during the six last years. The non official runs are the results obtained recently in the works of [135, 138]. Table 4.6 represents the comparative results. The results show that the ndcg@10 value of our approach is better than the best official runs of the four years (2011, 2012, 2013 and 2014) but lower than the best runs of the two years (2015 and 2016). This table shows also that our approach gave good results in the three first years compared to the non official runs.

4.4 Analysis

From the results, we have noticed that TBM model requires a smaller value of b for optimal performance whether for short or long queries compared to RBM model. This led

us to ask the following question :

Why document length normalization is required when using reviews as a document representation and not required when using tags as document representation?

As an answer to the question, this may be due to the fact that the users' reviews are a natural language text in which users can repeat freely the same term many times in the same review. Hence, the frequency value of any given term present in the query will be increased thereby increasing the relevance scores of long documents that contain long reviews. That is why document length normalization is required to penalize very long documents. Contrary to the reviews, the users' tags are keywords that users assign them to documents and the same user can not assign the same term many times to the same document. Hence, for long documents when the frequency value of any query term is great this means that this document was tagged by several users using the same term then it is relevant to the topic and it is unreasonable to penalize them. Therefore, the document length normalization is not required for tag representation.

4.5 Conclusion

Since documents can be represented with different kind of social information such as annotations, reviews and non-textual information like popularity, this chapter describes a simple way of adapting the BM25 ranking function to deal with the different social representations of books. We have studied the exploitation of user tags and reviews in social book search as well the sensitivity of the retrieval model to each one. Three indexes have been created, the first for tags, the second for reviews and the third which merges the two. Experiments showed that the Tag Based Model does not require a document length normalization especially for short queries. The best results of this model were obtained when the parameter b has very low values, this may be due to the fact that the number of tags assigned to books by users cannot be regarded as a length of text documents. However, the Review Based Model requires the document length normalization, this may be because the user reviews are long and be seen as a classical textual document. We have noticed that using two indexes for tags and reviews separately and combining the scores of the two models gives better results compared to when using a single index and a single model. We have also shown that the proposed combination has given satisfactory results, especially when using non-textual information to re-rank documents, and outperforms the best official runs submitted to SBS in the four first years from 2011 to 2014.

Chapter 5

Leveraging features extracted from social information to improve the retrieval performance

5.1 introduction

We have seen in previous chapters that social information retrieval has been applied to retrieve many kinds of data such as web pages, videos or books. The works carried out process social reviews in the same way as the text of documents in traditional information retrieval. Therefore, the documents are represented as a bag-of-words (BOW) disregarding features of entities reviewed. The models developed are term-based that tend to retrieve relevant documents based on the keywords issued by the user. However, social user reviews contain a wealth of latent information that could potentially be exploited in a retrieval process. Therefore in this chapter, we aim to leverage users' social reviews for app retrieval in which we take into account app features really required by users, such as functionalities, technical characteristics or characteristics related to the user interface of apps. We propose a term and feature-based approach that, in addition to terms, uses app features extracted from app description and social users' reviews, in order to retrieve the relevant apps to the query and meet the user's needs. The novelty of the proposed approach lies in the use of a representation by features to both apps and queries and the computation of the relevance score between them to get the feature-based score. In addition, our approach combines Feature-based score and Term-based score to get the relevance score of each app. We finally propose an effective techniques that extracts and weight features requested by user

in her query. The Experimental results indicate that the proposed approach is effective and outperforms the state-of-the-art retrieval models for app retrieval.

5.2 Motivations

The use of smartphones and other mobile devices is on the rise. This has led to increase the demand for mobile applications (apps); there were about 28.7 billion downloads in the second quarter of 2019 combined global Apple App Store and Google Play Store [121]. On the other side, the number of apps available for download in Google play store was about 2.9 million (Dec 2019) [120] and was about 2.2 million in Apple app store [119] (Jan 2017). Therefore, an efficient app search system is essential.

The Mobile app platforms allow the app's developers to upload their app and make a detailed description to explain the functionalities and features that the app has. They also allow online users to rate apps out of five stars. Online users can also evaluate apps in the form of reviews; they can write about features provided by these apps or point out the difficulties and problems related to use or installation. This important amount of information has drawn the interest of researchers toward mobile applications.

Mobile applications have drawn attention of researchers in information processing fields in order to analyse the description (information provided by developers) and user reviews (social information provided by online users) to extract useful information like features of applications. [69] defines the feature as the prominent or distinctive visible characteristic or quality of a product. Following [52], and after looking at app description and reviews, we noticed that app features can be any description of specific app functionality visible to the user (eg. "play music", "send message"), a specific characteristic related to the user interface (eg. "large font") , a general quality of the app (eg. "real time"), as well as specific technical characteristics (eg. "gps tracker").

The information extracted from description and reviews can be used to help users to find out the features and advantages of applications or help app developers to update their applications based on the opinion of users with regard to some features. Among these works, the work of [48] proposes a system that can analyze user ratings and comments in mobile app in order to identify reasons why users like or dislike a given app. At first, the authors applied a linear regression to model the relationship between review text and rating, then Latent Dirichlet Allocation (LDA) [15] was used to discover topics that correspond to the root causes of people's concerns toward apps. [52] used Natural

Language Processing (NLP) techniques to identify fine-grained app features in the reviews. Then they extract the user sentiments about the identified features and give them a general score across all reviews. MARA [63] (Mobile App Review Analyzer) is an automatic retrieval of mobile application features requested by users in their comments. A set of language rules has been defined to facilitate the identification of sentences referring to such requests. The authors of [125] proposed a keyword-based framework (MARK) for semi-automated review analysis. MARK allows an analyst describing his interests in one or several mobile apps by a set of keywords. As a response MARK returns the most relevant reviews to each given keyword. Most recently, [84] adopted NLP methods to identify comparative reviews of similar apps to be used to provide fine-grained app comparisons based on different aspects like performance, stability and usability.

Some research works have paid attention to mobile app retrieval [65, 79, 106]. We will give more details of these works in Section 5.3. Their main objective is to retrieve apps that best match a query user using description and reviews as a representation of apps. They rely on Term-based search engine that apply term matching technique to retrieve relevant apps based on the keywords issued by user. However, they lack the consideration of app features which are really required by users. To the best of our knowledge, no research has extracted features from reviews and description and used them in a retrieval system.

Actually, when a user looks for a particular app, she certainly has in her mind, a list of features that this app must provide. To seek this app in a search engine, she expresses her needs through a set of keywords. For instance, the owner of the query "simple notepad large font" requests apps that provide the two features "simple notepad" and "large font". Hence, the retrieval system must return a list of applications that provide these two features. Thus, we are interested to develop a Feature-based app retrieval system that finds applications that provide features requested by user in her query. As per our knowledge, this work [30] is the first to consider the app features requested by users, in their queries, in a search system for mobile apps. To achieve this goal, we propose to extract app features from description and reviews, extract features requested by user from her query and develop a retrieval model that matches between query and apps. The main covered research questions are:

- How can we extract app features from the description and the user reviews?
- How can we automatically formulate the user query from a list of keywords to a list of requested features?

- Can the combination of Feature-based and Term-based models improves the retrieval performance?

5.3 App retrieval

While the general information retrieval task is to retrieve and rank documents for a given query, the app retrieval focuses on apps instead of documents. The problem of Apps searching and retrieval is studied for the first time by [65] who provide a semantic-based search and ranking method for apps. They propose App Topic Model (ATM) in order to discover the latent semantics from app descriptions. To evaluate the relevance of an app with respect to a search query, they combined the textual score of the app (relevance of the app with regard to query terms using the traditional IR technique TF-IDF), the semantic search employing semantic similarity between terms and 12 static quality scores from the three perspectives of app popularity, developer reputation and link prestige. Experiments were carried out using a set of 1000 search queries and 1000 apps crawled from Google Play, Appgravity and AppBrain. [106] is the first work that rigorously studied the app retrieval problem. For that, the authors build a dataset that contains 43,041 App description enriched by 1,385,607 user reviews from Google Play store and 56 realistic queries generated by domain experts based on android forums. Reviews were added to represent applications because reviews and queries are written by users, so there may be less vocabulary gap between them. They performed experiments using state-of-the-art information retrieval models including BM25, query likelihood Language model and proposed a topic model based approach, appLDA. The proposed approach used LDA on both representations, description and user reviews and identified topics in reviews which were also mentioned in the app description. AppLDA achieved a significant improvement in retrieval performance compared to traditional information retrieval models. The same dataset has been used in [79] for mobile app retrieval and categorization. Word embeddings (Relemb) are learned with a neural network architecture using app description only. The learning of word embeddings is carried out based on the notion of relevance (for each word, the information contained in the top retrieved documents are utilized when the same word is used as a query to retrieval engine). Thereafter, the learned word embedding was used to expand queries by adding the top five terms closely related to the initial user query. The proposed word embeddings are effective for query expansion task and eventually improve retrieval effectiveness.

5.4 Feature extraction

Product features extraction from users reviews, has found wide use in recent years [6]. It can be defined as the process that identifies and extracts the set of most relevant features of the product that customers have talked about in their reviews. The objective is to help users to know the characteristics of the product before starting their purchase or help product manufacturers to improve the quality of their product. Since natural language processing (NLP) methods have shown success in extracting information from text like named entity recognition [9], most of the works that have been performed for product features extraction have used NLP techniques for candidate feature extraction followed by the application of some statistical measures in order to keep only the relevant and discard the irrelevant ones. For more details, we refer the reader to [6] which surveys recent works in in the field feature extraction.

One of the most-cited works in this domain is that of [58], in which the authors proposed a mining system that, first conducts POS tagging on the corpus, then runs the association rule miner to find all the frequent noun terms and noun phrases as candidate set. Because not all candidates generated by association mining are useful and in order to improve the precision of the extraction system, two types of pruning are applied (compactness pruning and redundancy pruning): the first method to check features that contain at least two words and the second one to remove redundant features that contain single words. Experimental results show that the proposed approach gives good results in term of recall (80%) and precision (70%).

Only few works used the mobile app reviews to extract the features requested by user or extract the existing features of mobile apps. [52] proposed an approach that produces a fine-grained list of features mentioned in app reviews. The Natural Language Toolkit, NLTK [14], was used for identifying and extracting the nouns, verbs, and adjectives from reviews. Afterwards, the features are extracted by applying a collocation finding algorithm provided by the same toolkit before filtering the less frequent. [128] noticed that most app functionalities are in the form of single nouns, noun phrases (noun+noun), and verb-object phrases (verb + noun, e.g., get direction). Based on this observation, they used the Stanford CoreNLP toolkit [93] to perform text preprocessing (tokenization, Part of Speech (POS) tagging (select noun, verb and adjective) and lemmatization). Finally, they kept only the single nouns, two-gram nouns, and two-gram verb-object that appear more than 10 times, and in less than 80% of the apps as function aspects. Mobile App Review Analyser (MARA) is a prototype that has been developed by [63] in order to

mine app reviews and retrieve the app features requested by users. MARA was designed to collect all the reviews available for a given app, mine the content of the reviews for identifying sentences or fragment of sentences that express feature requests, summarize feature requests, and finally present the results in a user-friendly manner. for feature request extraction, the authors of MARA defined a set of linguistic rules to allow the identification of sentences referring to such requests. Then, they apply LDA to identify topics among the feature request extracted. Similarly, SAFE [68] is an approach that extracts features from app description and users' reviews based on linguistic rules. The authors manually build sentence patterns and part-of-speech patterns that are often used in text referring to app features. Thereafter, these patterns are used to extract features from the app description and reviews.

[89] proposed an approach to automatically mine domain knowledge from App descriptions. By analyzing and summarizing the relationships between structures of sentences and features, they have defined a set of rules and used them to effectively extract features from app descriptions. Specifically, each sentence from a description text is transformed into a parsing tree using Stanford Parser, then the features are extracted from the tree based on the pre-defined rules, and the result is manually evaluated by domain analysts. In order to experiment and test their approach, they collected descriptions from 140 App products covering 7 main stream classes in Google Play, for a total of 2245 sentences.

To conclude, the existing works on features extraction can be used in retrieval systems to enhance their performance. However, none of the reported works uses app features for app retrieval. These gaps will be addressed with our proposed method.

5.5 The proposed approach

We are interested in helping mobile App users to search and retrieve apps in response to their queries and satisfy their needs. In real life, when a user looks for an app, she issues a text query q to a search engine. The keywords of q represent the search intent of the user in terms of functionalities and features of the sought app. Then, the search engine retrieves the relevant apps that satisfy the user intent, ranks them according to their relevance to q and presents them to the user as a final result to her query. The user usually prefers that the apps returned by the search engine provide all the features and functionalities that she requested.

Formally, we have M apps $A = \{a_1, ..., a_M\}$. Each app a_i is represented by a description

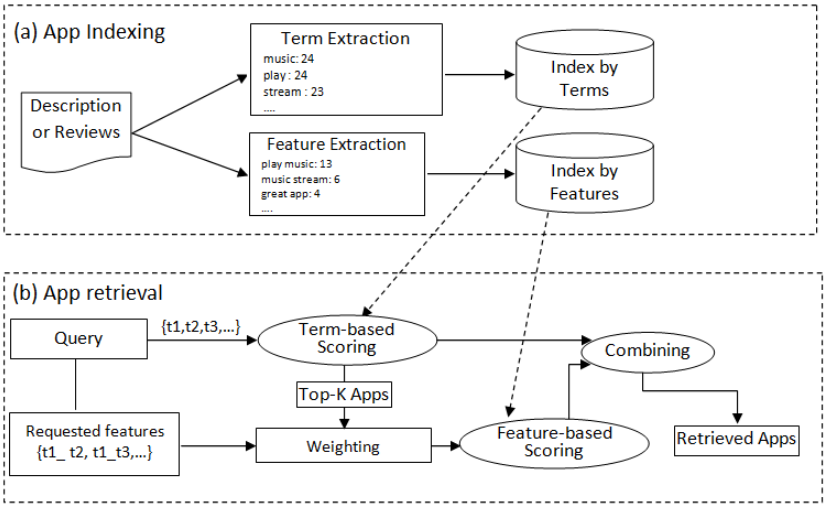

Figure 5.1: App Indexing and Retrieval framework based on terms and features. The same framework will be applied for both representations (description and reviews) separately.

(d_i) given by the app developer and user reviews (r_i) given by online users. Given a user query $q = \{t_1, t_2, ...t_n\}$, the purpose of the app retrieval system is to rank the apps according to their relevance to q, based on the descriptions and reviews of the apps. Since the users need apps that provide some specific features, our goal in this work is to extract app features from description and reviews to give a feature-based representation to apps, extract the features requested by the user in her query and finally match between query and apps to find relevant apps based on how well the app features mentioned in description and reviews match the features requested by user. Thus, our work included two steps: terms and features indexing and app retrieving. Both steps are outlined in Figure 5.1.

5.5.1 Terms and features indexing

As shown in Figure 5.1 (a), each app will be indexed using two types of indexes: indexing by terms and indexing by features. In the first indexing type, the terms extracted from description and reviews are used as a basis of the index process. Thus, we first extract tokens from app description and reviews by a standard indexing approach (tokenisation, stop word removal and then stemming). Then, we index the terms with their frequencies for each app into Desc_Term_Index and Review_Term_Index, which are the two term-based indexes of descriptions and reviews, respectively.

In feature indexing, the features extracted from social reviews and description are

used as a basis of the index. To identify and extract these features, we followed the same approach as in [52] study. They reached 91% precision and 73% recall for the whatsapp app. Averaged over all apps used in the experiments, the precision was approximately 60% and the recall was about 50%. The steps of our process are as follows :

- Use POS tags to identify and extract noun, verb and adjective from description and reviews since they are the most likely to be used together in natural language to describe features.
- Remove stopwords to eliminate terms that are very common in the English language.
- Use PorterStemmer algorithm to stem the terms in order to give the same root for terms that are semantically equal but syntactically different.
- Use collocation functionality to extract bigram words ($< Noun >< Verb >, < Noun >< Noun > and < Adjective >< Noun >$) that frequently occur together within a window size, regardless of whether these terms are adjacent or not. For example, when we look at the the following three sentences, "send the auto reply message","send out the automatic message" and "it sends messages", despite the fact that the distance between "send" and "message" is different in the three sentences but "send message" is considered to be a collocation and therefore considered as a feature. More details on the choice of window size will be discussed in the experimental section. We consider also that the word ordering is not important for describing features. For example, we consider that the pairs of words "send message" and "message send" reflect the same meaning.

This task was applied on descriptions and reviews separately. After the completion of the features extraction process, the features extracted from reviews are filtered by taking into consideration only those that appear in at least two reviews, so as to have a list of features with their frequencies for each app. The obtained result is used to construct the two indexes, Desc_Feat_Index and Review_Feat_Index, respectively, the index of descriptions based on features and the index of reviews based on features. Table 5.1 shows the features with their frequencies extracted from description and reviews of the application *airplay.android*, one of the most relevant apps of the query *"airplay music stream"*. From this table we can see that the frequency of features is higher in the reviews than in the description. This is the case, because several users can talk about the same feature in their reviews, whereas in description, the app developer does not repeat the

Representation	Extracted Features
Reviews	music play(13) googl music(9) googl play(9) airplay appl(6) music stream(6) airport express(5) app bought(5) give star(4)app great(4) app work(4)
Description	free player(2) airplay android(2) music play(2) googl play(2) music video(2) googl support(2) airplay video(2) speaker wireless(2) googl music(2) airplay music(2)

Table 5.1: Features with their frequencies extracted from description and reviews of the app *"airplay.android"*

same feature several times. We also notice that the features extracted from description and reviews match well with the user requested features ("airplay music","music stream"). This observation motivated us to perform a feature-based retrieval system.

5.5.2 Apps retrieval framework

The four obtained indexes will be used to retrieve apps based on terms and features extracted from description and reviews. Figure 5.1(b) shows an App retrieval framework used to retrieve apps that best match the query q. The same framework process will be applied to both description and review representations. In order to get the relevance score of apps based on description, we first apply a classical retrieval model using terms only. Then the top-ranked retrieved apps are used to weight the features extracted from query which are used for matching against the apps indexing features. Once term-based and feature-based scores are obtained, we combine them to get the description score $S_d(q, d)$ for each app. As for review representation, the same steps are repeated to compute the review score $S_r(q, r)$ for each app. Finally, the description and review scores are combined to get the final score. Thus, the score of each app a corresponding to a query q is calculated as follows :

$$S(q, a) = \alpha S(q, d) + (1 - \alpha)S(q, r) \tag{5.1}$$

where α is a parameter used to determine the proportions of description score $S(q, d)$ and review score $S(q, r)$ which are calculated as follow:

$$S(q, d) = \beta S_t(q, d) + (1 - \beta)S_f(q, d) \tag{5.2}$$

$$S(q, r) = \gamma S_t(q, r) + (1 - \gamma)S_f(q, r) \tag{5.3}$$

Where $S_t(q, d)$ and $S_t(q, r)$ are the two term-based scores using, respectively, description and reviews, and $S_f(q, d)$ and $S_f(q, r)$ are the two feature-based scores using, respectively,

Notation	Description
a, t, f, q	application a, term t, feature f, query q
d	application represented by description
r	application represented by reviews
$\|D_t\|$, $\|D_f\|$	description app lengths in terms of, respectively, terms and features
$\|R_t\|$, $\|R_f\|$	reviews app lengths in terms of, respectively, terms and features
C_{td}, C_{fd}	the entire collection using description in terms of, respevtively, terms and features
C_{tr}, C_{fr}	the entire collection using reviews in terms of, respectively, terms an features
$p_{ml}(t,d), p_{ml}(f,d)$	the maximum likelihood estimate (MLE) of t and f in app description
$p_{ml}(t,r), p_{ml}(f,r)$	MLE of t and f respectively in app reviews
$p_{ml}(t,C_{td}), p_{ml}(f,C_{fd})$	MLE of t and f respectively in the description collection
$p_{ml}(t,C_{td}), p_{ml}(f,C_{fd})$	MLE of t and f respectively in the reviews collection
$\|C_{td}\|, \|C_{fd}\|$	description collection lengths in terms of, respectively, terms and features
$\|C_{tr}\|, \|C_{fr}\|$	reviews collection lengths in terms of respectively, terms an features
$c(t,d), c(f,d)$	count of t and f in app description d
$c(t,r), c(f,r)$	count of t and f in app reviews r
$c(t,C_{td}), c(f,C_{fd})$	count of t and f in descriptions collections
$c(t,C_{tr}), c(f,C_{fr})$	count of t and f in reviews collections

Table 5.2: Table of notations

description and reviews. β and γ are two parameters to determine the proportion of term-based score and feature-based score using description and reviews respectively.

5.5.3 Retrieval model

The language model [57, 109] was used in several previous works and shown to have a strong performance in ad-hoc retrieval. We therefore use this model in our approach to estimate the scores mentioned above. The score of a document d with respect to query q is computed as follows :

$$S(q,d) = \prod_{t \in q} p(t \mid d) \tag{5.4}$$

where t is a term (unigram), $p(t|d)$ is the probability of t being in d. After applying Dirichlet smoothing technique [136] , in order to avoid zero-frequency problem of the unseen terms in document, the score of document d with respect to a query q is calculated as follows:

$$S(q|d) = \sum_{t \in q \cap d} log[1 + \frac{|D| p_{ml}(t,d)}{\mu p_{ml}(t,C)}] + nlog\frac{\mu}{|D| + \mu} \tag{5.5}$$

where n is the number of terms in the query, $|D|$ is the length of the document d, C is the set of all documents, μ is the Dirichlet smoothing parameter, $p_{ml}(t,d) = \frac{c(t,d)}{|D|}$ and $p_{ml}(t,C) = \frac{c(t,C)}{|C|}$ are the maximum likelihood estimate (MLE) of term t in, respectively,

the document and in the collection; where $c(t,d)$ and $c(t,C)$ denote the count of term t in, respectively, the document d and the collection C respectively, $|C|$ is the number of terms in the entire collection. Function 5.5 will be used to calculate the scores of apps. Table 5.2 depicts the summary of notations.

5.5.4 Term-based app score

We are given a query $q = \{t_1, t_2, ..., t_n\}$ made of a set of terms and an app a represented by the terms contained in the description d and the terms contained in user reviews r. According to equation 5.5 the two term-based scores of a corresponding to q are calculated as follows:

$$S_t(q,d) = \sum_{t \in q \cap d} log[1 + \frac{|D_t| p_{ml}(t,d)}{\mu_{td} p_{ml}(t, C_{td})}] + nlog\frac{\mu_{td}}{|D_t| + \mu_{td}} \tag{5.6}$$

$$S_t(q,r) = \sum_{t \in q \cap r} log[1 + \frac{|R_t| p_{lm}(t,r)}{\mu_{tr} p_{ml}(t, C_{tr})}] + nlog\frac{\mu_{tr}}{|R_t| + \mu_{tr}} \tag{5.7}$$

Where $p_{ml}(t,d) = \frac{c(t,d)}{|D_t|}, p_{ml}(t,r) = \frac{c(t,r)}{|D_r|}, p_{ml}(t,C_{td}) = \frac{c(t,C_{td})}{|C_{td}|}$ and $p_{ml}(t,C_{tr}) = \frac{c(t,C_{tr})}{|C_{tr}|}$. μ_{td} and μ_{tr} are the two smoothing parameters used for term-based scores of description and reviews.

5.5.5 Feature-based app score

Before introducing the function that calculates feature-based scores of apps, we first explain how extract the requested features from the query terms. We mentioned previously, when a user looks for a particular app, he certainly has a list of features that the app must provide. To seek this app, she issues a query in the form of terms to a search engine. For instance, the owner of the query "simple notepad large font" requests apps that provide the two features "simple notepad" and "large font". The challenge is how can we extract a list of requested features from a set of term given by a user so as to bring the most relevant results to the top ranks. To tackle this problem, we propose three techniques to select the most relevant requested features from the user query:

5.5.5.1 All pairs of query terms as requested features

We consider the set of all possible term pairs as the set of requested features. If we have n terms in the query, we will obtain $n(n-1)/2$ requested features. For example, if the query consists of $q = (t_1, t_2, t_3)$, we obtain the following set $F_q = \{f_1, f_2, f_3\}$ of requested

features, such as $f_1 = t_1\_t_2, f_2 = t_1\_t_3$ *and* $f_3 = t_2\_t_3$, with the ordering of terms being unimportant (similarly to the indexing process). The disadvantage of this technique is that it considers that all query pairs of terms are requested features and have the same weight ($w(f_i) = 1$), whereas if we take the query "simple notepad large font" the term pairs like "font simple" or "notepad large" should not be considered as features requested by the user who made the query.

5.5.5.2 Features in Top-k feedback apps as requested features

Pseudo relevance feedback [111, 114], also known as blind relevance feedback, is an important technique that can be used to improve the effectiveness of IR systems. The idea behind this technique is to assume that the initially returned *Top-k* ranked documents are relevant and therefore their terms are used to perform a new query. Intuitively, we assume that the *Top-k* ranked apps returned by the term based retrieval model, as described in section 5.5.4, are relevant to the user query, which makes it very likely that the features provided by these apps match well with the features requested by the user in his query. Consequently, the query term pairs that appear for at least one of the *Top-k* ranked apps are considered as requested features and the others are ignored. We give a weight for each term pair as follows :

$$w(f) = \begin{cases} 1, if\ \exists\ a_i \in Top\text{-}k,\ f\ appear\ in\ a_i \\ 0, otherwise \end{cases} \tag{5.8}$$

Where f is the requested feature whose weight we want to calculate and a_i is the i-th app in the *Top-k* ranked list represented either by description, or by reviews.

5.5.5.3 Likelihood ratio to weight requested features

The preceding technique has the disadvantage of being based on the presence of the query term pairs in the *Top-k* ranked applications, giving them the same weight $w(f) = 1$, but ignoring their importance in the description or reviews of application. However, some of the features extracted from apps as described in section 5.5.1 are observed due to the high frequency of the involved individual terms. This NLP problem is called *occurrence-by-chance* [70]. To avoid this problem and determine whether the term pair has some real structural importance and can be considered as a requested feature, we have adopted the "Likelihood Ratio" approach for hypothesis testing of independence [43], which takes into account the description/reviews of the application that has been considered for calculating

	t_2	not t_2	
t_1	o_{11}	o_{12}	R1
not t_1	o_{21}	o_{22}	R2
	C1	C2	N

Table 5.3: Contingency table of observed frequencies of t_1 and t_2

	t_2	not t_2	
t1	$e_{11} = \frac{R1C1}{N}$	$e_{12} = \frac{R1C2}{N}$	R1
not t1	$e_{21} = \frac{R2C1}{N}$	$e_{22} = \frac{R2C2}{N}$	R2
	C1	C2	N

Table 5.4: Contingency table of expected frequencies of t_1 and t_2

the frequency of the term pair as well as the frequency of the individual terms. The weight of each term pair will be calculated as follows :

$$w(f = t_1\_t_2) = \sum_{i=1}^{k} LR(f, a_i) \tag{5.9}$$

Where k is the number of $Top\text{-}k$ ranked apps and LR is the Log-Likelihood ratio of term pair f in the app a_i and is calculated based on the contingency tables of observed and expected frequencies, shown in Table 5.3 and Table 5.4, respectively.

$$LR(f, a) = 2 * \sum_{i,j} o_{ij} log(\frac{o_{ij}}{e_{ij}}) \tag{5.10}$$

Now that we have calculated the weights of each requested feature of the query, we only need to measure the two feature-based scores of each application using the two indexes Desc_Feat_Index and Review_Feat_Index. We are given a query $q = \{f_1, f_2, ..., f_n\}$ made of a set of requested features weighted using the three previous methods and an app a represented by the features contained in the description d and the features contained in user reviews r. According to Equation 5.5, the two feature-based scores of app

a corresponding to q are calculated as follows:

$$S_f(q,d) = \sum_{f \in q \cap d} log[1 + \frac{|D_f| p_{ml}(f,d)}{\mu_{fd} p_{ml}(t, C_{fd})}] + nlog\frac{\mu_{fd}}{|D_f| + \mu_{fd}} \tag{5.11}$$

$$S_f(q,r) = \sum_{f \in q \cap r} log[1 + \frac{|R_f| p_{lm}(f,r)}{\mu_{fr} p_{ml}(f, C_{tr})}] + nlog\frac{\mu_{fr}}{|R_f| + \mu_{fr}} \tag{5.12}$$

Where $p_{ml}(f, C_{td}) = \frac{c(f,C_{fd})}{|C_{fd}|}$ and $p_{ml}(f, C_{fr}) = \frac{c(f,C_{fr})}{|C_{fr}|}$. μ_{fd} and μ_{fr} are the two smoothing parameters used for the feature-based scores of description and reviews.

Since each requested feature has its weight, the maximum likelihood estimate of f in app reviews and description is calculated as follows:

$$p_{ml}(f,d) = \frac{w(f) * c(f,d)}{|D_f|} \tag{5.13}$$

$$p_{ml}(f,r) = \frac{w(f) * c(f,r)}{|D_r|} \tag{5.14}$$

Where $w(f)$ is the weight of the requested feature f calculated using the above three techniques, based on app description for calculating $S_f(q,d)$ and on app reviews for calculating $S_f(q,r)$.

5.6 Experiments and results

In this section, we describe the details of the dataset used to perform experiments and show the results obtained by our approach using terms and features to index applications.

5.6.1 Dataset and evaluation metric

To evaluate our approach, we used the same mobile app retrieval dataset used in [106]. This dataset contains 43,041 mobile app and 56 queries with their relevance judgments. Each app is represented by a description given by the app's developer and a set of reviews (max =50) given by online users. As for queries, each query is a set of keywords generated by domain experts from real requests made by users based on android forums. With regard to evaluation metrics, we also employed induced NDCG at $3, 5, 10, 20$, used in [106], allowing us to compare the results of our approach with the ones found in this work. The induced NDCG metric ignores unjudged apps from the ranked list and keeps only the judged apps

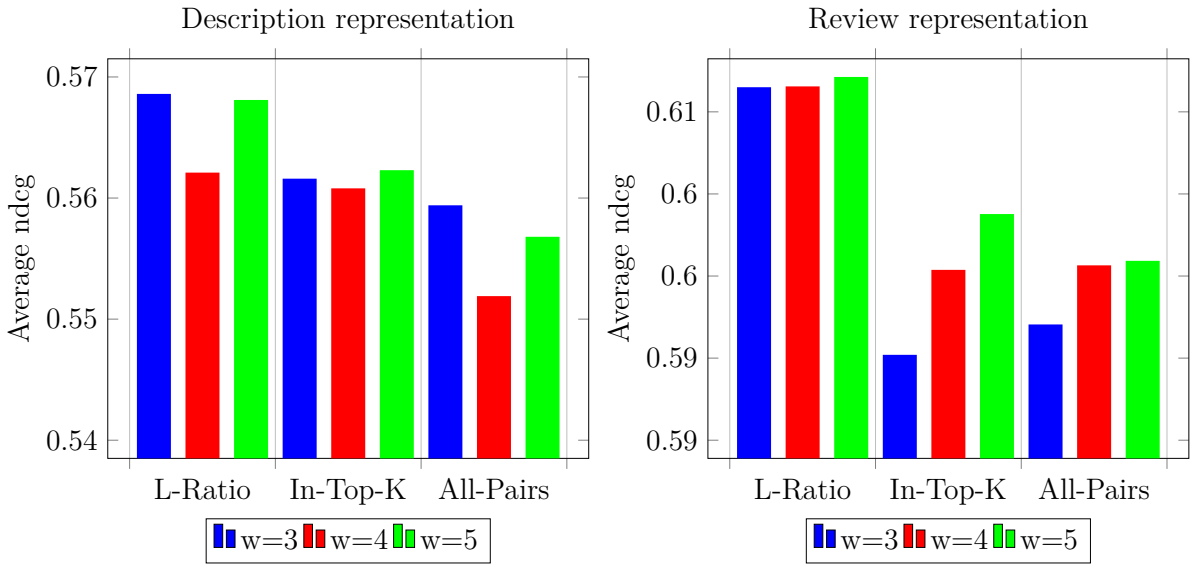

Figure 5.2: Avg-ndcg measures for different values of window size (w) and different weighting methods of requested features (L-Ratio, In-Top-K, All-Pairs) obtained from description representation(left) and review representation (right)

and is calculated in the same way as regular NDCG. More details on this metric can be found in [134].

5.6.2 Parameter setting

To get the final score of each app for a given query, by using Equation 5.1, we first have to compute several scores, employing the four indexes, before combing them. Thus, the computations require the tuning of several parameters. We tuned the parameters using the average of the four Induced NDCG at 3, 5, 10, 20 (avg-ndcg=$\frac{1}{4}\sum_i ndcg@i$). Thereby, we started our experiments by calculating the two term-based scores based on apps descriptions and reviews. for $S_t(q,d)$ and $S_t(q,r)$ we set μ_{td} and μ_{tr} to, respectively, 1000 and 300, since these values showed the best performance in [106]. Once completed the computation of term-based scores, we used the same values of μ_{td} and μ_{tr} to integrate the feature-based scores with equations 2 and 3. We mention here that we tried several window sizes to extract features from description and reviews, as described in section 5.5.1. Among the tested values (i.e $w \in [3,4,5]$), the best results were obtained, respectively, with $w = 3$ for description and $w = 5$ for reviews (Figure 5.2). For the number of apps used to weight the requested features, we set $k = 10$ and the best results were obtained when we applied the Likelihood ratio to weight the requested features (Figure 5.2). As for

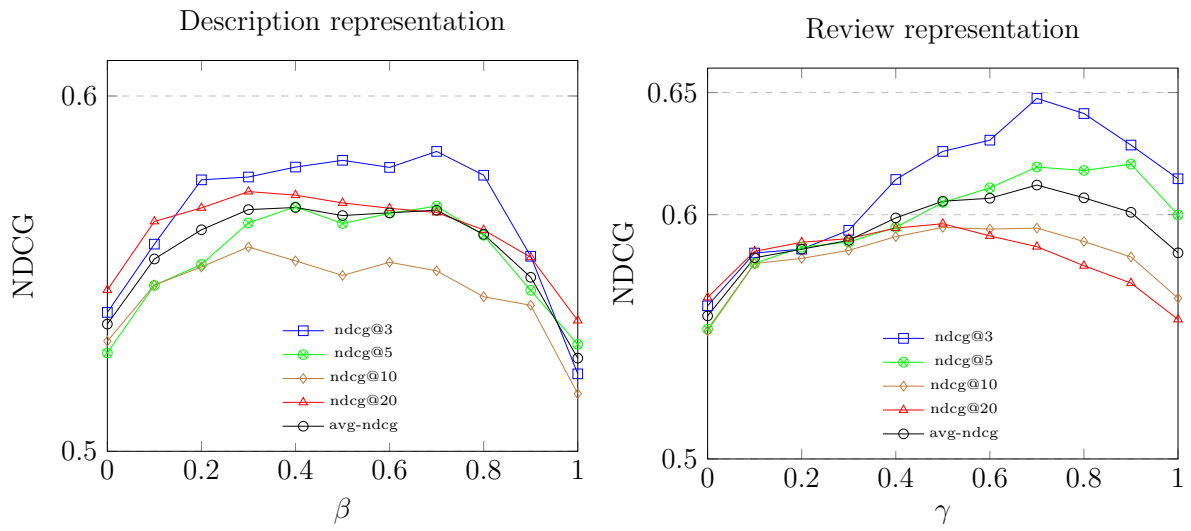

Figure 5.3: NDCG measures for different values of β in description representation (left) and different values of (γ) in review representation (right) to determine the impact of of feature-based score and term-based score on both representations.

μ_{fd} and μ_{fr}, we followed [100, 108], and set their values to the average collection length in features on description and reviews, respectively. The values β and γ were set to 0.4 and 0.7, respectively. Finally, the two scores $S(q, d)$ and $S(q, r)$ performed to measure the scores of the app using features and terms in description and reviews were combined, and α was set to 0.4 which showed the best results.

5.6.3 Analysis

In this section we present the results of experiments and show the impact of choosing the widow size to extract features, the impact of requested features weigthing, of combining terms and features and the impact of description and reviews on the model performance. We also compare our results against those obtained in previous works.

Table 5.5 shows the evaluation performance, in terms of ndcg@3, 5, 10, 20 and the average ndcg of the suggested methods compared with previous works [79, 106] as well as with Google Play's app search engine. We use symbols $+, \bullet$ and $*$ to mark if the improvement for integrating features is statistically significant (paired t-test with $p \leq 0.05$) in each measure over Term-based models using description, reviews and both description and reviews, respectively.

As expected, Table 5.5 shows that when integrating features in the score function has an influence on the retrieval performance. The models that combine terms and features

		ndcg@3	ndcg@5	ndcg@10	ndcg@20
Description	Term-only	0.522	0.530	0.516	0.537
	Terms + Features	0.580^{+}	0.569^{+}	0.554^{+}	0.572^{+}
Reviews	Term-only	0.615	0.600	0.566	0.557
	Term + Features	0.648	0.620	$0.595^{\bullet}$	$0.587^{\bullet}$
Description and Reviews	Term-only	0.649	0.632	0.612	0.615
	Terms + features	**0.680**	$\mathbf{0.665}^{*}$	$\mathbf{0.635}^{*}$	0.633
AppLDA [106]		0.651	0.656	0.627	**0.634**
Relemb [79]		0.577	0.563	0.556	/
Google Play [106]		0.589	0.575	0.568	0.566

Table 5.5: NDCG evaluation results obtained by different methods of our approach compared to related works.

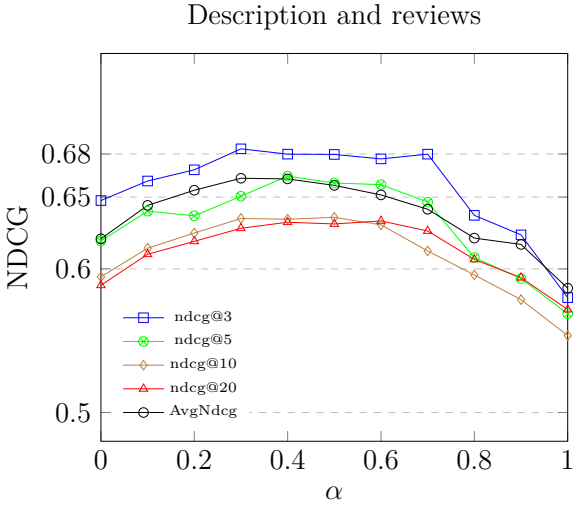

Figure 5.4: NDCG measures for different values of α to determine the impact of description score and review score

outperform term only model in all measures when either using description, reviews or combining the two representations to represent apps; indicating that apps may be better represented by terms and features than terms only. The same table also shows that the model that use reviews only perform better than the model that use app descriptions only; that means that social users' reviews are a good resource for app search compared to descriptions. For example when using terms only, ndcg@3 = 0.522 for description, whereas it equals 0.615 for reviews and when combining feature and terms, ndcg@3=0.580 for description, whereas it equals 0.648 for reviews. We also noticed that the impact of combining features and terms for description is more significant than for reviews. For example, ndcg@3 increases from 0.522 to 0.580 (11.11 %) for description, whereas it increases from 0.615 to 0.648 (5.36%) and it is not statistically significant in terms of this measure as shown in the table. In addition, Table 5.5 shows the good results obtained when window size is fixed to, respectively, 3 and 5 for description and reviews in the process of extracting features and using Likelihood-ratio to weight requested features.

Figure 5.2 shows the effects of weighting methods and the parameter w on the retrieval performance for the two representations description and reviews. We noticed that whatever the representation of the apps, the best results in terms of avg-ndcg are obtained when Likelihood-ratio (L-Ratio) is present as a weighting method compared to other methods. However, the worst results are obtained when all term pairs of the query terms are used as requested features. It is also clear that the best results are obtained when $w = 3$ for description and $w = 5$ for reviews regardless of the used weighting method. This difference may be due to the difference between the language styles used by the app developer to write the description and by online users to describe their opinion in reviews. Consider as an example the app "airplay.android" and the feature "airplay music". When we look at the description of this app, we find that the two terms of this feature are close to each other ("Magicplay brings ***airplay music*** and video support to Android", "**airplay music** from Android to WiFi speakers and receivers") but when we look at reviews, these terms are far from each other ("Latest update kinda broke **airplay** in Google play **music**", "show me the **airplay** code as it does when trying **music** broadcast") which explains the obtained results.

In order to understand the effects of features and terms, we further investigate performance when different values of β (descrription score) and γ (review score) are used. NDCG measures for different values of these parameters are shown in Figure 5.3. Here, when $\beta = 0$ or $\gamma = 0$ the models exploit terms only, whereas they exploit features only when $\beta = 1$ or $\gamma = 1$. From the figure, we notice that using features and terms together

yields even better performance than using terms only or features only for both representations. We notice also that for description ($\beta = 0.4$), the effect of feature score is stronger than the term score but the opposite is true for reviews ($\gamma = 0.7$).

Figure 5.4 shows the performance of our model when different values of α are used. Here, when $\alpha = 0$, the model exploits description only while it exploits reviews only when $\alpha = 1$. From the figure, we notice that using description and reviews together yields even better performance than using description only or reviews only. The best results are obtained when $\alpha = 0.4$, which means that the two representations are important and each complements the other.

Making a comparison with the results of related works, AppLDA [106] (exploits description and reviews), Relemb [79] (exploits description only) and Google Play's app search engine (unavailable information), We can see from Table 5.5 that, when using description and reviews together, our approach outperforms AppLDA in terms of all ndcg measures except for ndcg@20 when the results are almost the same (0.634 for AppLDA and 0.633 for our approach. As for the case of using description only, our model outperforms Relemb model in ndcg@3 and ndcg@5 and gives almost the same results as this approach in terms of ndcg@10.

5.7 Conclusion

In chapter, we conducted a study to build an app retrieval model based on terms and features extracted from the description given by the app developer and the social reviews given by online users. We started first by extracting and indexing terms and features from description and reviews using natural language processing techniques. Then, and in order to make the matching between the query and feature indexes, we proposed three techniques to extract and weight the requested features from the query terms. Finally, several scores using description and reviews as well as terms and features were calculated and combined to measure the relevance of each app to a given query. Evaluation results show that the models that incorporate features with terms outperform the models that use terms only regardless of whether reviews or descriptions are used, meaning that integrating features in the retrieval model effectively helps app retrieval. Using social reviews only gives better results than using descriptions only, meaning that the vocabulary gap between social reviews and users queries is smaller than the gap between descriptions and users queries. Extracting features from reviews requires the window size of co-occurrence to be

larger than the one used to extract features from description. This is probably due to the difference between the language styles used by the app developer to write description and by online users to describe their opinion in reviews. Evaluation results show also that statistical association measures, in our case Likelihood-ratio, can be used to weight the requested features extracted from query terms.

Conclusion and future works

In this thesis, we have addressed a recent area of information retrieval that emerged and gained popularity with the arrival of social media, called social information retrieval. The goal of this new area of research is to exploit the avalanche of social information that has exploded on the web in the retrieval process, in order to benefit from the experience of others users (*wisdom of the crowd*).
We started by giving a general overview of basic concepts of information retrieval systems, describing their main components such as document and query representation for indexing and matching. Afterward, we presented the different models proposed in the literature as well as the metrics proposed for evaluation. Next, we studied the state-of-the-art of social information retrieval, where we have categorized, surveyed and compared them. After several researches, we found that there are three different directions in which social information can be leveraged in the retrieval process:

- Social data as a complementary source of information to enrich documents and queries and enhance the search performance,
- Social interaction and collaboration as part of the search process, in which users find information by asking friends and other users for assistance in social forums,
- Social data as new information to be searched and in which users can gather recent information about a particular subject (e.g., product, hotel, event, etc.).

The works of this thesis are focused on the first and the second direction. In addition to exploiting the different types of social information in the search process, we also propose to generate automatic answers to users' requests extracted from social forums. Despite the large number of contributions proposed in the literature, there remain many research challenges that need to be addressed. The main challenges studied in this thesis are : (i) How to deal with natural language queries, usually posted by users in social forums to express their needs? (ii) How to adapt classical models to social information retrieval and

choose the suitable parameters for each different type of social information (Tags, reviews and social signals)? (iii) How to use social reviews to represent the sought entities? Using a standard bag of words representation or a new representation based on entity features discussed by users in their reviews.

Contributions

In our first contribution, we proposed two approaches to tackle the problem of long natural language queries submitted by users in online social forums. The first approach represents each query term by several features, namely, statistical, linguistic, fields, profile and example features. Then a learning to rank algorithm is used to weight and rank query terms and select the top ranked ones as input to the retrieval system. The second approach is a combination of query reduction and expansion techniques. We start by removing the extraneous terms from the query and keep only the the relevant ones that match user' needs. Then, we expand the query by adding terms from similar books mentioned by users in their queries. The two techniques yield improved search results compared to the original query.

In the second contribution, we studied how to represent books by tags and reviews in social book search and measured the sensitivity of the retrieval model parameters to each representation. This study is motivated by the fact that books can be represented by different types of social information such as tags or reviews. It allows us to know and choose for each representation the suitable model, thereby enhancing the retrieval performance. The experiments show that the Tag Based Model does not require a document length normalization, especially for short queries, for which the best results of this model were obtained when the length normalization parameter had a very low values. However, the Review Based Model requires the document length normalization. This may be caused by the fact that the user reviews are long and can thus be seen as classical textual documents. We have also noticed that using two indexes for tags and reviews separately and combining the results of the models gives better results compared to when using a single index and a single model.

As for the last contribution termed leveraging features extracted from social information to improve the retrieval performance, we propose a new feature-based representation

for both document (a mobile app in our case) and queries. Social user reviews and the description of app's developer are used to extract features that characterize the app. Then, we extract the request features from the user query. Finally, we match between the two representations in order to return the most relevant apps to the user query. Evaluation results show that the models that incorporate features with terms outperform the models that use a standard bag-of-words term-matching technique regardless of whether reviews or descriptions are used, meaning that integrating features in the retrieval model effectively helps app retrieval. Using social reviews only gives better results than using descriptions only, meaning that social reviews represent a valuable source of evidence for social information retrieval that minimize the vocabulary gap between documents and users queries.

Perspectives

Following recent successes of applying deep learning techniques, such as word2vec [96], LSTM [44] and BERT [39] which were shown to be effective for a wide range of text-based information retrieval applications like named entity recognition (NER) [34] and POS tagging [60], in future works, we plan to apply the neural embedding models for social information retrieval in two directions:

- In the first direction, the main challenge that we aim to tackle is the issue of complex and verbose natural language queries submitted by users in social forums which makes understanding users' needs and relevant terms extraction a challenging task. We aim to investigate an approach for dealing with this problem by applying a Bidirectional Encoder Representations from Transformers (BERT) model [39] to classify in order to classify query terms in two classes, relevant and non relevant. This is motivated by the success of such model in named entity recognition which is somewhat similar to our purpose.

- In addition, we also plan to use deep learning techniques in which we aim to represent the three elements of social information retrieval in a low dimensional vector space. Thus, user2vec to represent users based on their profiles, doc2vec for representing documents using tags and social reviews and query2vec which transforms each query to vector that represents the user's needs. These representations will

capture semantic and syntactic similarity relations between words as well as the non-linear relationship between documents, users and queries.

As for the feature-based approach, it provides interesting directions for future work. Firstly, not all the features mentioned by users in their reviews are actual features provided by the application, some of them can be features requested by users because the application does not provided them. This could potentially skew the results. Therefore, it is necessary to, first use a review analyser system that can distinguish between features provided by the app and features requested by online users before incorporating them into the retrieval model. Secondly, the application of a topic modeling algorithm like LDA to group related features in the same topic could possibly improve the performance by bridging the existing gap between the language present in social reviews and the language present in queries.

Bibliography

[1] Fabian Abel, Nicola Henze, and Daniel Krause. Optimizing search and ranking in folksonomy systems by exploiting context information. In José Cordeiro and Joaquim Filipe, editors, *Web Information Systems and Technologies - 5th International Conference, WEBIST 2009, Lisbon, Portugal, March 23-26, 2009, Revised Selected Papers*, volume 45 of *Lecture Notes in Business Information Processing*, pages 113–127. Springer, 2009.

[2] Ismail Badache and Mohand Boughanem. Document priors based on time-sensitive social signals. In Allan Hanbury, Gabriella Kazai, Andreas Rauber, and Norbert Fuhr, editors, *Advances in Information Retrieval - 37th European Conference on IR Research, ECIR 2015, Vienna, Austria, March 29 - April 2, 2015. Proceedings*, volume 9022 of *Lecture Notes in Computer Science*, pages 617–622, 2015.

[3] Ismail Badache and Mohand Boughanem. Fresh and diverse social signals: Any impacts on search? In Ragnar Nordlie, Nils Pharo, Luanne Freund, Birger Larsen, and Dan Russel, editors, *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval, CHIIR 2017, Oslo, Norway, March 7-11, 2017*, pages 155–164. ACM, 2017.

[4] R Baeza-Yates and B Ribeiro-Neto. Modern information retrieval: The concepts and technology behind search. 2011. *ISBN-13*, 978:0321416919, 2011.

[5] Peter Bailey, Ryen W White, Han Liu, and Giridhar Kumaran. Mining historic query trails to label long and rare search engine queries. *ACM Transactions on the Web (TWEB)*, 4(4):15, 2010.

[6] Noor Hasrina Bakar, Zarinah M Kasirun, Norsaremah Salleh, and Hamid A Jalab. Extracting features from online software reviews to aid requirements reuse. *Applied Soft Computing*, 49:1297–1315, 2016.

[7] Saeid Balaneshin-kordan and Alexander Kotov. Optimization Method for Weighting Explicit and Latent Concepts in Clinical Decision Support Queries. 2016.

[8] Shenghua Bao, Gui-Rong Xue, Xiaoyuan Wu, Yong Yu, Ben Fei, and Zhong Su. Optimizing web search using social annotations. In Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy, editors, *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 501–510. ACM, 2007.

[9] Jayendra Barua and Rajdeep Niyogi. Improving named entity recognition and disambiguation in news headlines. *International Journal of Intelligent Information and Database Systems*, 12(4):279–303, 2019.

[10] Michael Bendersky and W Bruce Croft. Discovering key concepts in verbose queries. *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 491–498, 2008.

[11] Michael Bendersky, Donald Metzler, and W Bruce Croft. Parameterized concept weighting in verbose queries. *Sigir*, pages 605–614, 2011.

[12] Michael Bendersky, Donald Metzler, and W Bruce Croft. Effective query formulation with multiple information sources. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 443–452. ACM, 2012.

[13] Chahinez Benkoussas, Hussam Hamdan, Shereen Albitar, Anaïs Ollagnier, and Patrice Bellot. Collaborative filtering for book recommandation. In *CLEF (Working Notes)*, pages 501–507, 2014.

[14] Steven Bird and Edward Loper. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain, jul 2004. Association for Computational Linguistics.

[15] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[16] Toine Bogers and Birger Larsen. Rslis at inex 2012: Social book search track. In *INEX*, volume 12, pages 97–108, 2012.

[17] Abraham Bookstein and Don R Swanson. Probabilistic models for automatic indexing. *Journal of the Association for Information Science and Technology*, 25(5):312–316, 1974.

[18] Mohamed Reda Bouadjenek. *Infrastructure and algorithms for information retrieval based on social network analysis/mining*. PhD thesis, University of Paris-Saclay, Versailles, 2013.

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

[20] Mohamed Reda Bouadjenek, Hakim Hacid, and Mokrane Bouzeghoub. Social networks and information retrieval, how are they converging? a survey, a taxonomy and an analysis of social information retrieval approaches and platforms. *Information Systems*, 56:1–18, 2016.

[21] Mohamed Reda Bouadjenek, Hakim Hacid, and Mokrane Bouzeghoub. Social networks and information retrieval, how are they converging? A survey, a taxonomy and an analysis of social information retrieval approaches and platforms. *Information Systems*, 56:1–18, 2016.

[22] Bert R Boyce, Donald H Kraft, Charles T Meadow, Donald H Kraft, Bert R Boyce, and Charles T Meadow. *Text information retrieval systems*. Elsevier, 2017.

[23] Peter Brusilovsky, Barry Smyth, and Bracha Shapira. Social search. In Peter Brusilovsky and Daqing He, editors, *Social Information Access - Systems and Technologies*, volume 10100 of *Lecture Notes in Computer Science*, pages 213–276. Springer, 2018.

[24] Vannevar Bush. As we may think. *Atl. Mon.*, 176(1):101–108, 1945.

[25] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In Zoubin Ghahramani, editor, *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML*

2007), Corvallis, Oregon, USA, June 20-24, 2007, volume 227 of *ACM International Conference Proceeding Series*, pages 129–136. ACM, 2007.

[26] Messaoud Chaa and Omar Nouali. CERIST at INEX 2015: Social book search track. In Linda Cappellato, Nicola Ferro, Gareth J. F. Jones, and Eric SanJuan, editors, *Working Notes of CLEF 2015 - Conference and Labs of the Evaluation forum, Toulouse, France, September 8-11, 2015*, volume 1391 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[27] Messaoud Chaa, Omar Nouali, and Patrice Bellot. Verbose query reduction by learning to rank for social book search track. In *CLEF (Working Notes)*, pages 1072–1078, 2016.

[28] Messaoud Chaa, Omar Nouali, and Patrice Bellot. New technique to deal with verbose queries in social book search. In *Proceedings of the International Conference on Web Intelligence*, pages 799–806. ACM, 2017.

[29] Messaoud Chaa, Omar Nouali, and Patrice Bellot. Combining tags and reviews to improve social book search performance. In Patrice Bellot, Chiraz Trabelsi, Josiane Mothe, Fionn Murtagh, Jian-Yun Nie, Laure Soulier, Eric SanJuan, Linda Cappellato, and Nicola Ferro, editors, *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 9th International Conference of the CLEF Association, CLEF 2018, Avignon, France, September 10-14, 2018, Proceedings*, volume 11018 of *Lecture Notes in Computer Science*, pages 64–75. Springer, 2018.

[30] Messaoud Chaa, Omar Nouali, and Patrice Bellot. Leveraging app features to improve mobile app retrieval. *Int. J. Intell. Inf. Database Syst.*, 14(2):177–197, 2021.

[31] Oscar Chaparro and Andrian Marcus. On the Reduction of Verbose Queries in Text Retrieval Based Software Maintenance. pages 716–718, 2016.

[32] Sergiu Chelaru, Claudia Orellana-Rodriguez, and Ismail Sengör Altingövde. How useful is social feedback for learning to rank youtube videos? *World Wide Web*, 17(5):997–1025, 2014.

[33] Yves Chiaramella and Philippe Mulhem. La recherche d'information. de la documentation automatique à la recherche d'information en contexte. *Document Numérique*, 10(1):11–38, 2007.

[34] Jason P. C. Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *Trans. Assoc. Comput. Linguistics*, 4:357–370, 2016.

[35] Gobinda Chowdhury. Social information retrieval systems: Emerging technologies and applications for searching the web effectively. *J. Assoc. Inf. Sci. Technol.*, 61(12):2587–2588, 2010.

[36] D Manning Christopher, Raghavan Prabhakar, Schütze Hinrich, et al. Introduction to information retrieval. *An Introduction To Information Retrieval*, 151(177):5, 2008.

[37] Ronan Cummins. A Study of Retrieval Models for Long Documents and Queries in Information Retrieval. 2016.

[38] Ronan Cummins and Colm O'Riordan. The effect of query length on normalisation in information retrieval. In *Irish Conference on Artificial Intelligence and Cognitive Science*, pages 26–32. Springer, 2009.

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

[40] Emanuele Di Buccio, Massimo Melucci, and Federica Moro. Detecting verbose queries and improving information retrieval. *Information Processing and Management*, 50(2):342–360, 2014.

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

[42] Pavel A. Dmitriev, Nadav Eiron, Marcus Fontoura, and Eugene J. Shekita. Using annotations in enterprise search. In Les Carr, David De Roure, Arun Iyengar, Carole A. Goble, and Michael Dahlin, editors, *Proceedings of the 15th international conference on World Wide Web, WWW 2006, Edinburgh, Scotland, UK, May 23-26, 2006*, pages 811–817. ACM, 2006.

[43] Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational linguistics*, 19(1):61–74, 1993.

[44] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 334–343. The Association for Computer Linguistics, 2015.

[45] Efthimis N. Efthimiadis. Interactive query expansion: A user-based evaluation in a relevance feedback environment. *J. Am. Soc. Inf. Sci.*, 51(11):989–1003, 2000.

[46] Shao-Hui Feng, Bo-Wen Zhang, Xu-Cheng Yin, Zan-Xia Jin, Jian-Lin Jin, Jian-Wei Wu, Le-Le Zhang, Hao-Jie Pan, Fan Fang, and Fang Zhou. Ustb at social book search 2016 suggestion task: Active books set and re-ranking. In *CLEF (Working Notes)*, pages 1089–1096, 2016.

[47] Larry Fitzpatrick and Mei Dent. Automatic feedback using past queries: Social searching? In Nicholas J. Belkin, A. Desai Narasimhalu, Peter Willett, William R. Hersh, Fazli Can, and Ellen M. Voorhees, editors, *SIGIR '97: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 27-31, 1997, Philadelphia, PA, USA*, pages 306–313. ACM, 1997.

[48] Bin Fu, Jialiu Lin, Lei Li, Christos Faloutsos, Jason Hong, and Norman Sadeh. Why people hate your app: Making sense of user feedback in a mobile app store. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1276–1284, New York, NY, USA, 2013. ACM.

[49] Dan Guo and Pengfei Gao. Complex-query web image search with concept-based relevance estimation. *World Wide Web*, 19(2):247–264, 2016.

[50] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. A deep relevance matching model for ad-hoc retrieval. *CoRR*, abs/1711.08611, 2017.

[51] Manish Gupta and Michael Bendersky. Information Retrieval with Verbose Queries. *Foundations and Trends® in Information Retrieval*, 9(3-4):209–354, 2015.

[52] Emitza Guzman and Walid Maalej. How do users like this feature? a fine grained sentiment analysis of app reviews. In *2014 IEEE 22nd international requirements engineering conference (RE)*, pages 153–162. IEEE, 2014.

[53] Meriem Hafsi, Mathias Géry, and Michel Beigbeder. Lahc at inex 2014: Social book search track. In *Working Notes for CLEF 2014 Conference*, 2014.

[54] Ben He and Iadh Ounis. Term frequency normalisation tuning for bm25 and dfr models. In *ECIR*, volume 5, pages 200–214. Springer, 2005.

[55] Paul Heymann, Georgia Koutrika, and Hector Garcia-Molina. Can social bookmarking improve web search? In Marc Najork, Andrei Z. Broder, and Soumen Chakrabarti, editors, *Proceedings of the International Conference on Web Search and Web Data Mining, WSDM 2008, Palo Alto, California, USA, February 11-12, 2008*, pages 195–206. ACM, 2008.

[56] Djoerd Hiemstra. Information retrieval models. In Ayse Göker and John Davies, editors, *Information Retrieval: Searching in the 21st Century*, pages 1–19. Wiley, 2009.

[57] Djoerd Hiemstra and Wessel Kraaij. Twenty-one at trec-7: Ad-hoc and cross-language track. In *TREC*, pages 174–185, 1998.

[58] Minqing Hu and Bing Liu. Mining opinion features in customer reviews. In *AAAI*, number 4, pages 755–760, 2004.

[59] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry P. Heck. Learning deep structured semantic models for web search using clickthrough data. In Qi He, Arun Iyengar, Wolfgang Nejdl, Jian Pei, and Rajeev Rastogi, editors, *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 2333–2338. ACM, 2013.

[60] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991, 2015.

[61] Bernardo A. Huberman and Lada A. Adamic. Novelty and social search in the world wide web. *CoRR*, cs.MA/9809025, 1998.

[62] Samuel Huston and W Bruce Croft. Evaluating verbose query processing techniques. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 291–298. ACM, 2010.

[63] Claudia Iacob and Rachel Harrison. Retrieving and analyzing mobile apps feature requests from online reviews. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 41–44. IEEE Press, 2013.

[64] Melanie Imhof, Ismail Badache, and Mohand Boughanem. Multimodal social book search. In *6th Conference on Multilingual and Multimodal Information Access Evaluation (CLEF 2015)*, pages pp–1, 2015.

[65] Di Jiang, Jan Vosecky, Kenneth Wai-Ting Leung, and Wilfred Ng. Panorama: a semantic-aware application search framework. In *Proceedings of the 16th International Conference on Extending Database Technology*, pages 371–382, New York, NY, USA, 2013. ACM.

[66] Song Jin, Hongfei Lin, and Sui Su. Query expansion based on folksonomy tag co-occurrence analysis. In *The 2009 IEEE International Conference on Granular Computing, GrC 2009, Lushan Mountain, Nanchang, China, 17-19 August 2009*, pages 300–305. IEEE Computer Society, 2009.

[67] Thorsten Joachims. Optimizing search engines using clickthrough data. *Kdd '02*, pages 133–142, 2002.

[68] Timo Johann, Christoph Stanik, Walid Maalej, et al. Safe: A simple approach for feature extraction from app descriptions and app reviews. In *2017 IEEE 25th International Requirements Engineering Conference (RE)*, pages 21–30. IEEE, 2017.

[69] Kyo C Kang, Sholom G Cohen, James A Hess, William E Novak, and A Spencer Peterson. Feature-oriented domain analysis (foda) feasibility study. Technical Report CMU/SEI-90-TR-021, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1990.

[70] Muhammad Asif Hossain Khan, Masayuki Iwai, and Kaoru Sezaki. A robust and scalable framework for detecting self-reported illness from twitter. In *2012 IEEE 14th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 303–308. IEEE, 2012.

[71] Angella J Kim and Eunju Ko. Do social media marketing activities enhance customer equity? an empirical study of luxury fashion brand. *Journal of Business Research*, 65(10):1480–1486, 2012.

[72] Jae Dong Kim, Hema Raghavan, and Rukmini Iyer. Predicting Term Importance in Queries for Improved Query-Ad Relevance Prediction . *Learning*, 2010.

[73] Sebastian Marius Kirsch, Melanie Gnasa, and Armin B. Cremers. Beyond the web: Retrieval in social information spaces. In Mounia Lalmas, Andy MacFarlane, Stefan M. Rüger, Anastasios Tombros, Theodora Tsikrika, and Alexei Yavlinsky, editors, *Advances in Information Retrieval, 28th European Conference on IR Research, ECIR 2006, London, UK, April 10-12, 2006, Proceedings*, volume 3936 of *Lecture Notes in Computer Science*, pages 84–95. Springer, 2006.

[74] Marijn Koolen. " user reviews in the search index? that'll never work!". In *ECIR*, volume 8416, pages 323–334. Springer, 2014.

[75] Marijn Koolen, Toine Bogers, Maria Gäde, Mark Hall, Iris Hendrickx, Hugo Huurdeman, Jaap Kamps, Mette Skov, Suzan Verberne, and David Walsh. Overview of the clef 2016 social book search lab. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 351–370. Springer, 2016.

[76] Marijn Koolen, Toine Bogers, Maria Gäde, Mark M. Hall, Hugo C. Huurdeman, Jaap Kamps, Mette Skov, Elaine Toms, and David Walsh. Overview of the CLEF 2015 social book search lab. In Josiane Mothe, Jacques Savoy, Jaap Kamps, Karen Pinel-Sauvagnat, Gareth J. F. Jones, Eric SanJuan, Linda Cappellato, and Nicola Ferro, editors, *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 6th International Conference of the CLEF Association, CLEF 2015, Toulouse, France, September 8-11, 2015, Proceedings*, volume 9283 of *Lecture Notes in Computer Science*, pages 545–564. Springer, 2015.

[77] Marijn Koolen, Jaap Kamps, and Gabriella Kazai. Social book search: comparing topical relevance judgements and book suggestions for evaluation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 185–194. ACM, 2012.

[78] Marijn Koolen, Gabriella Kazai, Jaap Kamps, Michael Preminger, Antoine Doucet, and Monica Landoni. Overview of the INEX 2012 social book search track. In Pamela

Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF 2012 Evaluation Labs and Workshop, Online Working Notes, Rome, Italy, September 17-20, 2012*, volume 1178 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[79] Shubham Krishna, Ahsaas Bajaj, Mukund Rungta, Vanraj Vala, and Hemant Tiwari. Relemb: A relevance-based application embedding for mobile app retrieval and categorization. *Computación y Sistemas*, 23(3):969–978, 2019.

[80] Giridhar Kumaran and James Allan. Effective and efficient user interaction for long queries. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 11–18. ACM, 2008.

[81] Giridhar Kumaran and Vitor R. Carvalho. Reducing long queries using query quality predictors. In James Allan, Javed A. Aslam, Mark Sanderson, ChengXiang Zhai, and Justin Zobel, editors, *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, pages 564–571. ACM, 2009.

[82] Giridhar Kumaran and Vitor R. Carvalho. Reducing long queries using query quality predictors. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09*, page 564, 2009.

[83] Matthew Lease, James Allan, and W. Bruce Croft. Regression rank: Learning to meet the opportunity of descriptive queries. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5478 LNCS:90–101, 2009.

[84] Yuanchun Li, Baoxiong Jia, Yao Guo, and Xiangqun Chen. Mining user reviews for mobile app comparisons. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol*, 1(3):1–15, September 2017.

[85] Shih Hsiang Lin, Ea Ee Jan, and Berlin Chen. Handling verbose queries for spoken document retrieval. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pages 5552–5555, 2011.

[86] Yuan Lin, Hongfei Lin, Song Jin, and Zheng Ye. Social annotation in query expansion: a machine learning approach. In Wei-Ying Ma, Jian-Yun Nie, Ricardo Baeza-Yates, Tat-Seng Chua, and W. Bruce Croft, editors, *Proceeding of the 34th International ACM*

SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011, pages 405–414. ACM, 2011.

[87] Tie-Yan Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, 2009.

[88] Tie-Yan Liu. *Learning to Rank for Information Retrieval*. Springer, 2011.

[89] Yuzhou Liu, Lei Liu, Huaxiao Liu, Xiaoyu Wang, and Hongji Yang. Mining domain knowledge from app descriptions. *Journal of Systems and Software*, 133:126–144, 2017.

[90] Rachel Tsz-Wai Lo, Ben He, and Iadh Ounis. Automatically Building a Stopword List for an Information Retrieval System. *Journal on Digital Information Management: Special Issue on the 5th Dutch-Belgian Information Retrieval Workshop (DIR)*, 5:17–24, 2005.

[91] Hans Peter Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317, 1957.

[92] Yuanhua Lv and ChengXiang Zhai. Lower-bounding term frequency normalization. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 7–16. ACM, 2011.

[93] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.

[94] Donald Metzler and W. Bruce Croft. Linear feature-based models for information retrieval. *Inf. Retr.*, 10(3):257–274, 2007.

[95] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[96] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[97] Bhaskar Mitra and Nick Craswell. An introduction to neural information retrieval. *Found. Trends Inf. Retr.*, 13(1):1–126, 2018.

[98] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. Learning to match using local and distributed representations of text for web search. In Rick Barrett, Rick Cummings, Eugene Agichtein, and Evgeniy Gabrilovich, editors, *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 1291–1299. ACM, 2017.

[99] Bhaskar Mitra, Eric T. Nalisnick, Nick Craswell, and Rich Caruana. A dual embedding space model for document ranking. *CoRR*, abs/1602.01137, 2016.

[100] Zaiqing Nie, Yunxiao Ma, Shuming Shi, Ji-Rong Wen, and Wei-Ying Ma. Web object retrieval. In *Proceedings of the 16th international conference on World Wide Web*, pages 81–90, 2007.

[101] Michael G. Noll and Christoph Meinel. The metadata triumvirate: Social annotations, anchor texts and search queries. In *2008 IEEE / WIC / ACM International Conference on Web Intelligence, WI 2008, 9-12 December 2008, Sydney, NSW, Australia, Main Conference Proceedings*, pages 640–647. IEEE Computer Society, 2008.

[102] Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Douglas Johnson. Terrier information retrieval platform. In *European Conference on Information Retrieval*, pages 517–519. Springer, 2005.

[103] Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Christina Lioma. Terrier: A high performance and scalable information retrieval platform. In *Proceedings of the OSIR Workshop*, pages 18–25, 2006.

[104] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[105] Jiaul H. Paik and Douglas W. Oard. A Fixed-Point Method for Weighting Terms in Verbose Informational Queries. *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management - CIKM '14*, pages 131–140, 2014.

[106] Dae Hoon Park, Mengwen Liu, ChengXiang Zhai, and Haohong Wang. Leveraging user reviews to improve accuracy for mobile app retrieval. In *Proceedings of the 38th*

International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 533–542. ACM, 2015.

[107] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014.

[108] Desislava Petkova and W Bruce Croft. Hierarchical language models for expert finding in enterprise corpora. *International Journal on Artificial Intelligence Tools*, 17(01):5–18, 2008.

[109] Jay M Ponte and W Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281. ACM, 1998.

[110] Stephen E Robertson. The probability ranking principle in ir. *Journal of documentation*, 1977.

[111] Stephen E Robertson and K Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146, 1976.

[112] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at TREC-2. In Donna K. Harman, editor, *Proceedings of The Second Text REtrieval Conference, TREC 1993, Gaithersburg, Maryland, USA, August 31 - September 2, 1993*, volume 500-215 of *NIST Special Publication*, pages 21–34. National Institute of Standards and Technology (NIST), 1993.

[113] J.J Rocchio. RELEVANCE FEEDBACK IN INFORMATION RETRIEVAL. *Prentice Hall*, pages 313– 323, 1965.

[114] Joseph John Rocchio. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*, 1971.

[115] Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. Using word embeddings for automatic query expansion. *CoRR*, abs/1606.07608, 2016.

[116] Gerald Salton. Automatic information organization and retrieval. 1968.

[117] Gerard Salton, Edward A. Fox, and Harry Wu. Extended boolean information retrieval. *Commun. ACM*, 26(11):1022–1036, 1983.

[118] Gerard Salton, Edward a. Fox, and Harry Wu. Extended Boolean information retrieval. *Communications of the ACM*, 26(11):1022–1036, 1983.

[119] Statista. Number of available applications in apple app store. [online] https://www.statista.com/statistics/263795/number-of-available-apps-in-the-apple-app-store/, 2020. (Accessed on: 02/02/2020).

[120] Statista. Number of available applications in the google play store. [online] https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/, 2020. (Accessed on: 02/02/2020).

[121] Statista. Number of downloads of applications. [online] https://www.statista.com/statistics/604343/number-of-apple-app-store-and-google-play-app-downloads-worldwide/, 2020. (Accessed : 02/02/2020).

[122] Gayatri Swamynathan, Christo Wilson, Bryce Boe, Kevin Almeroth, and Ben Y Zhao. Do social networks improve e-commerce?: a study on social marketplaces. In *Proceedings of the first workshop on Online social networks*, pages 1–6. ACM, 2008.

[123] Mona Taghavi, Ahmed Patel, Nikita Schmidt, Christopher Wills, and Yiqi Tew. An analysis of web proxy logs with query distribution pattern approach for search engines. *Computer Standards and Interfaces*, 34(1):162–170, 2012.

[124] C. J. van Rijsbergen. *Information Retrieval*. Butterworth, 1979.

[125] Phong Minh Vu, Tam The Nguyen, Hung Viet Pham, and Tung Thanh Nguyen. Mining user opinions in mobile app reviews: A keyword-based approach (t). In *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 749–759, Lincoln, NE, USA, 2015. IEEE.

[126] Annika Wærn. User involvement in automatic filtering: An experimental study. *User Model. User Adapt. Interact.*, 14(2-3):201–237, 2004.

[127] Xuan Wu, Dong Zhou, Yu Xu, and Séamus Lawless. Personalized query expansion utilizing multi-relational social data. In Mária Bieliková and Marián Simko, editors, *12th International Workshop on Semantic and Social Media Adaptation and Personalization, SMAP 2017, Bratislava, Slovakia, July 9-10, 2017*, pages 65–70. IEEE, 2017.

[128] Xiaoying Xu, Kaushik Dutta, Anindya Datta, and Chunmian Ge. Identifying functional aspects from user reviews for functionality-based mobile app recommendation. *Journal of the Association for Information Science and Technology*, 69(2):242–255, 2018.

[129] Xiaobing Xue and W Bruce Croft. Generating Reformulation Trees for Complex Queries. *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 525–534, 2012.

[130] Xiaobing Xue, Samuel Huston, and W. Bruce Croft. Improving verbose queries using subset distribution. *Proceedings of the 19th ACM international conference on Information and knowledge management - CIKM '10*, page 1059, 2010.

[131] Xueliang Yan, Guanglai Gao, Xiangdong Su, and Hongxi Wei. Hidden Markov Model for Term Weighting in Verbose Queries. pages 82–87, 2012.

[132] Bishan Yang, Nish Parikh, Gyanit Singh, and Neel Sundaresan. A study of query term deletion using large-scale E-commerce search logs. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8416 LNCS:235–246, 2014.

[133] Wai Gen Yee, Andrew Yates, Shizhu Liu, and Ophir Frieder. Are web user comments useful for search? In Claudio Lucchese, Gleb Skobeltsyn, and Wai Gen Yee, editors, *Proceedings of the 7th Workshop on Large-Scale Distributed Systems for Information Retrieval, co-located with ACM SIGIR 2009, LSDS-IR@SIGIR 2009, Boston, USA, July 23, 2009*, volume 480 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[134] Emine Yilmaz and Javed A Aslam. Estimating average precision with incomplete and imperfect judgments. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 102–111. ACM, 2006.

[135] Xu-Cheng Yin, Bo-Wen Zhang, Xiao-Ping Cui, Jiao Qu, Bin Geng, Fang Zhou, Li Song, and Hong-Wei Hao. Isart: A generic framework for searching books with social information. *PloS one*, 11(2):e0148479, 2016.

[136] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)*, 22(2):179–214, 2004.

[137] Bo-Wen Zhang, Xu-Cheng Yin, Xiao-Ping Cui, Jiao Qu, Bin Geng, Fang Zhou, and Hong-Wei Hao. Ustb at inex2014: Social book search track. In *CLEF (Working Notes)*, pages 536–542, 2014.

[138] Bo-Wen Zhang, Xu-Cheng Yin, Fang Zhou, and Jian-Lin Jin. Building your own reading list anytime via embedding relevance, quality, timeliness and diversity. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1109–1112. ACM, 2017.

[139] Dong Zhou, Séamus Lawless, and Vincent Wade. Improving search via personalized query expansion using social media. *Inf. Retr.*, 15(3-4):218–242, 2012.

[140] Dong Zhou, Xuan Wu, Wenyu Zhao, Séamus Lawless, and Jianxun Liu. Query expansion with enriched user profiles for personalized search utilizing folksonomy data. *IEEE Trans. Knowl. Data Eng.*, 29(7):1536–1548, 2017.

[141] Justin Zobel. What we talk about when we talk about information retrieval. *SIGIR Forum*, 51(3):18–26, 2017.

Abstract: The emergence of social media has revolutionized the Web, notably by allowing users to interact, exchange messages and share their knowledge with other users in the form of comments, annotations and ratings of resources. These tasks have led to a dramatic growth of information on the web. This new information, known as social information, has been a source of evidence, in the field of social information research, for estimating the relevance of documents and better responding to user requests. However, the use of social information to improve information retrieval has several challenges, the most important of which are (i) find a better representation of documents taking into account the social dimension, (ii) adapt the models of information retrieval to take into account the different types of social information such as comments and annotations, (iii) find a better representation of the user request which is generally complex and formulated in natural language in social forums. The main contributions of our work consist in proposing an approach based on reduction and expansion to process natural language queries and better understand user needs. We also propose to adapt and parametrize the IR models to suit the different types of social information. Finally, in order to better exploit users' reviews, we propose a new representation of documents, which combines terms and features extracted from user reviews. The proposed approaches were evaluated on two datasets, Social Book Search and App Retrieval, and the results clearly show the improvement in search performance.
Key-words: Social Information Retrieval, Social Media, Social Reviews, Social Tags, Natural Language Query , Natural Language Processing (NLP).

Résumé : L'émergence des média sociaux a révolutionné le Web en permettant notamment aux utilisateurs d'interagir, échanger des messages et de partager leurs connaissances avec d'autres utilisateurs sous forme de commentaires, annotations et notations sur des ressources. Ainsi, ces tâches ont conduit à une croissance spectaculaire de l'information sur le web. Cette nouvelle information dite information sociale a constitué une source d'évidence, dans le domaine de la recherche d'information sociale, permettant d'estimer la pertinence des documents et de mieux répondre aux requêtes des utilisateurs. Cependant, l'exploitation de l'information sociale pour améliorer la recherche d'information fait face plusieurs défis dont les plus important sont (i) trouver une meilleure représentation des documents en prenant en compte la dimension sociale, (ii) adapter les modèles de recherche d'information pour prendre en compte les différents types d'informations sociales tels que les commentaires et les annotations, (iii) trouver une meilleure représentation de la requête utilisateur qui est généralement complexe et formulée en langage naturel dans les forums sociaux.
Les principales contributions de notre travail ont consisté, dans un premier temps, à proposer une approche basée sur la réduction et l'expansion pour traiter les requêtes en langage naturel et mieux comprendre les besoins des utilisateurs. Par la suite, nous proposons d'adapter et de paramétrer les modèles de RI pour convenir les différents types d'informations sociales. Enfin, et afin de mieux exploiter les commentaires des utilisateurs, nous proposons une nouvelle représentation des documents, qui combine les termes et les caractéristiques extraites des commentaires des utilisateurs. Les approches proposées ont été évaluées sur deux datasets, Social Book Search et App Retrieval et les résultats montrent bien l'amélioration des performances de la recherche.
Mots clés: Recherche d'Information Sociale, Médias Sociaux, Commentaires Sociaux, Annotations Sociales, Requête en langue naturelle, Traitement Automatique de Langue (TAL).

ملخص: أحدث ظهور وسائل التواصل الاجتماعي ثورة في الويب ، لا سيما من خلال السماح للمستخدمين بالتفاعل وتبادل الرسائل ومشاركة معرفتهم مع مستخدمين آخرين في شكل تعليقات، علامات وتقييمات على الموارد . وبالتالي أدت هذه المهام إلى نمو كبير في المعلومات على الويب. شكلت هذه المعلومات الجديدة، المعروفة بإسم المعلومات الاجتماعية، مصدرًا للأدلة، في مجال أبحاث المعلومات الاجتماعية، لتقدير أهمية المستندات والاستجابة بشكل أفضل لطلبات المستخدمين. ومع ذلك فإن استخدام المعلومات الاجتماعية لتحسين استرجاع المعلومات له العديد من التحديات، وأهمها (1) إيجاد تمثيل أفضل للوثائق من خلال مراعاة البعد الاجتماعي ، (2) تكييف نماذج البحث المعلوماتي مع الأخذ بعين الاعتبار الأنواع المختلفة من المعلومات الاجتماعية مثل التعليقات والعلامات ، (3) إيجاد تمثيل أفضل لطلب المستخدم الذي يكون معقدًا بشكل عام ويتم صياغته باللغة الطبيعية في المنتديات الاجتماعية. تتمثل المساهمات الرئيسية لعملنا في المقام الأول في اقتراح مقاربة تعتمد على التخفيض والتوسع لمعالجة الطلبات المقدمة باللغة الطبيعية وفهم احتياجات المستخدم بشكل أفضل. ثم نقوم بتكييف نماذج إسترجاع المعلومات و ضبط متغيراتها لتلائم الأنواع المختلفة من المعلومات الاجتماعية. أخيرا ومن أجل استغلال تعليقات المستخدمين بشكل أفضل، نقترح تمثيلًا جديدًا للوثائق، والذي يجمع بين المصطلحات والخصائص المستخرجة من التعليقات. تم تقييم المناهج المقترحة على مجموعتي بيانات، "بحث الكتب" Social Book Search و"بحث التطبيقات "App Retrieval وتظهر النتائج بوضوح التحسن في أداء البحث.

كلمات البحث : البحث عن المعلومات الاجتماعية، وسائط التواصل الاجتماعي، التعليقات الاجتماعيّة، العلامات الاجتماعيّة، طلب باللغة الطبيعية، معالجة اللغات الطبيعية.