THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *15/06/2015* **par :**

Adrian-Gabriel CHIFU

Adapting information retrieval systems to contexts: the case of query difficulty
Adaptation des systèmes de recherche d'information aux contextes: le cas des requêtes difficiles

École doctorale et spécialité :
MITT : Image, Information, Hypermedia
Unité de Recherche :
Institut de Recherche en Informatique de Toulouse UMR 5505
Directeur de Thèse :
Josiane MOTHE
Rapporteurs :
Patrice BELLOT et *Jacques SAVOY*

Contents

List of figures

List of tables

List of abbreviations

IR Information Retrieval

IRS information retrieval system

TREC Text REtrieval Conference

QE Query Expansion

QF Query Feedback

TF Term Frequency

IDF Inverse Document Frequency

RI Robustness Index

AP Average Precision

MAP Mean Average Precision

DFR divergence from randomness

RM1 Relevance Model 1

RM3 Relevance Model 3

WN WordNet

API Application Programming Interface

PRF Pseudo Relevance Feedback

QL Query Likelihood

JSD Jensen-Shannon divergence

WS Word Sense

PRF pseudo-relevance feedback

LDA Latent Dirichlet allocation

Acknowledgements

The PhD thesis journey has come to its end. It has been an interesting, challenging and satisfying adventure. It has not been easy, but this enhances its beauty. First of all I would like to thank God for giving me power and patience. I was not alone during this period and I would like to address my acknowledgements to the people close to me. We have shared unforgettable moments, good or bad, happy or sad.

First of all, I would like to sincerely thank my adviser Josiane MOTHE whom has proven to be a great supervisor and manager. She is a precious mentor to me and she knew how to guide and motivate me. She has always treated me with patience and understanding. I am proud and happy to work with Josiane and she represents for me an example of determination, hard work, respect and optimism.

I also acknowledge my reviewers Patrice BELLOT and Jacques SAVOY for accepting to review my thesis and for their precious remarks.

The path for following a research career has been opened by Florentina HRISTEA, when I was a Master's student. I am grateful to Prof. HRISTEA for many things. She has been a valued mentor, warmly guiding me and often giving me useful advice. But most important for me, I thank Prof. HRISTEA for offering me her friendship.

I thank my parents with all my heart. They gave me their unconditional love, all the time, no matter what. They fully supported me, they encouraged and trusted me with my decisions. They would do anything for my well being and I cannot thank them enough. I am happy that they are proud of me and I am completely proud with them.

I thank Mariannic for these years spent in her house at Ramonville. From my landlord at first, she quickly became my mother from France. Always with a good advice, positive thinking and a warm smile, she helped me feel like home, far from home. I cannot thank her enough.

I would like to thank Ionel DAVID for guidance, support, patience, life lessons and more. He unfortunately passed away before I could finish my PhD. He has introduced me to the art of acting since I was a child and he has always been by my side through important moments.

I would like to thank my teachers from university, especially Cristinel MORTICI, Emil LUNGU, Dinu TEODORESCU and Alin POHOATA for their guidance and precious discussions.

I would like to thank my friends from home for their support from distance: Mihai, Richard, Adina, Vlad, Andrei, Corina, Ioana, Georgiana, Andra and all that are not mentioned here.

I am also thankful to my friends that I have met in Toulouse, for all the great moments spent together. Thank you, Dana, Madalina, Valentin, Laure, Cristina, Annette, Maria-Fernanda, Oana, Lavinia, Alexandra, Iulia, Andrei, Vali, Lavinia, Alex, Mihai.

I also would like to thank all people from IRIT for accepting me and helping me during my thesis, all my family members, colleagues and friends.

Special thanks to actual and former occupants of office 424. Thank you Léa, for being a good friend, colleague and for lending me your boyfriend to play drums in my band. Thank you Jonathan, for being such a good friend, confident, moral support and spiritual battery charger. Thanks, Anthony for being Anthony. Thank you Charlotte for the boost of good mood that you brought in our office. We have shared memorable moments and events together. Thanks Jeremy and Damien for being nice, good colleagues.

Infinite thanks for Alina, how has bravely endured my rhythm of life from the end of my thesis. She encouraged me, constantly supported and comforted me during this important period in my career. I thank Alina with all my heart for the great and unforgettable moments that she offered me.

I would like to address my most special thanks to my dear friend Radu. I have learned a lot from him, I respect him and I am happy to be his friend. Meeting Radu has been a life changing event. I owe him a lot of what I have become in the last few years. We have spent awesome moments together and I am looking forward for the moments to come. I also thank Andreea, his future wife, for being a good, supporting friend. She always knew how to bring joy and amusement when I was in trouble.

Abstract

In the search engine environment, users submit queries according to their information needs. In response to a query, the system retrieves and displays a list of documents which it considers of interest to the user. The user analyzes the retrieved results and decides what information is actually relevant to the information need. However, for some queries, the retrieved documents may not satisfy the user. Queries for which the search engine is not able to deliver relevant information are called difficult queries. Query difficulty represents the research context of this thesis. More specifically, we aim at adapting the information retrieval systems with respect to query difficulty, in order to improve the quality of search.

Term ambiguity may be the cause of query difficulty. For example, the query "orange" is ambiguous, since the engine does not know whether it refers to the fruit, to the color, or to the telephone company. As a first contribution, we have developed a re-ranking method for retrieved documents, based on query term disambiguation, in order to obtain better retrieval performance. This method works best on difficult queries, result that motivates our research in query difficulty prediction.

If it was possible to anticipate the difficulty of a query, the difficult queries could be handled in a special manner in order to optimize retrieval results. This observation, corroborated with the findings of our first contribution, led us to our second contribution, on the query difficulty. State-of-the-art predictors are not accurate enough for real applications. Thus, we propose combinations based on linear interpolations of difficulty predictors to improve the prediction quality of individual predictors.

Retrieval results can also be improved by the query expansion process which adds terms to the original query. Even if the query expansion is effective on average over several queries, some of the queries may be degraded. It would be interesting to know how to expand each query in order to improve performance. Our third contribution is an automatic learning method to classify queries by two query expansion variants. Learning is done on features extracted from query difficulty predictors.

Our fourth and last contribution represents an analysis conducted on the parameter optimization in the case of a query expansion model. The parameter that we aim to optimize adjusts the impact of the original query in the expanded query. We test several hypotheses with and without prior information, such as learning, pseudo relevance judgments, logistic regression and similarity measures. The results show that this optimization remains a difficult problem.

Résumé

Dans l'environnement des moteurs de recherche, les utilisateurs saisissent des requêtes en fonction de leurs besoins d'information. En réponse à une requête, le système récupère et affiche une liste de documents qu'il considère comme susceptibles d'intéresser l'utilisateur. L'utilisateur consulte les documents retrouvés et décide quelles informations sont réellement pertinentes par rapport au besoin d'information. Toutefois, les documents retrouvés pour certaines requêtes peuvent ne pas être satisfaisants pour l'utilisateur. Les requêtes pour lesquelles le moteur de recherche n'arrive pas à délivrer l'information pertinente sont appelées difficiles. La difficulté des requêtes représente le contexte de recherche de cette thèse. Nous visons plus spécifiquement à adapter le système de recherche d'information par rapport aux requêtes difficiles, afin d'améliorer la qualité de la recherche.

L'ambiguïté des termes peut-être la cause de la difficulté. Par exemple, la requête "orange" est ambiguë, puisque le moteur ne sait pas si elle se réfère au fruit, à la couleur, ou à la compagnie téléphonique. Comme première contribution, nous avons développé une méthode de ré-ordonnancement des documents retrouvés basée sur la désambiguïsation des termes des requêtes, dans le but d'obtenir de meilleurs résultats de recherche. Cette méthode fonctionne mieux sur les requêtes difficiles, résultat qui motive nos recherches dans la prédiction de la difficulté.

S'il était possible pour d'anticiper la difficulté d'une requête, les requêtes difficiles pourraient être traitées d'une manière particulière pour optimiser les résultats de recherche. Cette observation, complétée par les conclusions de notre première contribution, nous a menés vers notre deuxième contribution, sur la prédiction de la difficulté des requêtes. Les prédicteurs de l'état de l'art ne sont pas suffisamment précis pour des applications réelles. Ainsi, nous proposons des combinaisons basées sur des interpolations linéaires des prédicteurs de difficulté afin d'améliorer la qualité de prédiction.

Les résultats de recherche peuvent aussi être améliorés par l'expansion de requête, procédé qui rajoute des termes à la requête initiale. Même si l'expansion de requêtes est efficace en moyenne sur plusieurs requêtes, certaines requêtes peuvent être dégradées. Il serait donc intéressant de savoir comment réaliser cette expansion, pour chaque requête afin d'obtenir de meilleures performances de recherche. Notre troisième contribution est une méthode automatique d'apprentissage pour classifier les requêtes selon deux variantes d'expansion de requêtes. L'apprentissage est fait sur des caractéristiques extraites à partir des prédicteurs de difficulté.

Enfin, notre quatrième contribution représente une analyse portée sur l'optimisation d'un paramètre afin d'améliorer un modèle d'expansion des requêtes. Ce paramètre ajuste l'importance de la requête initiale dans le modèle étendu de requête. Nous vérifions plusieurs hypothèses sans et avec information a priori, comme l'apprentissage, les pseudo jugements de pertinence, la régression logistique et les mesures de similarité. Les résultats montrent que cette optimisation reste un problème difficile.

General introduction

The field of information retrieval (IR) studies the mechanisms to find relevant information in one or more document collections, in order to satisfy a user's information need. For an IR system, the information to find is represented by "documents" and the information need takes the form of a "query" formulated by the user.

The performance of an IR system depends on queries. Queries for which IR systems fail (little or no relevant documents retrieved) are called in the literature "difficult queries". This difficulty may be caused by term ambiguity, unclear query formulation, the lack of context for the information need, the nature and structure of the document collection, etc.

This thesis aims at adapting IR system to contexts, particularly in the case of difficult queries. The manuscript is organized into five main chapters, besides acknowledgements, general introduction, conclusions and perspectives.

The first chapter is an introduction to IR. We develop the concept of relevance, the retrieval models from the literature, the query expansion and the evaluation framework used in the experiments, which were used to validate our proposals. Each of the following chapters presents one of our contributions.

Each of the following chapters raises the research problem, indicates the related work, our theoretical proposals and their validation on benchmark collections.

In chapter two, we present our research on treating the ambiguous queries. The query term ambiguity can indeed lead to poor document retrieval of documents by the search engine. In the related work, the disambiguation methods that yield good performance are supervised [Zhong 2012], however such methods are not applicable in a real IR context, as they require the information which is normally unavailable. Moreover, in the literature, term disambiguation for IR is declared under optimal [Sanderson 1994], [Guyot 2008]. In this context, we propose an unsupervised query disambiguation method and show its effectiveness. Our approach is interdisciplinary between the fields of natural language processing and IR. The goal of our unsupervised disambiguation method is to give more importance to the documents retrieved by the search engine that contain the query terms with the specific meaning identified by disambiguation. This document re-ranking offers a new document list that contains more relevant documents to the user. We evaluated this document re-ranking method after disambiguation using two different classification techniques (Naïve Bayes [Chifu 2012] and spectral clustering [Chifu 2015], over three document collections and queries from the TREC competition (TREC7, TREC8, WT10G). We have shown that the disambiguation method we proposed works specifically well in the case of poorly performing queries (7.9% improvement compared to state-of-the-art methods).

In chapter three, we present the work focused on query difficulty prediction.

Indeed, if term ambiguity is a difficulty factor, it is not the only one. We completed the range of difficulty predictors by relying on the state-of-the-art predictors that are divided in two main categories: pre-retrieval and post-retrieval predictors. Post-retrieval predictors need the results of the retrieval process, such as document scores, or retrieved document ranks. The existing predictors are not sufficiently effective and stable to be used in concrete applications and therefore we introduce two new difficulty prediction measures that combine predictors by interpolating the value of a pre-retrieval predictor with the value of a post-retrieval predictor. Other researchers such as [Hauff 2009] or [Kurland 2012], have studied predictor combinations, but in different manners. We also propose a robust method to evaluate difficulty predictors. Using predictor combinations, on TREC7 and TREC8 collections, we obtain an improvement of 7.1% in terms of prediction quality, compared to the state-of-the-art [Chifu 2013].

In the fourth chapter we focus on the application of difficulty predictors. Specifically, we proposed a selective IR approach, that is to say, predictors are employed to decide which search engine, among many, would perform better for a query. In the literature, various applications employed learning based on predictors in different domains, such as cross-language Information Retrieval [Lee 2014], or query routing in the case of domain specific collections [Sarnikar 2014]. Our decision model is learned by SVM (Support Vector Machine). We tested our model on TREC benchmark collections (Robust, WT10G, GOV2). The learned model classified the test queries with over 90% accuracy. Furthermore, the research results were improved by more than 11% in terms of performance, compared to non-selective methods [Chifu 2014].

In the last chapter, we treated an important issue in the field of IR: the query expansion by adding terms. It is very difficult to predict the expansion parameters or to anticipate whether a query needs the expansion or not. We present our contribution to optimize the lambda parameter in the case of RM3 (a pseudo-relevance model for query expansion), per query. Lv et. al [Lv 2009] tried to solve the same optimization problem by the means of regression, without obtaining conclusive results. We manage to overcome their results, but our methods that surpass their performance require prior information. We tested several hypotheses, both with and without prior information. We are searching for the minimum amount of information necessary in order for the optimization of the expansion parameter to be possible. The results are not satisfactory, even though we used a wide range of methods such as SVM, regression, logistic regression and similarity measures. Some improvements with respect to baselines are noticeable. However, these improvements are not important enough to declare the problem as solved. Therefore, these findings may reinforce the conclusion regarding the difficulty of this optimization problem. The research was conducted not only during a three months research mobility at the Technion Institute in Haifa, Israel, in 2013, but thereafter, keeping in touch with the team of Technion. In Haifa, we worked with Professor Oren Kurland and PhD student Anna Shtok.

In conclusion, in this thesis we propose new methods to improve the performance of IR systems, based on the query difficulty. The results of the methods proposed in chapters two, three and four show significant improvements and open perspectives for future research. The analysis in chapter five confirms the difficulty of the optimization problem of the concerned parameter and encourages thorough investigation on selective query expansion settings.

In order to ease the reading, the lists of abbreviations, figures and tables are integrated at the beginning of this manuscript.

Chapter 1

Information retrieval framework

Contents

In this chapter we introduce the notions used throughout the entire manuscript. We start from the concept of relevance and its meaning. Thus, we discuss the concepts of relevance and evaluation in IR, the query expansion principles and models, we present the benchmark collections and the framework we used to implement our models.

1.1 Relevance

Humans have always been interested in knowing and their curiosity yielded the discoveries that lead to progress. A Latin aphorism says that "knowledge is power"[1] and the key for this knowledge is information. However, with the amount of information exponentially growing around us, finding good information has become a difficult task. The access to relevant information depends on the given task, on the

[1] "Scientia potentia est"

speed of retrieval, etc. In this context, computer science researchers from the field of Information Retrieval (IR) propose models in order to obtain information. "IR deals with the representation, storage, organization of, and access to information items" [Baeza-Yates 1999]. The organized information items and their representation should provide easy access to relevant information. From the IR point of view, relevance would represent how well a retrieved document (or set of documents) satisfies the information need a user has.

Relevance also represents a concept that appears in other fields of study, such as logic, library, cognitive sciences and information science. A definition given by researchers in socio-cognitive area would sound as follows: "Something (A) is relevant to a task (T) if it increases the likelihood of accomplishing the goal (G), which is implied by T" [Hjörland 2002].

The relevance of retrieved information has always preoccupied scientists and, in the 1950s, with the appearance of the first IR systems [Mooers 1950], researchers underlined the issue of retrieving irrelevant articles. B.C. Vickery explicitly mentioned the concept of relevance in 1958 at the International Conference on Scientific Information [Mizzaro 1997].

However, relevance is not of a single type, there are many kinds of relevance. In his paper [Mizzaro 1997], Mizzaro establishes a framework to classify the various types of relevance and defines two groups of entities. Relevance may be seen as the relation between no matter which two entities, each coming from one of the defined groups.

The first group contains the *document*, as a physical entity, its *representation* and the *information*, which represents what the user obtains while reading a document.

On the other hand, the second group has the following entities: the *problem* that a human is facing and that requires information in order to be solved, the *information need*, which is a representation of the problem, the *request* that is a human language representation of the information need and finally the *query* which is a representation of the information need in the language of computers.

Nevertheless, the IR process can be splitted in three components, which are *topic* (the subject area of interest), *task* (users activity with the retrieved documents) and *context* (everything that affects the way search and its evaluation take place).

The retrieval mechanism considers the query, which represents the information need, and matches it with the document representations. Then, the best candidate documents from the matching process are considered relevant by the system, thus they are proposed to the user. This scenario so far is a static one, however the IR process is englobed in a time interval, meaning that the time factor should be taken into account. For instance, a document may be irrelevant to a query, yet the same document may become relevant for the same query at a different time.

Thus, relevance can be seen as a point in a four-dimensional space that has the following dimensions [Mizzaro 1997]:

- document, surrogate, information;
- problem, information need, request, query;
- topic, task, context;
- the various time instances from the time elapsed between the appearance of the problem and its solution.

Having all these factors involved, there is place for errors. It is known that the user's information needs are formally expressed by queries which are submitted to an information retrieval system (IRS), in order to retrieve documents. For example, if a query does not represent well enough the user's information need, the search results will be poor. Therefore, badly formulated queries, or ambiguous queries, affect the retrieval performance. All queries that lead the IRS to performance failure are called *difficult queries*. Details on query difficulty will be provided in Chapter 3, which treats the matter of query difficulty prediction. Moreover, various systems can respond differently to submitted queries and it would be interesting to find out which system treats best a specific query. This represents the system variability issue in IR, reported in [Harman 2009]. We mention that query difficulty and system variability are challenges treated in this thesis. The system variability is related to a key factor for IR performance, which is the choice of the retrieval model.

In the following section we present several retrieval models from the literature that match documents to queries with the purpose of proposing presumably relevant information to users.

1.2 IR models

We present here several models employed to match documents and queries. The first model was proposed in the 70s by Lancaster in [Lancaster 1974] and it is called the boolean model. Based on the set theory, this model retrieves the documents which contain the query terms joined by logical operators, such as "OR", "AND", or "NOT". Even though this model has a clean formalism and its concept is intuitive and easy to implement, this exact matching method may retrieve too few or too many documents. Moreover, all terms are equally weighted and the output documents are difficult to rank. This is the reason why more recent models generally compute a similarity score between a document and a query, which is used to rank all retrieved documents for a query.

We present in the following sections the vector space model [Salton 1975], the BM25 model [Robertson 1994], the language model [Ponte 1998] and the divergence from randomness (DFR) models [Amati 2002], in order to provide a brief outline of IR models. The vector space and the BM25 models are described here because they represent milestones in IR. On the other hand, DFR models are employed in this thesis in Chapter 2 as retrieval models for strong baselines. The language model is used in Chapter 4 for selective query expansion and in Chapter 5 for parameter optimizing in language model-based query expansion.

The vector space model represents the document and the query as vectors in the term space and employs the cosinus measure to determine their similarity.

The probabilistic models, such as the BM25 model, the DFR models, or the language model, are based on the probability theory to measure the query-document similarity.

1.2.1 Vector space model

The vector space model [Salton 1975] allows to compute a similarity score between a query and a document. Queries and documents are represented by weight vectors of their terms. The ranking score is expressed by the cosinus between the query vector and the document vector (Definition 1).

Definition 1 (The vector space model)

The relevance score of a document d_i with respect to a query q is given by the cosinus similarity measure between the two vectors, as follows:

$$s\left(\mathbf{q}, \mathbf{d}_i\right) = \frac{\sum_{t=1}^{m}\left(w_{tq} \cdot w_{ti}\right)}{\sqrt{\sum_{t=1}^{m}\left(w_{tq}\right)^2}\sqrt{\sum_{t=1}^{m}\left(w_{ti}\right)^2}}, \tag{1.1}$$

where m is the number of terms, w_{tq} represents the weight of the term t in the query q, w_{ti} is the weight of the term t in the document d_i and $\mathbf{q} = [w_{1q} \ldots w_{mq}]$ and $\mathbf{d}_i = [w_{1i} \ldots w_{mi}]$ are the representations in the term vector space, for the query q and for the document d_i, respectively.

The term weight is expressed by the *TF.IDF* weight [Robertson 1976]. The TF.IDF measure is defined using the *Term Frequency (TF)* and *Inverse Document Frequency (IDF)*.

The TF represents the number of occurrences of a term in a document (or in a query).

The IDF represents the importance of a term from an entire set of documents (called corpus) and it is computed as follows:

$$IDF(t) = \log\left(\frac{N}{n_t + 1}\right), \tag{1.2}$$

where N is the total number of documents and n_t represents the number of documents that contain the term t. Thus, the *TF.IDF* weighting of a term t in a document d_i is the product between TF and IDF:

$$TF \cdot IDF\left(t, d_i\right) = TF\left(t, d_i\right) \cdot IDF\left(t\right) \tag{1.3}$$

This term weighting method considers that not all terms have the same discriminating power. The term importance is defined based on their frequency in documents, thus the terms which are too frequent are penalized.

The probabilistic models, such as BM25, DFR and language models, are more recent and are considered as more effective than the vector space model.

1.2.2 Okapi BM25 model

Robertson and Walker have proposed in 1994 the Okapi BM25 system, which used the BM25 probabilistic model [Robertson 1994]. This family of relevance models defines the similarity score $s\left(q, d_i\right)$ between a query q and a document d_i using the relevance and non relevance probabilities of documents, as follows:

$$s\left(q, d_i\right) = \frac{P\left(R|d_i\right)}{P\left(\bar{R}|d_i\right)}, \tag{1.4}$$

where $P\left(R|d_i\right)$ is the probability that the document d_i is relevant for the query q, while $P\left(\bar{R}|d_i\right)$ is the probability that d_i is not relevant for q. This score could be expressed as the sum of relevance weights w_j for the terms t_j (with $j \in \{1, \ldots, m\}$) found in the document:

$$s\left(q, d_i\right) = \sum_{j=1}^{m} w_j \mathbb{1}_{\{t_j \in d_i\}}, \tag{1.5}$$

where $\mathbb{1}_{\{t_j \in d_i\}}$ is the characteristic function which takes the value 1 if the term t_j is relevant for the document d_i and 0, otherwise. The relevance weights are expressed by relevance probabilities of terms for the document.

In [Robertson 1994] are proposed several approximations for these probabilities. The most commonly used version is the BM25 model (see Definition 2), based on *TF* and *IDF*.

The BM25 is widely used as baseline in IR, since it yields good results. Another widely employed probabilistic model, the divergence from randomness, is presented in the following section.

Definition 2 (The BM25 model)

The similarity score of the BM25 model between a query q and a document d_i, denoted $BM25\left(q, d_i\right)$, is defined as follows:

$$BM25\left(q, d_i\right) = \sum_{j=1}^{m} \frac{IDF\left(t_j\right) \cdot TF(t_j, d_i) \cdot \left(k_1 + 1\right)}{TF\left(t_j, d_j\right) + k_1 \cdot \left(1 - b + b\frac{|d_i|}{avdl}\right)}, \tag{1.6}$$

where TF and IDF are those defined in Section 1.2.1, $|d_i|$ is the number of terms in the document d_i, k_1 and b are tuning parameters and $avdl$ represents the average of term numbers, for all documents in the corpus.

1.2.3 Divergence from randomness model

The divergence from randomness (DFR) model [Amati 2004a] represents a generalized version of Harter's 2-Poisson indexing-model [Harter 1974]. The 2-Poisson model is one of the first IR models and is based on the hypothesis that the level of treatment of the informative terms is endorsed by a representative set of documents, in which these terms occur more often than in the rest of the documents.

Of course, there are terms that do not possess representative documents, and thus their frequency follows a random distribution. This is the single Poisson model. Harter's model, combined with the standard probabilistic model proposed by Robertson and Walker in [Robertson 1994], yielded the BM family of models, from which we mention the BM25 model, described in the previous section.

The idea behind the DFR models is the following: the divergence of the within-document term-frequency from its frequency within the collection is directly proportional with the information carried by the term t in the document d_i. Therefore, the term-weight is inversely proportional to the probability of term-frequency within the document d_i obtained by a randomness model M, as follows:

$$DFR\left(q, d_i\right) \propto -logProb_M\left(t \in d_i | Collection\right), \tag{1.7}$$

where M stands for the type of model of randomness used to compute the probability. IR is thus seen as probabilistic process, equivalent to the random placement of coloured balls into urns. Here, the balls are the terms and the urns are the documents, respectively. There are several models to choose M, which provide a basic DFR model. The basic models are presented in Table 1.1.

For the model M represented by the binomial distribution (P), the basic model formalized in Definition 3.

TABLE 1.1: Basic DFR models

Notation	Model
D	Divergence approximation of the binomial
P	Approximation of the binomial
B_E	Bose-Einstein distribution
G	Geometric approximation of the Bose-Einstein
$I\left(n\right)$	Inverse Document Frequency model
$I\left(F\right)$	Inverse Term Frequency model
$I\left(n_e\right)$	Inverse Expected Document Frequency model

Definition 3 (DFR - binomial distribution P)

The DFR model based on the approximation of the binomial, for a term t in a document d_i, denoted by P, is defined as follows:

$$-\,logProb_P\left(t, d_i|Collection\right) = -log\binom{tf}{TF}p^{TF}q^{tf-TF}, \tag{1.8}$$

where TF is the one defined in in Section 1.2.1, tf is the term-frequency of the term t in the $Collection$, N is the number of documents in the $Collection$ and $p = 1/N$ and $q = 1 - p$.

Similarly, if the model M is the geometric distribution, then the basic model G is formalized in Definition 4.

Definition 4 (DFR - geometric distribution G)

The DFR model based on the geometric distribution, for a term t in a document d_i, denoted by G, is defined as follows:

$$-\,logProb_G\left(t, d_i|Collection\right) = -log\left(\left(\frac{1}{1+\lambda}\right)\cdot\left(\frac{\lambda}{1+\lambda}\right)^{TF}\right), \tag{1.9}$$

where tf is the term-frequency of the term t in the $Collection$, N is the number of documents in the $Collection$ and $\lambda = F/N$.

TF can be normalized as follows:

$$TFn = TF\cdot log\left(1 + c\cdot\frac{sl}{d_i^l}\right), \tag{1.10}$$

where d_i^l represents the length of document d_i, sl stands for standard document length and c is a free parameter.

In the following section we present the language model, which also represents a widely used probabilistic model.

1.2.4 Language model

In 1998, Ponte proposed the application of language models in IR [Ponte 1998]. The idea is to associate each document to its characterizing model, which allows to compute the probability that a term sequence would be generated from a document. Considering a query q with m terms, then the probability that a document d_i is relevant for q is equivalent to the probability $P(q|d_i)$ that q has been generated by the document d_i. Assuming that the terms are independent, one can compute $P(q|d_i)$, as follows:

$$P(q|d_i) = \prod_{j=1}^{m} P(t_j|d_i), \tag{1.11}$$

where t_j are query terms, with $j \in \{1, \ldots, \}$. The most straightforward estimator for $P(t_j|d_i)$ is the maximum likelihood, $P_{ML}(t_j, d_i) = \frac{c(t_j, d_i)}{\sum_{k=1}^{m} c(t_k, d_i)}$, where $c(t_j, d_i)$ is the occurrence number of the term t_j in the document d_i. There is a major downside for this estimator which is the fact that it assigns a null probability for the terms that are not present, thus, if a document contains all the query terms except for one, its probability will be zero, even if in fact it potentially responds to a part of the query. In order to solve this issue, several smoothing methods have been proposed. Zhai et al. [Zhai 2001] have studied the properties of three smoothing methods: Dirichlet, Jelinek-Mercer and the Absolute Discounting, presented in definitions 5, 6 and 7, respectively.

Definition 5 (Dirichlet smoothing)

The probability that a term t_j has been generated by the document d_i by the Dirichlet smoothing is the following:

$$P_{DIR}(t_j|d_i) = \frac{c(t_j, d_i) + \mu P_{ML}(t_j|C)}{\sum_{k=1}^{m} c(t_k, d_i) + \mu}, \tag{1.12}$$

where $P_{ML}(t_j|C) = \frac{c(t_j, C)}{\sum_{k=1}^{m} c(t_k, C)}$, with $c(t_j, C)$ the occurrence number of the term t_j in the corpus C, is the maximum likelihood estimator for $P(t_j|C)$, which represents the probability that the term t_j has been generated by the corpus C. $\mu > 0$ represents controlling parameter for the smoothing.

Definition 6 (Jelinek-Mercer smoothing)

The probability that a term t_j has been generated by the document d_i by the Jelinek-Mercer smoothing is the following:

$$P_{JM}\left(t_j|d_i\right) = \left(1-\lambda\right) P_{ML}\left(t_j|d_i\right) + \lambda P_{ML}\left(t_j|C\right), \tag{1.13}$$

where $P\left(t_j|d_i\right)$ and $P\left(t_j|C\right)$ represent the probability that the term t_j has been generated by the document d_i and by the corpus C, respectively. $P_{ML}\left(t_j|d_i\right) = \frac{c\left(t_j,d_i\right)}{\sum_{k=1}^{m} c\left(t_k,d_i\right)}$, with $c\left(t_j,d_i\right)$ the occurrence number of the term t_j in the document d_i, is the maximum likelihood estimator for $P\left(t_j|C\right)$, which represents the probability that the term t_j has been generated by the corpus C. $P_{ML}\left(t_j|C\right) = \frac{c\left(t_j,C\right)}{\sum_{k=1}^{m} c\left(t_k,C\right)}$, with $c\left(t_j,C\right)$ the occurrence number of the term t_j in the corpus C, is the maximum likelihood estimator for $P\left(t_j|C\right)$, which represents the probability that the term t_j has been generated by the corpus C. $\lambda \in [0,1]$ represents a smoothing factor.

Definition 7 (Absolute Discounting smoothing)

The probability that a term t_j has been generated by the document d_i by the Absolute Discounting smoothing is the following:

$$P_{ABS}\left(t_j|d_i\right) = \frac{max\left(c\left(t_j,d_i\right)-\delta,0\right)+\delta\left|d_i\right|_u}{\sum_{k=1}^{m} c\left(t_k,d_i\right)}, \tag{1.14}$$

where $c\left(t_j,d_i\right)$ is the occurrence number of the term t_j in the document d_i, $\left|d_i\right|_u$ represents the number of unique terms in the document d_i and $\delta \in [0,1]$ is a constant which limits the importance of terms known by the language model.

In [Zhai 2001], the authors underline that the performance of language models is very dependent on the parameter tuning and also it varies with the query type. Thus, the Jelinek-Mercer smoothing yielded better results on long queries, as opposed to the other smoothing methods. However, it is difficult to decide which method is globally more efficient. Parameter optimization for language models is a matter also treated in this thesis (see Chapter 5).

Retrieval models have advanced significantly from the beginning to present day and researchers still seek for methods to push the limits of performance improvement. It is known that producing the query from the information need represents a crucial step for a successful retrieval [Lee 2009]. For this reason, there is research on query refinement, reformulation and expansion. Adding useful terms to an initial query should improve, in theory, the quality of the retrieved results. We discuss the query expansion process in the following section.

1.3 Query expansion

"The relative ineffectiveness of IR systems is largely caused by the inaccuracy with which a query formed by a few keyword models the actual user information need" [Carpineto 2012]. A solution for this issue is Query Expansion (QE), where the initial query is augmented by new similar meaning features.

The logical operator "OR" implicitly connects the query terms in most document ranking models. Having this in mind, one query expansion advantage is that there is more chance to retrieve a relevant document that does not contain some of the original query terms, fact which improves recall. However, the added terms may cause query drift, losing the focus of a search topic, involving a loss in precision. On the other hand, the effectiveness of IR systems is usually evaluated taking into account both recall and precision and in this context, improvements of more than 10% have been reported [Mitra 1998], [Liu 2004], [Lee 2008]. In [Mitra 1998], the authors used manually formulated boolean filters to help automatic QE via ad hoc feedback. They have tested their method on Text REtrieval Conference (TREC) 3-6 collections, with improvements between 6% and 13%. Liu et al. [Liu 2004] tested their method to propose candidate expansion terms on TREC 9, 10 and 12. The method uses WordNet to disambiguate the word senses for the query terms and its synonyms, hyponyms together with words from the definitions become candidate terms. The authors have reported 23%-31% of improvements. Another method, proposed in [Lee 2008], employed cluster-based resampling in order to obtain better pseudo-relevant documents for feedback. The authors have tested the method on TREC collections (GOV2, WT10G, Robust) with improvements between 6% and 26%. These claims support the hypothesis that QE is beneficial for IR, but these techniques might not suffice for approaches mainly interested in improving precision.

Moreover, these improvements are reported in average, over multiple queries. This means that performance of some queries, in terms of AP, may decrease after expansion [Sakai 2005]. Therefore, the performance of a QE system should be also checked whether it is robust, or not. Sakai et al. [Sakai 2005] have proposed a measure called Robustness Index, which takes into account the number of queries that are harmed by expansion and the queries that are improved by expansion. This measure is normalized by the total number of queries and the possible values are between -1 and 1. The value 1 characterizes the most robust results, with no queries harmed by the process of expansion. Contrarily, the value -1 suggests that all queries have a decreased AP after expansion. The Robustness Index is defined as follows:

Definition 8 (Robustness index)

For a set of queries Q the Robustness Index (RI) is computed by the formula:

$$RI\left(Q\right) = \frac{n_{+} - n_{-}}{|Q|}, \tag{1.15}$$

where n_{+} is the number of improved queries, n_{-} the number of degraded queries and $|Q|$ the total number of queries.

There are several techniques to generate and rank the candidate expansion terms. We discuss here about the analysis of feature distribution in top-ranked documents and about the query language modeling.

The idea for the feature distribution in top-ranked documents is to assume that the first retrieved documents in response to the original query are relevant (Pseudo-Relevance Feedback [Buckley 1994]). Expansion terms are extracted from these documents. Of course this process is highly dependent on the quality of the top retrieved documents. We present the main term-ranking functions based on term distribution for a set of pseudo-relevant documents in Table 1.2. The notations are the following: $w(t,d)$ is the weight of a term t in pseudo-relevant document d, $P(t|R)$ and $P(t|C)$ represent the probability of occurrence of t in the set of pseudo-relevant documents R and in the whole collection C, respectively.

Table 1.2: Main term-ranking functions based on the analysis of term distribution in pseudo-relevant documents by [Carpineto 2012]

Reference	Function	Mathematical formula
[Rocchio 1971]	Rocchio's weights	$\sum_{d \in R} w(t,d)$
[Robertson 1976]	Binary independence model (BIM)	$log \frac{P(t\|R)[1 - P(t\|C)]}{P(t\|C)[1 - P(t\|R)]}$
[Doszkocs 1978]	Chi-square	$\frac{[P(t\|R) - P(t\|C)]^2}{P(t\|C)}$
[Robertson 1990]	Robertson selective value (RSV)	$\sum_{d \in R} w(t,d) \cdot [P(t\|R) - P(t\|C)]$
[Carpineto 2001b]	Kullback-Leibler distance (KLD)	$p(t\|R) \cdot log \frac{P(t\|R)}{P(t\|C)}$

From the query language modeling we present here the relevance model proposed in [Lavrenko 2001], called Relevance Model 1 (RM1), because it is a widely used model and, moreover, we employ it in our experiments in chapters 4 and 5. Lavrenko and Croft use the query likelihood $P(q|d)$ as the weight for document d and they consider an average of the probability of term t given by each language model of documents. The formula to compute RM1 is the following:

$$P_{RM1}\left(t|q\right) \propto \sum_{\theta_d \in \Theta} P\left(t|\theta_d\right) P\left(\theta_d\right) \prod_{i=1}^{m} P\left(q_i|\theta_d\right), \tag{1.16}$$

where Θ represents the set of smoothed document models in the pseudo-relevant document set R and $q = \{q_1, q_2, \ldots, q_m\}$. In RM1, the Dirichlet smoothing method is used to smooth the language model of each pseudo-relevant document θ_d.

The relevance model $P_{RM1}(t|q)$ can be interpolated with the model of the initial query θ_q in order to obtain performance improvements [Abdul-Jaleel 2004]. Therefore, this interpolated relevance model, called Relevance Model 3 (RM3), is computed as follows:

$$P_{RM3}\left(t|\theta'_q\right) = (1-\lambda)\, P\left(t|\theta_q\right) + \lambda P_{RM1}(t|q) \tag{1.17}$$

We use RM1 and RM3 later on in our experiments, since they yield effective retrieval results, thus they represent strong baselines.

There is no optimal way to automatically expand the queries and the automatic QE process may actually harm system performances. However, when the right choices are made and when the parameters are properly tuned, the expansion is beneficial. For instance, in [Lv 2009] the authors aimed to optimize the balance parameter (λ from RM3 interpolation), by the means of logistic regression. Their improvements were, however, very close to the baseline (from 0.340 to 0.356 over TREC6, TREC7 and TREC8). Therefore, tuning the λ parameter in the RM3 interpolation remains an open problem that we also try to approach in this thesis.

The information relevance to a topic is measured using benchmark collections and the evaluation tasks started in the early 1960s with the Cranfield Experiments, then continued to this day with the tracks from the Text REtrieval Conference (TREC), which represents the main evaluation framework in IR. TREC is an annual workshop hosted by the US government's National Institute of Standards and Technology which provides the necessary infrastructure for the large-scale evaluation of text retrieval methods[2]. In the following sections we discuss the IR evaluation and the benchmark collections.

We have introduced here the classic models of query expansion. For a more detailed related work overview, see chapters 4 and 5.

1.4 Evaluation

The vector space model, the probabilistic model, or the language model, together with their parameter tuning, combinations and variants, represent the wide variety of IRS. This high rate of possible choices regarding IR algorithms raises the question of which model is better.

Yet, the notion of "better" depends of various aspects, such as efficiency and effectiveness. Firstly, the most common dimensions of system efficiency are time and space. A better system from this point of view is one which yields a shorter

[2]`http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=10667`

response time and one which uses a smaller hard disk space. These aspects represent the efficiency level of user satisfaction. The interface quality also plays an important role for user satisfaction [Speicher 2014]. Secondly, there is the quality of response according to the information need. This evaluation perspective measures how well the system managed to retrieve relevant information, with respect to the information need. This represents the effectiveness level of user satisfaction.

One approach to evaluate IRS, in terms of effectiveness, is the explicit feedback by the means of user study. The assumption, in the case of an user study, is that system performance and users' satisfaction are proportional. For this type of evaluation, users are hired, they complete some tasks using the systems and finally they report their subjective feeling. Even if this type of performance evaluation is close to reality, a user study is subjective and expensive in terms of time and money [Kelly 2009]. Of course, experiments on a smaller scale could be set up, but this implies biased results.

An alternative for the user study is the usage of test collections, built by following the Cranfield Paradigm. This paradigm was proposed in the early 1960s at Cranfield College of Aeronautics, England by Cyril Cleverdon, who conducted research on indexing methodologies and who showed that manual indexing was not needed. He also developed this evaluation methodology, which was quickly adopted by the IR community. The paradigm implies a test collection, which consists of a static set of documents, a set of topics and the set of known relevant documents for each of the topics, that is to say the relevance judgments. For the evaluation process it is required for a system to retrieve documents from the collection, for each topic. Evaluation measures are then applied on the retrieved list, on a per topic basis, against the relevance judgments. The scores for each topic are averaged and the resulting score represents the effectiveness score for the IRS. This evaluation paradigm should have an exhaustive assessment, which is expensive and unpractical for nowadays large collections, therefore the assessment based on pooling was introduced by TREC [Harman 1993] and it is employed in practice. The traditional pooling method is to select documents for assessment by running a set of representative retrieval systems against the collection, and pool their top-ranked results. The main advantage regarding the Cranfield approach is that the experiments are reproducible and comparable, a fact very important in a research context, since we need to test models and to compare performance with other systems.

Another facet of evaluation is the implicit feedback, based on user behavior (such as clicks), which obtains relevance judgments from user behavior. The implicit feedback, as the explicit feedback, also involves real users, it is cheaper than the Cranfield paradigm and much larger in terms of sample size [Joachims 2002]. However, it yields user behavior noise and long-tail search distribution. The long-tail distribution is induced by the existence of few popular choices together with a high number of rarely chosen documents, probably due to the fact that the user tends to click on the documents from the top of the retrieved list. The implicit feedback uses a small proportion of traffic for evaluation in two alternative ways:

by showing results from different retrieval methods alternatively, or by merging the results into a document list.

Regarding the evaluation in IR, we may summarize that the ground-truth is represented by the choice of real users, that method evaluation can be decomposed into efficiency, effectiveness and interface quality, that reusable test collections are useful and that user behavior (log) is really important and represents a kind of wealth. In our research we use Cranfield paradigm-based benchmarks, since they provide the full evaluation environment (documents, topics, relevance judgments). Moreover, the evaluation in this context is reproducible and comparable, key factors for research. On the other hand, robust user studies are expensive and difficult to set up, therefore the explicit feedback is not appropriate to evaluate our work. The click models and user logs, that is to say the implicit feedback, are more suitable for the web environment. However, the implicit feedback represents a resource which is not so often available and we do not employ click models and user logs in our contributions.

In the following section we discuss the evaluation measures in the TREC context, which employs the Cranfield Paradigm for evaluation.

1.4.1 Evaluation measures

In order to establish the effectiveness of an IRS, or to rank and compare systems, various performance measures have been proposed in the literature. In TREC, participant runs (see Section 1.4.5) are evaluated using the `trec_eval` package in order to rank the participant systems. This package provides performance measures, including some measures that are derived from the two basic measures in IR: recall and precision. The precision is the fraction of the retrieved documents that are relevant, while the recall represents the fraction of the documents relevant to the query that are successfully retrieved. We describe here the precision at a cut-off level (P@k), the Average Precision (AP) and the Mean Average Precision (MAP), since they are widely employed by the IR community and we usee all of them in our experiments.

The `trec_eval` package also implements the precision at certain cut-off levels. A cut-off level is a rank that defines the retrieved set. For example, a cut-off level of ten defines the retrieved set as the top ten documents in the ranked list (P@10). These top precision measures are important in a real life context, where the user is interested in the relevant documents among the first 5 or 10 retrieved documents. For instance, in the case of a web search, users naturally tend to click on the links from the top of the retrieved list, thus the need for improvements regarding these particular top results. In our experiments, we use three cut-off levels: P@5, P@10 and P@30, which are high precision measures. The cut-off level precision is defined in Definition 9.

Definition 9 (Precision P@k)

The precision P@k at a cut-off level k, for a query q, is defined as:

$$P@k(q) = \frac{|relev@k|}{k}, \tag{1.18}$$

where $relev@k$ represents the set of relevant documents retrieved until the rank k.

Average Precision (AP) is widely used in the literature as it involves both precision and recall. By plotting the precision $p(r)$ as a function of recall r, one would obtain the precision-recall curve. The AP computes the average of $p(r)$, over the interval $r \in [0, 1]$, as expressed in Equation 1.19:

$$AP = \int_0^1 p\left(r\right) dr \tag{1.19}$$

Equation 1.19 represents the area of the surface below the precision-recall curve. In practice, this integral is replaced with a finite sum over the positions in the rank list of retrieved documents, as follows:

$$AP = \sum_{k=1}^{R} P@k\Delta r\left(k\right), \tag{1.20}$$

where R is the number of retrieved documents, k is the rank, P@k is the precision of the top k retrieved documents and $\Delta r\left(k\right)$ is the change in recall from two consecutive retrieved documents $k-1$ and k.

This sum is equivalent to Definition 10.

Definition 10 (Average Precision (AP))

The Average Precision (AP) for a query q is defined as:

$$AP\left(q\right) = \frac{\sum_{k=1}^{R}\left[P@k \times rel\left(k\right)\right]}{relev\left(q\right)}, \tag{1.21}$$

where $relev\left(q\right)$ represents the number of documents relevant to the query q and $rel\left(k\right)$ equals 1 if the kth document is relevant and 0 otherwise.

The AP is a per topic measure, however, when we need to evaluate the performance of a system for a set of topics from a collection, a per system measure should be employed. Thus, the Mean Average Precision (MAP) stands for the mean of the average precision scores over queries. The MAP is a per system measure and its equation is presented in Definition 11.

Definition 11 (Mean Average Precision (MAP))

For a set of queries Q, the Mean Average Precision (MAP) is defined as follows:

$$MAP = \frac{1}{|Q|} \sum_{q=1}^{|Q|} AP(q). \tag{1.22}$$

We have presented here 3 measures for IRS evaluation, but in `trec_eval` there are 135 proposed measures. This fact raises the question of which or how many measures to use in order to obtain a proper evaluation process. Moreover, having this number of available measures, it would be interesting to find out whether these measures are somehow correlated one with each other, or not. In [Baccini 2012] the authors show that the measures can be clustered into highly correlated measure groups. They use 130 measures out of the 135 measures proposed in `trec_eval`. They also define a subset of weakly correlated performance measures by selecting a representative from each cluster. Thus, this representative group of measures could provide better insights regarding the system effectiveness. The authors used test collections from TREC2 to TREC8, together with the evaluations for the official participants at the corresponding TREC competitions. In the case of the most homogeneous clusters, any measure is a good representative of the cluster from the mathematical point of view. For the less homogeneous clusters, on the other hand, Baccini et al. suggest to use the centroid and to consider both the distance to the centroid and the popularity of the measure, in terms of usage. The authors mention that the most homogeneous clusters (compact clusters) are clusters 1, 3, 4 and 5. All the measure clusters are given in Table 1.3.

We notice that the cluster populations are not balanced and we underline that the AP belongs to the 4th cluster 4, while the P@5, P@10 and P@30 are in the 1st cluster. However, P@100 belongs to the 2nd cluster, therefore the P@k measures and the AP are not relatively redundant and the redundancy of P@k measures depends on k. Even though P@5, P@10 and P@30 belong to the same measure cluster, we employ these three measures in our contribution which re-ranks retrieved document lists based on query terms disambiguation (see Chapter 2). As a re-ranking method, our interest stresses on the changes occurred in the top of the document list. We aim to check how much improvement is obtained in the top 5 documents, this being also a realistic scenario when users check only the first retrieved documents. On the other hand, we also would like to know what happens further in terms of improvement, when the document limit reaches 30. All this justifies the usage of both P@k and AP in our research without redundancy.

Table 1.3: Performance measure clusters with their corresponding population, by [Baccini 2012]

Clusters
Cluster 1 (23 measures)
relative_unranked_avg_prec30 • relative_unranked_avg_prec20 • relative_prec30 • map_at_R • relative_unranked_avg_prec15 • relative_prec20 • **P30** • relative_prec15 • int_0.20R.prec • relative_unranked_avg_prec10 • X0.20R.prec • ircl_prn.0.10 • P20 • P15 • relative_prec10 • bpref_10 • **P10** • relative_unranked_avg_prec5 • relative_prec5 • **P5** • bpref_5 • recip_rank • ircl_prn.0.00
Cluster 2 (16 measures)
P100 • P200 • unranked_avg_prec500 • unranked_avg_prec1000 • bpref_num_ret • P500 • bpref_num_all • P1000 • num_rel_ret • exact_unranked_avg_prec • num_rel • exact_prec • bpref_num_correct • utility_1.0_.1.0_0.0_0.0 • exact_relative_unranked_avg_prec • bpref_num_possible
Cluster 3 (12 measures)
bpref_top10Rnonrel • bpref_retnonrel • relative_unranked_avg_prec500 • avg_relative_prec • recall500 • relative_prec500 • bpref_allnonrel • relative_unranked_avg_prec1000 • exact_recall • recall1000 • relative_prec1000 • exact_relative_prec
Cluster 4 (45 measures)
X1.20R.prec • ircl_prn.0.30 • X1.40R.prec • int_**map** • X1.00R.prec • R.prec • int_1.20R.prec • exact_int_R_rcl_prec • int_1.00R.prec infAP • avg_doc_prec • map • X11.pt_avg • X1.60R.prec • int_0.80R.prec • int_1.40R.prec • X0.80R.prec • old_bpref_top10pRnonrel • ircl_prn.0.40 • X1.80R.prec • int_1.60R.prec • X3.pt_avg bpref • X2.00R.prec • bpref_top25p2Rnonrel • old_bpref • bpref_top10pRnonrel • int_1.80R.prec • int_0.60R.prec • int_2.00R.prec • bpref_top25pRnonrel • X0.60R.prec • bpref_top50pRnonrel • bpref_top5Rnonrel • ircl_prn.0.20 • ircl_prn.0.50 • int_0.40R.prec • X0.40R.prec • int_map_at_R • ircl_prn.0.60 • unranked_avg_prec30 • ircl_prn.0.70 • ircl_prn.0.80 • unranked_avg_prec200 • unranked_avg_prec100
Cluster 5 (18 measures)
bpref_topnonrel • fallout_recall_42 • fallout_recall_28 • fallout_recall_56 • rcl_at_142_nonrel • fallout_recall_71 • fallout_recall_85 • relative_unranked_avg_prec100 • fallout_recall_99 • fallout_recall_113 • relative_prec100 • fallout_recall_127 • relative_unranked_avg_prec200 • fallout_recall_142 • recall100 • relative_prec200 • recall200 • bpref_retall
Cluster 6 (13 measures)
fallout_recall_14 • unranked_avg_prec20 • unranked_avg_prec15 • ircl_prn.0.90 • fallout_recall_0 • unranked_avg_prec10 • recall30 • ircl_prn.1.00 • recall20 • recall15 • unranked_avg_prec5 • recall10 • recall5
Cluster 7 (3 measures)
rank_first_rel • num_nonrel_judged_ret • num_ret

We have seen that evaluation measures are employed in order to establish the performance of an IRS. However, when one tries to improve performance, compared with a baseline result, the relative improvement percentage or the average improvement may be computed. This raises the question whether these improvements are statistically significant, or not. Thus, statistical tests are required. In the following section we present the correlation coefficient employed to check the correlation between two variables and the statistical significance test employed to check our experimental results, when compared to various baselines.

1.4.2 Statistical measures

When evaluating a model, we compare our results in terms of an evaluation measure (for example, AP or P@k) with baseline results. In order to check if the differences between these two sets of results are statistically different, statistical tests are required.

Whenever wanting to verify if two variables are related to each other, one can use a measure called correlation coefficient. For example, we propose in our contributions query difficulty predictors (see Chapter 3), thus we need to check the correlation between the predicted value and the true value of a difficulty measure, in order to evaluate the prediction quality. Another example would be checking the correlation between two lists of AP values.

We faced the choice between Pearson's and Spearman's correlation coefficients. When the data is basically elliptically distributed and with no important outliers, both of these coefficients yield similar results. However, the Spearman correlation is less sensitive to strong outliers than the Pearson correlation [Hauke 2011]. This fact is due to the outlier limitation to the value of its rank, in the case of Spearman's correlation. Our choice in terms of correlation coefficient is represented by the Spearman's coefficient, defined by Definition 12.

Definition 12 (Spearman correlation coefficient)

Given a sample of size n, the n raw scores X_i and Y_i are converted to ranks x_i and y_i and the Spearman correlation coefficient ρ is computed from:

$$\rho = 1 - \frac{6\sum d_i^2}{n\left(n^2 - 1\right)}, \tag{1.23}$$

where $d_i = x_i - y_i$ and represents the rank difference.

The Student's t-test can be used to establish whether two data sets are significantly different from each other, or not. This test has two parameters: the number of tails (1 or 2) and the type of test (paired and unpaired). The tails number regards the direction of the difference between the two data sets, meaning which one was predicted to be lower. In the case of the AP values per query, for the baseline and for the results, respectively), we do not predict which group is lower, thus the choice for a 2 tails test. A paired test is when the data sets represent multiple observations (values of an effectiveness measure for two IR systems, in our case) from the same individual (query, in our case). Thus, our choice is represented by the paired test.

The test statistics produced by t-test can be interpreted using the p-value. In a statistic test the *null hypothesis* represents an "uninteresting" statement, such as "no difference" between the two tested groups. In our case, the null hypothesis represents the statement that "the obtained results are not different from the baseline". The opposite is the *alternative hypothesis*, that is the "interesting" statement that would conclude an experiment, if the data allows it. The p-value is related to the null hypothesis and represents the probability of obtaining the experimental results when this null hypothesis is actually true. It is generally considered that results are statistically significant, in a conventional way, if the p-value is less than 0.05 (a threshold of 5%). In the case of two AP lists, a p-value less than 0.05 means that there are less than 5% chance that the lists are not different.

For example, we propose query difficulty predictors in our contributions, thus we need to check the correlation between the predicted value and the true value of a difficulty measure, in order to evaluate the prediction quality.

We have presented so far the concepts of relevance, IR models, query expansions techniques and the evaluation framework. In the following section we present the test collection on which the evaluation measures are applied in order to obtain the performance of our models.

1.4.3 Benchmark collections

For IR tasks there exist various benchmark collections for evaluation. The idea is that the IR community has a common ground in order to compare the performance of their methods, models or algorithms. One of the most important sets of test collections, which is widely used by academics, comes from the TREC Conference.

The TREC ad hoc and web tasks allow researchers to investigate the performance of systems that search a static set of documents using new information needs (called topics). We opted for TREC Robust collection for use in the ad hoc task and for WT10G and GOV2 in the case of web track. These collections are very popular in the literature and having two types of benchmarks, ad hoc and web, involves a wider evaluation perspective.

In the case of TREC Robust, the competition provided approximately 2 gigabytes worth of documents and a set of 250 natural language topic statements (per collection). Subsets of these 250 topics represent two test collections known by the name of TREC7, having 50 topics and TREC8, with 50 topics, respectively. The documents were articles from newspapers like the Financial Times, the Federal Register, the Foreign Broadcast Information Service and the LA Times. The WT10G collection provided approximately 10 gigabytes worth of Web/Blog page documents with its 100 corresponding topics. Finally, the GOV2 collection contains 25 million documents crawled from .gov sites, with 150 topics. The total size of the GOV2 collection is of 426 gigabytes. In Table 1.4 we present a summary of benchmark collections features regarding the topics, the documents, the disk space required for each set of documents and the MAP value of the best participant system for each campaign.

Table 1.4: Topic and document features from the data collections

Collection	No. of topics	Topic number	No. of documents	Space on disk	Best MAP (participant)
TREC Robust	250	301-450; 601-700	528,155	2GB	0.3330
TREC7	50	351-400	528,155	2GB	0.3701
TREC8	50	401-450	528,155	2GB	0.3207
WT10G	100	451-550	1,692,096	10GB	0.2226
GOV2	150	701-850	25,205,179	426GB	0.2840

1.4.4 TREC topics and queries

TREC distinguishes between a statement of information need (the *topic*) and the text that is actually processed by a retrieval system (the *query*). The TREC test collections provide topics. What is now considered the "standard" format of a TREC topic statement comprises a topic *ID*, a *title*, a *description* and a *narrative*. The title contains two or three words that could correspond to key words a user could have used as a query in a search engine. The description contains one or two sentences that describe the topic area. The narrative part gives a concise description of what makes a document relevant, or not relevant [Voorhees 1998]. Queries can be constructed out of any combination between the three topic parts. However, the trends in the IR community suggest the usage of the title part of the topic as query, since it is the closest to a real world scenario, when the user types few words in a search engine. Below, we present a sample query from the TREC collections.

Listing 1.1: TREC query sample

```
<top>

<num> Number: 301
<title> International Organized Crime

<desc> Description:
Identify organizations that participate in international criminal
activity, the activity, and, if possible, collaborating organizations
and the countries involved.

<narr> Narrative:
A relevant document must as a minimum identify the organization and
the type of illegal activity (e.g., Columbian cartel exporting
cocaine).
Vague references to international drug trade without identification
of the organization(s) involved would not be relevant.

</top>
```

The relevance judgements for each collection are also given. The relevance judgements files are called *qrels* and they contain on each line the query number, a document name and the relevance level of that document for the query in cause, each field separated by space. For the collections mentioned in this section there are binary relevance judgements, a document having the value 1 if it is relevant for the query and the value 0, otherwise.

1.4.5 TREC runs and participants

We have presented the TREC framework, with the documents, queries and relevance judgements. When an IRS runs retrieval over the benchmark documents, for queries it retrieves ranked documents per query. TREC requires these results in a specifically formatted file.

The results file has the format: *query_id*, *iter*, *docno*, *rank*, *score*, *run_name* delimited by spaces. *query_id* represents the query number, for example 351 is the first query of TREC7. The iteration constant *iter* is required. The document numbers *docno* are string values like FR940104-0-00001 and is to be found in collection documents, between <DOCNO> tags. The score value *score* is a float value and stands for the score obtained by the document from the field *docno* for the query represented by *query_id*. The rank field *rank* is an integer starting from 0 and it is used to rank the documents retrieved for a particular query. The last field stands for the name of the run, *run_name*. The whole result file must be sorted numerically by the field *query_id*. An example of several lines of a results file is given in Listing 1.2.

Listing 1.2: TREC run sample

```
351 Q0 FT941-9999 0 23.995955581970943 BB2c1.0
351 Q0 FT934-4848 1 23.020949953171677 BB2c1.0
351 Q0 FT922-15099 2 22.28826345449878 BB2c1.0
...
352 Q0 FT923-2780 727 6.909926252150617 BB2c1.0
352 Q0 FT923-1627 728 6.905332926462592 BB2c1.0
352 Q0 FT931-9306 729 6.905202601541681 BB2c1.0
...
400 Q0 FR940804-0-00041 997 12.194091569964083 BB2c1.0
400 Q0 LA080890-0042 998 12.19372652596313 BB2c1.0
400 Q0 LA081690-0008 999 12.192530792085257 BB2c1.0
```

For each TREC workshop, the organizers propose a set of test queries and the campaign participants must provide their result files, also named *runs*, after retrieving documents for those queries. All the files are evaluated and the participant runs are ranked with respect to their MAP results. Past participant runs are accessible through a payed subscription to TREC resources. We employ these kind of runs in our experiments for baselines and also for uery difficulty prediction (see Section 3). For instance, in TREC7 ad hoc track there have been 56 participants from 13 countries. This diversity of participants has ensured different IR approaches. The best participant results, per collection, are mentioned in Table 1.4 from Section 1.4.4.

1.4.6 IR platforms

By following the evaluation framework, we have used search engines with various parameter set ups in order to produce lists of retrieved documents, or runs. Our models also involve statistical, linguistic and classification tools. We present here the search engines and the software tools employed for our experiments.

1.4.6.1 Terrier

Our first choice regarding search engines is Terrier. Terrier is an open source platform that implements state-of-the-art indexing and retrieval models. Terrier is written in Java and it was developed at the School of Computing Science, University of Glasgow[3] [Ounis 2006].

With Terrier, document collections can be indexed, using different indexing strategies, such as single-pass or multi-pass and different stemming options. Regarding the retrieval part, Terrier offers several state-of-the art retrieval approaches such as Divergence From Randomness, BM25F [Zaragoza 2004] and term dependence proximity models. However, classical models, such as BM25 [Robertson 1994], are also implemented. We have presented several of these models in Section 1.2.

A very important aspect is that Terrier can index and perform batch retrieval over the TREC collections. Nevertheless, the possibility of customizing or implementing new search models is given by the available plugin architecture.

The query expansion functionality is present in Terrier's implementation. Different query expansion models, including Bo1, Bo2 and KL (all in [Amati 2004a]) are available. Besides that, several parameters are adjustable. For example, the number of documents considered for expansion, or the number of terms to be added to newly formed query, can be changed by the user.

Overall, Terrier represents a customizable and self contained retrieval environment, well adapted for our experimental needs.

1.4.6.2 Indri

The other search engine we used is Indri, an open source search engine that implements language model IR. Indri is a part of the Lemur Project and it is developed by the University of Massachusetts together with the Carnegie Mellon University[4]. Its API works with Java, C++, or PHP.

Indri is well suited for large text collections and, as Terrier, it can deal with TREC collections. A mild downside is that TREC topics must be converted to a certain XML-like format in order to be processed by Indri.

[3]`http://www.terrier.org`
[4]`http://www.lemurproject.org/indri/`

For the basic retrieval, Indri provides the query likelihood model (QL) with different parametrizable smoothing functions (such as Dirichlet smoothing), all introduced in Section 1.2.4. The queries can be expanded using the blind relevance feedback (RM1), or the Relevance Model 3 (RM3) [Lavrenko 2001], which represents an interpolation between RM1 and the initial query, as described in Section 1.3.

1.5 Conclusion

In this chapter we have presented the concept of relevance, from its definition and characteristics to the challenges raised by seeking relevant information.

Next, we detailed several IR models, with their principles and mathematical formulation. Query expansion is one technique to improve IR performance and we described several models from the literature.

We continued by explaining the IR evaluation process. We decomposed the notion of evaluation into different aspects and we presented the existing evaluation frameworks with their strong and weak points, respectively. Evaluation measures are also presented together with statistical significance tests and correlation coefficients, used in our studies and which quantify the significance of the obtained results.

Finally, we described the benchmark collections involved in our experiments and the IR platforms employed in our experimental framework throughout our work, in order to obtain a clear overview of the implementation process.

Next, we present our contributions and we start in the following chapter with our term disambiguation method which is used to re-rank retrieval results, in order to improve the top precision.

Chapter 2

Word sense discrimination and query ambiguity in information retrieval

Contents

Word sense ambiguity has been identified as a cause of poor precision in IRS [Schütze 1995]. Word Sense (WS) disambiguation and discrimination methods have been defined to help systems choose which documents should be retrieved in relation to an ambiguous query [Schütze 1995]. However, the only approaches that show a genuine benefit for word sense discrimination or disambiguation in IR are generally supervised ones [Zhong 2012]. Still, the supervised disambiguation cannot be used in practice on a large scale due to the absence of the necessary annotated corpora. In this chapter we propose a new unsupervised method that uses word sense discrimination in IR. The method we develop is based on unsupervised clustering and reorders an initially retrieved document list by boosting documents that are semantically similar to the target query. We consider two clustering techniques for our method, in order to analyze the impact of the clustering technique on the final results. The first clustering method is based on Naïve Bayes classification and the other is based on spectral clustering, which represents a state-of-the-art method. For several TREC ad hoc collections we show that our method is useful in the case of queries which contain ambiguous terms. We are interested in improving the level of precision after 5, 10 and 30 retrieved documents (P@5, P@10, P@30), since users are naturally interested in finding relevant information in the top of the retrieved list. We mention that the method with spectral clustering is more effective, since it surpasses the baselines in most cases, while the Naïve Bayes based method yields results beyond baseline only for queries with low performance. This motivates our focus on poor performing queries, opening the lead for this type of queries to become a central part in the following chapters of this thesis. Our method requires context for queries. We simulate this context by using the description and narrative part of TREC topics (see Section 2.4.1). However, in real applications, the context is not available, therefore we also propose an automatically generated context for queries. This research has been published in CEJCS[1] [Chifu 2012], for the Naïve Bayes method and in IPM[2] [Chifu 2015], for the spectral clustering method with a deeper analysis.

2.1 Introduction

According to Lin [Lin 1997], "given a word, its context and its possible meanings, the problem of Word Sense (WS) disambiguation is to determine the meaning of the word in that context"[3]. Although WS disambiguation is generally easy for humans, it represents an issue for computers. The problem becomes even more difficult to solve when an ambiguous word occurs in short chunks of texts, such as a query in

[1] Central European Journal of Computer Science

[2] Information Processing & Management

[3] For a complete discussion of state-of-the-art WS disambiguation see the monograph [Agirre 2006].

an IRS.

Applying WS disambiguation to improve IR results is a well studied problem, but with controversial results as evidenced in the literature. Several authors have concluded that WS disambiguation in IR does not lead to significant retrieval performance improvement [Sanderson 1994], [Guyot 2008]. Various studies [Krovetz 1992], [Voorhees 1993], [Uzuner 1999] have argued that the main problem in improving retrieval performance when using WS disambiguation is the inefficiency of the existing disambiguation algorithms, a problem which increases in the case of short queries.

In more recent years the issue remained "as to whether less than 90% accurate automated WS disambiguation can lead to improvements in retrieval effectiveness" [Stokoe 2003]. This remark refers primarily to the traditional task of WS disambiguation which identifies the meaning of the ambiguous words in context. This type of WS disambiguation is generally based on external sources, such as dictionaries or WordNet (WN)-like knowledge bases for labeling senses [Guyot 2008], [Carpineto 2012] and is therefore knowledge-based.

Attempts to use knowledge-based WS disambiguation in IR have been numerous. In [Gonzalo 1998] as well as in [Mihalcea 2000] positive results were reported. These studies made use of semantic indexing based on WN synsets. However, they were all conducted on small data sets. As commented in [Ng 2011], the evaluation is scaled up to a large test collection in [Stokoe 2003] but the reported improvements are from a weak baseline. Positive results are also reported in [Kim 2004], although the quantum of improvements is small.

Zhi Zhong and Hwee Tou Ng [Zhong 2012] are among the few authors who more recently have expressed a growing belief in the benefits brought by WS disambiguation to IR - when using a supervised WS disambiguation technique, based on annotated parallel corpora. However, the annotated or parallel corpora are not available in a real context, making unsupervised methods unpractical.

In contrast to all these authors, we are suggesting and investigating the usage of an unsupervised WS disambiguation technique. In this chapter, we present an approach that aims at identifying clusters from similar contexts, where each cluster shows a polysemous word being used for a particular meaning. In this manner, this type of analysis is more suited for the application in IR. Our approach is therefore not concerned with performing a straightforward WS disambiguation, but rather with differentiating among the meanings of an ambiguous word. Considering WS discrimination rather than straightforward WS disambiguation avoids the use of external sources such as dictionaries or WN type synsets which are commonly used [Carpineto 2012].

We hereby propose a new WS discrimination method for IR based on Naïve Bayes and spectral clustering. Firstly, from the wide range of unsupervised learning techniques that could be applied to our problem, we have chosen to use a parametric model to assign a sense group to an ambiguous occurrence of a so-called target word. In each case we shall assign the most probable group given the context as defined by the Naïve Bayes model, where the parameter estimates are formulated via unsupervised techniques. We employ WN-based feature selection, which has provided good disambiguation results in the case of all parts of speech [Hristea 2008], [Hristea 2009a].

On the other hand, there is the spectral clustering. This state-of-the-art clustering technique is now a hot topic; for example, in [Bellugi 2015] is presented a spectral clustering search algorithm for predicting shallow landslide size and location, in [Wang 2014] the authors recently discussed about the constrained spectral clustering and its applications, while Borjigin and Guo [Borjigin 2012] studied the cluster number determination in spectral clustering. Spectral clustering has been used in WS discrimination for the first time by Popescu and Hristea [Popescu 2011] who point out the importance of the clustering method used in unsupervised WS disambiguation. This importance would also be reflected in the IR application results. For instance, in the case of the Naïve Bayes model, in spite of performing WN-based feature selection, we were not able to overcome the baseline when considering all queries, and therefore we only targeted the subset with lowest precision. On the other hand, when using spectral clustering, which has its own feature weighting, the same baseline is, in most cases, surpassed.

The WS discrimination-based re-ranking method is summarized in Algorithm 2.1.

Algorithm 2.1 Re-ranking method based on unsupervised WS discrimination

```
Input: TREC topics, document collection
 1: Consider the title part of a TREC topic as a query
 2: Identify the part of speech for each query term
 3: Check each query term if it is ambiguous[4]
 4: Select the queries that contain at least one ambiguous term.
 5: Retrieve the set of documents for each ambiguous query.
 6: Add context to queries (descriptive and narrative topic parts).
 7: Add contextualized queries from step 6 to the retrieved document sets.
 8: Obtain document clusters (# clusters = # WN synsets), by performing WS
    discrimination for each query and for each ambiguous term, with one of the
    methods:
      • Naïve Bayes classification
      • Spectral clustering
 9: For each ambiguous query term, select the document cluster where the query
    itself was assigned.
10: If there are more than one ambiguous terms per query, combine the document
    clusters selected at step 9 for each individual term.
11: Re-rank initial retrieved document lists by giving more importance to docu-
    ments that are in the query cluster.
Output : Re-ranked document lists for each query
```

[4]A term is ambiguous if it has more than one synsets in WN, for its part of speech (step 2).

Our re-ranking method based on unsupervised WS discrimination requires context for queries. We simulate this context by employing the text from the description and the narrative parts of TREC topics (see Section 2.4.1). However, in real applications, the queries are short (2 or 3 words) and the context is not available, therefore we also propose here a method to automatically generate context for short queries, based on pseudo-relevance feedback. The re-ranking method is tested with this automatically generated query context.

The present chapter is organized as follows: in Section 2.2 we present the related works on WS disambiguation in IR; the focus is on unsupervised methods. Section 2.3 presents word sense discrimination based on Naïve Bayes classification and spectral clustering. Section 2.4 presents the two step IR process using the proposed WS discrimination model. The first step is performing the query WS discrimination and the second step re-ranks the retrieved documents according to the query WS discrimination results. The evaluation is presented in Section 2.5. A more thorough analysis of the obtained results with spectral clustering is performed in Section 2.7, by analyzing the improvements by precision intervals and by the number of ambiguous terms in queries, respectively. The disambiguation context is represented by the description and narrative parts of a TREC topic (see Section 1.4.4 for details). In Section 2.8 we introduce an automatically generated context and we study its lays out the on the results of our spectral clustering-based method. Section 2.9 concludes this chapter.

2.2 Related work

WS ambiguity is a central concern in natural language processing (NLP). SENSEVAL defined the first evaluation framework for WS disambiguation in NLP [Kilgarriff 1997]. According to Kilgarriff and Rosenzweig [Kilgarriff 2000], SENSEVAL participants defined systems that can be classified into two categories: supervised systems, which use training instances of sense-tagged words and non-supervised systems. According to [Navigli 2009], supervised systems are typically employed when a restricted number of words have to be disambiguated, while this type of system encounters more difficulties when all open-class words from a text have to be disambiguated. In addition to general WS disambiguation, many recent papers consider disambiguation of person names [Artiles 2007], [Piskorski 2009], [D'Angelo 2011] and disambiguation of place names [Leidner 2007]. Indeed, WS disambiguation has many applications, such as text processing, machine translation and IR, for which this type of disambiguation – proper names – can be useful (although not sufficient).

Multiple approaches from the literature, including ours, are using external linguistic resources such as WN. WN [Miller 1995] is a large lexical database of English. We mention that the project is developped for other languages through the Global WordNet Associaton. This computational linguistics and natural language processing tool was started by George Miller, at the Princeton University. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms called synsets, each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations[5]. The English version of the software is available freely for research and for commercial usage and one can use the stand alone application, the online application, or the API in order to integrate WN features in newly developed software. WN resembles a thesaurus, since it groups words together based on their meanings. However, there are some differences, such as labeled semantic relations between words and the semantic disambiguation for words in proximity, in the case of WN. The majority of the WN's relations connect words from the same part of speech (POS). Thus, WN really consists of four sub-nets: for nouns, for verbs, for adjectives and for adverbs, with few cross-POS pointers.

Krovetz and Croft [Krovetz 1992] were among the first to conduct a thorough analysis of ambiguity in IR. They used the CACM and TIME test collections and compared query word sense with word senses in retrieved documents. They found that sense mismatch occurs more often when the document is non-relevant to the query and when there are few common words bridging the query and the retrieved document. Another large scale study of word sense disambiguation in IR was conducted by Voorhees [Voorhees 1993]. The automatic indexing process she developed used the "is-a" relations from WN and constructed vectors of senses to represent documents and queries. This approach was compared to a stem-based approach for 5 small collections (CACM, CISI, CRAN, MED, TIME). The results showed that the stem-based approach was superior overall, although the sense-based approach improved the results for some queries [Voorhees 1993]. Sanderson [Sanderson 1994] used the Reuters collection in his experiments and showed that disambiguation accuracy should be of at least 90% in order for it to be of practical use. He used pseudo-words in his experiments.

Schütze introduced word sense discrimination in IR [Schütze 1995], [Schütze 1998]. Schütze considers that, in some cases, WS disambiguation can be defined as a two-stage process: first sense discrimination, then sense labeling. Sense discrimination aims at classifying the occurrences of a word into categories that share the same word sense. This type of approach is quite distinct from the traditional task of WS disambiguation, which, as already mentioned, classifies words relative to existing senses. Schütze and Pedersen [Schütze 1995] created a lexical co-occurrence based thesaurus. They associated each ambiguous term with a word vector where coordinates correspond to co-occurring term frequencies. Words with the same meaning were assumed to have similar vectors. Word vectors were clustered together to determine the word uses. Similarity was based on the cosine measure. The application in IR consisted of modifying the standard word-based vector-space model. The words from the "bag of words" text representation were replaced by word senses. Evaluation of TREC 1 showed that average precision is improved when using sense-based retrieval rather than word-based retrieval. Combining word and sense-

[5]`http://wordnet.princeton.edu/`

based retrieval improves precision as well. They were the first to demonstrate that disambiguation, even if imperfect, can indeed improve text retrieval performance [Schütze 1995].

Schütze's [Schütze 1998] context group discrimination uses a form of average link clustering known as McQuitty's Similarity Analysis. Schütze adapts LSI/LSA so that it represents entire contexts rather than single word types using second-order co-occurrences of lexical features. The created clusters are made up of contexts that represent a similar or related sense. In [Schütze 1998] it is again shown that unlabeled clusters of occurrences of a word representing the same sense result in improved IR. The tests are conducted over the TREC-1, Category B collection and the reported results are up to 14.4% improvement in terms of AP. We reimplement this method and we use it as comparison with our results.

Unlike that described in [Schütze 1995] and [Schütze 1998], the method we propose here is based on re-ranking and not on modifying document representation. Re-indexing the entire corpus when new queries are submitted, due to changes in documents, would represents an important time and resource consuming event.

The approaches that show benefit for are generally supervised ones. Zhi Zhong and Hwee Tou Ng [Zhong 2012] constructed their supervised WS disambiguation system directly from parallel corpora. Experimental results on standard TREC collections show that, using the word senses tagged by this supervised WSD system, significant improvements over a state-of-the-art IR system can be obtained [Zhong 2012].

To overcome the fact that supervised methods are unpractical, we propose here an efficient unsupervised method for WS discrimination in IR.

This method consists in proposing the application of two different clustering techniques to implement our method. Our aim is not only to stress on the benefits of unsupervised WS discrimination in IR, but also to point out the importance of the clustering technique involved in this task. While in the case of the Naïve Bayes model, in spite of performing WN-based feature selection, we were not able to move beyond the baseline when considering all queries, and therefore only targeted the lowest precision ones, we hereby show that, when using spectral clustering (that performs its own feature weighting) the same baseline is, in most cases, surpassed.

Analysis of the results provided by the proposed method, with both clustering techniques, will be carried out (see Section 2.6) against the other major approach existing in the literature [Schütze 1995]. The obtained results will be shown as promising in sustaining the concept of sense discrimination being beneficial for IR applications, especially when used from a re-ranking perspective.

2.3 Clustering-based WS discrimination

WS discrimination can be considered as a clustering problem since a way to solve it is to group the contexts of an ambiguous word into a number of groups and to discriminate between these groups without labeling them. However, clustering linguistic data is a difficult task due to its highly complex structure. Recently, a variety of clustering algorithms have been proposed in order to deal with situations where the data is complex, not linearly separable and where the clusters are non-convex. We present here our choice in terms of clustering method, that is Naïve Bayes classification and spectral clustering.

When using a Bayesian classifier in the context of WS discrimination, the algorithm looks at the words around an ambiguous word within the so-called context window. Each content word contributes with useful information concerning which sense of the ambiguous word is more likely to be used with it. The feature in this context are all content words occurring in the context window of the target word. The classifier does no feature selection, instead it combines the evidence coming from all features [Manning 1999]. This fact generates a so-called "bag of words model", based on the Naïve Bayes assumption that all these content words are conditionally independent. The assumption is clearly not true in the case of natural language, but has provided very good practical disambiguation results. As commented in [Manning 1999], there is a surprisingly large number of cases in which the Naïve Bayes assumption does well, "partly because the decisions made can still be optimal even if the probability estimates are inaccurate due to feature dependence".

Still in the context of not linearly separable data and non-convex clusters, two related families of methods, kernel and spectral methods, have proven to be very effective in solving different tasks.

In computational linguistics, spectral clustering has been used for machine translation [Gangadharaiah 2006], [Zhao 2005], name disambiguation for author citation identification [Han 2005], and in unsupervised WSD [Popescu 2011].

Spectral clustering has been used in WSD for the first time by Popescu and Hristea [Popescu 2011] who pointed out the importance of the clustering method used in unsupervised WSD. Spectral clustering has been shown [Popescu 2011] as strong enough to make up for the lack of external knowledge of all types, solving many problems on its own, including that of feature selection for WSD. Disambiguation results, after using an unsupervised algorithm based on spectral clustering (that uses its own feature weighting) were superior to those obtained using a classical unsupervised algorithm (with an underlying Naïve Bayes model, for which feature selection was performed) for all parts of speech [Popescu 2011].

The disambiguation accuracy obtained when using clustering in unsupervised WSD, relative to all parts of speech, encouraged us to adopt these two clustering techniques for WS discrimination in the context of IR.

2.3.1 Unsupervised WS discrimination with an underlying Naïve Bayes model

The algorithm for WSD that is used here exemplifies an important theoretical approach in statistical language processing: Bayesian classification [Gale 1992].

In order to formalize the Bayesian model in the context of WSD, we shall follow [Hristea 2008] and we shall present the probability structure of the corpus $\mathcal{C}$. The following notations will be used: w is the word to be disambiguated (target word); $s_1, ..., s_K$ are possible senses for w; the $c_1, ..., c_I$ are contexts of w in a corpus $\mathcal{C}$, as described in Section 2.3.1.1; and the $v_1, ..., v_J$ are words used as contextual features for the disambiguation of w.

2.3.1.1 The probability model of the corpus, the Bayes classifier and parameter estimation

Let us note that the contextual features could be some attributes (morphological, syntactical, etc.), or they could be actual "neighboring" content words of the target word. The contextual features occur in a fixed position near w, in a window of fixed length, centered or not on w. In what follows, a window of size n will denote taking into consideration n content words to the left and n content words to the right of the target word, whenever possible. The total number of words taken into consideration for disambiguation will therefore be $2n+1$. When not enough features are available, the entire sentence in which the target word occurs will represent the window of context.

The probability structure of the corpus is based the assumption that *the contexts* $\{c_i, i = 1, ..., I\}$ *in the corpus* $\mathcal{C}$ *are independent.* Hence, the likelihood of $\mathcal{C}$ is given by the product defined in (2.1):

$$P(\mathcal{C}) = \prod_{i=1}^{I} P(c_i) \tag{2.1}$$

This assumption is quite natural, as the contexts are not connected because they occur at significant distance one from another in $\mathcal{C}$.

Considering the possible senses of each context, one gets

$$P(\mathcal{C}) = \prod_{i=1}^{I} \sum_{k=1}^{K} P(s_k) \cdot P(c_i \mid s_k) \tag{2.2}$$

A model with independent features (usually known as the Naïve Bayes model) assumes that the contextual features are conditionally independent. That is:

$$P\left(c_i \mid s_k\right) = \prod_{v_j \ in \ c_i} P\left(v_j \mid s_k\right) = \prod_{j=1}^{J} \left(P\left(v_j \mid s_k\right)\right)^{|v_j \ in \ c_i|}, \tag{2.3}$$

where $|v_j \ in \ c_i|$ denote the number of occurrences of feature v_j in context c_i. The likelihood of the corpus $\mathcal{C}$ is then

$$P\left(\mathcal{C}\right) = \prod_{i=1}^{I} \sum_{k=1}^{K} P\left(s_k\right) \prod_{j=1}^{J} \left(P\left(v_j \mid s_k\right)\right)^{|v_j \ in \ c_i|} \tag{2.4}$$

The parameters of the probability model with independent features are:

$$\left\{P\left(s_k\right), k = 1, ..., K \ \ and \ \ P\left(v_j \mid s_k\right), j = 1, ..., J, k = 1, ..., K\right\},$$

where

- $P\left(s_k\right) = \alpha_k$, $k = 1, ..., K$, $\alpha_k \geq 0$ for all k, $\sum_{k=1}^{K} \alpha_k = 1$;
- $P\left(v_j \mid s_k\right) = \theta_{kj}$, $k = 1, ..., K$, $j = 1, ..., J$, $\theta_{kj} \geq 0$ for all k and j, $\sum_{j=1}^{J} \theta_{kj} = 1$ for all $k = 1, ..., K$.

With this notation, the likelihood of the corpus $\mathcal{C}$ can be written as:

$$P\left(\mathcal{C}\right) = \prod_{i=1}^{I} \sum_{k=1}^{K} \alpha_k \prod_{j=1}^{J} \left(\theta_{kj}\right)^{|v_j \ in \ c_i|}. \tag{2.5}$$

The well known Bayes classifier involves a posteriori probabilities of the senses, calculated by the Bayes formula for a specified context c,

$$P\left(s_k \mid c\right) = \frac{P\left(s_k\right) \cdot P\left(c \mid s_k\right)}{\sum\limits_{k=1}^{K} P\left(s_k\right) \cdot P\left(c \mid s_k\right)} = \frac{P\left(s_k\right) \cdot P\left(c \mid s_k\right)}{P\left(c\right)}, \tag{2.6}$$

with the denominator independent of senses.

The Bayes classifier chooses the sense s' for which the a posteriori probability is maximal (sometimes called the Maximum A Posteriori classifier)

$$s' = \underset{k=1,...,K}{\arg\max} P\left(s_k \mid c\right) \tag{2.7}$$

Taking into account the previous Bayes formula, one can define the Bayes classifier by the equivalent formula

$$s' = \underset{k=1,...,K}{\arg\max} \left(\log P\left(s_k\right) + \log P\left(c \mid s_k\right)\right) \tag{2.8}$$

Of course, when implementing a Bayes classifier, one has to estimate the parameters first.

Parameter estimation is performed by the Maximum Likelihood Method, for the available corpus $\mathcal{C}$. That is, one has to solve the optimization problem

$$\max\left(\log P\left(\mathcal{C}\right) \mid \left\{P\left(s_k\right), k=1,...,K \;\; and \;\; P\left(v_j \mid s_k\right), j=1,...,J, k=1,...,K\right\}\right).$$

For the Naïve Bayes model, the problem can be written as

$$\max\left(\sum_{i=1}^{I}\log\left(\sum_{k=1}^{K}\alpha_k\prod_{j=1}^{J}\left(\theta_{kj}\right)^{|v_j \;\; in \;\; c_i|}\right)\right) \tag{2.9}$$

with the constraints:

$$\sum_{k=1}^{K}\alpha_k = 1$$
$$\sum_{j=1}^{J}\theta_{kj} = 1 \;\; for \;\; all \;\; k=1,...,K$$

For unsupervised disambiguation, where no annotated training corpus is available, the maximum likelihood estimates of the parameters are usually constructed by means of the Expectation - Maximization (EM) algorithm [Dempster 1977]. The EM algorithm is known as a very successful iterative method, very well fitted for models with missing data (which, in our case, are the senses of the ambiguous words). It has been used by us as well for parameter estimation, closely following [Hristea 2009b].

If, for simplicity, we denote the vector of parameters[6] by

$$\psi = \left(\alpha_1,...,\alpha_K,\theta_{11},...,\theta_{KJ}\right) \tag{2.10}$$

then it is well known that the EM iterations $\left(\psi^{(r)}\right)_r$ converge to the Maximum Likelihood Estimate

$$\widehat{\psi} = \left(\widehat{\alpha}_1,...,\widehat{\alpha}_K,\widehat{\theta}_{11},...,\widehat{\theta}_{KJ}\right). \tag{2.11}$$

Once the parameters of the model have been estimated[7], we can disambiguate contexts of w by computing the probability of each of the senses based on features v_j occurring in the context c. Making the Naïve Bayes assumption and using the Bayes decision rule, we can decide s' if

[6]Let us notice that the number of independent components (parameters) is $(K-1)+(KJ-K)=KJ-1$.

[7]For a detailed presentation of the involved computation, see [Hristea 2009b].

$$s' = \underset{k=1,\dots,K}{\arg\max} \left(\log \widehat{\alpha}_k + \sum_{j=1}^{J} |v_j \; in \; c| \cdot \log \widehat{\theta}_{kj} \right) \tag{2.12}$$

When the Naïve Bayes model is applied to supervised disambiguation, the actual words occurring in the context window are usually used as features. This type of framework generates a great number of features and, implicitly, a great number of parameters. As noted in [Hristea 2008], this can dramatically decrease the model performance since the available data is usually insufficient for the estimation of the great number of resulting parameters. A situation that becomes even more drastic in the case of unsupervised disambiguation, where parameters must be estimated in the presence of missing data (the sense labels). In order to overcome this problem, the various existing unsupervised approaches to WSD implicitly or explicitly perform a feature selection. Of the possible ways of performing feature selection the present study implements WN-based feature selection as described in [Hristea 2008]. This approach to WSD places the disambiguation process at the border between unsupervised and knowledge-based techniques. It is based on a set of features formed by the actual words occurring near the target word (within the context window) and reduces the size of this feature set by performing knowledge-based feature selection that relies entirely on WN. The WN semantic network provides the words considered relevant for the set of senses taken into consideration corresponding to the target word. In our experiments, presented in Section 2.6, WordNet 3.0 has been used.

First of all, words occurring in the same WN synsets as the target word (WN synonyms) have been chosen, corresponding to all senses of the target. Additionally, the words occurring in synsets related (through explicit relations provided in WN) to those containing the target word have also been considered as part of the vocabulary used for disambiguation. Synsets and relations were restricted to those associated with the part of speech of the target word [Hristea 2008]. The content words of the glosses of all types of synsets participating in the disambiguation process, using the corresponding example strings as well, have equally been taken into consideration. The latter choice has been made since previous studies [Banerjee 2003], performed for knowledge-based disambiguation, have come to the conclusion that the "example relation" (which simply returns the example string associated with the input synset) seems to provide useful information in the case of all parts of speech. The authors tested their method over SENSEVAL-2 words and reported absolute improvements up to 16.3%.

In the following section we present the other unsupervised clustering technique chosen for our proposed method, which is spectral clustering.

2.3.2 Spectral clustering

Besides the Naïve Bayes method, we employ spectral clustering as classification technique in order to show the decisive importance of the clustering method choice reflected in top precision results. Indeed, in Section 2.6 we show that the spectral clustering-based method performs best. The method of spectral clustering is briefly presented here. For more details and justification of the method the reader is referred to [von Luxburg 2007] and [Hastie 2008].

2.3.2.1 Spectral clustering method

Given a set of observations $x_1, ..., x_n$ and some notion of similarity $s_{ij} \geq 0$ between all pairs of observations x_i and x_j, the intuitive goal of clustering is to divide the observations into several groups such that observations in the same group are similar and observations in different groups are dissimilar to each other. One possible way to represent the pairwise similarities between observations is via an undirected *similarity graph* $G = (V, E)$. The vertices of the graph represent the observations (the vertex v_i represents the observation x_i). Two vertices are connected if the similarity s_{ij} between the corresponding observations x_i and x_j is positive (or exceeds some threshold). The edges are weighted by the s_{ij} values. The problem of clustering can then be reformulated as a graph-partition problem, where we identify connected components with clusters. Our intention is to find a partition of the graph such that the edges between different groups have very low weights (which means that observations in different clusters are dissimilar to each other) and the edges within a group have high weights (which means that observations within the same cluster are similar to each other).

An important element in spectral clustering is to construct similarity graphs that reflect the local neighborhood relationships between observations. Starting from a similarity matrix, there are many ways to define a similarity graph that reflects local behavior: ε-neighborhood graph, k-nearest neighbor graphs, fully connected graph [von Luxburg 2007] and [Maier 2009]. One of the most popular graphs and the one that we will use for unsupervised WSD, is the mutual k-nearest-neighbor graph. The vertex v_i is connected to the vertex v_j if, according to the similarity matrix s_{ij}, the observation x_i is among the k-nearest neighbors of the observation x_j or the observation x_j is among the k-nearest neighbors of the observation x_i. The weight of the edge $v_i v_j$ will be $w_{ij} = s_{ij}$ in this case.

In order to formally present the method of spectral clustering we introduce the following notations.

Let $G = (V, E)$ be an undirected graph with vertex set $V = v_1, ..., v_n$. In the following we assume that the graph G is weighted, that each edge between two vertices v_i and v_j carries a non-negative weight $w_{ij} \geq 0$. The weighted adjacency matrix of the graph is the matrix $W = (w_{ij})_{i,j=1,...,n}$. $w_{ij} = 0$ means that the vertices v_i and v_j are not connected by an edge. As G is undirected, we require that $w_{ij} = w_{ji}$. The degree of a vertex $v_i \in V$ is defined as $d_i = \sum_{j=1}^{n} w_{ij}$. The degree matrix D will be the diagonal matrix with the degrees $d_1, ..., d_n$ on the diagonal.

Given a subset of vertices $A \subseteq V$, we denote its complement $V \setminus A$ by $\overline{A}$ and its cardinal by $|A|$. For two not necessarily disjoint sets $A, B \subseteq V$ we define

$$W(A,B) = \sum_{v_i \in A, v_j \in B} w_{ij}. \tag{2.13}$$

We can now formulate the graph-partition problem in relation to spectral clustering. For a given number k of subsets (clusters) there is a partition of V $A_1, ..., A_k$ which minimizes[8]:

$$\text{RatioCut}(A_1, ..., A_k) = \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{|A_i|} \tag{2.14}$$

Unfortunately, the above optimization problem is NP hard [von Luxburg 2007]. Spectral clustering solves a relaxed version of this problem. Relaxing "RatioCut" leads to unnormalized spectral clustering.

The unnormalized graph Laplacian matrix of a similarity graph G is defined as:

$$L = D - W \tag{2.15}$$

Spectral clustering finds the m eigenvectors $U_{n \times m}$ that correspond to the m smallest eigenvalues of L (ignoring the trivial constant eigenvector corresponding to the eigenvalue 0). Using a standard method like K-means, the rows of U are clustered, giving a clustering of the original observations.

The unnormalized spectral clustering algorithm is summarized in Algorithm 2.2.

2.3.2.2 Using spectral clustering for unsupervised WS discrimination

There are a number of issues that must be dealt with when applying spectral clustering in practice. One must choose how to compute the similarity between observations and how to transform these similarities into a similarity graph. In the case of the mutual k-nearest-neighbor graph, the parameter k, representing the number of nearest neighbors, must be set. In the light of all these issues we follow the approach adopted by [Popescu 2011] where spectral clustering was used in WSD for the first time.

In unsupervised WS disambiguation, the observations are represented by contexts of the ambiguous word. The contextual features are given by the actual

[8]The "RatioCut" is not the only objective function optimized in spectral clustering. See [von Luxburg 2007] for other variants such as "Ncut".

Algorithm 2.2 Unnormalized spectral clustering algorithm

```
Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number $k$ of clusters to construct.
1: Construct a similarity graph in one of the standard ways, for example by using
   the mutual $k$-nearest-neighbor graph. Let $W$ be its weighted adjacency matrix.
2: Compute the unnormalized Laplacian $L$.
3: Compute the first $k-1$ eigenvectors $u_1, ..., u_{k-1}$ of $L$ corresponding to the $k-1$
   smallest eigenvalues of $L$ (ignoring the trivial constant eigenvector correspond-
   ing to the eigenvalue 0).
4: Let $U \in \mathbb{R}^{n \times (k-1)}$ be the matrix containing the vectors $u_1, ..., u_{k-1}$ as columns.
5: For $i = 1, ..., n$, let $y_i \in \mathbb{R}^{k-1}$ be the vector corresponding to the $i$-th row of $U$.
6: Cluster the points $(y_i)_{i=1,...,n}$ in $\mathbb{R}^{k-1}$ with the k-means algorithm into clusters
   $C_1, ..., C_k$.
Output : Clusters $A_1, ..., A_k$ with $A_i = \{j \mid y_j \in C_i\}$.
```

"neighboring" content words of the target (ambiguous) word. They occur in a fixed position near the target, in a window of fixed length, centered or not centered on the target. A window of size n denotes the consideration of n content words to the left and n content words to the right of the target, whenever possible. The total number of words considered for disambiguation is therefore $2n + 1$. When not enough features are available, the entire sentence in which the target word occurs represents the context window. The classical window size, generally used in WS disambiguation, and also in our case, is 25 ($n = 25$). Within this representation, the value of a feature is given by the number of occurrences of the corresponding word in the given context window. Thus, a context is represented as a feature vector and the similarity between two contexts is given by the value of the dot product of the corresponding feature vectors. The dot product was chosen as the measure of similarity between feature vectors because of the success of the linear kernel in supervised WS disambiguation [Màrquez 2006].

As a method for building the similarity graph from the similarity matrix we use the mutual k-nearest-neighbor graph method. This involves the choice of the parameter k, the number of neighbors. As in [Popescu 2011], we use a value of 30 for the number of neighbors[9].

2.4 WS discrimination in IR

The main contribution of this chapter is the proposal of a new unsupervised clustering-based method for performing word sense discrimination for IR. This method aims to increase the top level precision for queries which contain ambiguous words. Our suggested approach reorders an initially retrieved document list by pushing to the

[9]See [Popescu 2011] for a justification concerning the choice of this number of neighbors.

fore documents that are semantically similar to the target query.

To start with, for each query we retrieve a set of documents by means of a state-of-the-art search engine; documents are ordered according to their scores. Our objective is then to pinpoint the documents which are more relevant to the information need because they share the term sense with the query; and to enable these documents to improve their position at the top of the document list.

The first phase in the method is based on clustering the retrieved document set with respect to the senses of each ambiguous word and to decide which documents share the query term sense. In this phase a classification technique is employed. First, we try the Naïve Bayes classification and then the spectral clustering. This means that we have the same WS discrimination algorithm with two variants, one for each corresponding clustering technique. In the second phase, we reorder the initially retrieved document list by boosting documents belonging to the selected cluster.

2.4.1 Query WS discrimination

Before we can apply the WS discrimination technique described in Section 2.3.2.2 to the queries, we need to process the data within a preprocessing step. The preprocessing step identifies the polysemous words of a query (as a result of their occurrence in multiple WN synsets).

Term discrimination uses a subset of documents. More precisely, for each query, we consider the first n documents retrieved by the IR system. For each document we thus know the score which indicates how similar that document is to a specific query. The query is added to this set of retrieved documents as if it was another document. The feature set for each document is then calculated by creating an incidence matrix with rows representing the documents and columns representing the features. Each element of this matrix is either 1 or 0, depending whether or not the feature indicated by the column index is present in the document indicated by the row index. The number of WN senses and the incidence matrix obtained after data preprocessing is used as input for the WS discrimination algorithm. Thus, the WS discrimination process is performed on $n + 1$ documents (n initially retrieved documents and the query itself).

Let us now describe in more details the entire WS discrimination process in relation to a single polysemous target word.

The first step is to build the corresponding feature set. The first processing step is to eliminate the stopwords. The remaining words are stemmed using the Porter stemmer algorithm [Porter 1980]. The stem corresponding to the target word is not retained, while the remaining stems, alphabetically ordered, represent the final set of features that are used in the WS discrimination process.

The second step is to build the incidence matrix that indicates what features occur in each document. We determine the position of the target word within each of the documents. In our experiments we used a context window of size 25, as suggested in [Hristea 2008] in order to obtain the best possible disambiguation accuracy. The features that occur in the context window are stored in the row of the matrix that corresponds to the analyzed document. If a certain document contains the target several times, we only consider its first occurrence. This is done in accordance with the "one-sense-per-discourse" heuristic [Gale 1992] which is largely used in WS disambiguation and which states the tendency of a word to preserve its meaning across all its occurrences in a given discourse.

For each query and for each ambiguous term occurring in that query, we cluster the retrieved documents into a number of clusters equal to the number of senses the ambiguous term has, according to a lexical database (WN), which will be used as sense inventory. The final task of the WS discrimination process for a given polysemous word is to determine the document clusters relative to that word. At this moment the resulting clusters depend on the applied clustering technique, Naïve Bayes or spectral clustering, respectively. Each obtained cluster corresponds to a specific sense of the polysemous target word, with one of the clusters containing the query itself. Note that two documents are similar (and thus belong to the same cluster), from the WS discrimination point of view, if the polysemous word has the same sense in both documents. Therefore, disambiguating a polysemous term results in retaining only those documents occurring in the same cluster as the query.

A query can contain several ambiguous terms. In this case, as many clusters of documents as the number of ambiguous words in the query are retained. In order to form a unique list of documents, we fuse these sets of documents; we consider the initial values obtained by the search engine to be the document scores. Various fusion functions that can be used for this purpose have been defined in the literature [Shaw 1995]. In order to obtain a unique cluster per query, we apply the fusion function CombMNZ [Shaw 1995].

The CombMNZ function computes the final document scores as follows:

$$S_f^i = c_i \sum_{j=0}^{c_i} (S_j^i) \ , \tag{2.16}$$

where S_f^i represents the final score for each document d_i, S_j^i represents the score of the document d_i from the cluster j (if the document d_i does not exist in one particular cluster j, then $S_j^i = 0$), and c_i represents the number of nonzero scores for each document i ($c_i = k$ if the document d_i occurs in k clusters).

We have also tested, for scores fusion, the CombSum function [Shaw 1995], which has the formula:

$$S_f^i = \sum_{j=0}^{c_i} (S_j^i) \ . \tag{2.17}$$

One can notice that this function, unlike CombMNZ, does not boost the final score when multiplying the scores sum by the number of occurrences in all the clusters. Thus, the results obtained using CombSum are weaker than in the case of CombMNZ, as we will present later on in Section 2.6.1.1.

We should point out that our method performs WS discrimination and therefore does not give the actual word sense (since we do not know which cluster refers to a specific sense). However, it is not necessary to pair clusters with senses, as document clusters are sufficient for explicit automatic disambiguation in IR.

2.4.2 Document re-ranking

Our approach aims to improve the top retrieved document list. The first phase in the discussed method leads to a set of documents extracted by the search engine, corresponding to each query, and in the clusters of documents obtained as described in Section 2.4.1. Our main purpose for using a WS discrimination technique in IR is to find the most probable relevant documents and to assign them a higher rank in the initial document list. The second phase of the method thus corresponds to a re-ranking method [Meister 2011]. We mention that we have also tested our method as a filtering method, keeping only the clusters of documents obtained through the query WS discrimination. We have obtained results that do not overcome the baselines (see Section 2.6.1.1) and we believe this fact is due to the possible loss of relevant documents through filtering.

Thus, in our approach, the way to improve the top retrieved document list is to modify the order of the retrieved documents by pushing those documents to the fore that are semantically similar to the query, as defined by the WS discrimination results. To do this, we merge the initial set of documents with those obtained as a result of clustering. Unlike the fusion step in the case of multiple ambiguous terms per query (see Section 2.4.1), which aims at obtaining one document set per query by fusing the document lists for each ambiguous query term, we merge here the initial set of retrieved documents with the clustering document list, per on a per query basis. According to the method discussed here, the documents obtained by the search engine and the set of documents obtained after clustering have different levels of importance in the final results. We therefore use a parameter to assign a weight to the fusion function.

This function has the following structure:

$$\begin{array}{l} S_f^i = S_1^i + \alpha S_2^i \text{ with:} \\ S_1^i = score(d_i) \\ S_2^i = \begin{cases} score(d_i), & \text{if } d_i \text{ exists in } Clust \\ 0, & \text{otherwise} \end{cases} \end{array} \tag{2.18}$$

where S_f^i represents the final score of a document d_i, $score(d_i)$ represents the score of that document d_i when considered in the initially retrieved document set, *Clust* is the document cluster containing the query itself and $\alpha \in [0, 1]$ represents the weight of the clustering method for the final results.

As reported in the evaluation section (Section 2.5), we started with $\alpha = 0$ and then increased this parameter by 0.01 at each trial.

Finally, in Section 2.7, the method is used on subgroups of queries with the purpose of analyzing its behavior with regard to different types of queries. The criterion for creating these subgroups of queries is their performance after being sent to the search engine.

2.5 Evaluation framework

2.5.1 Data collection features

To evaluate the two versions of the described method we have used data collections from the TREC competition. We opted for three collections for use in the ad hoc task: TREC7, TREC8 and WT10G. For TREC7 and TREC8, the competition provided approximately 2 gigabytes worth of documents and a set of 50 natural language topic statements (per collection).

The TREC test collections provide topics. What is now considered the "standard" format of a TREC topic statement comprises a topic *ID*, a *title*, a *description* and a *narrative*. The title contains two or three words that represent the key words a user could have used to send a query to a search engine. Both the descriptive and the narrative parts can offer clues about the word senses used in the title part.

For more details about TREC data sets, see Section 1.4.3.

2.5.2 Ambiguous queries and ambiguous terms

In our approach, we search for ambiguous terms in the title part of the TREC topic. The ambiguous terms were detected using the WN knowledge database. If the term occurred in multiple WN synsets, then it was considered as an ambiguous term. A query is defined as ambiguous if it contains at least one ambiguous term. The queries from the three collections contained from zero to four ambiguous words, as presented in Table 2.1, with most of them being nouns.

2.5.3 Evaluation measures

As detailed in Section 1.4.1, TREC ad hoc tasks are evaluated using the `trec_eval` package. This package provides various performance measures, including some single valued summary measures that are derived from the two basic measures in IR: recall and precision. The precision is the fraction of the retrieved documents

TABLE 2.1: The number of ambiguous queries for the data collections

Collection	**Number of queries with X ambiguous terms**					**Total number of ambiguous queries**
	X=0	**X=1**	**X=2**	**X=3**	**X=4**	
TREC7	15	22	10	3	0	35
TREC8	13	22	6	9	0	37
WT10G	18	15	11	5	1	32

that are relevant, while the recall represents the fraction of the documents relevant to the query that are successfully retrieved. The average precision is defined as:

$$AP\left(q\right) = \frac{\sum_{r=1}^{R}\left[p\left(r\right) rel\left(r\right)\right]}{relev\left(q\right)} \tag{2.19}$$

where $relev\left(q\right)$ represents the number of documents relevant to the query q, R is the number of retrieved documents, r is the rank, $p\left(r\right)$ is the precision of the top r retrieved documents and $rel\left(r\right)$ equals 1 if the rth document is relevant and 0 otherwise. The MAP (Mean Average Precision) stands for the mean of the average precision scores for each query. The `trec_eval` package also implements the precision at certain cut-off levels. A cut-off level is a rank that defines the retrieved set. For example, a cut-off level of ten defines the retrieved set as the top ten documents in the ranked list (P@10).

This study uses three cut-off levels: P@5, P@10 and P@30, which are high precision measures. It is worth mentioning that, since we re-ordered an initial retrieved document list, we were unable to retrieve documents that would not have initially been retrieved (no recall improvement). We therefore target high precision improvements.

2.5.4 Baselines

The present study is based on runs constructed by Terrier (see Section 1.4.6). The sets of documents retrieved by Terrier are the first 1,000 ranked documents returned by the search engine. We tried several configurations of the Terrier parameters. For each collection, we chose as our baselines the settings with the highest MAP. The MAP values for our baselines are consistent with the literature [He 2005], [Zhong 2012]. Runs (associated with the baselines), which determine the set of documents to be used by our WS discrimination method, were constructed as additional baselines.

The best configuration for TREC7 was the following: a two step indexation (both direct and inverted index), with active indexation by block and the use of the BB2 (parameter $c = 1$) as a weighting model. BB2 is a DFR model (see Section 1.2.3), implemented in Terrier. As a query expansion model our choice was the parameter-free KL model (KLbfree). The configuration required 3 documents to be used for the query expansion. A term has to occur in two documents in order to be considered relevant. Finally, the number of terms to be added to the query for the process of query expansion was set at 10. The queries use all the three topic parts (title, descriptive and narrative). However, for TREC8 and WT10G, the parameter configuration with the best results in terms of MAP stays in place, except for two differences: the weighting model which is changed with the DFRee model (no parameters) and the narrative part of the topic which is not taken into account when the query is constructed.

The values of the MAP corresponding to the best initial results for the data collections we use are presented in Table 2.6, in addition to some collection features, such as the number of topics and the number of documents.

Table 2.2: Topic and document features from the data collections

Collection	No. of topics	Topic number	No. of documents	Baseline MAP
TREC7	50	351-400	528 155	0.2851
TREC8	50	401-451	528 155	0.2577
WT10G	50	451-500	1 692 096	0.1733

2.5.5 WS discrimination settings

For each query, the set of the top 1,000 documents retrieved by the best settings for Terrier was considered. These 1,000 documents are the documents considered as the most similar to the information need, sorted by their obtained score, in descending order. The method we promote aims at filtering and reordering those documents before retrieving them for the user.

The target terms for the described WS discrimination process are taken from the title part of the TREC topic only. However, our approach needs to have a context for the term. Since topic titles are generally too short, in order to form the context window for the ambiguous terms in the title, all three parts of the topic were used (title, narrative and descriptive). The stopwords from the resulting text were removed (Terrier's stopwords list is used).

2.6 Results

2.6.1 Results using Naïve Bayes classification

We have tested our method both as a filtering technique and as a re-ranking one, over the TREC7 and TREC8 collections. We discuss here the choices we have made and the obtained results. We compare our performances with the considered baseline and we show that the method only as filtering technique is not effective, thus re-ranking would be required in order to improve IR performance.

2.6.1.1 The filtering method for all the ambiguous queries

Our WS discrimination analysis has shown that the queries (topic title) taken into account for our study may contain from zero up to three ambiguous words, for TREC7 and TREC8 (see Table 2.1). In the case of the TREC7 data, the analysis results for the considered topics show 15 queries with no polysemous words, 22 queries with only one polysemous word, 10 queries with two polysemous words and 3 queries with three polysemous words. Corresponding to the TREC8 data, the analysis results identify 13 queries with no polysemous words, 22 queries with only one polysemous word, 6 queries with two polysemous words and 9 queries with three polysemous words.

The average number of senses for the polysemous words of the 35 TREC7 ambiguous queries is 3.47 and the average number of features is 128.5. There are three adjectives (human, commercial, organic), two verbs (teaching, dismantling), and forty-six nouns (see Table 2.3). In the case of the TREC8 data, the average number of senses is 4.03 and the average number of features is 138.07. There are four adjectives, five verbs and fifty-one nouns. In the preprocessing stage we did not identify any adverbs, both corresponding to the TREC7 and to the TREC8 data, this fact being consistent with other studies in natural language processing finding that adverbs, in general, are less ambiguous and more rare in natural language than other parts of speech. The English nouns, however, have a high degree of polysemy; even two proper names that are polysemous have been detected in both collections: "El Nino" and "Amazon" in TREC7 and "Cuba" and "Triangle" (from the Golden Triangle) in TREC8, respectively.

Table 2.3: Number of part-of-speech elements from the ambiguous queries, for TREC7 and TREC8

Collection	**Number of**		
	adjectives	**verbs**	**nouns**
TREC7	3	2	46
TREC8	4	5	45

We have considered as scores for the list of documents forming the clusters the initial scores obtained by Terrier.

Only one list of retrieved documents per query must be proposed, however, in the case of multiple ambiguous terms from the same query, we obtain a list per ambiguous term. Thus, a score fusion is required in order to fuse all the lists for each ambiguous query term coming from the same query. Regarding the fusion function in the case of several ambiguous words, we have evaluated CombSum and CombMNZ [Shaw 1995], presented in Section 2.4.1.

We have constructed TREC runs for our filtering method using both fusion functions. We present the obtained results for the TREC7 data in Table 2.10. The same results corresponding to the TREC8 data are presented in Table 2.5.

The CombMNZ function behaves constantly better than the CombSum function,

Table 2.4: Baseline, CombSum and ComMNZ results for all the ambiguous queries (TREC7)

Precision	Baseline	CombSum	CombMNZ
P@5	**0.6343**	0.5257	0.5429
P@10	**0.5600**	0.4629	0.4914
P@30	**0.4095**	0.3571	0.3714

Table 2.5: Baseline, CombSum and ComMNZ results for all the ambiguous queries (TREC8)

Precision	Baseline	CombSum	CombMNZ
P@5	**0.4960**	0.4171	0.4686
P@10	**0.4660**	0.3514	0.3829
P@30	**0.3707**	0.2476	0.2752

but the baseline is not even reached (both in the case of the TREC7 data and in that of the TREC8 data) because of the possible loss of relevant documents after filtering. Therefore, we evaluate our method as a re-ranking method.

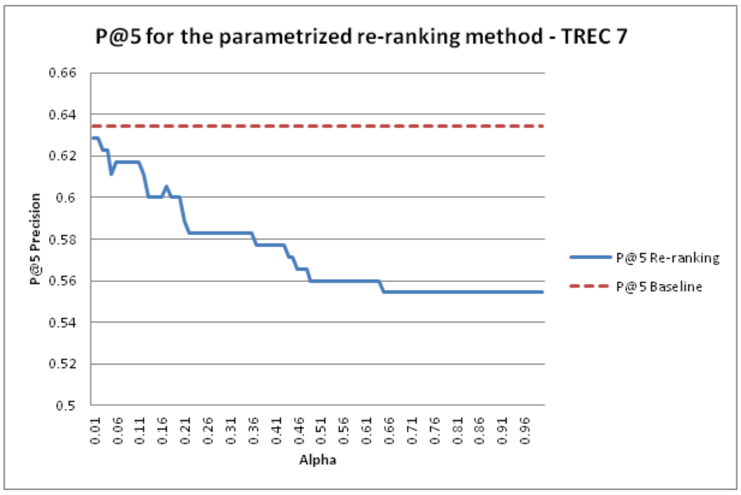
2.6.1.2 The re-ranking method

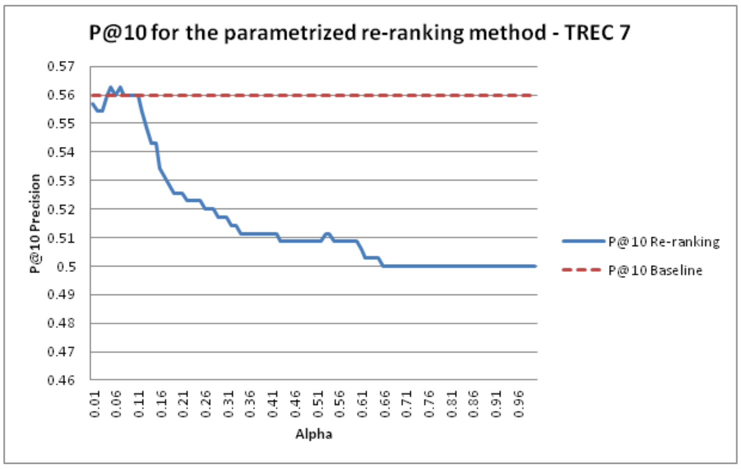
All the ambiguous queries

The re-ranking method is based on an additional fusion step: the initial set of documents, retrieved by Terrier, is fused with the list of documents obtained by clustering (see Section 2.4.2). We test the most favorable contribution of the two sets of documents to the final results by parameterizing the fusion function, as presented in Equation 2.18.

To evaluate our choices we have constructed the corresponding TREC runs, and we have determined the overall precisions for each alpha parameter value in the weighted sum $S_f^i = S_1^i + \alpha S_2^i$ (see Equation 2.18).

Figures 2.1 and 2.2 show that the higher the alpha parameter value is, the lower is the performance. There are some improvements for alpha between 0.1 and 0.2, but the difference between the results and the baseline is usually less than 0.01. This conclusion holds for both the TREC7 and the TREC8 test data. In Figure 2.1 we can notice that the obtained results do not even surpass the baseline in the case of P@5. The best improvements are obtained for the P@30, for a low value of alpha, between 0.05 and 0.13. On the other hand, for P@30 of TREC8 (Figure 2.2), the results are only slightly above the baseline, for a low alpha, in order to descend below the baseline with the growth of alpha. For the P@5, however, on the same collection, there are some improvements for a low alpha (0.01, 0.06), in contrast with P@5 results for TREC7.





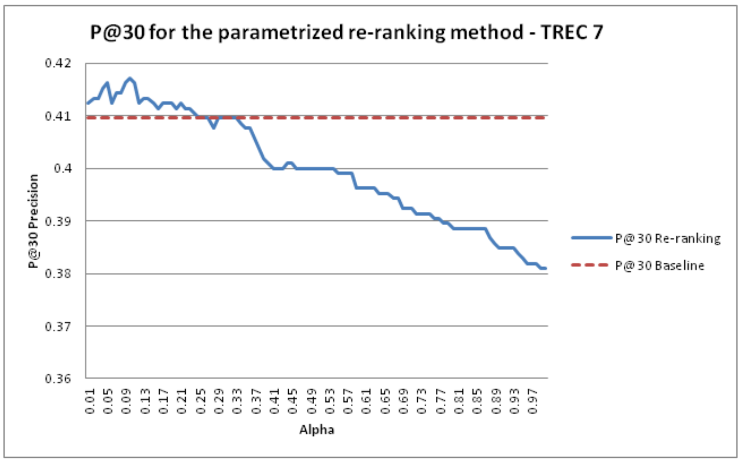
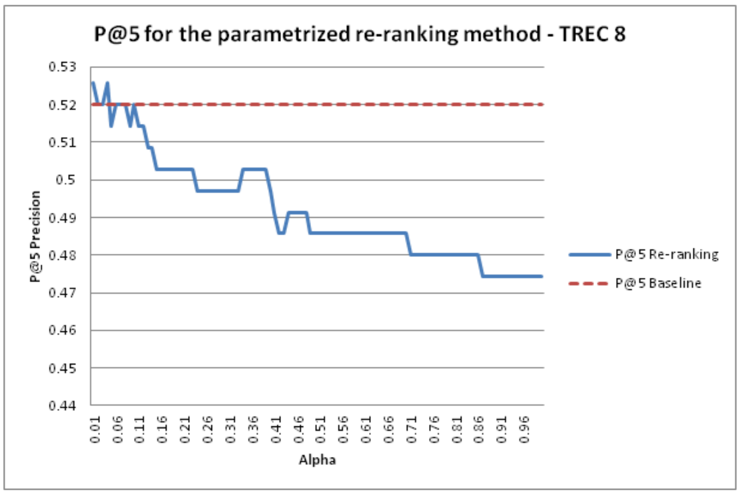
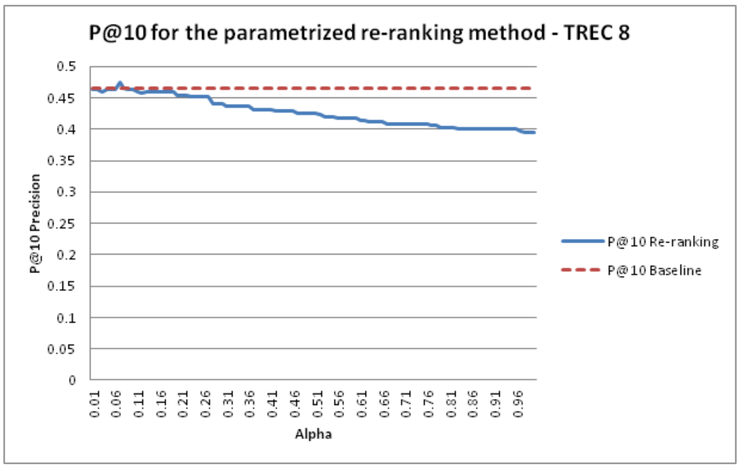


FIGURE 2.1: Precision after the first 5, 10 and 30 retrieved documents, compared to the baseline, for the parametrized method, over TREC7





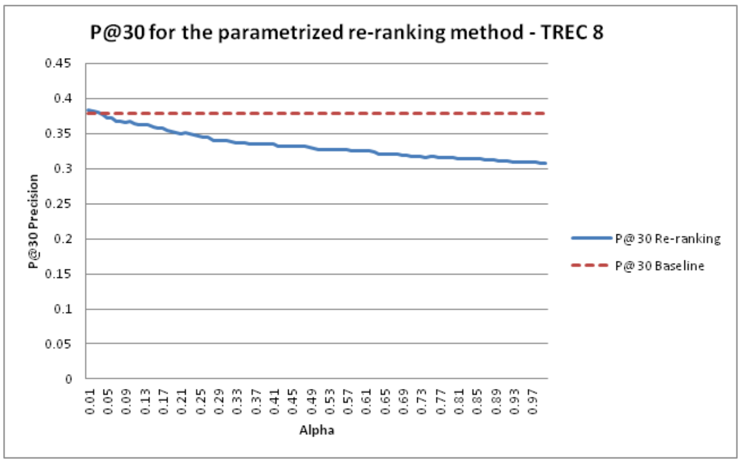


FIGURE 2.2: Precision after the first 5, 10 and 30 retrieved documents, compared to the baseline, for the parametrized method, over TREC8

Table 2.6: Baseline and best results for the lowest precision queries, over TREC7 (* marks statistically significant results, p-value < 0.05)

Precision	Baseline	Best results
P@5	0.2000	**0.2222***
P@10	0.1667	**0.1778***
P@30	0.1370	**0.1481***

Table 2.7: Baseline and best results for the lowest precision queries, over TREC8 (* marks statistically significant results, p-value < 0.05)

Precision	Baseline	Best results
P@5	0.0364	**0.1455***
P@10	0.1091	**0.1545***
P@30	0.1545	**0.1636***

Lowest precision queries

Rather than evaluating the result on all the ambiguous queries, we consider poorly performing topics only. Previous work have shown that clustering topics according to their difficulty in terms of precision provides interesting insight [Bigot 2011].

With respect to quartiles, we have considered the TREC7 35 ambiguous queries and the TREC8 35 ambiguous queries, respectively and have grouped them by their overall precision for the top 5 documents obtained with Terrier. The first quartile, the one with the lowest precision, contains 9 queries (TREC7) and 11 queries (TREC8), respectively.

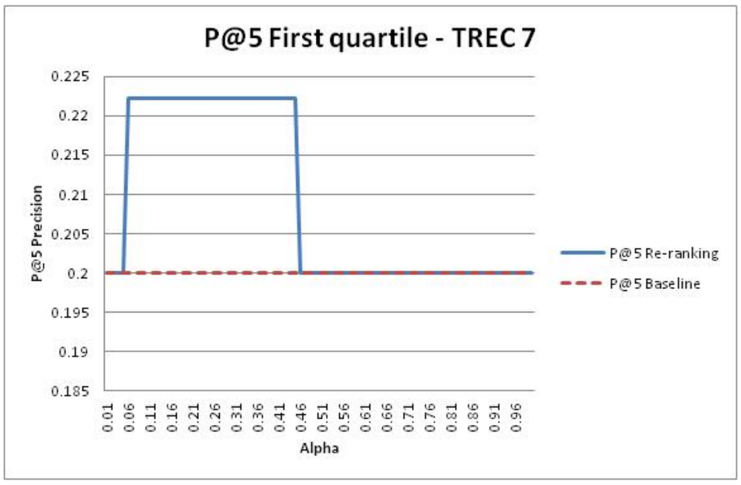
We present the corresponding results of the parametrized fusion function in Figures 2.3 and 2.4. The P@5 and P@10 results for the first quartile, on TREC8 (Figure 2.4), are constantly above the baseline. Baseline is also constantly surpassed on TREC7 (Figure 2.3), but for P@10 results.

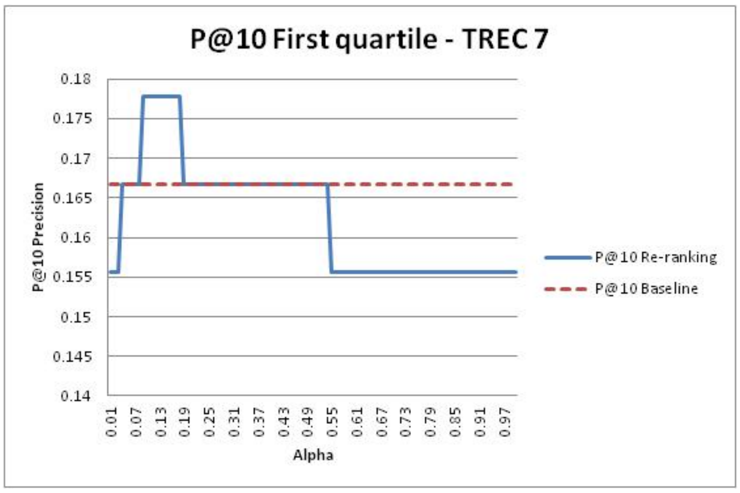
The best results were obtained with the alpha parameter having values between 0.1 and 0.2 in the case of both datasets (Figures 2.3 and 2.4). Since alpha is the parameter assigning a weight to the clustering method for the final results, one can notice that it is best for the cluster documents to participate with 1/5 of their scores in the final document score. Regarding the TREC8 results (Figure 2.4) we can notice, at P@5, a significant improvement of precision, compared to the baseline, for all the values of alpha greater than 0.03. In this case, the highest difference between our results and the baseline is 0.1091.

The baseline precisions and the best results obtained for the lowest precision queries are shown in Table 2.6 (TREC7) and in Table 2.7 (TREC8), respectively.

2.6.2 Results using spectral clustering

In this section we present the results of our method while employing the spectral clustering technique. The test collections are TREC7, TREC8 and WT10G. These





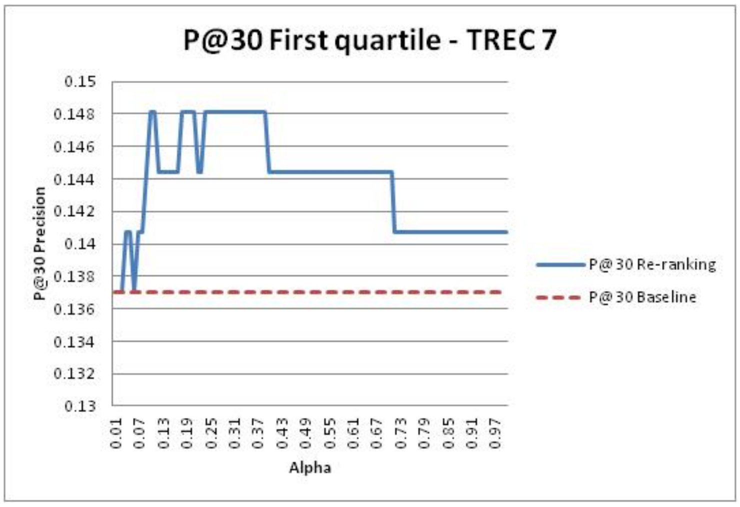
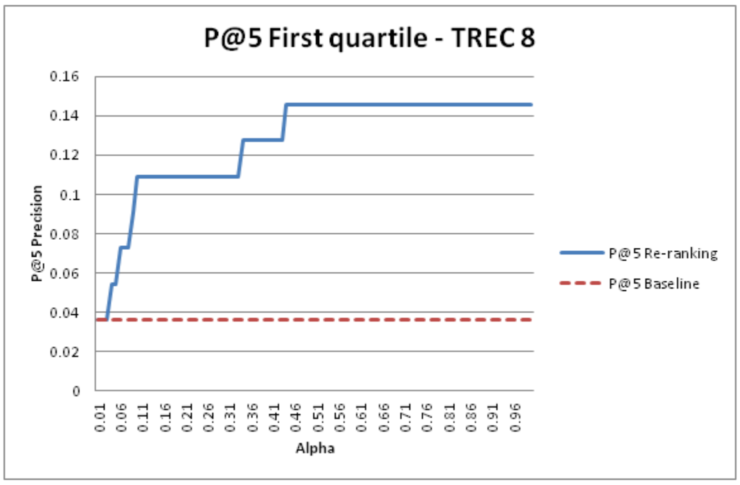
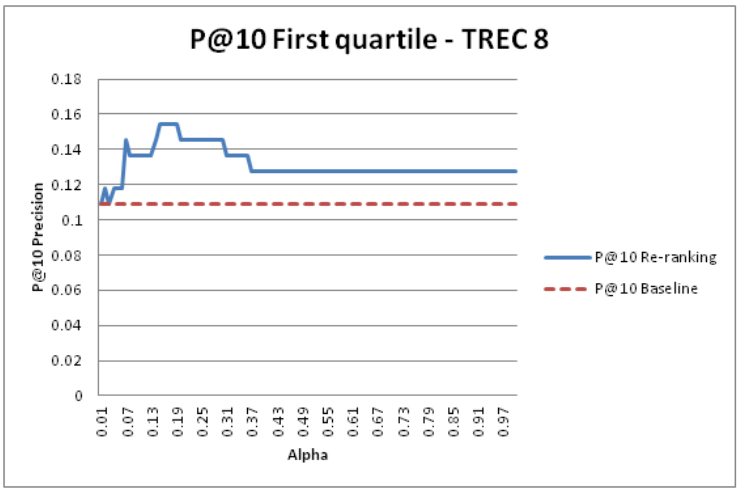


FIGURE 2.3: Precision after the first 5, 10 and 30 retrieved documents, obtained for the lowest precision queries, compared to the baseline, for the parametrized method, over TREC8





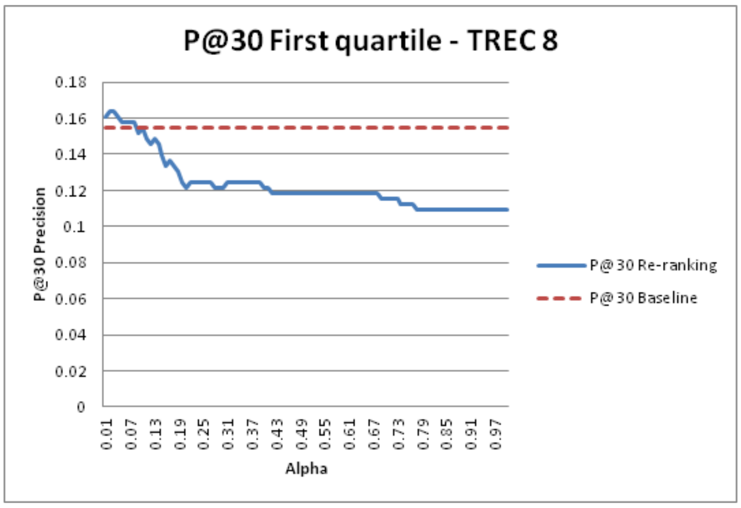


FIGURE 2.4: Precision after the first 5, 10 and 30 retrieved documents, obtained for the lowest precision queries, compared to the baseline, for the parametrized method, over TREC8

results as compared to the baselines (see Section 2.5.4) are presented in Figure 2.5. The graphs illustrate the manner in which the top levels of precision evolve with respect to the alpha parameter. Alpha is the same parameter that gives a greater or smaller level of importance, in the final score, to the scores of the documents in the cluster, as presented in Section 2.4.2 (Equation 2.18). On the first row, the results for P@5 are shown, in comparison with the three collections. The next two rows present the results for P@10 and P@30 respectively. For each cut-off level, the vertical axis is recalibrated in order to obtain a clearer view.

Figure 2.5 shows that the best results were obtained when the value of the alpha parameter was between 0.02 and 0.20. This observation holds for all the top levels of precision (P@5, P@10 and P@30) and for all the three collections involved in the study. p-values smaller than 10^{-4} of t-tests have confirmed the statistical significance of our results. t-tests have used the two following populations: the baseline value and our results per alpha, respectively. It is also noticeable that for an alpha parameter which is greater than 0.2, the results usually fall below the baselines. Since alpha is the parameter that assigns the importance of the clustering method for the final scores, one can notice that it is best for the cluster documents to participate with not more than 1/5 in the final document score. The alpha parameter is similar to the lambda interpolation parameter in the case of RM3 (see Section 1.3, Equation 1.17), since the optimal lambda values for the short queries vary between 0.01 and 0.4 [Zhai 2004].

The baselines were surpassed by the obtained results for each collection and for each top level of precision. While for P@5 and P@10 the difference between the results and the baselines is clear, for P@30 the curve representing the results remains closer to the baseline. The best improvement occurred for the WT10G collection in the case of P@10. A precision value of 0.2937 was obtained (the baseline was 0.2688), which represents an improvement of 8.48%, and is statistically significant with a p-value $<10^{-6}$ (t-test).

Baselines, so far, consisted of initial runs that we try to improve by re-ranking based on WS discrimination (see Section 2.5.4). More than that, analysis of the results has been carried out against the major approach existing in the literature. We compare our results with those obtained when implementing the disambiguation method proposed in [Schütze 1995], as well as with the results we obtained while using our Naïve Bayes classification-based method for the unsupervised clustering step.

In the case of the terms co-occurrence-based method [Schütze 1995], which is described in Section 2.2, we have reimplemented this method and have organized the same testing setup as the one originally used by Schütze and Pedersen [Schütze 1995]. The stop words have been removed (Terrier stop-words list) and the target term-centered context window was set to size 40 (20 terms before the target term, 20 terms after the target term). We have identified $1,466,983$ unique terms that induced the $1,466,983 \times 1,466,983$ sparse term co-occurrence matrix.

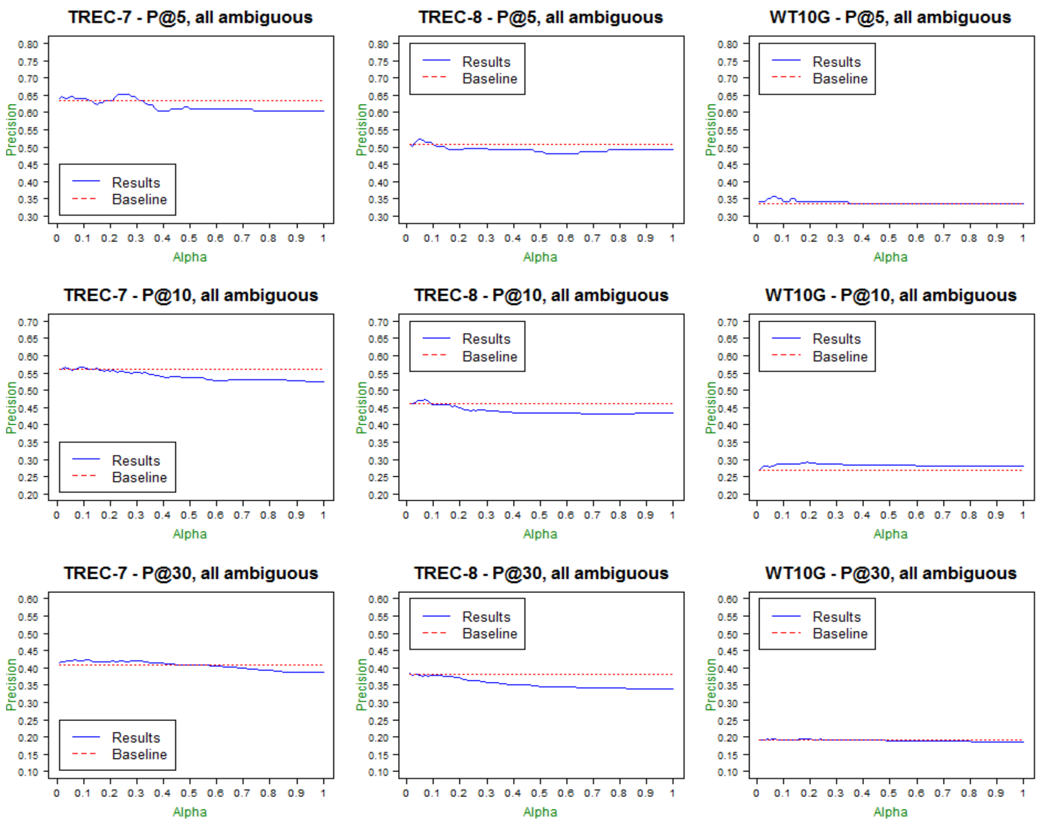


FIGURE 2.5: The results for the three test collections by the top levels of precision

We mention that, due to the high number of vocabulary terms, the co-occurrence matrix is difficult to handle from a computational point of view. The SVD for the co-occurrence matrix was set to 100 dimensions and the reduced matrix was computed using the `irlba` package of `R`[10] (with 100 iterations). To classify the context vectors we used the Buckshot algorithm implemented using R (packages `hc` and `kmeans` of `R`), with 10% sampling for the initial hierarchical clustering step. The query terms that occurred less than 100 times in the corpus were not considered ambiguous since, according to its authors [Schütze 1995], the method uses $f/50$ as the number of senses, with f denoting the occurrence number of the target term in the corpus. Creating context vectors for each target word occurrence, as well as reindexing after replacing words with their senses (for each set of queries), also represent time and resource consuming operations.

In Table 2.8 we present the results of our comparison in terms of high precision. *Best Run* represents the best run obtained with Terrier, treated as baseline (see Section 2.5.4) and also treated as the word-based retrieval for the terms co-occurrence-based method. *Naïve Bayes* represents our method with the Naïve Bayes

[10]http://www.r-project.org/

classifier, described in Section 2.3.1. *Spectral Clustering* is our method, described in Section 2.3.2. *Sense-based* represents the terms co-occurrence-based method of [Schütze 1995] and *CombRank* represents the modified co-occurrence method, also presented in [Schütze 1995], which considers the sum of ranks from the word-based retrieval and from the sense-based retrieval as the final rank for a retrieved document. It was reported [Schütze 1995] as better than using the sense-based method alone. *CMNZ-WB-SB* represents the combined document list resulting from the word-based and sense-based retrievals, using the CombMNZ function [Shaw 1995]. *CMNZ-WB-SB-alpha* represents the *CMNZ-WB-SB* results with the sense-based retrieved list weighted by an *alpha* parameter. We tested various *alpha* values. The best turned out to be 0.1.

Table 2.8: Comparison with the co-occurrence-based methods and with the Naïve Bayes-based method, by the top precisions, for the TREC7 collection (** represents p-value $<10^{-6}$), compared to *Best Run*

Prec.	Best	Naïve Bayes		Spectral Clustering		Schütze & Pedersen			
	Run	Peak Res.	Average Res.	Peak Res.	Average Res.	Sense-based	CombRank	CMNZ-WB-SB	CMNZ-WB-SB-0.1
P@5	0.6343	0.6286	0.5736	**0.6514****	0.6193	0.3829	0.4400	0.4286	0.4914
P@10	0.5600	0.5629	0.5152	**0.5657****	0.5409	0.3400	0.3686	0.3743	0.4257
P@30	0.4095	0.4171**	0.3999	**0.4248****	0.4068	0.2438	0.2629	0.2752	0.3210

The various combinations help to improve the initial performance of Sense-based results, although the baseline results (*Best Run*) are not surpassed. The number of the improved queries with respect to *Best Run* was also computed for the *CMNZ-*0.1 (which is the best alpha for co-occurrence-based results) and for the *Spectral Clustering* results, with the same *alpha* value of 0.1. The results are presented in Table 2.9. Only very few queries are improved by the terms co-occurrence-based method.

As opposed to the *Sense-based* model, the peak results of *Spectral Clustering* outperform the *Best Run* baseline, for all the levels of high precision (from 1.01% to 3.73%). The average results do not overcome the baseline due to the performance decrease after a certain value of alpha (see Figure2.5).

The *Naïve Bayes* peak results outperform the *Best Run* only for P@10 and P@30 (0.51% and 1.85%, respectively). In addition, our method outperforms the Naïve Bayes method both on average and on peak results. The average is computed across all alpha parameter values, considering all the ambiguous queries. This again suggests the importance of the clustering technique used in unsupervised WSD for IR. We hereby conclude that spectral clustering is an appropriate clustering method for the purpose of sense discrimination in IR.

The present method was also tested for 5,000 document runs, but the results were not improved. We think the reason is that, once more documents per run are taken into account, a significant amount of noise is also introduced and the re-ranking method cannot reach efficiency at the top level of precision (P@5, P@10 and P@30). We also considered a two cluster model in which documents could either be clustered in the query cluster if similar enough to the query, or in the

Table 2.9: The number of improved queries, by the top precisions, for *CMNZ-WB-SB*-0.1 (Schütze) and *Spectral Clustering*-0.1

Prec.	**Number of improved queries**	
	CMNZ-WB-SB-0.1 (Schütze)	**Spectral Clustering-**0.1
P@5	3	18
P@10	3	19
P@30	5	17

non-query cluster. Results were better when as many clusters as WN senses were considered.

2.7 Further analysis of the spectral clustering results

This section aims to deepen the analysis of the spectral clustering results obtained when considering two types of query clusters: those based on the query performance and those taking into account the number of ambiguous terms per query.

2.7.1 Improvements in baseline precision intervals

Following previous research showing that results can differ according to query difficulty [Bigot 2011] and with the purpose of observing where the proposed method behaves most accurately, (independently for each collection), all the results from all the three test collections were gathered into a single data set. All the runs corresponding to the 104 ambiguous queries were divided into 5 groups, according to the baseline precision (0.0 − 0.2, 0.2 − 0.4, . . . , 0.8 − 1.0) and for each of the top levels of precision being investigated (P@5, P@10 and P@30, respectively).

The P@5 results for all the intervals are depicted in Figure 2.6. For the queries with low or very low performance (intervals 0.0 – 0.2 and 0.2 – 0.4) we obtained significant improvements. This fact suggests that the reordering of the poorly performing list of documents retrieved by the search engine puts more relevant documents at the top of the list. On the other hand, for the queries with a good or very good performance (0.6 – 0.8 and 0.8 – 1.0), our re-ranking method could not bring more relevant documents to the fore because the search engine results were either already as good as possible (0.8 for the interval 0.6 – 0.8, or 1.0 for the interval 0.8 – 1.0), or very close to this. The results can hence overcome the baseline only by chance. All the comparisons are statistically significant, with the p-values $<10^{-6}$ (t-test for columns *Baseline* vs. *Peak Result* and *Baseline* vs. *Average Result*, respectively). The conclusions for P@5 are also consistent with P@10. The good results for P@10 can be explained by the fact that there is a higher chance of obtaining new relevant documents in a list of 10 documents than in a list of 5. However, for P@30, the improvements were not as significant as for the other top levels of precision. In order to improve the performance in a list of 30 retrieved documents it would be necessary to bring more than 1 or 2 new relevant documents from the re-ranking

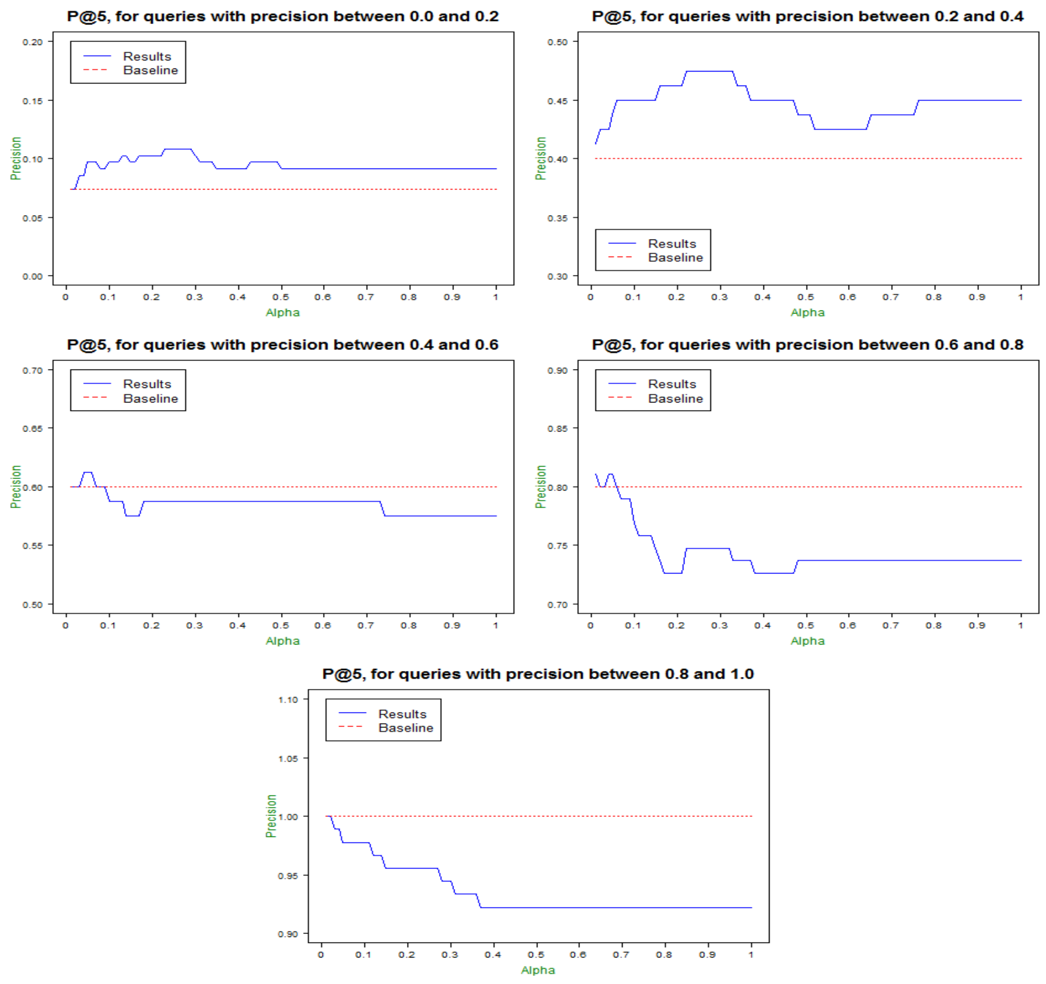


FIGURE 2.6: P@5 by the intervals of performance corresponding to the TREC7, TREC8 and WT10G collections

process. (For P@5, 1 new relevant document represents a 20% improvement, while for P@30, 1 new relevant document represents only a 3.33% improvement).

In Table 2.10 we present the peak (the best results) and average improvements for each baseline precision interval in the case of P@5 and P@10 respectively.

2.7.2 Detailed results after taking into account the number of ambiguous terms

The queries from the data set utilized in this study contain from 0 to 4 ambiguous terms (see Table 2.1). A high number of ambiguous terms also suggests an increased level of query difficulty caused by multiple possible combinations of senses between terms. Keeping this aspect in mind, we investigated the behavior of our spectral clustering method over clusters of queries classified by the number of ambiguous

TABLE 2.10: The peak and average improvements for each baseline precision interval (** represents p-value $<10^{-4}$)

Precision	Interval	Baseline	Peak Result (Spectral)	Peak Improvement	Average Result	Average Improvement
P@5	0.0 – 0.2	0.0742	0.1085	**46.15%****	**0.0942**	**26.95%****
P@5	0.2 – 0.4	0.4000	0.4750	**18.75%****	**0.4477**	**11.92%****
P@5	0.4 – 0.6	0.6000	0.6125	2.08%**	0.5851	-2.48%**
P@5	0.6 – 0.8	0.8000	0.8105	1.31%	0.7433	-7.08%**
P@5	0.8 – 1.0	1.0000	1.0000	0.00%	0.9360	-6.40%**
P@10	0.0 – 0.2	0.0696	0.1121	**60.87%****	**0.1048**	**50.57%****
P@10	0.2 – 0.4	0.3560	0.3920	**10.11%****	**0.3731**	**4.80%****
P@10	0.4 – 0.6	0.5312	0.5437	2.35%**	0.4916	-7.45%**
P@10	0.6 – 0.8	0.7533	0.7600	0.88%	0.6546	-13.10%**
P@10	0.8 – 1.0	0.9533	0.9533	0.00%	0.9227	-3.20%**

terms. We proceeded as in Section 2.7.1 (independently for each collection) by grouping all three data sets into a single one. All of the 104 ambiguous queries were divided into three classes: queries that contain 1 ambiguous term, 2 ambiguous terms and 3 ambiguous terms respectively. Our method was not applied to the queries that contained no ambiguous terms (see Section 2.5.2). The population for the cluster corresponding to queries with 4 ambiguous terms was very weak (only one query) and therefore was also not taken into account.

The peak spectral clustering results (with optimal alpha) and the percentages of improvements for each cluster, by the top levels of precision, are presented in Table 2.11.

TABLE 2.11: The peak improvements, by the number of ambiguous terms, for each top precision (** represents p-value $< 10^{-4}$)

Precision	1 Ambiguous Term		
	Baseline	Peak Res.	Peak Improv.
P@5	0.5767	0.5936	2.93%**
P@10	0.5238	0.5404	3.17%**
P@30	0.3911	0.4001	2.31%**

Precision	2 Ambiguous Terms		
	Baseline	Peak Res.	Peak Improv.
P@5	0.4000	0.4148	3.70%**
P@10	0.3307	0.3445	4.17%**
P@30	0.2323	0.2434	4.78%**

Precision	3 Ambiguous Terms		
	Baseline	Peak Res.	Peak Improv.
P@5	0.3882	0.4118	**6.06%****
P@10	0.3176	0.3353	**5.57%****
P@30	0.2727	0.2941	**7.91%****

The highest values were obtained for the clusters of queries containing 3 ambiguous terms, which suggests that our method best improves the most ambiguous queries. For P@30 the improvement was almost 8%. The results are statistically significant with p-values $<10^{-6}$ (t-test for columns *Baseline* vs. *Peak Res.*). It is also worth mentioning that constant improvements were also obtained for the other two clusters being investigated.

2.8 The spectral clustering method using automatically generated context

The method we propose uses all the three parts of the TREC topics, title, description and narrative (TDN), as disambiguation context. TDN implies the assumption that a context exists for the query, which is not the case in real world applications. For this reason, in this section we automatically build a context in order to validate our approach. This automatic context is not optimal (weaker performance than for TDN) and it is not optimized since our point was only to validate that our method still works with automatic context. We present the automatic contextualization method and we discuss the obtained results.

2.8.1 Automatic contextualization using pseudo relevance feedback

We chose a straightforward pseudo relevance feedback (PRF) approach in order to obtain the context [Attar 1977, Buckley 1994]. The option for this type of contextualization is motivated by the assumption that the first retrieved documents have high chances to be relevant and thus they presumably contain the target words with the correct sense.

First of all, we run retrieval on the initial query (title part of the TREC topic) over the TREC document collections and we retain the first three retrieved documents, as it was done in the baseline (see Section 2.5.4). This parameter value is used in query expansion models [He 2009] based on the assumption that, when taking into account more than five documents, the probability of treating irrelevant documents increases. Having these top documents, we concatenate the texts, we remove the stopwords and we search for the presence of at least two query terms in a moving context window of 50 words. If this presence occurs, we keep the text in the context window and add it to our context. The search for at least two query terms together is motivated by the assumption that two ambiguous words tend to disambiguate each other when found together, for example "java" and "island" [Andrews 2011].

External sources such as Wikipedia were avoided when building the context due to differences in terms of actuality. Moreover, the relevance judgments were constructed considering the information in the description and narrative parts of the topic, suggesting some kind of a closed circuit. For instance, supposing that we have obtained a context with senses for target words different than the senses suggested for pooling, this would lead the evaluation of the disambiguation process to complete failure.

The usage of our PRF-based context and insights regarding the performance are presented in the following subsection.

2.8.2 Experiments and results

In order to prove the effectiveness of our method in the case of automatically generated context we created four TREC runs, as follows:

- **Title**: retrieved documents when the query represents only the title part of the topic;
- **Title+Context**: retrieved documents when the query represents the title part of the topic, together with the automatically built context;
- **Spectral-Title**: the re-ranked documents after applying the spectral clustering method, when the query is represented only by the title part of the topic;
- **Spectral-Title+Context**: the re-ranked documents after using the automatic context as WSD context for the spectral clustering method.

For few queries in each collection, our method was not able to provide any context either due to a title part of the TREC topic formed only by one term, or due to the complete nonexistence of co-occurrences of at least two terms in the context window, in the retained text. Hence, we considered only the queries containing ambiguous terms and for which the automatic method was able to provide a context, as follows: 29 out of 35 ambiguous queries in TREC7, 35 out of 35 ambiguous queries in TREC8 and 28 out of 32 ambiguous queries in WT10G, respectively.

Tables 2.12, 2.13 and 2.14 provide precision values at 5, 10 and 30 retrieved documents after evaluating the above mentioned runs, for each collection. For comparison we recall the results obtained using the reformulated TD(N) runs (from Section 2.5). We mention that the queries without automatic context were also removed from the TD(N) evaluations, in order to maintain the same comparison basis. The best values per collection are written in bold. Statistical significance of results (t-test between the basic *Title* run and *Spectral-Title+Context*) is also marked with asterisks in the tables (p-value $<10^{-6}$).

TABLE 2.12: TREC7, P@k for the considered runs

Precision	**TD(N)**	**Spectral-TD(N)**	**Title**	**Title+ Context**	**Spectral-Title**	**Spectral-Title+ Context**
P@5	0.6429	0.6571	0.5310	0.5448	0.5379	**0.5655(6.5%)****
P@10	0.5679	0.5714	0.4586	0.4103	0.4586	**0.4724(3.0%)****
P@30	0.4226	0.4381	0.3425	0.2770	0.3402	**0.3460(1.0%)****

TABLE 2.13: TREC8, P@k for the considered runs

Precision	TD(N)	Spectral-TD(N)	Title	Title+ Context	Spectral-Title	Spectral-Title+ Context
P@5	0.5081	0.5243	0.5371	0.3829	0.5486	**0.5543(3.2%)****
P@10	0.4622	0.4730	0.4800	0.2543	**0.5114**	0.5029(4.8%)**
P@30	0.3811	0.3829	0.3848	0.1524	0.3886	**0.4019(4.4%)****

TABLE 2.14: WT10G, P@k for the considered runs

Precision	TD(N)	Spectral-TD(N)	Title	Title+ Context	Spectral-Title	Spectral-Title+ Context
P@5	0.3538	0.3692	0.3500	0.2071	0.3357	**0.3571(2.0%)****
P@10	0.2885	0.3115	0.2714	0.1357	0.2679	**0.2786(2.7%)****
P@30	0.1923	0.1949	0.1917	0.0679	0.1857	**0.1929(0.6%)****

In terms of top level precision, the *Spectral-Title+context* run is better than the *Title* run, which is better than the *Title+Context* run. This suggests that the generated context is harmful for the retrieval process itself but beneficial for the spectral clustering method (*Spectral-Title+Context* run is generally better than *Spectral-Title*). We believe that this is due to the amount of "noisy" terms in the context. Unlike the feature selection process using a Naïve Bayes technique [Chifu 2012], spectral clustering automatically selects its useful features, hence "noise" filters out and it remains less of a problem than for the retrieval process.

We notice that in 89% of cases the *Spectral-Title+Context* run has the greatest performance. Even if the relative improvement (0.6% - 6.5%) is not very high, this improvement allows us to state that our method remains effective even with automatic contextualization. In addition, the results we obtained in Section 2.6 show that using a better context would improve the results even more.

The least improved results are to be noticed in the case of WT10G (Table 2.14). Here the initial retrieval (Title only) has the lowest performance among all three considered collections, therefore the context quality decreases, since the P@3 is relatively low (TREC7: 0.5977, TREC8: 0.5619, WT10G: 0.3810). Having a poor context implies a less performing WSD process.

2.9 Conclusion

In this chapter we presented a re-ranking method for IR, based on WS discrimination. In order to discriminate the term senses, the WS discrimination requires unsupervised clustering, for which we have chosen Naïve Bayes classification and spectral clustering. Our method shows a remarkable improvement in high rank precision for ambiguous queries. We believe that this represents a very important aspect, considering the fact that IR systems are predisposed to failure in the case of this particular type of queries [Stokoe 2003], [Mothe 2007].

Several previous studies [Sanderson 1994], [Guyot 2008] have failed to prove the usefulness of WS disambiguation in IR. On the contrary, we show that unsupervised WS disambiguation, namely WS discrimination, can improve IR results. We are of the opinion that WS discrimination is sufficient in IR and that WS disambiguation is not compulsory, as opposed to text translation, for example. We analyze the obtained results with respect to a major approach existing in the literature [Schütze 1995], as detailed in Section 2.5. When using a Naïve Bayes-based clustering technique, we demonstrated that WS discrimination can improve IR performance. However, unlike for the case of spectral clustering, we only recorded very small improvement and only on some sub-cases, hence the importance of the clustering technique used for WS discrimination in IR, another point which we have made here.

Our method exploits TREC topic definitions which have a level of detail in their description, a level which is not normally available in an IRS; the topic definition is used to provide a context to the query. Other works from the literature use these structural elements and most often the descriptive part of the query helps in improving the results [He 2004]. However, using the complete statement of the topic could lead to valid criticism of experiments such as ours because we exploit this detail. We prove that, if this level of detail is available, then it can be used in a beneficial way to improve retrieval effectiveness. However, even if our goal was not to develop mechanisms which can capture in an optimal way the needed level of detail, we do propose a method to capture the context of the query and show that our own method for WS discrimination in IR remains useful. Such a method of contextualization, namely the usage of PRF [Buckley 1994], has been employed by us in Section 2.8 for validating our conclusions in the presence of automatically generated context. Indeed, contextualizing short texts, such as tweet contextualization [SanJuan 2011] and query expansion [Ogilvie 2009] is an active research domain and we think it will be worth considering new contextualization techniques in our WS discrimination method as a future goal.

The research conducted for the results in this chapter opens the lead to include in our study query difficulty prediction, which is already an active research area [Mothe 2005], [Pehcevski 2010], [Carmel 2010], [Sarnikar 2014] yet having a lack in terms of applications, unfortunately. The fact that our method rather improves poor performing queries (Section 2.7.1), especially those with multiple ambiguous terms (Section 2.7.2), should drive in-depth research along this path.

This research has been published in CEJCS[11] [Chifu 2012], for the Naïve Bayes method and in IPM[12] [Chifu 2015], for the spectral clustering method with a deeper analysis.

[11]Central European Journal of Computer Science
[12]Information Processing & Management

Chapter 3

Query difficulty prediction

Contents

We underlined in the previous chapter (Chapter 2) that our disambiguation methods produce higher improvements in terms of high precision, in the case of poorly performing queries. Thus, it would be interesting to analyze the possible mechanisms to predict the query performance.

In this chapter we present the context of query difficulty prediction. We begin with the notion of query difficulty and we continue with the description of query difficulty predictors from the literature.

Next, we develop our contributions on the query difficulty prediction, that have been published in [Chifu 2013], in CORIA[1] conference. The first contribution consist in proposing two linear combinations of difficulty predictors with the purpose

[1]COnférence en Recherche dInformation et Applications

of obtaining a better correlation between the prediction value and the performance measure. We obtain improvements up to 9.5% in terms of correlation coefficient with respect to the best individual predictor. The second contribution regards the robustness of the performance measure in relation with the correlation of the prediction value. The robustness refers to the stability of the measure when changing the retrieval systems or the test collections. We propose a more robust performance measure that allows us to check the prediction quality more accurately.

3.1 Introduction

The search engine performance depends on many variables, such as the parameter tuning, the retrieval model, the query formulation, the query difficulty, the document collection and so on. It is well known that for certain queries, most IRS do not manage to retrieve relevent documents, this making them difficult. In the TREC context, "a topic is considered difficult when the median of the average precision (AP) scores of all participants for that topic is below a given threshold (i.e., half of the systems score lower than the threshold)" [Carmel 2006]. We are interested in the query difficulty aspects, since it is important to avoid the system failure that yields bad performance. This difficulty could be caused by ambiguity, unclear or too vague formulation, lack of context, or the nature and the structure of the document collection [Carmel 2006], [Mothe 2005].

Some researchers have studied the characteristics of these queries so called "difficult". In [Cronen-Townsend 2002b], the authors suggest that the coherence of the language models of the documents that are likely to generate the query is a difficulty indicator. Harman and Buckley ([Harman 2009]) underline the issue of topic variability in the TREC tracks. They discuss the topic variation in terms of AP for best performing systems, the wide variation in performance across topics for a given system, the wide performance variation across topics regarding the effectiveness of some methods such as relevance feedback, and the wide variation between two variants of the same IRS with respect to the rank of the same retrieved document. In [Carmel 2010], the authors seek for the reasons that cause search engines to fail for some queries and then they review various approaches that estimate the query difficulty.

We also showed that query difficulty estimation could have useful applications. For example, in Chapter 2 we laid stress on the fact that the disambiguation process improves the results for queries with low performance in terms of $P@k$, that is to say the difficult queries. Thus, it seems interesting to deal with difficult queries apart, in a specific, particular way, in order to obtain better performance. For this, it is necessary to find ways to identify them.

Various predictors of heterogeneous nature have been suggested in the literature. Thus the predictors may be classified by their nature, such as linguistic or statistical, or as pre-retrieval and post-retrieval, whether they take into account the retrieval results [Carmel 2010].

In our study, we propose to test the correlation between certain difficulty predictors and a query difficulty measure, based on AP. AP is a performance measure widely used in literature, especially in query difficulty prediction area. We show that the correlation for each predictor, individually, is not high. Our hypothesis is that as the predictors are different in nature, combining them, we could get a more correlated measurement with the difficulty. We are proposing two linear interpolations between two predictors: one pre-retrieval and one post-retrieval. In addition, we take into account the efficiency in terms of computation time, to ensure the applicability in a practical use of a prediction system.

The other contribution is a robust method, meaning a stable method with respect to changes of the retrieval method or the test collections, in order to check the correlation of predictors with the query difficulty. We have shown that for systems with a similar MAP, the correlation coefficients with the same difficulty predictor may be very different. We show that it is more robust to consider the average of several systems as a measure of difficulty, in order to avoid the variability yielded by choosing only one system.

The rest of this chapter is structured as follows: in Section 3.2 we present the existing research in the field of query difficulty prediction. The predictors are presented with respect to the retrieval process. If a predictor does not consider any information resulted after the retrieval process, such as the ranked document lists or the retrieved document scores, then it is a pre-retrieval predictor. Otherwise, it is a post-retrieval predictor. In Section 3.3 we give details regarding the predictors chosen to be employed further in our study. We have chosen predictors of heterogeneous nature and from both pre and post retrieval types, in order to cover a vast predictor range. Section 3.4 is dedicated to our proposed predictor combinations. We evaluate our propositions in In Section 3.5

3.2 Related work on query difficulty prediction

In this section, we present the types of prediction measures, the predictors used in the literature and the connection between the query difficulty and AP. This connection is verified by the value of a correlation coefficient (Spearman's, Pearson's, or Kendall's *tau*) between the predictor value and AP. A high correlation coefficient, either positive or negative, suggests a good prediction quality.

By the involvement of the retrieval results in the computations, the difficulty predictors can be divided in two categories: **pre-retrieval** predictors and **post-retrieval** predictors [Carmel 2010].

Since pre-retrieval predictors do not require the retrieval results, they are faster to compute and retrieval independent, thus they are more interesting for real applications. However, until now, the post-retrieval predictors have been proven to be more effective than pre-retrieval predictors.

3.2.1 Pre-retrieval predictors

Pre-retrieval prediction estimates the query difficulty before the retrieval process takes place. Because of that, only the query terms together with certain predefined statistics extracted from the documents can be utilized for prediction.

For example, one could consider the pre-retrieval predictor based on the query length. The query length represents the number of unique words in the query, after eliminating the stop words. We could imagine that the longer the query is, the easier it is for the IRS to respond, having in mind the contextual contribution of the words. In [He 2004], the authors have shown that, contrarily to this hypothesis, there is no correlation between the query length and the system performance. They have used TREC4, TREC7 and TREC8 collections for their tests. In addition, treating long queries may be difficult, while they generally contain considerable amounts of noise (terms considered relevant by the users, that in fact create confusion for the system). The authors have reported a weaker correlation for the Simplified Clarity Score predictor, which will be described later in this section, in the case of long queries and they assumed that the cause might be the maximum likelihood of the query model, which becomes not reliable when the query length increases.

The pre-retrieval predictors can be extracted either based on linguistic methods, or on statistical methods. The linguistic methods apply natural language processing (NLP) techniques and analyze the query expression, in order to obtain query difficulty indicators such as level of ambiguity or polysemy.

Mothe and Tanguy [Mothe 2005] have extracted 16 different linguistic query features. These features include morphological features such as the average number of morphemes per query word. Another linguistic feature studied by Mothe and Tanguy consists in polysemy measured by the average number of synsets in WordNet, per word. They have shown that the majority of linguistic features, such as the average number of proper nouns, the average number of numerical values, or the average number of prepositions, are not strongly correlated with IRS performance. In the same manner, negative results have been reported by Hauff [Hauff 2010b], who has tested the average semantic distance of query terms to predict performance. The semantic distance or semantic relatedness between terms can be computed from co-occurence statistics in document collections or from WordNet. It measures how related two terms are.

Statistical predictors can be classified in four main categories [Carmel 2010], according to the query characteristics that they try to estimate: query specificity, the common terms between the query and the collection, the coherence of the query term distribution and the relationship strength between the query terms.

The statistical methods analyze the query terms distribution in the document collection, searching for deviations in the distribution of the query terms frequency.

In this context, the pre-retrieval statistical predictors have been extensively studied [He 2004], [He 2008], [Zhao 2008], [Hauff 2008]. Two term-based statistics have been frequently used: the inverse document frequency (*idf*) and the inverse collection term frequency (*ictf*) [Hauff 2010a]. Hauff has used several TREC collections, such as TREC Robust, WT10G and GOV2 for experiments; correlation values she reports are up to 0.480 in the case of *idf* and up to 0.465 in the case of *ictf*, respectively. The Kendall's τ correlation coefficient has been employed for obtaining the predictor performance.

The query scope (*QS*) predictor, proposed by He and Ounis [He 2004], measures the percentage of documents containing at least one of the query terms in the collection. The query scope yielded performances up to 0.363 in terms of correlation. The simplified clarity score (*SCS*) measures the Kullback-Leibler divergence between the simplified query language model and the collection language model, as a query specificity indicator. The simplified query language model is given by the equation $\frac{qtf}{ql}$, where qtf is the number of occurrences of a query term in the query and ql is the query length. The correlation coefficient values for the *SCS* have been up to 0.448. The experiments have been conducted over TREC4, TREC7 and TREC8 collections and the Spearman's correlation coefficient has been used.

In [Zhao 2008], the authors have measured the similarity based on a vector space model between the query and the collection, while considering the collection as a single large document composed of concatenation of all the documents. They have conducted experiments on TREC Robust, WT10G and GOV2 benchmark collections, with reported result values up to 0.640 on a subset of queries and up to 0.478 in general, in terms of Spearman's correlation coefficient.

Regarding the coherence, He et al. [He 2008] have studied the potential of query-coherence-based measures to predict query difficulty. The query coherence with this respect is related to the inter-similarity of documents containing the query terms. This type of measures require a heavy analysis during the indexation process, in order to be exploited during search. Zhao et al. [Zhao 2008] have proposed a less expensive approach, which measures the variance of the term weights over the documents containing it within the collection.

To finish with the statistical measures, we mention the term relatedness based predictors. They explore term co-occurrences statistics. If the query terms co-occur frequently in the collection, that query is supposed to have a good performance, assuming all query terms are related to the same topic [Metzler 2005], [Hauff 2010a].

In conclusion, the pre-retrieval predictors are faster to compute since they are retrieval process independent. However, even if their correlation with average precision is significant, they still do not reach correlation levels high enough to state that they represent efficient prediction methods. The pre-retrieval predictors based on *idf* yield the best results, with correlation coefficient values up to 0.48 [Hauff 2010b]. Next, we present the post-retrieval predictors, which are more correlated with average precision than the pre-retrieval predictors.

3.2.2 Post-retrieval predictors

In contrast with the pre-retrieval methods, the post-retrieval prediction methods analyze the retrieval results, that is, the top retrieved documents in response to a query. We have to mention that the prediction quality strictly depends on the retrieval process, as different results are expected while using different retrieval methods [He 2004], [Zhao 2008].

The post-retrieval methods can be classified as follows: "Clarity"-based methods, robustness-based methods and score distribution based methods.

The Clarity approach for performance prediction of an IRS is based on measuring the coherence (clarity) of the result-list with respect to the corpus. The Clarity measure applied by Cronen-Townsend et al. [Cronen-Townsend 2002b], is based on the KL-divergence between the language model of the result set and the language model of the entire collection. They have obtained a Spearman's correlation coefficient value up to 0.577, over TREC4-8 collections and over AP88+89 collection.

Zhou and Croft [Zhou 2007] have proposed a framework for measuring query robustness, named Query Feedback (QF), which models retrieval as a communication channel problem. From the initial query, a new query is constructed, by adding noisy terms, and the overlap between the new retrieval and the initial results is considered as a robustness measure. A robust query represents an easy query. The Query Feedback (QF) yielded Pearson's correlation coefficient values up to 0.480, tested over Robust04, Robust05 adhoc and Terabyte TB06 adhoc tracks.

Query perturbation has also been examined by Vinay et al. [Vinay 2006], who has studied the effect of small modifications to the query term weights on the search results. Similarly, if the results were changed drastically then the query is presumed to be difficult. The experiments conducted over 200 topics (301-450 and 601-650) and their corresponding documents from the TREC Robust collection have leaded to results up to 0.521 in terms of Kendall τ correlation coefficient.

The effect of document perturbations on the top retrieved document list is another form of robustness estimation [Vinay 2006], [Zhou 2006]. This robustness is about the stability of the retrieved document list with respect to noise injection in the query. The top retrieved documents are injected with noise (for example, by adding or removing terms). Afterwards, these documents are re-ranked. High similarity between the original list and the re-ranked list reveals query robustness. A robust query from this point of view should be an easy query, therefore the robustness measurement is a difficulty predictor. In [Zhou 2006] the tests have been done over TREC1-5, TREC Robust and Terabyte04&05, with results up to 0.612 in terms of Pearson's correlation coefficient.

Regarding the retrieval perturbation, Aslam and Pavlu [Aslam 2007] have studied query robustness with respect to using different retrieval methods. They have proven that the disagreement between result lists retrieved for the query by multiple scoring functions is an indication for query difficulty. Experimental results on TREC data sets (TREC5-8, TREC Robust and Terabyte04&05) show a relatively strong correlation between the agreement of systems and query performance. Analyzing the submission of all the participants at several TREC tracks, Aslam and Pavlu have shown that their diversity predictor, the Jensen-Shannon divergence, is strongly correlated to query difficulty, difficulty measured by the median system (in terms of MAP) among all the participants. They have reported correlations up to 0.623, in terms of Kendall τ correlation coefficient.

Finally, the score distribution analysis for query difficulty prediction is fast, in terms of computational time, because it is not necessary to treat the document terms, unlike Clarity or robustness. A proposed predictor would be the Weighted Information Gain (*WIG*) [Zhou 2007]. WIG essentially measures the divergence between the mean retrieval score of top-ranked documents and that of the entire corpus. The reported results reach Pearson's correlation coefficient values up to 0.478, over TREC Robust and GOV2 collections.

The Normalized Query Commitment (*NQC*) predictor [Shtok 2009] measures the standard deviation of the retrieved documents, normalized by the score of the entire collection. The reported performance of *NQC* was up to 0.7 in terms of Pearson's correlation coefficient. The experiments have been conducted over TREC1-3, TREC4-5, TREC Robust, WT10G and GOV2.

Kurland et al. proposed in [Kurland 2012] a probabilistic framework in order to derive and explain existing predictors, showing that even though some predictors seem very different, they share the same formal basis. They use the same framework to integrate pre-retrieval and post-retrieval predictors. The experiments have been conducted over TREC5, TREC Robust, WT10G, GOV2 and parts of ClueWeb09 and Clueb10 collections. They have reported results that overcome, over several collections, the level of 0.70, in terms of Pearson's correlation coefficient.

In conclusion, the post-retrieval predictors generally yield better prediction performance than the pre-retrieval ones. We underline here the NQC [Shtok 2009], which is reported to obtain correlation coefficients up to 0.70. Another strong post-retrieval predictor would be QF.

A summary of the presented predictors is shown in Table 3.1.

We also summarize the maximum correlations reported for the main results from the literature (see Table 3.2). We mention the authors, the name of the predictor (where it is the case), the correlation values, the employed correlation coefficients and the test collections. We stress on the fact that results are not comparable, since they have been produced using different correlation coefficients, different test collections and different AP values as difficulty measures.

In the following section we present several approaches from the literature that

Table 3.1: Summary on query difficulty predictors

Authors	Pre-ret.	Post-ret.	Predictors
[He 2004]	x		query length
[Mothe 2005]	x		16 linguistic features, polysemy included
[Hauff 2010a]	x		average semantic distance of query terms
idf	x		term statistics: inverse document frequency
[Hauff 2010b] (ictf)	x		term statistics: inverse collection term frequency
[He 2004] (QS)	x		documents containing at least one query term
[Zhao 2008]	x		vector space model; similarity query-collection
[He 2008]	x		inter-similarity of documents containing query terms
[Zhao 2008]	x		variance of term weights over the documents
[Metzler 2005]	x		statistical measure: term co-occurence
[Hauff 2010b]	x		statistical measure: term co-occurence
[Cronen-Townsend 2002a] (Clarity)		x	KL-divergence, language model
[Zhou 2007] (QF)		x	overlap of two retrieved lists
[Vinay 2006]		x	impact of small changes in queries
[Aslam 2007]		x	disagreement between result lists
[Zhou 2007]		x	divergence between the mean retrieval scores and the corpus
[Shtok 2009] (NQC)		x	normalized standard deviation of retrieval scores
[Kurland 2012]	x	x	Probabilistic experimental framework

combine query difficulty predictors.

3.2.3 Predictor combinations

In [Hauff 2009], the authors combine pre-retrieval predictors in order to better predict the query performance. They use regression to combine the predictors and the method is tested over TREC6, TREC7 and TREC8 considered together, WT10G and GOV2. The results show that the predictor combinations were comparable the best single predictors.

Besides the probabilistic framework for query difficulty predictors, Kurland et al. [Kurland 2012] propose predictor combinations based on sum, average or product. They also propose a method to integrate pre- and post-retrieval predictors, based on probabilities. Their contribution is tested over various collections, such as TREC5, Robust, WT10G, GOV2, ClueWeb09, and ClueWeb10. They report important improvements in the case of combinations, compared to single predictors. For instance, over Robust, the average *idf* has a Pearson's correlation coefficient of 0.466 and the QF predictor has a correlation of 0.500. Their combination yields a correlation coefficient of 0.602.

TABLE 3.2: Results of query difficulty predictors from the literature

Predictor	Maximal correlation	Correlation coefficient	Collections
Pre-retrieval predictors			
[Hauff 2010b](idf)	0.480	Kendal's τ	Robust, WT10G, GOV2
[Hauff 2010b](ictf)	0.465	Kendal's τ	Robust, WT10G, GOV2
[He 2004](QS)	0.363	Spearman	TREC4, TREC7, TREC8
[He 2004](SCS)	0.448	Spearman	TREC4, TREC7, TREC8
[Zhao 2008]	0.478	Spearman	Robust, WT10G, GOV2
[Mothe 2005] (SYNSETS)	-0.354	Pearson	TREC3, TREC5, TREC6, TREC7
Post-retrieval predictors			
[Cronen-Townsend 2004b] (Clarity)	0.577	Spearman	TREC4-8, AP88-89
[Zhou 2007](QF)	0.480	Pearson	Robust, Terabyte06
[Vinay 2006]	0.521	Kendall's τ	Robust
[Zhou 2006]	0.612	Pearson	TREC1-5, Robust, Terabyte04&05
[Aslam 2007]	0.623	Kendall's τ	TREC5-8, Robust, Terabyte04&05
[Zhou 2007](WIG)	0.478	Pearson	Robust, GOV2
[Shtok 2009](NQC)	0.700	Pearson	TREC1-3, TREC4-5, Robust, WT10G, GOV2
[Kurland 2012]	0.700	Pearson	TREC5, Robust, WT10G, GOV2, ClueWeb09, ClueWeb10

In [Lee 2014], the authors use SVM based on features extracted from multiple predictors with the purpose of choosing between inter-language and intra-language PRF in the context of cross-language IR. They have manually created 50 queries in English that are then used to retrieve documents written in Chinese. They reported improvements up to 27%, compared to monolingual baseline approaches. In Chapter 4, we also employ SVM learning with features based on query difficulty predictors in the context of query expansion.

Sarnikar et al. [Sarnikar 2014] propose methods that combine various predictors, using them as features for learning. By regression and SVM, the authors try to route queries, in the case of multiple collections, classified by domains. Their models are tested over CACM, CISI, CRAN, TIME and TREC9 collections. The reported results show minor improvements in terms of MAP (from 0.2755 to 0.2783 in the case of TREC9).

We have to mention the variety of applications for query difficulty predictors and for predictor combinations, such as selective query expansion, query routing in domain-specific repositories, or cross-language IR.

3.3 The retained predictors

Individual query difficulty predictors are, in most cases, not applicable in real applications, since they are insufficiently correlated with the difficulty method. However, due to their heterogeneous nature, they cover various aspects of difficulty, as presented in Section 3.2. This drives our research interest to combine predictors in order to obtain better correlation coefficients with query difficulty.

We use both pre-retrieval and post-retrieval predictors in the contributions we propose [Chifu 2013], [Chifu 2014]. Moreover, the retained predictors are very different in nature (linguistic, statistical, based on term distribution, based on term similarity). The hypothesis is that heterogeneous predictors would cover more efficiently the various aspects of query difficulty. In addition, the retained predictors are among the best in the literature, such as idf, NQC, or QF (see Section 3.2).

Pre-retrieval predictors may be calculated either using exclusively information on the query terms, or also using the document collection itself. When based on the query terms only, pre-retrieval predictors can be computed easier than when the entire collection is involved. However, the information they carry is rather weak since it is based on few information (only the query text and some features on the terms it consists of). On the contrary, post-retrieval predictors treat the retrieved document lists which contain extra information (document scores, ranks). Even though this type of predictors involve a retrieval process in order to treat the result list, they do not require any supplementary collection processing, meaning that the amount of resources, in terms of time and computations, stays reasonable.

In the following sections, we give more details regarding the query difficulty predictions that we have selected for our models. The predictors retained for our experiments are among the best from literature and they yield the higher correlation with query difficulty, except for the predictor based on WordNet senses [Mothe 2005], for which the authors report a weak correlation. However, this polysemy predictor remains the most correlated with difficulty from its class of linguistic predictors and, moreover, we will see later on that it is useful when combined with predictors of other nature (see Section 3.6).

3.3.1 WordNet Senses

WordNet Senses (*WNS*) [Mothe 2005] represents a pre-retrieval linguistic predictor, it is an ambiguity measure and it computes the average number of WordNet[2] senses for all the query terms of a query q:

$$WNS(q) = \frac{1}{|q|} \sum_{t \in q} senses_t, \tag{3.1}$$

[2]http://wordnet.princeton.edu/

where *senses*$_t$ represents the number of WordNet synsets for the query term t from q. This predictor will be employed in one of the proposed predictor combinations, described in Section 3.4.

3.3.2 Inverse Document Frequency

Inverse Document Frequency (*IDF*) is a statistical pre-retrieval predictor and it measures if a term is either rare or common in the corpus. Its value for a certain query represents the average of *idf*s for all the query terms. The IDF for a query q ($IDF(q)$) is computed as follows:

$$IDF(q) = \frac{1}{|q|} \sum_{t \in q} \log_{10} \left(\frac{N}{N_t + 1} \right), \tag{3.2}$$

where N is the total number of documents in the collection, N_t the number of documents containing the term t. This predictor is employed in the second proposed predictor combination, presented in Section 3.4.

3.3.3 Standard Deviation

Standard Deviation (*STD*) represents a statistical post-retrieval predictor that measures the degree of variation with respect to the average for the list of scores assigned to the retrieved documents, corresponding to a query. It is a variant of NQC [Shtok 2009], without normalization, since we do not consider the score for the entire collection as a document. For a query q and for the first N_q retrieved documents, the STD is calculated by the formula:

$$STD(q) = \left(\frac{1}{N_q} \sum_{i=1}^{N_q} \Big(score(d_i^q) - \frac{1}{N_q} \sum_{j=1}^{N_q} score(d_j^q) \Big)^2 \right)^{\frac{1}{2}}, \tag{3.3}$$

where $score(D_i^q)$ represents the score of the i^{th} retrieved document for q and N_q stands for the number of documents retrieved by the IRS for q. While *WNS* and *IDF* are retrieval model independent, for STD the retrieval model represents a parameter. This predictor represents the main term for both predictor combinations proposed in Section 3.4.

3.3.4 Query Feedback

Query Feedback (*QF*), proposed in [Zhou 2007] (see Section 3.2.2), is a post-retrieval predictor which models retrieval as a communication channel problem. It computes the overlap between two retrieval lists obtained by processing an initial query and the same query expanded by additional terms, respectively. Having the two retrieved lists, the overlap represents the number of documents that these lists have in common. A cut-off level x could be established for the document lists, that means the first x retrieved documents only are considered for the overlap. In order to obtain a measure between 0 and 1, the overlap is normalized by x. The parameters for QF are the cut-off level and the retrieval model. For a query q and the two retrieved document lists L_1 and L_2, with a cut-off level x, the QF is computed as follows:

$$QF(L_1^x(q), L_2^x(q)) = \frac{|L_1^x(q) \cap L_2^x(q)|}{x}. \tag{3.4}$$

Since we mentioned in Section 3.2 that reported results from the literature are not comparable, we reimplemented the retained predictors and we measured their Spearman's correlation coefficient with a difficulty measure, over TREC7 and TREC8. The values are reported in Table 3.3. The best predictor is STD and the weakest is WNS.

TABLE 3.3: Correlation coefficients between the reimplemented predictors and the difficulty measure, for TREC7 and TREC8

Predictor	**TREC7**		**TREC8**	
	Corr.	***p*-value**	**Corr.**	***p*-value**
WNS	−0.1796017	0.08747	−0.2814032	0.05086
IDF	0.2986315	0.01082	0.3175586	0.00165
STD	0.5531333	0.03515	0.5446819	0.00135
QF	0.4464256	0.00131	0.4927206	0.00019

In the following section we propose two linear combinations between predictors presented above, with the purpose of improving the prediction quality of individual predictors. This predictor will be employed in Chapters 4 and 5 to generate learning features.

3.4 Combinations of difficulty predictors

Even if single predictors are not sufficiently correlated with the AP as a difficulty measure such that they may be applied in real applications, we believe th at combining them will boost the predicting performance. Moreover, a heterogeneous nature of predictors targeted for combination will supposedly induce a higher correlation. We therefore propose two linear combinations between predictors described in section 3.3. The ease of implementation and the possibility of weighting the combination components made the linear combination to seem an appropriate choice. Intuitively, the importance of the predictors within the combinations must not be equal and shall be tuned. Most probably, a better predictor will have a higher importance.

The first combination takes into account the STD and the WNS creating a post/pre-retrieval combination. Even though the pre-retrieval predictor WNS is weakly correlated to AP, the hypothesis is that in combination it might enhance the prediction quality. The computations for the first combination, for a query q are the following:

$$COMB_1(q) = \lambda \frac{STD(q)}{max(STD)} + (1 - \lambda) \frac{1}{WNS(q)}, \tag{3.5}$$

where $max(STD)$ represents the maximal STD value over all queries from a collection, in order to normalize the ratio corresponding to STD between 0 and 1. λ is a weighting coefficient which gives a varies the importance of STD and WNS in the final result. Here we consider the inverse of WNS, since the hypothesis is that a more ambiguous query (higher WNS) will yield a lower performance.

The second predictor combination concerns the STD and IDF measures and is computed by the following equation:

$$COMB_2(q) = \lambda \frac{STD(q)}{max(STD)} + (1 - \lambda) \frac{IDF(q)}{max(IDF)}, \tag{3.6}$$

where $max(STD)$ and $max(IDF)$ represent the normalization factor, that is the maximal values per collection, of STD and IDF, respectively. λ has the same role as for the $COMB_1$ predictor.

3.5 Evaluation

This section describes the experimental framework we propose, with the collections and the parameter settings for predictors, used in our work.

In order to evaluate the predictors we mentioned, we used two TREC benchmark collections, TREC7 and TREC8, respectively (see Section 1.4.3). They are widely used test collections.

3.5.1 Parameter setup for predictors

For the STD predictor, we used runs built by Terrier (see Section 1.4.6), trying various parameter configurations. For each collection, we opted for the parameter setting that gives the best MAP. We used the set of retrieved documents, representing the first 1000 documents which have obtained the highest scores for the established parameter configuration. The MAP values, of 0.2988 for TREC7 and

0.2315 for TREC8, corresponding to the chosen runs, are consistent with other work from the literature [He 2005], [Zhong 2012]. We present the retained configurations in the following paragraph.

The configuration which yielded the best MAP results, for TREC7, is as follows: a two step indexing (with both and direct and inverted indexes), using the BB2 weighting model (with parameter $c = 1$). As for the query expansion model, the model was the parameter free KL model (KLbfree). This configuration considered 3 documents for expansion. A term must appear in at least 2 documents in order to be valid for query expansion. Finally, the number of query expansion terms to be added was set to 10. The queries used all the three topic parts (title, description and narrative), that are described in Section 1.4.4. On the other hand, for TREC8, the best parameter configuration stands in place, except for two differences: the query expansion model is the parameter free DFR (DFRee) and the narrative part of the topic is not involved anymore in the query construction.

The post-retrieval predictor STD uses the document scores from the list retrievad by Terrier, with the purpose of computing the standard deviation. Contrarily, the other two predictors involved in our combinations (WNS and IDF) do not imply the retrieval process in order to predict the query difficulty.

For the IDF we need to know the total number of documents in which a query term t appears (N_t, in Equation 3.2). Terrier provides the lexicon access after the indexing step. Throughout the information concerning the terms vocabulary we also can find the N_t, which is required. We considered the vocabulary after stemming, in order to eliminate various forms of a term.

Regarding the WNS, we used a part of speech tagger (Stanford POS Tagger) to identify the part of speech for every query term. We then search in WordNet (see Section 2.2) the number of synsets for a given term, with that particular part of speech identified by the part of speech tagger.

3.5.2 Prediction quality

Our first contribution in the query difficulty context context has been the proposition of linear combinations of heterogeneous predictors in order to improve the prediction accuracy. We present here our second contribution which consists in using the average over several systems, in terms of AP per query, in order to obtain a more robust difficulty measure, as ground truth to test predictor correlations.

To analyze the prediction quality, we computed the Spearman's correlation coefficient (see Section 1.4.2) for two variables: the predictor in cause and the measure of query difficulty. The Spearman's correlation coefficient is less sensitive to strong outliers than Pearson's correlation coefficient. As a measure for difficulty we have chosen the average AP, per query, for all participant systems at TREC7 and at TREC8, respectively. Research in this area have shown that this averaged-based measure is robust in terms of collection independence and system variability

[Bigot 2011], [Carmel 2006]. [Carmel 2006] et [Aslam 2007] propose the median system in terms of AP as a difficulty measure. However, in Section 3.6.1, we will show that it is more robust to consider an average MAP than the MAP of a single system. Indeed, the result variability is very important, depending on the system choice.

For *IDF*, we find a positive correlation based on Spearman's correlation coefficient [Chifu 2013]. In the same way, a greater *IDF* value indicates an easier query. A positive correlation is also to be found in the case of the *STD* [Chifu 2013] and of the *QF* [Zhou 2007], respectively. Contrarily, the correlation between *WNS* and AP is negative, meaning that a higher number of ambiguous query terms involves a more difficult query.

The correlation results for the predictors, both individual and in combinations, are underlined in the following section.

3.6 Results and discussion

The obtained results are presented in this section. We first analyze the variability of correlations between a difficulty predictor and several AP values, corresponding to several reference systems. The analysis of this variability represents an argument for proposing our contribution regarding the robustness of a difficulty measure, when considering the AP average over multiple systems. Then we comment on the correlations between the predictors we proposed and the new difficulty measure, described in Section 3.5.2 (average AP over TREC participants).

3.6.1 Variability in terms of predictor correlations

Unlike several studies that use the AP values of a single system to evaluate their predictors [Aslam 2007], [Cronen-Townsend 2002b], we propose here to use the average in terms of AP for multiple systems in the calculations. We justify this approach by the variability of correlations between predictors and the AP values for different IRS. In this section, we illustrate this hypothesis through the correlation analysis of the *IDF* predictor and the AP. We chose the predictor *IDF* as a reference, because its value is independent of the system used, being a pre-retrieval predictor. Moreover, *IDF* is among the best pre-retrieval predictors [Hauff 2010a].

Regarding the retained systems, we selected 9 systems that participated to the TREC7 and TREC8 competitions, respectively. See Section 1.4.5 for more details on TREC participants. The criterion to choose the systems was their performance in terms of MAP, which must cover all the range of MAP values, from worst to best. Among the 10 considered systems we have the best participant system, the weakest participant system, and also the simulated system which would have obtained the average AP in per query, over multiple systems, that represents our approach. In this scenario the simulated system represents the average for all participants in TREC7 and TREC8, respectivelly. We computed, for each system, the Spearman correlation coefficient and the p-value (see Section 1.4.2) between IDF and the difficulty measure, in terms of AP. Correlations are to be found in Table 3.4 and in Table 3.5. Results are sorted in descending order with respect to MAP. The system representing the average of all systems is called *AvgMAP*.

The correlation coefficient values, as well as the p-values, vary considerably. For example, for systems with MAP values between 0.2609 and 0.3346, the correlations vary between 0.22 and 0.34 with their corresponding p-values between 0.0008 and 0.01857, for TREC8 (see Table 3.5). In addition, for TREC7 (see Table 3.4) the best systems had a p-value higher than 0.05 (0.10650 and 0.11830, respectively), thus the results are not statistically significant, meaning that the correlation value is not reliable. Regarding the systems with a low MAP the confidence in correlations is not very interesting, because they represent systems which in any case are not effective. For a MAP value of 0.0287 we have a correlation of 0.5083825 with a p-value of 0.15610 and for a very close MAP value of 0.0273, we obtained a correlation of 0.2158393 and a p-value of 0.00167 (see Table 3.5), therefore, the variation is very important.

Table 3.4: Correlations between IDF and AP, per systems, for TREC7. The MAP for each system is also mentioned

TREC7			
Name	**MAP**	**Corr.**	***p*-value**
CLARIT98COMB	0.3702	0.2329948	0.10650
t7miti1	0.3675	0.2505162	0.11830
uoftimgr	0.2755	0.2632893	0.02930
ok7as	0.2614	0.2039376	0.12190
FLab7atE	0.2020	0.2692437	0.04071
AvgMAP	*0.1992*	*0.2986315*	*0.01082*
Brkly24	0.1714	0.3014238	0.01299
APL985L	0.1576	0.2406300	0.03061
KD70000	0.0250	0.1737295	0.00688
dsir07a01	0.0117	0.2579675	0.46870

Table 3.5: Correlations between IDF and AP, per systems, for TREC8. The MAP for each system is also mentioned

TREC8			
Name	**MAP**	**Corr.**	***p*-value**
perf-class	0.4726	0.3430094	0.00697
CL99SDopt2	0.3520	0.2228625	0.03048
8manexT3D1N0	0.3346	0.2201786	0.01857
ok8amxc	0.3169	0.2789503	0.00647
ibmg99b	0.2609	0.3445461	0.00080
AvgMAP	*0.2533*	*0.3175586*	*0.00165*
Dm8TFbn	0.1630	0.3768819	0.00010
AntHoc1	0.0287	0.5083825	0.15610
isa25t	0.0273	0.2158393	0.00167
isa25	0.0026	0.3642227	7.57E-06

This analysis shows that the correlations vary dramatically depending on the selected systems as a reference, even if we consider an overall effective system in terms of MAP. This led us to choose rather a system constructed from the average AP of several systems, as the basis for calculation. This will represent the reference used in the following section to measure predictor correlation and efficiency.

3.6.2 Correlations of predictor combinations

After establishing that is more robust with respect to system variability to choose the AP average as a difficulty measure, we present in Table 3.6 the correlations between this measure and predictors we retained, for TREC7 and TREC8 collections. For the $COMB_1$ (the interpolation between STD and WNS from Equation 3.5) and the $COMB_2$ (the interpolation between STD and IDF from Equation 3.6) predictor combination, presented in Section 3.4, we tested various λ parameter values between 0 and 1. We retained the values that yielded the best correlation coefficients, $\lambda = 0.7$ for TREC7 and $\lambda = 0.78$ for TREC8, respectively.

Table 3.6: Correlation coefficients between the proposed predictors and the difficulty measure, for TREC7 and TREC8

Predictor	**TREC7**		**TREC8**	
	Corr.	***p*-value**	**Corr.**	***p*-value**
STD	0.5531333	0.03515	0.5446819	0.00135
WNS	−0.1796017	0.08747	−0.2814032	0.05086
IDF	0.2986315	0.01082	0.3175586	0.00165
$COMB_1$	0.4529652	0.01986	**0.5965426**	0.00023
$COMB_2$	**0.5601441**	**0.00567**	0.5849220	**0.00010**

Among the three proposed predictors (STD, WNS and IDF), STD is the most correlated with the difficulty measure, while WNS is the most weakly correlated. The p-values for WNS show that its correlations are not statistically significant. On the contrary, the combination between STD and WNS ($COMB_1$) yields a high correlation coefficient, with also a high degree of confidence ($Corr$=0.5965426 and p-value=0.000231, for TREC8).

The IDF predictor is more strongly correlated than the WNS predictor and this is certainly the reason why the STD and IDF combination ($COMB_2$) performs best.

With regard to the *lambda* parameter, the value that generated the best results was 0.78 for $COMB_1$ and 0.7 for $COMB_2$, respectively. The best performing predictor, STD therefore provides more than 70% in the combination predictors. For $COMB_1$, WNS having a weak individual correlation, STD shall take even more weight (0.78).

For the same benchmark collections (TREC7 and TREC8), the "Clarity Score" [Cronen-Townsend 2002a] yields an average correlation coefficient of 0.535. The $COMB_2$ predictor we propose allows to obtain an average correlation coefficient of 0.573, in the same context, with respect to the AP of a single system, not specified by the authors. "Clarity Score" computes the relative entropy between the query language model and the collection language model. This process is computationally heavy since it calculates the relevance scores for the query model [He 2004]. On the contrary, the STD directly uses the retrieved documents scores and the IDF uses the term frequencies from the vocabulary created by the search engine, thus the $COMB_2$ predictor is faster to compute.

3.7 Conclusion

In conclusion, in this chapter we presented the existing query difficulty predictors, we proposed new predictor combinations and a query difficulty measure to test its correlations with the predictors.

We showed that it is more robust to consider the average of several systems as a measure of difficulty to compare the correlations with different predictors, in order to avoid the variability generated by only one system. We have shown that for systems with a similar MAP, the correlation coefficients with the same predictor of difficulty can be very different. In addition, for TREC7, the correlation between the AP and the difficulty predictor, when the best system is chosen, is not statistically significant (p-value).

On the other hand, we proved that if the predictors are heterogeneous in nature (statistical, linguistic, pre-retrieval, post-retrieval, etc.), their combination may produce a stronger correlation with the measure of difficulty. We considered here two linear combinations, each based on the interpolation of two predictors. $COMB_1$ is the interpolation between STD and WNS and $COMB_2$ is the interpolation between STD and IDF. $COMB_1$ yields better correlation coefficient over TREC7, while $COMB_2$ yields better correlation coefficient over TREC8. Moreover, $COMB_1$ is more stable with respect to the p-value in the case of both test collections, suggesting that this combinations is more reliable.

Finally, the idea of successfully combining query difficulty predictors suggests an application that consists in learning to decide which system would respond best for a particular query, when various systems are available, all based on query difficulty prediction queries. For instance, since the query expansion may harm the performance of some queries, while improving performance of others (see Section 1.3), it would be interesting to be able to learn which query expansion method should be applied for each query. This premise is tested in the next chapter, dedicated to selective query expansion.

Chapter 4

Selective query expansion

Contents

We have seen in the Chapter 3 that combining query difficulty predictors of heterogeneous nature may enhance the prediction accuracy. Moreover, even though an individual predictor is weakly correlated with a difficulty measure, it can still have a positive impact on prediction quality when used in combinations. On the other hand, choosing which query expansion method should be employed, in order to obtain better retrieval performance, represents a difficult problem. It would be interesting to be able to learn which expansion to use for each query. Thus, in this chapter we propose to bring together query difficulty prediction and machine learning with the purpose of obtaining a selective query expansion method. We propose a SVM-based approach with features based on various query difficulty predictors, in order to choose between two query expansion types. This research was published in [Chifu 2014], in CORIA conference.

4.1 Introduction

We noticed so far that IRS performance is strongly dependent on queries. In Chapter 2, the proposed query disambiguation-based method was proved as beneficial to IR by yielding improvements on high precision, with even more impact on queries with low performance. In Chapter 3 we tried to obtain a better correlation between predictors and difficulty, by interpolating the values of individual predictors of heterogeneous nature, with the purpose of making difficulty prediction more useful in real applications. One possible application could be selecting which system or what method should be applied for a particular query, that is to say selective IR.

We introduce in this chapter a selective IR method that learns to choose between two types of QE, one based on pseudo-relevance feedback (PRF) and the other based on simulated manual refinement of pseudo-relevance feedback (PRF). The learning mechanism is based on features extracted from query difficulty predictors. First, we briefly present the selective IR field and then we focus on the selective QE according to query difficulty, which represents the context of our work in this chapter.

Selective IR refers to the selection of the most appropriate search engine configuration according to the context and the query. This challenge can find one of its origins in the consideration of query and system variability [Buckley 2004] [Lv 2009]. Because of variability in performance across queries and systems, it is natural to think of methods that are query-dependent. On the other hand, query difficulty is another feature that should be considered in systems [De Loupy 2000], [Carpineto 2001a]. Indeed, in the case of easy queries, one can consider that any reasonable configuration can be used since any of them will perform well on the user's point of view. On the contrary, in the case of difficult queries, the choice of configuration can make the difference between good or bad results, since for such queries most systems fail to retrieve relevant documents. Intuitively, one can maximize the gain in terms of relevant retrieved documents by using appropriate techniques.

Selecting the best configuration on a per query basis can lead to high performance improvement. Bigot et al. show that it is possible to learn the best configuration system using a sample of documents and relevance judgments [Bigot 2011].For example, Peng et al. propose to select the best learning to rank function on a per query basis [Peng 2010].

Other approaches focus on difficult queries and the case of query expansion (QE). One of the main hypotheses is that QE and specifically QE based on pseudo-relevance feedback (PRF) can lower results for poor performing queries since QE will be based on non relevant documents. Several authors propose to selectively apply QE based on query difficulty predictors as we do. The idea is to avoid the application of QE when it is likely to decrease performance [Amati 2004b], [Cronen-Townsend 2004a], [Chen 2012].

Rather than just deciding whether QE should be applied or not, some approaches try to apply QE differently according to queries [He 2004], [Cao 2008], [Lv 2009]. This also represents the context of our contribution. Lv and Zhai study the impact of the initial query in the PRF by the means of logistic regression. The features are based on queries and feedback documents [Lv 2009]. He and Ounis propose a method for selecting the most appropriate term-weighting model as a pre-retrieval technique [He 2004]. Cao et al. propose a term classification method to predict the usefulness of expansion term candidates [Cao 2008]. On the other hand, we propose a machine learning method two select between two types of QE, based on features extracted from query difficulty predictors.

Other approaches also rely on the capability of distinguishing difficult queries. We reviewed the query difficulty prediction in Chapter 3. Current query difficulty prediction approaches (ours included) tend to combine various evidence, generally based on some statistics [Kurland 2012], [Sarnikar 2014]. Many individual predictors such as the clarity score [Cronen-Townsend 2002b], the query scope [He 2004], the weighted information gain [Zhou 2007], or the normalized query commitment [Shtok 2009] are based on some statistics and correspond either to pre-retrieval or to post-retrieval predictors. Linguistics oriented predictors have also been studied [Mothe 2005]. These predictors can be used as evidence for selective application of QE or other search component. For more details on existing query difficulty predictors, see Section 3.2.

In our contribution, we consider QE as a key issue in selective IR processes, depending on the query. There are many ways QE can be achieved. In his survey, Carpineto distinguishes between RF, interactive query refinement, word sense disambiguation, and search result clustering [Carpineto 2012]. We employed word sense disambiguation and clustering retrieved documents in Chapter 2. Here, we focus on the first two types. RF uses information from the retrieved documents judged as relevant. Alternatively, PRF considers the top retrieved documents as relevant. While the first method implies caption of users' judgment or interaction, PRF can be fully automatic. On the other hand, interactive query refinement or expansion implies to use user studies or to consider query logs. Alternative solutions consist in simulating query refinement [Zhao 2012].

In our approach we consider a longer query (simulated by using the descriptive part of the queries) as a kind of query refinement. We hypothesize that the two types of QE are complementary and can be used in a selective way, depending on the query. Previous approaches using QE in a selective process are of two kinds: some consider selective QE (some queries will be expanded other will not); others decide which QE should be used. Our approach follows this second trend, considering that a difficult query should require a certain type of expansion accordingly to difficulty nature. Moreover, compared to other approaches, our method combines evidences extracted from the retrieved documents and thus from the document collection (using PRF) and other types of evidences, in our case collected from the descriptive part of the topics (see Section 1.4.4 on TREC topics). We use several query difficulty predictors. As developed in Chapter 3, each predictor focuses on a particular aspect of query difficulty, therefore several prediction measures combined should offer a complete and comprehensive characterization of a query. Some predictors are similar to the one used in previous studies [He 2004], [Chen 2012], others considering the importance of linguistic features [Mothe 2005]. The efficiency of this method is analyzed in terms of learning accuracy, MAP and query expansion robustness (see the robustness index introduced in Section 1.3).

This chapter is organized as follows: in section 4.2, we review related work. Section 4.3 overviews the selective QE method and presents the detailed query features and the query classification. Section 4.4 describes the evaluation framework. We present here the test collections, the retrieval parameters, the learning mechanism configuration and the evaluation process. Results and discussion are presented in Section 4.5. Section 4.6 concludes this chapter.

In the following section we present we the existing work in the field of selective query expansion.

4.2 Related work on selective query expansion

Considering various systems or various system configurations to answer users' needs has first been investigated in the case of meta-search in which various system results are fused to improve the ranked list of retrieved documents, either considering reinforcement on system decisions like in Comb-like functions [Fox 1994] or in order to favor diversity [Liu 2012, Santos 2010]. We employed re-ranking and Comb-like functions in Chapter 2, in our query disambiguisation method. These data fusion techniques however are based on the assumption that any query can be treated the same way. As opposed to this, some approaches have investigated the use of a query-dependent functions [He 2003], [Amati 2004b], [Cronen-Townsend 2004a], [Wilkins 2006] [Lv 2009], [Chen 2012].

Many selective approaches make the assumption that difficult queries should be treated differently from the others. Predicting query difficulty can be based on pre-retrieval and post-retrieval predictors, as presented in Section 3.2. Many researchers showed interest regarding this matter. In Section 3.2 we give details on difficulty predictors and here we briefly recall them. The *average Idf of query terms* is one simple pre-retrieval predictor: it measures the discriminative power of query terms. On the other hand, the *clarity score* aims at quantifying the level of ambiguity of a query and corresponds to the relative entropy between the query language model and the document collection language model [Cronen-Townsend 2002b]. The *Query Scope* [He 2004], measures the percentage of collection documents that contain at least one of the query terms. The *average number of query term senses* and the *complexity of the query* are other pre-retrieval predictors that are linguistics-based [Mothe 2005]. The post-retrieval predictor *weighted information gain* [Zhou 2007] measures the divergence between the average score of the top retrieved list and the entire corpus. Yom-Tov et al. suggest a method based on the agreement between the results from the full query and the ones obtained when sub-queries are considered [Yom-Tov 2005]. *Normalized query commitment* [Shtok 2009] is a post-retrieval predictor which measures the standard deviation for the scores of the retrieved documents.

Selective QE has been studied as a possible application of query difficult prediction: the system decides whether QE should be applied or not, based on the difficulty of the query.

In Amati et al. [Amati 2004b], the decision is based on InfoQ, an information theoretic function that combines query length, idf, and other features. They show that when considering a performing QE model, the selective approach can improve MAP and optimize the number of queries with no relevant documents on the top 10. The tests were conducted on 150 queries of TREC Robust and their corresponding documents. They slightly improve the MAP, for example from a MAP of 0.2479 in the case of expansion for all queries, to a MAP of 0.2524 in the case of selective QE.

Cronen-Townsend et al. [Cronen-Townsend 2004a] compare the language model obtained from the retrieved documents using QE and using non expanded queries. The term clarity score is used to classify queries in two groups: the ones which would benefit from QE and the ones that would not. They tested the method over the queries from TREC1 to TREC3 considered together, over TREC5, TREC6, TREC7 and TREC8. They also slightly improve the MAP compared to the baseline method that expands all queries. For instance, on TREC8 they improved the MAP from 0.2715 to 0.2812.

Other approaches try to optimize QE or other function implied in the search process for each query. Lv and Zhai [Lv 2009] suggest a learning method to predict the coefficient that balances the initial query and the feedback information in PRF. They use different features such as discrimination of a query (query length, query entropy and query clarity), discrimination of feedback documents (feedback length, entropy, clarity), and divergence between the query and feedback documents; logistic regression is used to learn the balance coefficient. The authors show that the three components are complementary and capture different aspects of information. Moreover, the authors argue that the adaptive coefficient is more accurate than a fixed coefficient when training and testing sets are not similar. The authors tested over TREC6, TREC7 and TREC8 collection and they reported only marginal MAP improvements (from 0.340 to 0.356 in the case of TREC7+8).

He and Ounis [He 2004] suggest a method to select among several term-weighting models depending on the query. Queries are characterized by various features which are used to cluster them using agglomerative hierarchical clustering. Training associates the best term-weighting schema with each query cluster. After training, a new query is first clustered into the existing clusters and the pre-trained system is used to process it. This is a pre-retrieval method since the query characteristics are calculated from the query itself and from general characteristics of the document collection (query length, term idf, and clarity/ambiguity of the query). When evaluated on Robust TREC, the method slightly improves MAP when using QE based models if enough training queries are used.

Chen and al. [Chen 2012] also consider various features to predict performance and apply selective QE. The authors use query features such as query length, query coherence, query entropy, as well as other features such as blind RF features (probabilities associated with expanded terms) and explicit RF features (feedback document length, entropy, rank, ...). The predictive model is based on logistic regression that aims at predicting if blind RF will be better than combining explicit and blind RF. The choise of the QE method is based on this prediction. Using ClueWeb09 (Category B), the precision of the prediction model is from 53% up to 65%. The method improves MAP compared to any of the methods where a single QE method is used.

Cao et al. [Cao 2008] propose a supervised method to decide whether expansion term candidates are good or not according to their impact on retrieval. The process is viewed as a term classification problem which is solved using SVM. We also employ SVM in our approach, however the decision is different in our case, since we choose between two QE methods and not among expansion terms.

In the method we propose, query difficulty predictors are used to cluster queries and associate a QE model with the query clusters. Query difficulty predictors have been used in the past to predict query difficulty, but no concrete application of the prediction have been proposed apart from reducing long queries [Kumaran 2009]. The authors have developed a learning to rank method that uses query quality measures as features for all the sub-sets of an original query. The original long query is replaced by the sub-query chosen by the ranker.

Like in [He 2004] or [Chen 2012], we consider query features to make the query-dependent decision on which function to use; however our predictors include some linguistics-based predictors that have been shown to be correlated to query difficulty [Mothe 2007]. Similar features and also the SVM learning mechanism have been employed in [Lee 2014]. However, their application differs from ours since their decision aims at choosing between inter-language and intra-language PRF in the context of cross-language IR. In our method, we consider a selective function that chose among various QE methods. More specifically, we consider RF and query refinement. These two methods are user oriented.

When a user study is not possible because of missing resources and for reproducibility reasons, it is possible to simulate them [Lin 2008], [Zhao 2012]. We thus simulate these two types of QE. With regard to RF we use blind RF that considers the first x retrieved documents as relevant. In addition, we simulate query refinement by considering the descriptive part of the topics.

Zhao and Callan [Zhao 2012] use diagnosis of term mismatch in order to suggest manual query reformulation or to guide other types of query expansion. Indeed, many various query reformulation techniques have been developed in the literature [Fox 1994]. Query reformulation can be manual or automatic, such as blind relevance feedback [Cronen-Townsend 2002b] for example. However, evaluating manually reformulated query face reproducibility; for this reason, it can be simulated. Human behavior simulation has been used in the past in IR [Zhao 2012], [Lin 2008].

In the following section, we present our selective QE method based on learning with features based on query difficulty predictors.

4.3 SVM-based learning using query difficulty predictors

In this section we present the proposed method and we develop on the query features, by difficulty predictor types. Next, we describe the criterion for query classification and the learning mechanism that we employ. We start by the method overview.

4.3.1 Method overview

Our method has as input TREC topics, from which we construct 4 query variants: the initial query (title part of topic), the humanly refined query (title+description together), the automatically expanded query (title expanded by PRF) and the fully expanded query (title+description, all expanded by PRF). Having the query variants, we compute the difficulty predictor values. From predictor results we generate the feature data set. For each topic, we assign a class corresponding to the best type of expansion. We then split the by the 10-fold cross-validation principle. We use cross-validation to optimize the parameters of the learning method. The model is then learned on the training data set folds and then tested. We finally obtain the predicted class for each topic of the test data set.

We summarize the method in Algorithm 4.1.

Algorithm 4.1 Selective QE based on SVM with query difficulty prediction features

```
Input: TREC topics, document collection
 1: Generate query variants from topics[1]
 2: Compute predictor values corresponding to generated queries.
 3: Build the feature data set, based on predictor values and combinations.
 4: Assign a class to each topic, corresponding to the AP of best expansion type.
 5: Separate the data set in train/test (10 folds).
 6: Optimize learning parameters by cross-validation.
 7: Learn the model on train data.
 8: Apply model on test data.
Output : Class value per topic, for test data.
```

Queries are characterized by features that are computed using the query text and the retrieved document list; these features are derived from 4 query difficulty predictors from the literature, which are presented in Section 3.3. They are effective predictors of heterogeneous nature.

[1]For details regarding the query variants, see Section 4.4.1.

4.3.2 Detailed query features

From these four core predictors, we have calculated a total of 31 features by adjusting their parameters or by combining them.

Term ambiguity is represented by 6 features derived from the WNS predictor: the maximum, average and sum for the number of senses, over query terms. The features are calculated both for the initial query and the refined query.

Term discrimination is represented by features derived from the IDF predictor: the minimum, maximum, average, and sum over query terms. IDFs are calculated both for the initial query and the refined query.

Document list homogeneity: query features are derived from the STD predictor. We consider the STD value for the list of documents retrieved from the initial query and the refined query and their STD values difference.

Divergence between lists: query features are derived from the QF predictor considering the various initial query and QE list combinaison and for 4 different cutoffs (5, 10, 50, 100).

Pre/post-retrieval combinations provide 2 features: the product between the WNS of the initial query and the STD difference as well as the IDF multiplied by STD difference.

4.3.3 Query classification

The query classification aims at taking the decision between the fully expanded (refined query + RF) and the RF expanded query variants. In the data set, a query belongs to the class 0 if the Average Precision (AP) of the RF expanded query is greater than the AP of the fully expanded query, otherwise the query belongs to class 1. Learning is based on SVM, which is an adjustable learning method, well fitted for binary classification.

The first stage uses cross-validation in order to automatically obtain the optimal parameter values. After having the necessary parameters tuned, the learning algorithm creates the model. A test query is associated with one class according to its feature similarity to training queries and then the learned corresponding QE model is applied. The method is evaluated in terms of MAP and robustness. We use cross validation. For each collection, we consider 90% of the queries for training and 10% for testing. We use the 10-folds cross-validation principle and we average the performance over all folds.

4.4 Evaluation framework

4.4.1 Test collections and retrieval parameters

We evaluate our method on three collections from TREC (Text REtrieval Conference)[2] ad hoc tasks: Robust, WT10G and GOV2. For details on the test collections, see Section 1.4.3.

We employ the 250 topics of Robust, the 100 topics of WT10G and the 150 topics of GOV2. Regarding the query formulation, we have chosen the title part of the topic to stand for the initial short query and, by adding up the description part we simulate the *humanly generated* expanded query. We do not use the narrative part of a topic in our evaluation framework.

Our selective approach implements two QE methods, while the search engine remains unchanged. The used search engine represents a key factor for several stages of our method and its retrieval parameters have a crucial influence over the outcome of the selective IR approach. For instance, the post-retrieval predictors obtain their values through computations made over the list of retrieved documents returned by a search engine. Moreover, the runs for performance evaluations are also produced through a search engine, hence the importance of the search engine's choice. The present study is based on runs constructed by Indri.The retrieved document lists consist of the first 1000 ranked documents returned by Indri.

The collections were indexed, considering stopwords removal and using the Krovetz stemmer. For the basic retrieval we used the query likelihood model (QL) with a Dirichlet smoothing ($\mu = 1000$). To expand the queries we used the Relevance Model 3 (RM3) [Lavrenko 2001], which represents an interpolation between the blind relevance feedback (RM1) and the initial query. Choosing the parameter for this interpolation represents an open problem. We have set this parameter to 0.5 assuming an equal importance of the initial query and the automatic expansion. Moreover, fixing this parameter value implies a robust baseline (Robustness Index greater than 0.56). The number of documents for feedback is set to 100 and the number of terms for feedback is set as well to 100.

Four query variants contribute to the method to provide learning features:

1. the *initial short query* (denoted **T**) is generated from the title part of the TREC topic after the suppression of stopwords and a stemming process;

2. the humanly *refined query* (denoted **TD**) is simulated by considering the descriptive part of topics and consists of the title along with the descriptive of TREC topic, having stopwords removed and the terms stemmed;

3. the *automatically expanded* initial query using blind RF and denoted **TRF**, which is T expanded by the search engine expansion mode, and

[2]http://mitpress.mit.edu/books/trec

4. the *fully expanded* query that combines the two last QE denoted **TDRF**: TD expanded by the search engine expansion model.

We have tried several configurations of Terrier's parameters. Corresponding to each collection, we have chosen the settings which produced the highest MAP. These MAP values are consistent with the literature [He 2004], [Zhong 2012] and they also represent baselines to evaluate the efficiency of our selective approach.

Table 4.1 provides the MAP for each individual runs.

Table 4.1: The baseline MAP values, for each collection

Collection	MAP T	MAP TD	MAP TRF	MAP TDRF
Robust	0.233	0.260	0.260	0.296
WT10G	0.194	0.215	0.220	0.244
GOV2	0.292	0.294	0.332	0.329

4.4.2 SVM configuration

As mentioned in Section 4.3.2, the feature matrix for the SVM model contains the values of features based on query difficulty predictors variants and combinations. The data set contains 31 features in total. From the retained predictors (WNS, IDF, STD and QF), we have derived different variants and combinations. See Section 3.3 for details on the retained predictors. Our full feature list is presented in Table 4.2.

Table 4.2: The learning feature list based on query difficuly predictors

WNS	IDF	STD
max WNS for T average WNS for T sum WNS for T max WNS for TD average WNS for TD sum WNS for TD,	max IDF for T min IDF for T average IDF for T sum IDF for T max IDF for TD min IDF for TD average IDF for TD sum IDF for TD	STD for T STD for TD (STD T)-(STD TD)

QF	Combinations
QF for T and TD QF for T and TRF QF for TD and TDRF (each QF with cutoffs: 5, 10, 50, 100)	(average WNS for T)*(difference STD) (average IDF for T)*(difference STD) WNS-c, with c$\in \{1, 1.5, 2, 2.5\}$

For the STD, there were considered as well the STD for the T run and the STD of the TD run, each of them as a standalone feature.

The QF predictor has multiple variants generated by the existence of two parameters. The cut-off level for the result lists represents the number of documents from the top result lists that are considered for the overlapping. We have chosen a various number of cut-off levels between 5 and 200. The second parameter is the choice of which results lists are overlapped, for example the results from the T run with the results from the TRF run.

Regarding the predictor combinations, we have proposed $IDF{*}STD\_diff$ (the difference between the STD of the T run and the STD of the TD run), $WNS * STD\_diff$ and $WNS{-}c$, where c represents a threshold for the permitted amiguity level and $c \in \{1, 1.5, 2, 2.5\}$.

When creating the feature matrix on the training set, the class per query was established with respect to the AP value for the TRF and TDRF expanded versions: 0 if the AP of the TRF version is greater than the AP of the TDRF version and 1, otherwise.

Regarding the SVM kernels [Aizerman 1964], we have tried several possibilities, such as Gaussian (Radial basis function), sigmoid and polynomial (up to third degree). The usage of a non linear kernel allows the algorithm to fit the maximum-margin hyperplane in a transformed feature space. The mathematical equations for the various kernels k are listed in Table 4.3.

Table 4.3: The mathematical expression for several SVM kernels

SVM kernel	Formula
Gaussian	$k(x, x') = e^{-\gamma \|\|x-x'\|\|^2}$
Polynomial	$k(x, x') = (\gamma x^T x)^d$
Sigmoid	$k(x, x') = tanh(\gamma x^T x)$

The representation of the query in the feature space is denoted by x, d represents the polynomial degree, and γ is an adjustable parameter. Another adjustable parameter is the *cost*, which represents the cost of constraint violation. For the values of *cost* and γ parameters, we have used an automatic sampling method of 10-fold cross validation. The best values obtained are the following: $\gamma = 0.015625$ and $cost = 4$. These values remain stable for all the different learning trials.

4.4.3 Evaluation method

For each collection, we use 10 fold cross validation to check the accuracy of the learnt model. We use 90% of the queries for training and the rest for tests. The accuracy measure represents the average accuracy over the 10 testing folds.

Regarding the decision, for the testing queries, with respect to the classification established by the SVM, one type of QE (TRF or TDRF) is assigned for each query. The classified query group is submitted to the search engine. The retrieval results (retrieved document lists) represent new evaluation runs and these new runs are evaluated, by computing the MAP. Then, the results are compared to baselines, in terms of MAP, in order to measure the improvements.

QE has always involved a matter of robustness. In average, the expansion improves the performances, only that the performance of some queries might decrease in terms of AP. We introduced the Robustness Index (RI) [Sakai 2005] in Section 1.3. We recall its definition here:

$$RI(q) = \frac{n_+ - n_-}{|q|}, \tag{4.1}$$

where q is the set of queries over the RI has to be calculated, n_+ is the number of improved queries, n_- the number of degraded queries and $|q|$ the total number of queries. For our method, we have computed the RI and we have compared this robustness measures with the RI of runs without the selective premise, in order to obtain the improvement percentage.

The obtained results and comments are presented in the following subsection.

4.5 Results and discussion

In this section we present the performance of our method in terms of SVM accuracy, MAP improvements and robustness, along with result comments, arguments and observations.

As Guyon and Elisseeff stated in [Guyon 2003], redundant data could be helpful for obtaining a more accurate learning model. We have tested this hypothesis and, indeed, a dataset with a smaller number of features has performed worse.

Regarding the SVM kernels, we have tested several kernel types as presented in section 4.4.2. The Gaussian kernel has performed clearly and constantly better than all the other kernels. For instance, the failure of the linear kernel suggests that there is no linear separation with respect to the two query classes for the query representations in the feature space. We present here the results obtained by using the Gaussian kernel, since it yielded the best learning performance.

Table 4.4 presents the MAP improvements, with respect to non selective retrieval, for test queries in the 3 collections, the p-value for the statistically significance test (t-test) and the model accuracy are also displayed.

TABLE 4.4: The improvements in terms of MAP (*: p-value< 0.05; **: p-value< 0.001)

	Robust	**WT10G**	**GOV2**
Baseline TRF	0.265	0.234	0.331
Baseline TDRF	0.305	0.264	0.334
Result with decision	**0.317**	**0.296**	**0.372**
Improvement TRF	**19.41%****	**24.65%****	**12.59%****
Improvement TDRF	**3.90%****	**12.27%****	**11.35%****
SVM accuracy	97.20%	100%	100%
Collins-Thompson: Baseline Expansion	0.244	0.183	0.291
Collins-Thompson: Result (Robust Feedback)	0.245	0.199	0.300
Collins-Thompson: Improvement	0.40%	8.73%	3.09%

The improvement range is between 3.90% and 24.65% for the presented test examples and all the results are statistically significant. He and Ounis [He 2007] report MAP improvements up to 10.83% for WT10G and up to 7.40% for GOV2, using their combining fields method for adaptive query expansion. The selective method proposed by Lv and Zhai [Lv 2009] improve MAP with about 4%. Collins-Thompson [Collins-Thompson 2009] formulates query expansion as a robust optimization problem and his method aims to reduce the risk-reward trade-off of expansion. His baselines and obtained results are presented in 4.4. The highest MAP improvement with respect to the baseline RF expansion is of 8.73%, for WT10G. It is worth mentioning that his RF baseline was lower.

Selective QE is beneficial in average, however some queries might lose performance, in terms of AP. For this reason, a robustness study should be involved. Therefore, we have computed the robustness improvement in terms of RI. We have compared our QE decision to the constant decision for TRF expansion and to the constant decision for TDRF expansion, respectively. The results are displayed in Table 4.5 together with the RI values obtained by Collins-Thompson [Collins-Thompson 2009].

Table 4.5: The robustness analysis (*: p-value< 0.05; **: p-value< 0.001)

	Robust	**WT10G**	**GOV2**
RI(TRF)	0.384	0.200	0.413
RI(TDRF)	0.576	0.360	0.187
RI(SVM)	**0.752**	**0.620**	**0.653**
Collins-Thompson: RI	0.377	0.270	0.262
Improv. RI(TRF)	**95.83%****	**210%****	**58.07%****
Improv. RI(TDRF)	**30.56%****	**72.22%****	**250%****

The baselines for the RI are consistent with the literature [Collins-Thompson 2007] and the RI improvement starts from 30% and goes up to 250%, this means that the selective IR method is also helpful in terms of robustness.

4.6 Conclusion

We have proposed a selective IR method that utilizes query difficulty predictors in order to learn how to select between several types of query expansion. Our method takes advantage of a large range of query features, including linguistics-based ones. The method focuses on the query, while other selective IR methods from the literature deal with the collection documents, or with the search engine types.

The method is effective in terms of learning accuracy, MAP and RI. We have obtained a learning accuracy constantly over 80%, MAP improvements up to 24.65% in average and robustness improvements that reach 250%, results that are statistically significant.

The query difficulty prediction research obtains more and more efficient and accurate measures, although these predictors are not enough exploited and utilized in IR systems. We consider that the selective IR represents a realistic and fruitful framework for query difficulty prediction applications.

Regarding future works, we are interested in building up new difficulty predictors along with new possible predictor combinations. The SVM has represented a choice for the learning mechanism and we aim to test other learning techniques such as decision trees. Also the pre/post retrieval similarity analysis opens another lead for query-focused selective IR.

Chapter 5

Failure analysis: parameter optimization for a pseudo-relevance feedback model

Contents

Query expansion improvements are usually reported on average, over multiple queries, meaning that some queries may actually be harmed (see Section 1.3). For this reason, it is useful to find out how to expand each query. In this context, in Chapter 4 we proposed a per query method to choose between two expansion types, by learning based on difficulty predictors.

In this chapter we remain in the field of query expansion and we present our trial to optimize the interpolation parameter (lambda) in the case of RM3 query expansion model (see Section 1.3), on a per query basis. This parameter varies the importance of the initial query in the expanded model. It is a difficult problem

[Lv 2009] and we believe that it might not be solvable without prior information. We test several hypothesis, both with and without prior information, and we also try to establish the minimum amount of necessary information in order to be able to optimize the expansion parameter. We employ linear regression, SVM, pseudo-qrels, Jensen-Shannon divergence and logistic regression to test our hypothesis. Some improvements with respect to baselines are noticeable. However, these improvements are not important enough to declare the problem as solved.

This research has been conducted not only during a three months research mobility at Technion Institute from Haifa, Israel, in 2013, but also afterwards, by keeping a close contact with the team from Technion. In Haifa we have worked with Prof. Oren Kurland and with PhD. candidate Anna Shtok.

5.1 Introduction

Nowadays, the efficiency of QE based on pseudo-relevance feedback (PRF) is widely acknowledged by the IR community. Following this principle, the initial query is submitted to the search engine yielding the retrieved document list. Next, by analyzing this list, expansion terms are found and then used to expand the initial query. The expanded query is finally submitted to the search engine in order to obtain a new, supposedly improved, retrieved document list.

This type of approach, which basically dates back since about 20 years ago [Buckley 1995], extends the idea of query reformulation based on explicit relevance judgements from users [Rocchio 1971, Harman 1992]. Various studies have underlined the benefits brought by PRF. However, He and Ounis have shown that there is no significant difference in terms of performance between the results obtained by injecting relevance through all the first 10 initially retrieved documents and only the truly relevant documents among these first 10 retrieved documents [He 2009]. In the case of language models, which also is our interest here, [Lavrenko 2001] have reported improvements between 2 and 5% in terms of precision when 10 initially retrieved documents are considered as relevant, over TREC collections. On the other hand, [Deveaud 2013] propose the employment of external resources in order to obtain the expansion terms. Based on language model, this approach interrogates Wikipedia or the web and the first results are considered as expansion documents. He shows that this method improves results, moreover when multiple external sources are combined.

It is very important to mention that the expansion improvements are global, meaning that the query expansion is beneficial on average, over all the queries from a dataset. Thus, several queries are improved in this way, while other queries are harmed in terms of performance [Harman 2009], [Bigot 2011]. Therefore, various researchers are interested in adapted query reformulation. We also proposed a contribution for selective QE, based on learning with features based on query difficulty predictors (see Chapter 4). This selective way of expanding queries consists, thus, in deciding how, or whether to expand the initial query. The decision lies generally on features extracted from the queries themselves. In [Amati 2004a], the authors combine various query features, such as query length, the terms *idf*, aiming to decide if the expansion should be applied. They showed that the precision could be improved this way and also that few queries did not have any relevant documents among the 10 top retrieved documents.

In this query expansion optimization context, we focus our research on optimizing a parameter from a language model based query expansion, on a per query basis. The purpose is to improve the AP value for each query, this implying a more robust retrieval system (see the robustness index from Section 1.3). We believe that this parameter should be optimized for each query and not for an entire system, that is over all queries. The Query Likelihood (QL) is employed for retrieval in the case of the initial query, while Relevance Model 1 (RM1) considers the query formed by the expansion terms. The model we try to optimize is called Relevance Model 3 (RM3) and represents an interpolation between QL and RM1. All these models are described in Section 1.2.4. We only restate here few details that will be useful in this chapter.

The IR language models assume that each document is generated from a different model [Ponte 1998].

A relevance model could be defined as a model which is representative for relevant documents. In [Lavrenko 2001], the authors define a *relevance model as a model that determines the probability of observing a word in the documents that are relevant to a query.*

We are trying to optimize the Relevance Model 3 (RM3) [Abdul-Jaleel 2004], which is the Relevance Model 1 (RM1) [Lavrenko 2001] interpolated with the model of the initial query, in order to obtain performance improvements. Thus, the interpolated relevance model RM3 is computed as follows:

$$P_RM3\left(t|\theta'_q\right) = (1-\lambda)\,P\left(t|\theta_q\right) + \lambda P_RM1\left(t|q\right) \tag{5.1}$$

The central figure of our analysis is the lambda parameter from RM3, parameter which we try to optimize for each query in order to obtain a better AP performance. This interpolation parameter can have real values, between 0 and 1. When the value is set to 0, this means that the RM1 has all the importance. Contrarily, when the value is set to 1, QL has all the importance.

This chapter is structured as follows: in Section 5.2 we present the research related work. The evaluation framework is described in Section 5.3. The optimization hypothesis without prior information are presented and tested in Section 5.4, while the hypothesis with prior information are presented and tested in Section 5.5. In Section 5.6 we share some additional insights regarding the optimization problem. Section 5.7 concludes this chapter.

5.2 Related work

The robustness issue that query expansion is raising when degrading some queries, while improving others has driven IR researchers to find ways to learn how and if to expand the queries. Most of the approaches for optimizing query expansion performance are presented in Section 4.2. We revisit here only the approaches that try to optimize the query expansion based on pseudo-relevance feedback, in the case of relevance models. Only one approach is close to our propositions [Lv 2009].

From the selective query expansion perspective, in [Cronen-Townsend 2004b], the authors proposed a framework with three methods. The idea is to use the expanded queries only when they are straying from the initial ones, therefore it represents a selective query expansion approach in the sense of choosing whether to expand a query or not. The framework consisted in three methods to make the expansion decision. The first is the clarity score [Cronen-Townsend 2002b] and the hypothesis is that if the clarity score is high, then expansion would be unwise to make. The second approach is based on the overlap between the unexpanded and expanded ranked list, as in the case of QF difficulty predictor [Zhou 2007]. A high value of the overlap score would suggest a beneficial expansion. Finally, the third and the best method is the model comparison. The main idea is to compare the language models of the unexpanded and expanded retrieval results. From this comparison they try to predict when the expanded query drifted from the original query sense. The authors employed the language model retrieval and expansion techniques implemented in Indri over the AP88-90, TREC123, TREC5, TREC6, TREC7, TREC8 and Query Track Aggregate test collections. Some small Mean Average Precision (MAP) improvements were reported, only for TREC5 and TREC6 collections, respectively (from 0.1609 to 0.1644 and from 0.2013 to 0.2115).

Collins-Thomson and Callan [Collins-Thompson 2007] aim to estimate and to use the uncertainty in the pseudo-relevance feedback. They resample the top retrieved documents of a query in order to estimate a posterior feedback model distribution. Their meta-feedback algorithm uses query variants, with the assumption that various aspects of the original query might be important. These query variants consist in: using the original query, leaving one term out, or keeping a single term from the query. We also employ query variants to compute features based on query difficulty predictors. These features are required for the application of several hypothesis stated in Section 5.4. Finally, the enhanced feedback models from multiple query variants are combined. Indri was used as a search engine and Indri query language was employed to generate query variants. The authors conducted experiments over the TREC1&2, TREC7, TREC8 and WT10G collections and reported improvements in terms of MAP and RI. For instance, in the case of the TREC8 collection the MAP values go from 0.1829 (baseline represented by feedback without re-sampling) to 0.1946.

Seeing the query expansion as a risk-reward optimization, Collins-Thompson

[Collins-Thompson 2009] proposed a robust constrained optimization approach. This selective query expansion method models the expansion benefit and risk, by optimizing over uncertainty sets from data. The authors tested their method over several TREC collections, such as TREC1&2, TREC7, TREC8, WT10G, Robust and GOV2. The evaluation measures were MAP, precision at a cutoff of 20 (P@20) and robustness index (RI). We mention that the search engine, as in our study, was Indri and the query expansion model was RM3. Their baselines represent the retrieval results without query expansion and the retrieval results with query expansion for all queries. The reported improvements are quite close to the baseline with query expansion, both in terms of MAP and RI. The highest MAP improvement was for WT10G, from 0.1830 to 0.1990, while the highest RI improvement was from -0.0270 to 0.2703.

The approach which is the closest to our objective is presented in [Lv 2009]. The authors proposed a method to adapt the relevance feedback. The aim is to optimize the balance parameter (the lambda interpolation parameter from RM3) for each query and for each set of feedback documents. By the means of logistic regression with various models with their corresponding features the adaptive mechanism is tested over TREC6, TREC7 and TREC8 collections. Query features are based on the query length, on query entropy and on query clarity, while feedback documents features are based on feedback length, feedback radius, on entropy of feedback documents and on clarity of feedback documents. Also the divergence between query and feedback documents is taken into account through the absolute and relative divergence, respectively. However, the MAP improvements are statistically significant only for the training set of TREC6 and TREC7 together (from 0.340 to 0.356) and for TREC6, TREC7 and TREC8 all together (from 0.340 to 0.354), respectively.

For the logistic regression, the authors employed features without any prior information about the retrieved lists quality for the logistic regression. Since the improvements are not satisfying enough, we tried to check the necessary amount of information required for a better parameter prediction.

In the following section we introduce the evaluation framework established to test the optimization hypothesis.

5.3 Evaluation framework

In this section we establish the evaluation framework, thus we present the benchmark collections employed for tests, we define the baselines and also we mention the parameter setup for the retrieval system we used in our experiments.

5.3.1 Data collections

The main data sets are TREC Robust, WT10G and GOV2 (see Section 1.4.3 for details). However, for initial, small scale experiments, we used in a complementary way the TREC123 and TREC5 collections.

TREC Robust, WT10G and GOV2 are described in Section 1.4.3. On the other hand, TREC123 is a benchmark which consists in 100 topics addressed to Disks 1, 2 and 3 from TREC ad hoc. TREC5 ad hoc collection has 50 topics (251 - 300) and documents from Disks 2 and 4 of TREC disks.

5.3.2 Baselines

In order to check the performance of our results, we established two baseline configurations. The first baseline is a strong one and represents the AP per query, obtained with the best lambda for that particular query. This represent the optimal system and we try to reach as close as possible to its performance. The second baseline is weaker and represents the AP values per query, obtained this time with the lambda value that yields the best results on average, across all the queries. This baseline that fixes a best lambda across all the queries should normally be surpassed by the results when predicting the lambda parameter on a per query basis. We also compare the results to performances of QL and RM1, respectively.

5.3.3 Retrieval parameters

The search engine used for experiments in this chapter is Indri (see Section 1.4.6.2).

Firstly, the collections have been indexed by Indri. The stopwords have been removed and we used the Krovetz stemmer to stem the terms.

Having the indexes, we submit the queries to the search engine in order for it to retrieve the documents supposedly relevant. The queries consist in the title part of the TREC topics. We considered 11 runs per collection: the RM1 (the equivalent of RM3 with $\lambda = 1$), the QL (the equivalent of RM3 with $\lambda = 0$) and 9 more runs corresponding to lambda values from 0.1 to 0.9, with a step of 0.1.

For the basic QL retrieval we opted for the Dirichlet smoothing with $\mu = 1000$. The top retrieved documents by QL are used as feedback documents for RM1 and RM3.

For the RM1 and RM3 models we have tested various parameter combinations regarding the number of feedback documents and feedback terms and since the values were collection dependant, we decided to keep 100 feedback documents and 100 feedback terms. The smoothing method stands in place, that is Dirichlet smoothing with $\mu = 1000$.

In the following sections we present the optimization proposals regarding the lambda parameter, in the case of query expansion by the means of RM3. Firstly, we try to predict the lambda value without any prior information regarding the quality of the initially retrieved list QL and of the RM1 retrieved list, which are interpolated in order to obtain RM3. Secondly, we try to find the minimum amount of necessary information to predict the lambda parameter, since the hypothesis without prior information seem to fail.

5.4 Optimization hypothesis without prior information

As hypothesis without prior information on the quality of the retrieved lists, we firstly employ regression and SVM based on query difficulty prediction features and then we use pseudo-qrels in order to estimate the quality of various lambda configurations aiming to decide which one is better.

5.4.1 Linear regression and SVM with query difficulty prediction features

We underlined the capabilities of query difficulty prediction features for selective IR, in Chapter 4. Thus, we also test two learning techniques with the purpose of predicting the quality of retrieved document lists. These two techniques are linear regression and SVM, respectively.

The learning hypothesis was tested using various feature configurations. The features are based on query difficulty predictors, as in Chapter 4. We consider both pre-retrieval and post-retrieval predictors in our feature models.

Regarding the pre-retrieval predictors we employed the following predictors: the maximum of collection query similarity (MaxSCQTFIDF) and maximum term weight variability (MaxVARTFIDF), both derived from [Zhao 2008], together with the maximum IDF value.

For the post post-retrieval predictors we considered the following: WIG from [Zhou 2007], we computed the NQC from [Shtok 2009] by normalizing the STD predictor with respect to the entire collection score, the QF predictor and the geometric mean, the minimum value, the maximum value, the entropy and the average of the retrieved documents scores. In addition, for some experiments we also took into account a mutation of QF, which computes the overlap of multiple retrieved document lists. However, the difference in terms of results after adding these feature was not significant. Tests showed that even though the multiple lists QF is the best feature for some collection, the final results are not significantly improved.

We also try different feature combinations and configurations, such as separating pre-retrieval from post-retrieval predictors, different normalization applied over several features, feature selection, or different feature variations stemming from the same predictor (mean, min, max, etc.).

The first investigations, realized both with regression and with SVM, regard two retrieved document lists QL and RM3, with a fixed $\lambda = 0.5$. The regression method should predict the AP of the best query variant (QL or RM3), while the SVM binary classification should choose the best query variant itself, also between QL and RM3.

The actual lambda parameter optimization is done in other experiments, by multiclass SVM. There are two class separation hypothesis, one with three classes corresponding to lambda values in intervals (0.1-0.3, 0.3-0.6 and 0.7-0.9) and one with 9 classes, each class corresponding to a lambda value from 0.1 to 0.9, incremented with a step of 0.1. The value of $\lambda = 0$ and $\lambda = 1$ correspond to RM1 and QL, respectively.

For the result presentation, we mention the correlation between the AP predicted by regression and the true value of the best AP per query. We also present the SVM accuracy over the test dataset and the train dataset, as well as the best features for every considered scenario.

As a preliminary analysis, we check the predictor correlations between the AP of QL, RM1 and RM3 (with fixed $\lambda = 0.5$) and the predictor values, for Robust and WT10G collections. Results for the Pearson's correlation coefficient are displayed in Table 5.1.

We notice that there are predictors which are very weakly correlated with AP of no matter which list, such as WIG, with values starting from 0.03 in absolute value. On the other hand, NQC is highly correlated, having Pearson's coefficient up to 0.58. Variation in terms of correlation are to be noticed when checking across collections. In this case, for WIG, we can obtain correlations of -0.18 for Robust, while for WT10G the correlation coefficient has the value of 0.08. Moreover, several predictors yield a negative correlation, for example GeoMeanLog.

We firstly discuss the proposed regression method which tries to predict the best AP value per query. The choice is between the AP of QL and RM3 with fixed $\lambda =$ 0.5, respectively. This is inspired by the selective query expansion (see Chapter 4) and we test this three choices in order to have some first insights regarding the discriminant power of the features, reflected in learning results. However, before obtaining the prediction results we should analyze the features and their impact on the model.

Having many features of heterogeneous nature may raise the question of which of these features are the best in terms of their importance in the prediction model. Therefore, we look at the regression weights of our features and we highlight. We tried three regression models per collection, each corresponding to predicting the AP of QL, RM1 and RM3, respectively. The highlights are shown in Table 5.2. We highlight and we write in bold the ten best features per prediction model, in absolute value, since a high negative value is also discriminant therefore important for the model. We notice that while some predictors never manage to be in the top 10 features (WIG), or they are in top only for one model (Min), other features are

Table 5.1: Individual predictor correlation with AP, for Robust and WT10G

Predictors	**Robust**			**WT10G**		
	QL	**RM1**	**RM3**	**QL**	**RM1**	**RM3**
wigQL/WIG10	0.53	0.36	0.43	0.37	0.18	0.22
wigQL/WIG50	0.41	0.27	0.32	0.27	0.12	0.14
wigQL/WIG100	0.34	0.21	0.26	0.21	0.08	0.09
wigRM1_exp/WIG10	-0.18	-0.12	-0.16	-0.04	0.03	0.08
wigRM1_exp/WIG50	-0.33	-0.26	-0.30	0.06	0.13	0.18
wigRM1_exp/WIG100	-0.37	-0.30	-0.34	0.10	0.16	0.21
wigRM3/WIG10	0.14	0.05	0.11	0.21	0.10	0.06
wigRM3/WIG50	0.33	0.22	0.30	-0.09	0.00	0.05
wigRM3/WIG100	0.38	0.28	0.35	-0.03	0.05	0.11
QL_RM1/QF5	0.26	0.26	0.24	0.31	0.23	0.18
QL_RM1/QF10	0.32	0.36	0.31	0.37	0.31	0.27
QL_RM1/QF50	0.26	0.29	0.22	0.50	0.42	0.37
QL_RM3/QF5	0.28	0.24	0.24	0.34	0.18	0.18
QL_RM3/QF10	0.29	0.26	0.25	0.30	0.14	0.14
QL_RM3/QF50	0.24	0.24	0.19	0.40	0.30	0.27
RM1_RM3/QF5	0.04	0.16	0.08	0.14	0.18	0.10
RM1_RM3/QF10	0.14	0.21	0.16	0.31	0.35	0.30
RM1_RM3/QF100	0.16	0.19	0.15	0.34	0.30	0.23
QL/NQC50	0.53	0.42	0.48	0.43	0.26	0.30
QL/NQC100	0.54	0.44	0.49	0.49	0.33	0.36
QL/NQC500	0.39	0.32	0.32	0.36	0.22	0.23
RM1/NQC50	0.47	0.45	0.46	0.30	0.29	0.30
RM1/NQC100	0.48	0.49	0.48	0.32	0.28	0.29
RM1/NQC500	0.37	0.39	0.36	0.29	0.19	0.20
RM3/NQC50	0.55	0.51	0.54	0.27	0.20	0.23
RM3/NQC100	0.58	0.57	0.58	0.36	0.29	0.31
RM3/NQC500	0.43	0.44	0.41	0.26	0.14	0.16
QL_RM1_RM3/QF10	0.32	0.36	0.31	0.37	0.31	0.27
QL_RM1_RM3/QF50	0.26	0.29	0.22	0.50	0.42	0.37
QL_RM1_RM3/QF100	0.18	0.21	0.14	0.42	0.33	0.29
pre/MaxIDF	0.45	0.28	0.36	0.22	0.11	0.16
pre/MaxSCQTFIDF	0.36	0.31	0.29	0.44	0.30	0.38
pre/MaxVarTFIDF	0.44	0.28	0.33	0.46	0.31	0.38
q_rep/Entropy	0.30	0.26	0.24	0.24	0.14	0.14
q_rep/GeoMean	0.40	0.35	0.33	0.30	0.19	0.17
q_rep/Max	0.36	0.29	0.28	0.22	0.11	0.10
q_rep/Min	0.35	0.30	0.30	0.29	0.19	0.17
q_rep/GeoMeanLog	-0.37	-0.35	-0.32	-0.33	-0.19	-0.16
q_rep/Mean	0.41	0.35	0.33	0.28	0.17	0.15

in top almost for every model and collection (QF100).

The train/test splits have been created by 10-fold cross-validation, and the presented results are the average results of all 10-fold results.

Regarding the feature selection, we selected at first the 6 best features, based on their weights in Table 5.2. In Table 5.3 we compare the regression results when considering 6 features and the results when considering all the features. The correlation is higher than all the predictors taken individually, but still not high enough

Table 5.2: Linear regression feature weights, for Robust and WT10G collections

Predictors	Robust			WT10G		
	QL AP	**RM1 AP**	**RM3 AP**	**QL AP**	**RM1 AP**	**RM3 AP**
wigQL/WIG10	6.34	-2.70	2.65	0.44	-0.06	0.37
wigQL/WIG50	1.95	-2.13	1.24	-0.07	-0.38	0.05
wigQL/WIG100	3.70	-1.93	0.77	-0.39	-0.74	-0.29
wigRM1_exp/WIG10	-0.67	-4.85	-1.48	1.34	0.89	1.39
wigRM1_exp/WIG50	0.22	-3.03	-2.22	**1.76**	**1.87**	**2.06**
wigRM1_exp/WIG100	-0.93	-4.66	-4.14	1.76	**1.93**	**2.16**
wigRM3/WIG10	-1.86	-6.19	-4.77	0.79	0.60	0.58
wigRM3/WIG50	3.17	-0.03	1.36	**2.25**	**2.46**	**2.49**
wigRM3/WIG100	4.25	2.07	3.69	**2.48**	**3.20**	**3.24**
QL_RM1/QF5	**-7.01**	-6.96	**-7.34**	1.14	0.40	0.48
QL_RM1/QF10	**8.41**	**10.37**	**8.82**	0.18	0.16	0.04
QL_RM1/QF50	5.13	**7.31**	3.86	**2.34**	**3.95**	**3.71**
QL_RM3/QF5	6.75	6.86	5.52	1.59	0.55	1.02
QL_RM3/QF10	**7.01**	3.02	4.58	0.33	-1.28	-1.33
QL_RM3/QF50	-3.34	-5.96	-2.60	0.89	0.81	0.41
RM1_RM3/QF5	-6.14	2.04	1.58	-1.17	-0.61	-0.65
RM1_RM3/QF10	1.32	2.44	3.10	0.67	1.43	1.51
RM1_RM3/QF100	5.04	**9.33**	**11.72**	**2.35**	**2.44**	**2.77**
QL/NQC50	**15.45**	**12.73**	**17.18**	**2.54**	1.37	1.75
QL/NQC100	0.65	2.07	5.98	**2.39**	**1.97**	**2.35**
QL/NQC500	**-13.02**	-5.57	**-6.45**	-0.71	-0.32	-0.47
RM1/NQC50	1.13	0.57	3.42	0.19	1.04	0.95
RM1/NQC100	6.94	**8.36**	**9.76**	-0.48	0.21	0.40
RM1/NQC500	1.21	-2.88	-4.17	0.28	0.32	0.70
RM3/NQC50	**8.69**	**14.83**	**14.83**	-0.09	-0.03	0.01
RM3/NQC100	**12.86**	**20.20**	**20.87**	-0.10	0.68	0.78
RM3/NQC500	-1.54	3.37	-0.32	-0.02	-1.14	-1.48
QL_RM1_RM3/QF10	**7.75**	**10.38**	**8.83**	0.18	0.16	0.04
QL_RM1_RM3/QF50	5.14	**7.57**	3.97	**2.34**	**3.95**	**3.71**
QL_RM1_RM3/QF100	5.24	3.66	4.14	**2.65**	**2.97**	**2.92**
pre/MaxIDF	**16.40**	**11.06**	**12.45**	0.71	1.32	1.44
pre/MaxSCQTFIDF	1.04	5.59	3.24	1.25	1.17	1.69
pre/MaxVarTFIDF	**11.41**	-1.49	1.01	1.75	**2.36**	**2.46**
q_rep/Entropy	-5.99	1.78	-5.40	1.66	1.16	1.08
q_rep/GeoMean	3.58	6.93	3.90	-0.32	-1.03	-0.94
q_rep/Max	4.72	-4.15	-1.34	**-1.89**	-0.96	-0.72
q_rep/Min	1.78	0.28	0.40	1.19	1.22	1.09
q_rep/GeoMeanLog	-0.63	-4.68	-2.81	0.23	0.88	0.78
q_rep/Mean	4.34	-0.66	-0.15	-1.28	-1.23	-1.25

to be able to state an accurate and clear prediction. In the case of 6 features the test performance is closer to the train performance than when considering all the features. On the other hand, when considering all the features, the trained model is better, however the test performance slightly decreases. This suggest over-fitting.

For the SVM, we used `svmlight` [Joachims 1999] and we kept the same feature setup as for the regression. In Table 5.4 we present the SVM accuracy for test/train splits, over Robust and WT10G, while using 6 features and all features, respectively. We mention that the train accuracy represents only an indication for possible over-fitting and it does not represent a reliable estimator of the real accuracy rate. The accuracy is not high, classification results being close to chance, in some cases

Table 5.3: Correlation values in the case of regression, for different number of features, for train and test sets, over Robust and WT10G collections

# of features	**Average (standard deviation)**			
	Robust		**WT10G**	
	Train	**Test**	**Train**	**Test**
6 features	0.663 (0.017)	0.635 (0.150)	0.641 (0.023)	0.526 (0.281)
All features	0.671 (0.013)	0.593 (0.210)	0.632 (0.028)	0.587 (0.163)

havinèg worse performance than chance. For instance, the test splits accuracy for the WT10G collection is 45.429% with 6 features and 48.571% for all the features. However, by observing the created classes, one could notice that the classifier tends to assign all the queries to the first class. Close values between train and test also suggest that there is no over-fitting involved.

Table 5.4: SVM accuracy, for different number of features, for train and test sets, over Robust and WT10G collections

# of features	**Accuracy**			
	Robust		**WT10G**	
	Train	**Test**	**Train**	**Test**
6 features	64.259%	64.283%	54.462%	45.429%
All features	64.257%	64.250%	56.596%	48.571%

Empirically following an exploring trial/error experimental process, we also tried kernel methods for SVM, such as the Gaussian kernel. On the other hand, regarding the multiclass SVM we tried the following class separation: three classes corresponding to lambda values in intervals (0.1-0.3, 0.3-0.6 and 0.7-0.9) and 9 classes, each class corresponding to a lambda value from 0.1 to 0.9, incremented with a step of 0.1. All these results were in the same area, of 60% in terms of accuracy, which is not enough to state that efficient prediction is possible with for our optimization problem.

In the following section we present the pseudo-qrels hypothesis and we discuss the obtained results.

5.4.2 Pseudo-relevance information

For the pseudo-relevance information (denoted here pseudo-qrels) hypothesis, the following retrieved lists are known: QL, RM1 and various RM3 lists corresponding to different lambda parameter values. We would like to predict the quality of the RM3 lists, without having the true relevance judgements. Therefore, we simulate these relevance judgements by fusing all the retrieved lists, cutting at a certain cut-off k and considering these k top documents as relevant. Finally, with the pseudo-qrels, we can compute a pseudo AP, for each RM3 retrieved list. We then keep the lambda parameter for its corresponding list, with the best pseudo AP value.

We analyze the results when considering several cut-offs levels and fusion functions. We have tried all cutoffs from 1 to 500, plus the 1000 cutoff levels. This cutoff represents the number of documents from the top of each list which are considered as relevant. We show in Table 5.5 only the results for the cutoff levels of 10, 50, 100 and 1000 documents. Next to each MAP value in the table we specify between brackets the number of considered queries. For a low cutoff there are high chances that some queries in a collection do not have any retrieved documents in common to construct the pseudo-qrels. The best results are written in bold. We also display in Table 5.5 the Robustness Index (RI) for each setup, computed against the initial query run, QL.

Underlining the best results across different cutoffs, we notice that the cutoff which yields the most best results, both in terms of AP and RI, is 100. For all collections we are getting close to the baseline defined by the best fixed lambda on average. For instance, in the case of TREC5 the best obtained AP result is 0.167, while the baseline with average fixed lambda gives 0.169. The best possible results are still out of reach. However, one could notice high improvements in terms of RI compared to the same average baseline, such as from 0.427 to 0.693, over TREC123 collection.

TABLE 5.5: Predicting λ using pseudo-qrels, over Robust, WT10G, TREC123 and TREC5 collections ("cons.q" stands for "considered queries")

Cutoff	**Robust**		**WT10G**	
	MAP	**RI (cons.q.)**	**MAP**	**RI (cons.q.)**
10	0.288 (241)	0.237 (241)	0.219 (94)	0.106 (94)
50	0.283 (all)	0.325 (all)	**0.225 (all)**	0.134 (all)
100	**0.285 (all)**	**0.341 (all)**	**0.225 (all)**	**0.175 (all)**
1000	0.279 (all)	0.060 (all)	0.205 (all)	-0.196 (all)
		Baselines		
Best possible	0.324	x	0.254	x
Best average fixed	0.292	0.365 (all)	0.229	0.155 (all)
QL	0.249	x	0.198	x
RM1	0.267	0.012 (all)	0.193	-0.237 (all)

Cutoff	**TREC123**		**TREC5**	
	MAP	**RI(cons.q.)**	**MAP**	**RI(cons.q.)**
10	0.298 (145)	0.614 (145)	0.160 (all)	0.200 (all)
50	0.288 (all)	**0.693 (all)**	**0.167 (all)**	0.360 (all)
100	0.288 (all)	0.680 (all)	0.162 (all)	**0.400 (all)**
1000	**0.293 (all)**	0.413 (all)	0.159 (all)	0.160 (all)
		Baselines		
Best possible	0.317	x	0.189	x
Best average fixed	0.297	0.427 (all)	0.169	0.440 (all)
QL	0.229	x	0.149	x
RM1	0.287	0.347 (all)	0.154	0.080 (all)

In the following section we present and then test the hypothesis that require prior information.

5.5 Hypothesis with prior information

Due to the difficulty of predicting the value of the lambda parameter without any information about the quality of the retrieved list, we try to establish the minimum amount of necessary information in order to obtain a good prediction. This extra information could refer to which list yields a better AP, between QL and RM1, for example. This is only small amount of information, compared to the case when the AP values of QL and RM1 are actually known, for instance.

Using this information, we propose two ways of predicting the lambda parameter, one based on Jensen-Shannon divergence between lists, in terms of AP and the other by employing logistic regression over various features.

5.5.1 Jensen-Shannon divergence

The available information consists in the AP values, per query, of the QL and RM1 retrieved document lists. The starting point is to give QL and RM1 weights. The weight of QL is the following:

$$w\left(QL\right) = \frac{AP\left(QL\right)}{AP\left(QL\right) + AP\left(RM1\right)}. \tag{5.2}$$

The weight of RM1 is complementary and it is computed as follows:

$$w\left(RM1\right) = 1 - w\left(QL\right). \tag{5.3}$$

Afterwards, for each considered lambda parameter, we compute a similarity score. The score for a particular RM3 list, denoted $RM3_\lambda$, depends on the QL and RM1 lists and has the formula:

$$score\left(RM3_\lambda\right) = \frac{sim\left(RM3_\lambda, QL\right)}{sim\left(RM3_\lambda, QL\right) + sim\left(RM3_\lambda, RM1\right)}. \tag{5.4}$$

The similarity function sim, chosen for our experiments, is the Jensen-Shannon divergence (JSD). The similarity score which is the closest to the $w\left(QL\right)$ is retained and its corresponding lambda value represents our prediction.

The results for the Jensen-Shannon hypothesis are presented in Table 5.6, for Robust, WT10G and GOV2. In the same table we show the number of queries that achieve the perfect lambda prediction, as well as the number of queries that predict lambda with an error of 0.1 and 0.2 distance from the best, in absolute value. The MAP values of QL and RM1 are clearly surpassed, however the prediction of the best lambda prediction remains an unreached target. Regarding the number of queries with perfect match, about 20% success is reached in the case of Robust, with 45 out of 249 matches. For the same collection, by allowing a prediction error of 0.2, 65% of the queries are matching, with 162 out of 249 queries.

Table 5.6: Predicting λ knowing the AP of QL and RM1

	Robust	**WT10G**	**GOV2**
MAP QL	0.2490	0.1982	0.2917
MAP RM1	0.2668	0.1934	0.2597
MAP for "JSD" lambda	0.3052	0.2393	0.3398
MAP for optimal lambda	**0.3247**	**0.2546**	**0.3540**
# of queries with perfect match λ	**45**	**14**	**25**
# of queries with partial match λ (0.1 err.)	**110**	**38**	**70**
# of queries with partial match λ (0.2 err.)	**162**	**58**	**106**
Total # of queries	249	97	148

In the following section we present and analyze the hypothesis based on logistic regression.

5.5.2 Logistic regression

Logistic regression is a type of probabilistic statistical classification model and it estimates the probability of an event occurring. Frequently, logistic regression is used to refer specifically to the problem in which the variable to predict is binary, meaning that the number of available categories is two. However, logistic regression can treat problems with multiple categories and this type of regression is referred to as multinomial logistic regression. This is our case, since we try to predict the optimal lambda value among 11 possible (from 0 to 1, with a step of 0.1).

Our hypothesis considers the AP for QL as given information denoted AP(QL) and AP(RM1), respectively. Based on this, we build seven models (M1 to M7), each based on several features. For model M6 we consider also the number of relevant documents per query as given information. We apply the logistic regression algorithm on each model, aiming to predict the best lambda parameter value for RM3.

The feature list for is presented below:

M1 AP(RM1), AP(QL)

M2 AP(RM1), AP(QL), AP(QL)/AP(RM1)

M3 AP(RM1), AP(QL), JSD(QL, RM1) (cutoff 100)

M4 0 or 1 (0 if AP(RM1)>AP(QL), 1 otherwise)

M5 features from **M2** and **M3** together

M6 AP(RM1)*R, AP(QL)*R, where R is the number of relevant documents per query.

M7 AP(QL)/AP(RM1)

The obtained results for all of the above models are displayed in Table 5.7. We compare our results with the optimal lambda results. We notice that the best results for Robust (0.3141) and GOV2 (0.3482) are obtained with the model M2, which requires the ratio in terms of AP between QL and RM1. This comes in addition to the M1 features represented by the performance of the two lists. We mention that these results are not far from the optimal lambda, which is 0.3247 for Robust and 0.3540 for GOV2 Moreover, the basic QL and RM1 results are clearly surpassed. However, this is a method requiring the AP values for both QL and RM1 lists, which means a lot of prior information, being unrealistic in a real world scenario.

Table 5.7: Predicting λ using logistic regression with various feature models

MAP	**Robust**	**WT10G**	**GOV2**
QL	0.2490	0.1982	0.2917
RM1	0.2668	0.1934	0.2597
Logistic regression M1	0.3130	0.2430	0.3457
Logistic regression M2	0.3141	0.2437	0.3482
Logistic regression M3	0.3130	0.2433	0.3455
Logistic regression M4	0.3110	0.2416	0.3435
Logistic regression M5	0.3131	0.2452	0.3481
Logistic regression M6	0.3090	0.2372	0.3400
Logistic regression M7	0.3035	0.2416	0.3419
Optimal lambda	**0.3247**	**0.2546**	**0.3540**

5.6 Other insights

By plotting the lambda value variations for several queries, we searched for a pattern concerning the connection between the parameter value that yields the best AP and the fact that either QL or RM1. Figures 5.1 and 5.2 depict the AP variation with respect to lambda, on a per query basis.

A clear pattern is not to be found, queries having unpredictable behaviour. However, in Figure 5.1 we can still notice that, in many cases, the maximum AP value is reached in the proximity of the best performing list, between QL and RM1, respectively. This cannot be generalized, since there are many exceptions. For instance, in Figure 5.2, one can notice that, for query 616, the AP of QL is close to 0.9, while the AP of RM1 is only 0.65. Regardless of this fact, the best lambda value is 0.2, therefore closer to the weaker RM1. This observation is valid for several queries, in the same figure.

In this context, we tried to assigned a fixed lambda value per query depending on which list is better out of QL and RM1. For instance, if QL is performing better

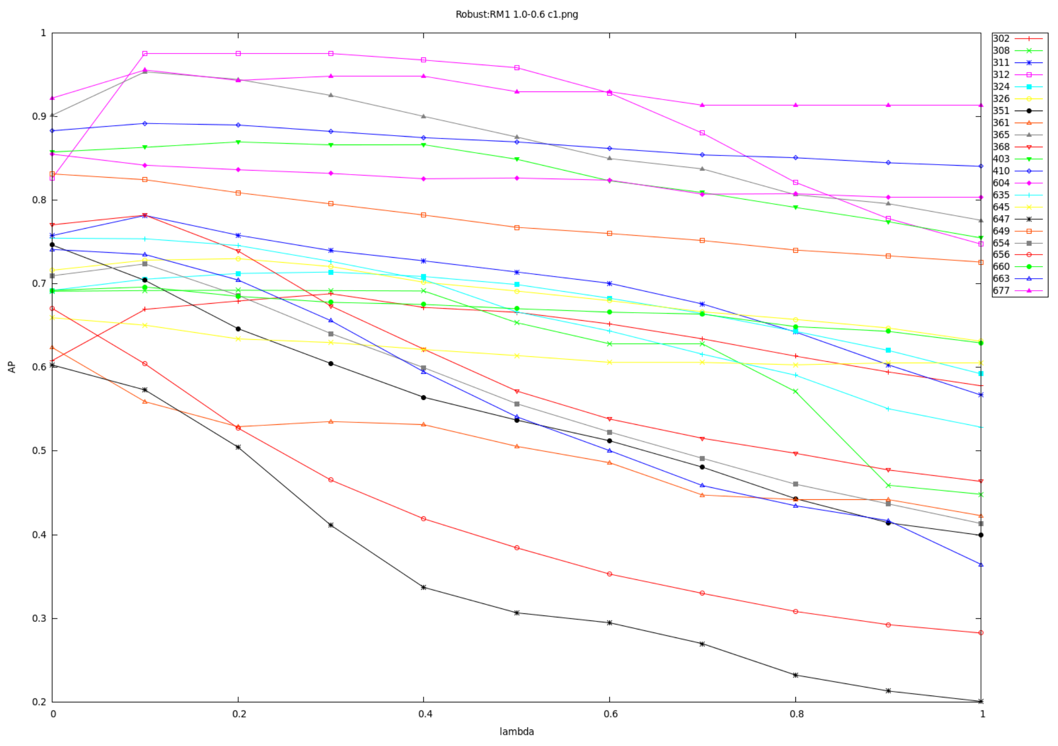


FIGURE 5.1: The results for the three test collections by the top levels of precision

we fix $\lambda = 0.2$ and $\lambda = 0.8$, otherwise. The results for this kind of binary selection are shown in Table 5.8. The first column, "$\lambda1$;$\lambda2$" represents the lambda values to fix with respect to which list is better between QL and RM1. If RM1 yields a better AP, then the choice is for $\lambda1$, otherwise the choice is for $\lambda2$. We highlight the best results per collection and we notice that the best lambda setup is different for all the three collections. For GOV2 the best result in terms of AP gets the closest to the optimal value.

TABLE 5.8: Predicting lambda by fixing values

$\lambda1$;$\lambda2$		**MAP**	
	Robust	**WT10G**	**GOV2**
0.1;0.6	0.3102	0.2418	0.3404
0.1;0.7	0.3110	**0.2425**	0.3412
0.1;0.8	**0.3116**	0.2417	0.3400
0.2;0.6	0.3044	0.2417	0.3428
0.2;0.7	0.3052	0.2424	**0.3435**
0.2;0.8	0.3058	0.2416	0.3424
Optimal lambda	**0.3247**	**0.2546**	**0.3540**

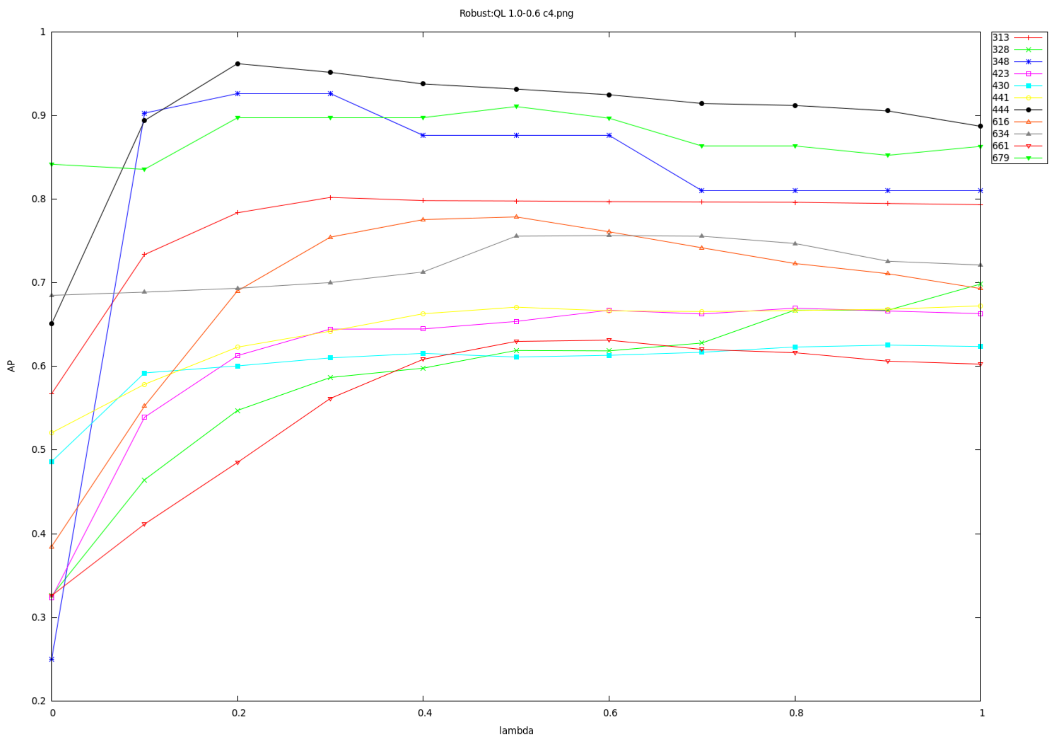


Figure 5.2: The results for the three test collections by the top levels of precision

5.7 Conclusion

In this chapter we explored the optimization possibilities for the lambda parameter in the case of language model-based query expansion, more precisely Relevance Model 3 (RM3).

This research started during a three months research mobility at Technion in Haifa, Israel and it continued long after the mobility was ended. The optimization problem we approached represents a hard challenge, raised by the robustness issue of query expansion methods. If one would be able to predict which query to expand and how, the expansion will surely be more effective than treating all the queries in the same way.

In this chapter, we thus presented four approaches to optimize RM3 expansion, which can be divided into approaches without prior information and approaches with prior information. For the approaches without any prior information we considered fitting the data with regression and classifying with SVM. The regression method predicts AP better than individual predictors, however the correlation coefficient is only around 0.60. We also mention that the SVM accuracy is close to results obtained by chance. The second approach, based on pseudo-qrels yields some improvements, especially in terms of Robustness Index (RI).

The Jensen-Shannon similarity based approach and the logistic regression approach require information on the quality of QL and RM1 lists. The results in these cases are getting closer to the optimal values. However, the required information would not be available in a real world context. Moreover, we believe that the methods require too much information compared to what they can give back in terms result improvements.

Therefore, the problem of optimizing RM3 parameter remains a hard, open problem. Of course, some improvements were noticeable, especially in the case of logistic regression.

Chapter 6

Conclusion and future work

Contents

This manuscript gathers the research results obtained during the three years thesis period. Our work is focused on adapting information retrieval systems to contexts with the purpose of improving retrieval performance. In this chapter we summarize our contributions, we recall the proposed methods and their corresponding results and, finally, we pitch the possible research continuity in the future work.

6.1 Conclusion

Our research is positioned in the field of Information Retrieval (IR) and, more precisely, it aims at adapting IR systems to contexts. The specific context we are focused on concerns the difficult queries. A query is difficult if it leads IR systems to failure in terms of performance measures. In our contributions we disambiguate query terms and we employ query difficulty prediction to selectively treat queries, all with the purpose of improving retrieval efficiency. More precisely, we proposed a term disambiguation method employed to re-rank retrieved document lists. Next, we predict query difficulty by combining existing predictors, with the purpose of improving prediction quality. In the next contribution, the query difficulty predictors are used as features for learning to discriminate between two different query expansion types. Finally, we approach a hard optimization problem, also in the field of query expansion, this time for language modeling-based relevance models.

We present several hypothesis for this optimization, together with their empirical results, which however are not conclusive. In the following sections we summarize our contributions and results.

6.1.1 Word sense discrimination in information retrieval

Our first contribution, presented in Chapter 2, represents a re-ranking method for IR, based on Word Sense (WS) discrimination. WS disambiguation in IR has been questioned and declared as ineffective [Sanderson 1994], [Guyot 2008]. In this context, we have proposed an unsupervised method of WS disambiguation, namely WS discrimination, and we have proven its efficiency in IR. In order to discriminate the term senses, the WS discrimination requires unsupervised clustering, for which we have chosen Naïve Bayes classification and spectral clustering. Our method shows a remarkable improvement in high rank precision for ambiguous queries. We believe that this represents a very important aspect, considering the fact that IR systems are predisposed to failure in the case of this particular type of queries [Stokoe 2003], [Mothe 2007]. The WordNet linguistic resource has been employed to check the term ambiguity, by considering the number of senses.

The experiment have been conducted over three TREC collections, TREC7, TREC8 and WT10G, respectively. We have analyzed the obtained results with respect to a major approach existing in the literature [Schütze 1995]. When using a Naïve Bayes-based clustering technique, we have shown that WS discrimination is indeed able to improve IR performance. However, unlike for the case of spectral clustering, when the baselines are generally surpassed for all the ambiguous queries, for the Naïve Bayes-based clustering we have only recorded very small improvement and only on some particular cases, hence the importance of the clustering technique used for WS discrimination in IR, which represents one of our conclusions.

Our method exploits the descriptive part of the TREC topics to extract query context, a level of detail which is not normally available in an IRS. Other works from the literature use these structural elements and most often the descriptive part of the query helps in improving the results [He 2004]. However, using the complete statement of the topic could lead to valid criticism of experiments such as ours because we exploit this detail. We prove that, if this level of detail is available, then it can be used in a beneficial way to improve retrieval effectiveness. However, even if our goal was not to develop mechanisms which can capture in an optimal way the needed level of detail, we do propose a method to capture the context of the query and show that our own method for WS discrimination in IR remains useful. Such a method of contextualization, namely the usage of PRF [Buckley 1994], has been employed for validating our conclusions in the presence of automatically generated context. Indeed, contextualizing short texts, such as tweet contextualization [SanJuan 2011] and query expansion [Ogilvie 2009] is an active research domain and we think it will be worth considering new contextualization techniques in our WS discrimination method as a future goal.

The research conducted in this chapter opens the lead to include in our study query difficulty prediction, which is already an active research area [Mothe 2005], [Pehcevski 2010], [Carmel 2010], [Sarnikar 2014], still having a lack in terms of applications, unfortunately. The fact that our method rather improves poor performing queries, especially those with multiple ambiguous terms,has driven our research along the path of query difficulty prediction.

6.1.2 Query difficulty predictors and predictor combinations

Improving poorly performing queries as our disambiguation method does, implies the identification of difficult queries. We have started research in the field of query difficulty prediction. Our contribution in this area is divided into two axis.

Firstly, we have shown that it is more robust to consider the average of several systems as a measure of difficulty to compare the correlations with different predictors, in order to avoid the variability generated by only one system. We have shown that for systems with a similar Mean Average Precision, the correlation coefficients with the same predictor of difficulty can be very different. In addition, for TREC7, the correlation between the Average Precision and the difficulty predictor, when the best system is chosen, is not statistically significant (p-value).

Secondly, we have shown that if the predictors are heterogeneous in nature (statistical, linguistic, pre-retrieval, post-retrieval, etc.), their combination may produce a stronger correlation with the measure of difficulty. We have proposed two linear combinations, each based on the interpolation of two predictors. $COMB_1$ is the interpolation between STD and WNS and $COMB_2$ is the interpolation between STD and IDF. The combinations have been tested over TREC7 and TREC8 collections and, $COMB_1$ has yielded better correlation coefficient over TREC7, while $COMB_2$ has yielded better correlation coefficient over TREC8. Moreover, $COMB_1$ has been more stable with respect to the p-value in the case of both test collections, suggesting that this combination is more reliable.

The idea of successfully combining query difficulty predictors suggests an application that consists in learning to decide which system would respond best for a particular query, when various systems are available, all based on query difficulty prediction queries. This result has paved the road towards our next contribution, focused on selective IR.

6.1.3 Selective information retrieval based on query difficulty predictors

In Chapter 4, we have proposed a selective IR method that employs query difficulty predictors, in order to learn how to discriminate between several types of query expansion. Our method takes advantage of a large range of query features, including linguistics features. The method focuses on the query and the ways to expand it, while other selective IR methods from the literature deal with other aspects, such as cross-language IR [Lee 2014], or with query routing in the case of domain specific collections [Sarnikar 2014]. The SVM has represented a choice for the learning mechanism.

We have tested our method over Robust, WT10G and GOV2 collections and we have proven its effectiveness in terms of learning accuracy, Mean Average Precision and Robustness Index. We have obtained a learning accuracy rate constantly over 80%, MAP improvements up to 24.65% in average and robustness improvements that reach 250%, results that are statistically significant.

The query difficulty prediction research obtains more and more efficient and accurate measures, although these predictors are not exploited and utilized enough in IR systems. We consider that the selective IR represents a realistic and fruitful framework for query difficulty prediction applications.

6.1.4 Parameter optimization for language modelling-based query expansion

Our last contribution represents an analysis of the optimization possibilities for the lambda parameter, in the case of language model-based query expansion, namely Relevance Model 3 (RM3). This research has been realized in collaboration with the Technion Institute, from Haifa, Israel.

Other researchers have approached this optimization problem [Lv 2009], unfortunately reporting only marginal results. Tuning the lambda parameter, represents a hard challenge, raised by the robustness issue of query expansion methods. If one would be able to predict which query to expand and how, the expansion will surely be more effective than treating all the queries in the same way.

To approach the research problem, we have presented in Chapter 5 four approaches to optimize RM3 expansion, which can be divided into approaches with prior information and approaches without prior information. For the latter approaches we considered fitting the data with regression and classifying with SVM. The regression method has predicted AP better than individual predictors, however the correlation coefficient is only around 0.60. We also mention that the SVM accuracy has been not very different to results obtained by chance. The second approach, based on pseudo-qrels has yielded some improvements, especially in terms of Robustness Index (RI).

The Jensen-Shannon similarity based approach and the logistic regression approach require information on the quality of QL and RM1 lists. The results in these cases have been closer to the optimal values. However, the required information would not be available in a real world context. Moreover, we believe that the methods require too much information compared to what they can give back in terms result improvements. The most noticeable improvements have been obtained in the case of logistic regression.

Therefore, the problem of optimizing RM3 parameter remains a hard, open problem.

6.2 Future work

The research conducted during this thesis have yielded good and interesting results. We are currently investigating several hypothesis that have emerged from the work presented in this manuscript. The obtained results have raised research questions that would be interesting to verify. We detail in the following sections our research leads for future work.

6.2.1 Redefining difficulty

Query difficulty could be defined in many ways, depending on the context. For instance in the TREC context, a topic can be considered difficult when "the median of the average precision (AP) scores of all participants for that topic is below a given threshold (i.e., half of the systems score lower than the threshold)" [Carmel 2006]. From another perspective, a topic could be considered difficult when it is too short, or when it contains ambiguous terms. However, a query could be easy for an IR system, but hard for another, making the system dependant difficulty worth to be studied.

Having all this in mind, we could try to find better ways to characterize query difficulty. One idea would be to see the terms as vertexes in a co-occurrence graph and to analyze the queries employing graph theory methods. Another lead would be to check the correlation between query difficulty and performance measures other than Average Precision, which is generally employed.

6.2.2 Latent Dirichlet allocation for selective IR

Query difficulty prediction is also related to estimating the retrieval score of an IRS, since a query is considered to be difficult or easy for a specific IRS depending on the score of the respective system. Furthermore, a method for estimating (or predicting) the score of an IRS can also be used for selective IR. Understanding the behavior of a good IRS and the behavior of a bad IRS is the key in solving this problem. Intuitively, a "good" system should return results on a specific topic (related to the query), while an average (or "bad") system will return results from various topics, since not all the documents will be relevant. This statement also represents the basis of the clarity score [Cronen-Townsend 2002b].

In this context, we aim at employing Latent Dirichlet allocation (LDA) to model the topics within a document collection in our future work. Then, by analyzing the global topic distribution generated by the results retrieved by an IRS, we can figure out if the behaviour of the system is "good" or "bad". A closed formula based on the global topic distribution could be used to determine the behaviour of the IRS. In the context of selective IR, we will choose the system that shows the best behavior on each query. Alternatively, we could use the closed formula to determine whether a query is difficult or not for a specific system, or to rank a set of IR systems.

6.2.3 Features for learning in selective IR

We have employed features extracted [Chifu 2014] (see Chapter 5). Based on a current research project, we try to extend our feature set, based on features from learning to rank data sets. We believe that useful information could be extracted from this kind of features.

Another feature extraction method that we are currently exploring is the application of convolutional neural networks over textual information from documents and queries. This type of neural network is widely employed in computer vision, but also in natural language processing. The network processes the raw data and extracts its own low level features, also called "deep" features.

Based on the obtained features, we could learn how to select the best IR system parameter configuration, or to classify queries by difficulty.

Author's publication list

International journal papers

[**Chifu 2015**] Adrian Chifu, Florentina Hristea, Josiane Mothe, Marius Popescu. *Word Sense Discrimination in Information Retrieval: A Spectral Clustering-based Approach.* In: Information Processing & Management, Elsevier, Vol. 51, p. 16-31, march 2015.
Access: `http://oatao.univ-toulouse.fr/13247/`

[**Chifu 2012**] Adrian Chifu, Radu Tudor Ionescu. *Word Sense Disambiguation to Improve Precision for Ambiguous Queries.* In: Central European Journal of Computer Science, Versita, co-editor Springer Verlag, London - GB, Vol. 2 N. 4, p. 398-411, december 2012.
Access: `http://www.irit.fr/publis/SIG/2012_CEJCS_CI.pdf`

National conference papers

[**Chifu 2015a**] Adrian Chifu, Léa Laporte, Josiane Mothe. *La prédiction efficace de la difficulté des requêtes : une tâche impossible?* (présentation courte) (regular paper). In: Conférence francophone en Recherche d'Information et Applications (CORIA 2015), Paris, Association Francophone de Recherche d'Information et Applications (ARIA), p. 189-204, March 2015.
Access: `http://www.irit.fr/publis/SIG/2015_CORIA_CLM.pdf`

[**Chifu 2014a**] Julie Ayter, Cecile Desclaux, Adrian Chifu, Josiane Mothe, Sébastien Déjean. *Performance Analysis of Information Retrieval Systems* (regular paper). In: Spanish Conference on Information Retrieval, Coruna, Springer-Verlag, (electronic format), June 2014.
Access: `http://oatao.univ-toulouse.fr/13047/`

[**Chifu 2014**] Adrian Chifu, Josiane Mothe. *Expansion sélective de requêtes par apprentissage* (regular paper). In: Conférence francophone en Recherche d'Information et Applications (CORIA 2014), Nancy, France, 19/03/2014-21/03/2014, LORIA, p. 231-246, March 2014.
Access: `http://www.irit.fr/publis/SIG/2014_CORIA_CM.pdf`

[**Chifu 2013**] Adrian Chifu. *Prédire la difficulté des requêtes : la combinaison de mesures statistiques et sémantiques* (short paper). In: Conférence francophone en Recherche d'Information et Applications (CORIA 2013), Neuchatel, Suisse, Université de Neuchâtel, p. 191-200, April 2013.
Access: `http://www.irit.fr/publis/SIG/2013_CORIA_C.pdf`

Bibliography

[Abdul-Jaleel 2004] Nasreen Abdul-Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah S. Larkey, Xiaoyan Li, Mark D. Smucker et Courtney Wade. *UMass at TREC 2004: Novelty and HARD.* In Ellen M. Voorhees et Lori P. Buckland, editeurs, TREC, volume Special Publication 500-261. National Institute of Standards and Technology (NIST), 2004. 24, 109

[Agirre 2006] Eneko Agirre et Philip Edmonds. Word sense disambiguation: Algorithms and applications. Springer Publishing Company, Incorporated, 1st édition, 2006. 38

[Aizerman 1964] M. A. Aizerman, E. M. Braverman et L. I. Rozonoér. *Theoretical foundations of the potential function method in pattern recognition learning.* Automation and Remote Control, vol. 25, pages 821–837, 1964. 103

[Amati 2002] Gianni Amati et Cornelis Joost Van Rijsbergen. *Probabilistic Models of Information Retrieval Based on Measuring the Divergence from Randomness.* ACM Trans. Inf. Syst., vol. 20, no. 4, pages 357–389, Octobre 2002. 16

[Amati 2004a] Giambattista Amati. *Probability models for information retrieval based on divergence from randomness.* PhD thesis, 2004. 18, 34, 109

[Amati 2004b] Giambattista Amati, Claudio Carpineto et Giovanni Romano. *Query Difficulty, Robustness, and Selective Application of Query Expansion.* Advances in Information Retrieval, pages 127–137, 2004. 94, 96, 97

[Andrews 2011] Pierre Andrews, Juan Pane et Ilya Zaihrayeu. *Semantic Disambiguation in Folksonomy: A Case Study.* In Proceedings of the 2009 International Conference on Advanced Language Technologies for Digital Libraries, NLP4DL'09/AT4DL'09, pages 114–134, Berlin, Heidelberg, 2011. Springer-Verlag. 71

[Artiles 2007] Javier Artiles, Julio Gonzalo et Satoshi Sekine. *The SemEval-2007 WePS Evaluation: Establishing a benchmark for the Web People Search Task.* In Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007). ACL, 2007. 41

[Aslam 2007] Javed A Aslam et Virgiliu Pavlu. *Query Hardness Estimation Using Jensen-Shannon Divergence Among Multiple Scoring Functions.* In Giambattista Amati, Claudio Carpineto et Giovanni Romano, editeurs, Advances in Information Retrieval, 29th European Conference on IR Research, ECIR 2007, Rome, Italy, April 2-5, 2007, Proceedings, volume 4425 of *Lecture Notes in Computer Science*, pages 198–209. Springer, 2007. 80, 82, 83, 89

[Attar 1977] Attar et Fraenkel. *Local Feedback in Full-Text Retrieval Systems.* JACM: Journal of the ACM, vol. 24, 1977. 71

[Baccini 2012] Alain Baccini, Sébastien Déjean, Laetitia Lafage et Josiane Mothe. *How many performance measures to evaluate information retrieval systems?* Knowledge and Information Systems, vol. 30, no. 3, pages 693–713, 2012. vii, 28, 29

[Baeza-Yates 1999] Ricardo A. Baeza-Yates et Berthier Ribeiro-Neto. Modern information retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999. 14

[Banerjee 2003] Satanjeev Banerjee et Ted Pedersen. *Extended gloss overlaps as a measure of semantic relatedness.* pages 805–810, Aot 2003. 48

[Bellugi 2015] Dino Bellugi, David G. Milledge, William E. Dietrich, Jim A. McKean, J. Taylor Perron, Erik B. Sudderth et Brian Kazian. *A spectral clustering search algorithm for predicting shallow landslide size and location.* Journal of Geophysical Research: Earth Surface, vol. 120, no. 2, pages 300–324, 2015. 40

[Bigot 2011] Anthony Bigot, Claude Chrisment, Taoufiq Dkaki, Gilles Hubert et Josiane Mothe. *Fusing different information retrieval systems according to query-topics: a study based on correlation in information retrieval systems and TREC topics.* Information Retrieval, vol. 14, no. 6, pages 617–648, dec 2011. 62, 68, 89, 94, 108

[Borjigin 2012] Sumuya Borjigin et Chonghui Guo. *Non-unique cluster numbers determination methods based on stability in spectral clustering.* Knowledge and Information Systems, Septembre 2012. 40

[Buckley 1994] Chris Buckley, Gerard Salton, James Allan et Amit Singhal. *Automatic Query Expansion Using SMART: TREC 3.* In TREC, pages 69–80, 1994. 23, 71, 74, 126

[Buckley 1995] Chris Buckley et Gerard Salton. *Optimization of Relevance Feedback Weights.* In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR ’95, pages 351–357, New York, NY, USA, 1995. ACM. 108

[Buckley 2004] Chris Buckley. *Why current IR engines fail.* In Proceedings of SIGIR’04, pages 584–585. ACM, 2004. 94

[Cao 2008] Guihong Cao, Jian-Yun Nie, Jianfeng Gao et Stephen Robertson. *Selecting good expansion terms for pseudo-relevance feedback.* In Proceedings of SIGIR’08, pages 243–250. ACM, 2008. 94, 95, 98

[Carmel 2006] David Carmel, Elad Yom-Tov, Adam Darlow et Dan Pelleg. *What makes a query difficult?* In Proceedings of SIGIR'06, pages 390 – 397. ACM Press, Aot 2006. 76, 89, 129

[Carmel 2010] David Carmel et Elad Yom-Tov. *Estimating the query difficulty for information retrieval.* In Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '10, pages 911–911, New York, NY, USA, 2010. ACM. 74, 76, 77, 78, 127

[Carpineto 2001a] Claudio Carpineto, Renato de Mori, Giovanni Romano et Brigitte Bigi. *An information-theoretic approach to automatic query expansion.* ACM Trans. Inf. Syst., vol. 19, no. 1, pages 1–27, Janvier 2001. 94

[Carpineto 2001b] Claudio Carpineto, de Mori, Renato, Giovanni Romano et Brigitte Bigi. *An information-theoretic approach to automatic query expansion.* ACM Transactions on Information Systems, vol. 19, no. 1, pages 1–27, 2001. 23

[Carpineto 2012] Claudio Carpineto et Giovanni Romano. *A Survey of Automatic Query Expansion in Information Retrieval.* ACM Computing Surveys, vol. 44, no. 1, pages 1–50, Janvier 2012. vii, 22, 23, 39, 95

[Chen 2012] Chen Chen, Hou Chunyan et Yuan Xiaojie. *Relevance Feedback Fusion via Query Expansion.* In Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Volume 03, WI-IAT '12, pages 122–126, Washington, DC, USA, 2012. IEEE Computer Society. 94, 95, 96, 97, 98

[Chifu 2012] Adrian Chifu et Radu Tudor Ionescu. *Word Sense Disambiguation to Improve Precision for Ambiguous Queries.* Central European Journal of Computer Science, vol. 2, no. 4, pages 398–411, 2012. 9, 38, 73, 74

[Chifu 2013] Adrian Chifu. *Prédire la difficulté des requêtes : la combinaison de mesures statistiques et sémantiques (in French).* In Proceedings of CORIA, Neuchatel, Switzerland, pages 191–200, April 2013. 10, 75, 84, 89

[Chifu 2014] Adrian Chifu et Josiane Mothe. *Expansion sélective de requêtes par apprentissage (regular paper).* In and, editeur, ConfÃľrence francophone en Recherche d'Information et Applications (CORIA), Nancy, France, 19/03/14-21/03/14, pages 231–246, http://www.loria.fr, mars 2014. LORIA. 10, 84, 93, 130

[Chifu 2015] Adrian-Gabriel Chifu, Florentina Hristea, Josiane Mothe et Marius Popescu. *Word sense discrimination in information retrieval: A spectral clustering-based approach.* Inf. Process. Manage, vol. 51, no. 2, pages 16–31, 2015. 9, 38, 74

[Collins-Thompson 2007] Kevyn Collins-Thompson et Jamie Callan. *Estimation and use of uncertainty in pseudo-relevance feedback.* In Proceedings of SIGIR'07, page 303. ACM Press, Juillet 2007. 105, 110

[Collins-Thompson 2009] Kevyn Collins-Thompson. *Reducing the risk of query expansion via robust constrained optimization.* In CIKM, pages 837–846, 2009. 105, 111

[Cronen-Townsend 2002a] Steve Cronen-Townsend et W. Bruce Croft. *Quantifying query ambiguity.* pages 104–109, Mars 2002. 82, 91

[Cronen-Townsend 2002b] Steve Cronen-Townsend, Yun Zhou et W. Bruce Croft. *Predicting query performance.* In Proceedings of SIGIR'02, pages 299 – 306. ACM Press, Aot 2002. 76, 80, 89, 95, 96, 98, 110, 129

[Cronen-Townsend 2004a] Steve Cronen-Townsend, Yun Zhou et W. Bruce Croft. *A framework for selective query expansion.* In Proceedings of the thirteenth ACM international conference on Information and knowledge management, CIKM '04, pages 236–237. ACM, 2004. 94, 96, 97

[Cronen-Townsend 2004b] Steve Cronen-Townsend, Yun Zhou et W. Bruce Croft. *A language modeling framework for selective query expansion.* Rapport technique, 2004. 83, 110

[D'Angelo 2011] Ciriaco Andrea D'Angelo, Cristiano Giuffrida et Giovanni Abramo. *A heuristic approach to author name disambiguation in bibliometrics databases for large-scale research assessments.* JASIST, vol. 62, no. 2, pages 257–269, 2011. 41

[De Loupy 2000] Claude De Loupy et Patrice Bellot. *Evaluation of document retrieval systems and query difficultyl.* In Proceedings of the Workshop Using Evaluation within HLT Programs : Results and trends, pages 31–38, 2000. 94

[Dempster 1977] A. P. Dempster, N. M. Laird et Donald B. Rubin. *Maximum likelihood from incomplete data via the EM algorithm.* Journal of the Royal Statistical Society, Series B, vol. 39, pages 1–38, 1977. 47

[Deveaud 2013] Romain Deveaud, Ludovic Bonnefoy et Patrice Bellot. *Quantification et identification des concepts implicites d'une requête.* In CORIA 2013 - Conférence en Recherche d'Infomations et Applications - 10th French Information Retrieval Conference, Neuchâtel, Suisse, April 3-5, 2013., pages 159–174, 2013. 108

[Doszkocs 1978] Tamas E. Doszkocs. *An Associative Interactive Dictionary for Online Bibliographic Searching.* In Jerusalem Conference on Information Technology, pages 489–492, 1978. 23

[Fox 1994] Edward A. Fox et J. A. Shaw. *Combination of multiple searches.* In Proc. TREC-2, pages 243–249, 1994. 96, 98

[Gale 1992] William A. Gale, Kenneth W. Church et David Yarowsky. *One Sense Per Discourse.* In Proceedings of the Workshop on Speech and Natural Language, HLT '91, pages 233–237, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics. 45, 53

[Gangadharaiah 2006] Rashmi Gangadharaiah, Ralf D. Brown et Jaime G. Carbonell. *Spectral Clustering for Example Based Machine Translation.* In HLT-NAACL'06, pages –1–1, 2006. 44

[Gonzalo 1998] Julio Gonzalo, Felisa Verdejo, Irina Chugur et Juan Cigarrin. *Indexing with WordNet synsets can improve text retrieval.* In CoRR, pages 38–44, 1998. 39

[Guyon 2003] Isabelle Guyon et André Elisseeff. *An introduction to variable and feature selection.* The Journal of Machine Learning Research, vol. 3, pages 1157–1182, Mars 2003. 104

[Guyot 2008] Jacques Guyot, Gilles Falquet, Saïd Radhouani et Karim Benzineb. *Analysis of Word Sense Disambiguation-Based Information Retrieval.* In Carol Peters, Thomas Deselaers, Nicola Ferro, Julio Gonzalo, Gareth J. F. Jones, Mikko Kurimo, Thomas Mandl, Anselmo Peñas et Vivien Petras, editeurs, CLEF, volume 5706 of *Lecture Notes in Computer Science*, pages 146–154. Springer, 2008. 9, 39, 73, 126

[Han 2005] Hui Han, Hongyuan Zha et C. Lee Giles. *Name Disambiguation in Author Citations using a K-way Spectral Clustering Method.* In Proceedings of the Fifth ACM/IEEE-CS Joint Conference on Digital Libraries, 2005. 44

[Harman 1992] Donna Harman. *Relevance Feedback Revisited.* In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '92, pages 1–10, New York, NY, USA, 1992. ACM. 108

[Harman 1993] Donna Harman. *Overview of the First TREC Conference.* In Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '93, pages 36–47, New York, NY, USA, 1993. ACM. 25

[Harman 2009] Donna Harman et Chris Buckley. *Overview of the Reliable Information Access Workshop.* Information Retrieval, Springer, vol. 12, no. 6, pages 615–641, 2009. 15, 76, 108

[Harter 1974] S.P. Harter. *A probabilistic approach to automatic keyword indexing.* PhD thesis, 1974. 18

[Hastie 2008] Trevor Hastie, Robert Tibshirani et Jerome Friedman. The elements of statistical learning: data mining, inference and prediction. Springer, 2008. 49

[Hauff 2008] Claudia Hauff, Djoerd Hiemstra et Franciska de Jong. *A survey of pre-retrieval query performance predictors.* In James G Shanahan, Sihem Amer-Yahia, Ioana Manolescu, Yi Zhang, David A Evans, Aleksander Kolcz, Key-Sun Choi et Abdur Chowdhury, editeurs, Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008, pages 1419–1420. ACM, 2008. 79

[Hauff 2009] Claudia Hauff, Leif Azzopardi et Djoerd Hiemstra. *The Combination and Evaluation of Query Performance Prediction Methods.* In Advances in Information Retrieval, 31th European Conference on IR Research, ECIR 2009, Toulouse, France, April 6-9, 2009. Proceedings, pages 301–312, 2009. 10, 82

[Hauff 2010a] C. Hauff. *Predicting the Effectiveness of Queries and Retrieval Systems.* PhD thesis, Centre for Telematics and Information Technology University of Twente, Janvier 2010. 79, 82, 89

[Hauff 2010b] Claudia Hauff. *Predicting the effectiveness of queries and retrieval systems.* SIGIR Forum, vol. 44, no. 1, page 88, Aot 2010. 78, 79, 82, 83

[Hauke 2011] Jan Hauke et Kossowski Tomasz. *Comparison of Values of Pearson's and Spearman's Correlation Coefficients on the Same Sets of Data.* Quaestiones Geographicae, vol. 30, no. 2, pages 87–93, 2011. 30

[He 2003] Ben He et Iadh Ounis. *University of Glasgow at the Robust Track- A Query-based Model Selection Approach for the Poorly-Performing Queries.* In TREC, pages 636–645, 2003. 96

[He 2004] Ben He et Iadh Ounis. *Inferring query performance using pre-retrieval predictors.* In 11th International Conference, SPIRE 2004, Proceedings, pages 43 – 54, Padova, 2004. Springer Berlin Heidelberg. 74, 78, 79, 80, 82, 83, 92, 94, 95, 96, 97, 98, 102, 126

[He 2005] Ben He et Iadh Ounis. *Term Frequency Normalisation Tuning for BM25 and DFR Models.* In David E. Losada et Juan M. Fernández-Luna, editeurs, ECIR - Advances in Information Retrieval, 27th European Conference on IR Research, ECIR 2005, Santiago de Compostela, Spain, March 21-23, 2005, Proceedings, volume 3408 of *Lecture Notes in Computer Science*, pages 200–214. Springer, 2005. 56, 88

[He 2007] Ben He et Iadh Ounis. *Combining fields for query expansion and adaptive query expansion.* Information Processing & Management, vol. 43, no. 5, pages 1294–1307, Septembre 2007. 104

[He 2008] Jiyin He, Martha Larson et Maarten de Rijke. *Using Coherence-Based Measures to Predict Query Difficulty.* In Advances in Information Retrieval, 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings, volume 4956 of *Lecture Notes in Computer Science*, pages 689–694. Springer, 2008. 79, 82

[He 2009] Ben He et Iadh Ounis. Studying Query Expansion Effectiveness, volume 5478 of *Lecture Notes in Computer Science.* Springer Berlin Heidelberg, Berlin, Heidelberg, Avril 2009. 71, 108

[Hjörland 2002] Birger Hjörland et Frank Sejer Christensen. *Work tasks and socio-cognitive relevance: A specific example.* Journal of the American Society for Information Science and Technology, vol. 53, no. 11, pages 960–965, 2002. 14

[Hristea 2008] Florentina Hristea, Marius Popescu et Monica Dumitrescu. *Performing word sense disambiguation at the border between unsupervised and knowledge-based techniques.* Artif. Intell. Rev, vol. 30, no. 1-4, pages 67–86, 2008. 40, 45, 48, 53

[Hristea 2009a] F. Hristea. *Recent advances Concerning the Usage of the Naïve Bayes Model in Unsupervised Word Sense Disambiguation.* Internationl Review on Computers and Software, vol. 4, no. 1, pages 58–67, 2009. 40

[Hristea 2009b] Florentina Hristea et Marius Popescu. *Adjective Sense Disambiguation at the Border Between Unsupervised and Knowledge-Based Techniques.* Fundam. Inf., vol. 91, no. 3-4, pages 547–562, Aot 2009. 47

[Joachims 1999] T. Joachims. *Making large-Scale SVM Learning Practical.* In B. Schölkopf, C. Burges et A. Smola, editeurs, Advances in Kernel Methods - Support Vector Learning, chapitre 11, pages 169–184. MIT Press, Cambridge, MA, 1999. 116

[Joachims 2002] Thorsten Joachims. *Optimizing Search Engines Using Clickthrough Data.* In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '02, pages 133–142, New York, NY, USA, 2002. ACM. 25

[Kelly 2009] Diane Kelly. *Methods for Evaluating Interactive Information Retrieval Systems with Users.* Found. Trends Inf. Retr., vol. 3, no. 1–2, pages 1–224, Janvier 2009. 25

[Kilgarriff 1997] Adam Kilgarriff. *What is word sense disambiguation good for?* CoRR, vol. cmp-lg/9712008, 1997. 41

[Kilgarriff 2000] Adam Kilgarriff et Joseph Rosenzweig. *Framework and Results for English SENSEVAL.* Computers and the Humanities, vol. 34, no. 1-2, pages 15–48, 2000. 41

[Kim 2004] Sang-Bum Kim, Hee-Cheol Seo et Hae-Chang Rim. *Information retrieval using word senses.* In Proceedings of SIGIR'04, page 258. ACM Press, Juillet 2004. 39

[Krovetz 1992] R. Krovetz et B. Croft. *Lexical Ambiguity and Information Retrieval.* ACM Transactions on Information Systems, vol. 10, no. 2, pages 115–141, 1992. 39, 42

[Kumaran 2009] Giridhar Kumaran et Vitor R. Carvalho. *Reducing long queries using query quality predictors.* In James Allan, Javed A. Aslam, Mark Sanderson, ChengXiang Zhai et Justin Zobel, editeurs, SIGIR, pages 564–571. ACM, 2009. 98

[Kurland 2012] Oren Kurland, Anna Shtok, Shay Hummel, Fiana Raiber, David Carmel et Ofri Rom. *Back to the Roots: A Probabilistic Framework for Query-performance Prediction.* In Proceedings of, CIKM '12, pages 823–832, 2012. 10, 81, 82, 83, 95

[Lancaster 1974] F. W. Lancaster et E. G. Fayen. *Information Retrieval On-Line.* Journal of the American Society for Information Science, vol. 25, no. 5, pages 336–337, 1974. 15

[Lavrenko 2001] Victor Lavrenko et W. Bruce Croft. *Relevance based language models.* In Proceedings of SIGIR'01, pages 120–127. ACM Press, Septembre 2001. 23, 35, 101, 108, 109

[Lee 2008] Kyung-Soon Lee, W. Bruce Croft et James Allan. *A cluster-based resampling method for pseudo-relevance feedback.* In Sung-Hyon Myaeng, Douglas W. Oard, Fabrizio Sebastiani 0001, Tat-Seng Chua et Mun-Kew Leong, editeurs, SIGIR, pages 235–242. ACM, 2008. 22

[Lee 2009] Chia-Jung Lee, Yi-Chun Lin, Ruey-Cheng Chen et Pu-Jen Cheng. *Selecting Effective Terms for Query Formulation.* In GaryGeunbae Lee, Dawei Song, Chin-Yew Lin, Akiko Aizawa, Kazuko Kuriyama, Masaharu Yoshioka et Tetsuya Sakai, editeurs, Information Retrieval Technology, volume 5839 of *Lecture Notes in Computer Science*, pages 168–180. Springer Berlin Heidelberg, 2009. 21

[Lee 2014] Chia-Jung Lee et W. Bruce Croft. *Cross-Language Pseudo-Relevance Feedback Techniques for Informal Text.* In Advances in Information Retrieval - 36th European Conference on IR Research, ECIR 2014, Amsterdam, The Netherlands, April 13-16, 2014. Proceedings, pages 260–272, 2014. 10, 83, 98, 128

[Leidner 2007] Jochen L. Leidner. *Toponym resolution in text: annotation, evaluation and applications of spatial grounding.* SIGIR Forum, vol. 41, no. 2, pages 124–126, 2007. 41

[Lin 1997] Dekang Lin. *Using Syntactic Dependency as Local Context to Resolve Word Sense Ambiguity.* In Meeting of the Association for Computation Linguistics, pages 64–71, 1997. 38

[Lin 2008] Jimmy J. Lin et Mark D. Smucker. *How do users find things with PubMed?: towards automatic utility evaluation with user simulations.* In SIGIR, pages 19–26. ACM, 2008. 98

[Liu 2004] Shuang Liu, Fang Liu, Clement T. Yu et Weiyi Meng. *An effective approach to document retrieval via utilizing WordNet and recognizing phrases.* In Mark Sanderson, Kalervo Järvelin, James Allan et Peter Bruza, editeurs, SIGIR, pages 266–272. ACM, 2004. 22

[Liu 2012] Hongzhi Liu, Zhonghai Wu et D. Frank Hsu. *Combination of Multiple Retrieval Systems Using Rank-Score Function and Cognitive Diversity.* In Leonard Barolli, Tomoya Enokido, Fatos Xhafa et Makoto Takizawa, editeurs, AINA, pages 167–174. IEEE, 2012. 96

[Lv 2009] Yuanhua Lv et ChengXiang Zhai. *Adaptive Relevance Feedback in Information Retrieval.* In Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM ’09, pages 255–264, New York, NY, USA, 2009. ACM. 10, 24, 94, 96, 97, 105, 108, 110, 111, 128

[Maier 2009] Markus Maier, Matthias Hein et Ulrike von Luxburg. *Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters.* Theoretical Computer Science, vol. 410, no. 19, pages 1749–1764, Avril 2009. 49

[Manning 1999] Christopher D. Manning et Hinrich Schütze. Foundations of statistical natural language processing. MIT Press, Cambridge, MA, USA, 1999. 44

[Màrquez 2006] Lluís Màrquez, Gerard Escudero, David Martínez et German Rigau. *Supervised Corpus-Based Methods for WSD.* In Eneko Agirre et Philip Edmonds, editeurs, Word Sense Disambiguation Algorithms and Applications, volume 33 of *Text, Speech and Language Technology*, pages 167 – 216. Springer Netherlands, Dordrecht, 2006. 51

[Meister 2011] Lior Meister, Oren Kurland et Inna Gelfer Kalmanovich. *Re-ranking search results using an additional retrieved list.* Inf. Retr, vol. 14, no. 4, pages 413–437, 2011. 54

[Metzler 2005] Donald Metzler et W Bruce Croft. *A Markov random field model for term dependencies.* In Proceedings of SIGIR’05, pages 472–479. ACM Press, 2005. 79, 82

[Mihalcea 2000] Rada Mihalcea et Dan Moldovan. *Semantic indexing using WordNet senses.* In Proceedings of the ACL-2000 workshop on Recent advances in natural language processing and information retrieval held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics -, volume 11, page 35, Morristown, NJ, USA, Octobre 2000. Association for Computational Linguistics. 39

[Miller 1995] George A. Miller. *WordNet: A Lexical Database for English.* Commun. ACM, vol. 38, no. 11, pages 39–41, Novembre 1995. 41

[Mitra 1998] M. Mitra, A. Singhal et C. Buckley. *Improving Automatic Query Expansion.* In ACM SIGIR, Melbourne, Austrailia, aug 1998. 22

[Mizzaro 1997] Stefano Mizzaro. *Relevance: The whole history.* Journal of the American Society for Information Science, vol. 48, no. 9, pages 810–832, 1997. 14, 15

[Mooers 1950] C. N. Mooers. *The theory of digital handling of non-numerical information and its implications to machine economics.* In Proceedings of the Association for Computing Machinery Conference, 1950. 14

[Mothe 2005] Josiane Mothe et Ludovic Tanguy. *Linguistic features to predict query difficulty.* In ACM Conference on research and Development in Information Retrieval, SIGIR, Predicting query difficulty - methods and applications workshop , Salvador de Bahia, Brésil, 15/08/05-19/08/05, pages 7–10. ACM, 2005. 74, 76, 78, 82, 83, 84, 95, 96, 127

[Mothe 2007] Josiane Mothe et Ludovic Tanguy. *Linguistic Analysis of Users' Queries: Towards an Adaptive Information Retrieval System.* In Signal-Image Technologies and Internet-Based System, pages 77–84, 2007. 73, 98, 126

[Navigli 2009] Roberto Navigli. *Word sense disambiguation: A survey.* ACM Comput. Surv, vol. 41, no. 2, 2009. 41

[Ng 2011] Hwee Tou Ng. *Does word sense disambiguation improve information retrieval?* In Proceedings of the fourth workshop on Exploiting semantic annotations in information retrieval - ESAIR '11, page 17, New York, New York, USA, Octobre 2011. ACM Press. 39

[Ogilvie 2009] Paul Ogilvie, Ellen Voorhees et Jamie Callan. *On the number of terms used in automatic query expansion.* Information Retrieval, vol. 12, no. 6, pages 666–679, Juillet 2009. 74, 126

[Ounis 2006] I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald et C. Lioma. *Terrier: A High Performance and Scalable Information Retrieval Platform.* In Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006), 2006. 34

[Pehcevski 2010] Jovan Pehcevski, James Thom, Anne-Marie Vercoustre et Vladimir Naumovski. *Entity ranking in Wikipedia: utilising categories, links and topic difficulty prediction.* Information Retrieval, vol. 13, pages 568–600, 2010. 10.1007/s10791-009-9125-9. 74, 127

[Peng 2010] Jie Peng, Craig Macdonald et Iadh Ounis. *Learning to select a ranking function.* In Proceedings of the 32nd European conference on Advances in Information Retrieval, ECIR'2010, pages 114–126, Berlin, Heidelberg, 2010. Springer-Verlag. 94

[Piskorski 2009] Jakub Piskorski, Karol Wieloch et Marcin Sydow. *On knowledge-poor methods for person name matching and lemmatization for highly inflectional languages.* Information Retrieval, vol. 12, no. 3, pages 275–299, 2009. 41

[Ponte 1998] Jay M. Ponte et 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, SIGIR '98, pages 275–281, New York, NY, USA, 1998. ACM. 15, 20, 109

[Popescu 2011] Marius Popescu et Florentina Hristea. *State of the art versus classical clustering for unsupervised word sense disambiguation.* Artif. Intell. Rev, vol. 35, no. 3, pages 241–264, 2011. 40, 44, 50, 51

[Porter 1980] M. F. Porter. *An algorithm for suffix striping.* Program, vol. 14, no. 3, pages 130–137, 1980. 52

[Robertson 1976] Stephen E. Robertson et Karen Sparck Jones. *Relevance weighting of search terms.* Journal of the American Society for Information Science, vol. 27, pages 129–146, 1976. 16, 23

[Robertson 1990] S. Robertson. *On term selection for query expansion.* Journal of Documentation, vol. Vol. 46, pages 359–364, 1990. 23

[Robertson 1994] S. E. Robertson et S. Walker. *Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval.* In Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '94, pages 232–241, New York, NY, USA, 1994. Springer-Verlag New York, Inc. 15, 17, 18, 34

[Rocchio 1971] J. J. Rocchio. *Relevance feedback in information retrieval.* In G. Salton, editeur, The Smart retrieval system - experiments in automatic document processing, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall, 1971. 23, 108

[Sakai 2005] Tetsuya Sakai, Toshihiko Manabe et Makoto Koyama. *Flexible pseudo-relevance feedback via selective sampling.* ACM Transactions on Asian Language Information Processing, vol. 4, no. 2, pages 111–135, Juin 2005. 22, 104

[Salton 1975] G. Salton, A. Wong et C. S. Yang. *A Vector Space Model for Automatic Indexing.* Commun. ACM, vol. 18, no. 11, pages 613–620, Novembre 1975. 15, 16

[Sanderson 1994] Mark Sanderson. *Word sense disambiguation and information retrieval.* In Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval, pages 49–57, Dublin, IE, 1994. 9, 39, 42, 73, 126

[SanJuan 2011] Eric SanJuan, Patrice Bellot, Véronique Moriceau et Xavier Tannier. *Overview of the INEX 2010 question answering track (QA@INEX).* In Proceedings of the 9th international conference on Initiative for the evaluation of XML retrieval: comparative evaluation of focused retrieval, INEX'10, pages 269–281, Berlin, Heidelberg, 2011. Springer-Verlag. 74, 126

[Santos 2010] Rodrygo L. T. Santos, Craig Macdonald et Iadh Ounis. *Selectively diversifying web search results.* In Jimmy Huang, Nick Koudas, Gareth J. F. Jones, Xindong Wu, Kevyn Collins-Thompson et Aijun An, editeurs, CIKM, pages 1179–1188. ACM, 2010. 96

[Sarnikar 2014] Surendra Sarnikar, Zhu Zhang et J. Leon Zhao. *Query-performance prediction for effective query routing in domain-specific repositories.* JASIST, vol. 65, no. 8, pages 1597–1614, 2014. 10, 74, 83, 95, 127, 128

[Schütze 1995] Hinrich Schütze et Jan Pedersen. *Information Retrieval based on Word Senses.* In Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval, pages 161–175, Las Vegas, USA, 1995. 38, 42, 43, 65, 66, 67, 74, 126

[Schütze 1998] Heinrich Schütze. *Automatic Word Sense Discrimination.* Computational Linguistics, vol. 24, no. 1, pages 97–123, 1998. 42, 43

[Shaw 1995] J. A. Shaw et Edward A. Fox. *Combination of Multiple Searches.* In Overview of the Third Text REtrieval Conference (TREC-3), pages 105–108. NIST Gaithersburg, 1995. 53, 58, 67

[Shtok 2009] Anna Shtok, Oren Kurland et David Carmel. *Predicting Query Performance by Query-Drift Estimation.* In 2nd International Conference on the Theory of Information Retrieval, pages 305–312, 2009. 81, 82, 83, 85, 95, 96, 113

[Speicher 2014] Maximilian Speicher, Andreas Both et Martin Gaedke. *Ensuring Web Interface Quality through Usability-Based Split Testing.* In Sven Casteleyn, Gustavo Rossi et Marco Winckler, editeurs, Web Engineering, volume 8541 of *Lecture Notes in Computer Science*, pages 93–110. Springer International Publishing, 2014. 25

[Stokoe 2003] Christopher Stokoe, Michael P. Oakes et John Tait. *Word sense disambiguation in information retrieval revisited.* In SIGIR, pages 159–166. ACM, 2003. 39, 73, 126

[Uzuner 1999] Ozlem Uzuner, Boris Katz et Deniz Yuret. *Word sense disambiguation for information retrieval.* In Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence, AAAI '99/IAAI '99, pages 985–, Menlo Park, CA, USA, 1999. American Association for Artificial Intelligence. 39

[Vinay 2006] V Vinay, I J Cox, N Milic-Frayling et K Wood. *On Ranking the Effectiveness of Searches.* ACM Press, 2006. 80, 82, 83

[von Luxburg 2007] Ulrike von Luxburg. *A tutorial on spectral clustering.* Statistics and Computing, vol. 17, no. 4, pages 395–416, 2007. 49, 50

[Voorhees 1993] Ellen Voorhees. *Using WordNet to disambiguate word senses for text retrieval.* In Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '93, pages 171–180, New York, NY, USA, 1993. ACM. 39, 42

[Voorhees 1998] E. Voorhees et D. Harman. *Overview of the Seventh Text REtrieval Conference (TREC-7).* In Text REtrieval Conference (TREC) TREC-7 Proceedings, pages 1–23. Department of Commerce, National Institute of Standards and Technology, 1998. NIST Special Publication 500-242: The Seventh Text REtrieval Conference (TREC 7). 32

[Wang 2014] Xiang Wang, Buyue Qian et Ian Davidson. *On constrained spectral clustering and its applications.* Data Mining and Knowledge Discovery, vol. 28, no. 1, pages 1–30, 2014. 40

[Wilkins 2006] Peter Wilkins, Paul Ferguson et Alan F. Smeaton. *Using score distributions for query-time fusion in multimediaretrieval.* In James Ze Wang, Nozha Boujemaa et Yixin Chen, editeurs, Multimedia Information Retrieval, pages 51–60. ACM, 2006. 96

[Yom-Tov 2005] Elad Yom-Tov, Shai Fine, David Carmel et Adam Darlow. *Learning to estimate query difficulty.* In Proceedings of SIGIR'05, page 512. ACM Press, Aot 2005. 96

[Zaragoza 2004] Hugo Zaragoza, Nick Craswell, Michael Taylor, Suchi Saria et Stephen Robertson. *Microsoft Cambridge at TREC-13: Web and HARD tracks.* In IN PROCEEDINGS OF TREC 2004, 2004. 34

[Zhai 2001] Chengxiang Zhai et John Lafferty. *A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval.* In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '01, pages 334–342, New York, NY, USA, 2001. ACM. 20, 21

[Zhai 2004] Chengxiang Zhai et John Lafferty. *A Study of Smoothing Methods for Language Models Applied to Information Retrieval.* ACM Trans. Inf. Syst., vol. 22, no. 2, pages 179–214, Avril 2004. 65

[Zhao 2005] Bing Zhao, Eric P. Xing et Alex Waibel. *Bilingual Word Spectral Clustering for Statistical Machine Translation.* In Proceedings of the ACL Workshop on Building and Using Parallel Texts, pages 25–32, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. 44

[Zhao 2008] Ying Zhao, Falk Scholer et Yohannes Tsegay. *Effective Pre-retrieval Query Performance Prediction Using Similarity and Variability Evidence.* In Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven et Ryen W White, editeurs, Advances in Information Retrieval, 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings, volume 4956 of *Lecture Notes in Computer Science*, pages 52–64. Springer, 2008. 79, 80, 82, 83, 113

[Zhao 2012] Le Zhao et Jamie Callan. *Automatic term mismatch diagnosis for selective query expansion.* In Proceedings of SIGIR'12, page 515. ACM Press, Aot 2012. 95, 98

[Zhong 2012] Zhi Zhong et Hwee Tou Ng. *Word Sense Disambiguation Improves Information Retrieval.* In ACL (1), pages 273–282. The Association for Computer Linguistics, 2012. 9, 38, 39, 43, 56, 88, 102

[Zhou 2006] Yun Zhou et W Bruce Croft. *Ranking Robustness: A Novel Framework to Predict Query Performance.* In Proc. CIKM'06, 2006. 80, 83

[Zhou 2007] Yun Zhou et W Bruce Croft. *Query performance prediction in web search environments.* In Wessel Kraaij, Arjen P de Vries, Charles L A Clarke, Norbert Fuhr et Noriko Kando, editeurs, Proceedings of SIGIR'07, pages 543–550. ACM, 2007. 80, 81, 82, 83, 85, 89, 95, 96, 110, 113