LEVERAGING USER INTERACTION TO IMPROVE SEARCH EXPERIENCE WITH DIFFICULT AND EXPLORATORY QUERIES

BY

ALEXANDER SERGEYEVICH KOTOV

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science in the Graduate College of the University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Doctoral Committee:

Associate Professor ChengXiang Zhai, Chair
Professor Jiawei Han
Associate Professor Kevin Chang
Assistant Professor Luo Si, Purdue University
Abstract

The query-based search paradigm is based on the assumption that the searchers are able to come up with the effective differentiator terms to make their queries specific and precise. In reality, however, a large number of queries are problematic return either too many or no relevant documents in the initial search results. Existing search systems provide no assistance to the users when they cannot formulate an effective keyword query and receive the search results of poor quality. In some cases, the users may intentionally formulate broad or exploratory queries (for example, when they want to explore a particular topic without having a clear search goal). In other cases, the users may not know the domain of the search problem sufficiently well and their queries may suffer from the problems, of which they may not be aware, such as ambiguity or vocabulary mismatch. Although the quality of search results can be improved by reformulating the queries, finding a good reformulation is often non-trivial and takes time. Therefore, in addition to the existing work on using the relevant documents from the top-ranked initially retrieved results to retrieve more relevant documents, it is important from both theoretical and practical points of view to also develop an interactive retrieval model, which would allow the search systems to improve the users’ search experience with exploratory queries, which return too many relevant documents, and difficult queries, which return no relevant documents in the initial search results. In this thesis, we propose and study three methods for interactive feedback that allow the search systems to interactively improve the quality of retrieval results for difficult and exploratory queries: question feedback, sense feedback and concept feedback. All three methods are based on a novel question-guided interactive retrieval model, in which a search system collaborates with the users in achieving their search goals by generating the natural language refinement questions.

The first method, question feedback is aimed at interactive refinement of short, exploratory keyword-based queries by automatically generating a list clarification questions, which can be presented next to the standard ranked list of the retrieved documents. Clarification questions place the broad query terms into a specific context and help the user focus on and explore a particular aspect of the query topic. By clicking on a question, the users are presented with an answer to it and by clicking on the answer they can be redirected to the document containing the answer for further exploration. Therefore, clarification questions
can be considered as shortcuts to specific answers. Questions also provide a more natural mechanism to elicit relevance feedback from the users. A query can be expanded by adding the terms from the clicked question and resubmitted to the search system, generating a new set of questions and documents retrieved with the expanded query. Enabling interactive question-based retrieval requires major changes to all components of the retrieval process: from more sophisticated methods of content analysis to ranking and feedback. Specifically, we propose the methods to locate and index the content, which can be used for question generation, and to generate and rank well-formed and meaningful questions in response to user queries. We implemented the prototype of a question-guided search system on a subset of Wikipedia and conducted the user studies, which demonstrated the effectiveness of the question-based feedback strategy.

The second method, sense feedback, is aimed at clarifying the intended sense of ambiguous query terms with automatically generated clarification questions in the form of “Did you mean \( \{ \text{ambiguous query term} \} \) as \( \{ \text{sense label} \} \)?”, where the sense label can be a single term or a phrase. Our approach to sense detection is based on the assumption that the senses of a word can be differentiated by grouping and analyzing all the contexts, in which a given word appears in the collection. We propose to detect the senses of a query term by clustering the global (based on the entire collection) graph of relationships of a query term with other terms in the collection vocabulary. We conducted simulation experiments with two graph clustering algorithms and two methods for calculating the strength of relationship between the terms in the graph to determine the upper bound for the retrieval effectiveness of sense feedback and the best method for detecting the senses. We also proposed several alternative methods to represent the discovered senses and conducted a user study to evaluate the effectiveness of each representation method with the actual retrieval performance of user sense selections.

The third method, concept feedback, utilizes ConceptNet, an on-line commonsense knowledge base and natural language processing toolkit. As opposed to ontologies and other knowledge bases, such as WordNet and Wikipedia, ConceptNet is not limited to hyponym/hypernym relations and features a more diverse relational ontology as well as a graph-based knowledge representation model, which allows to make more complex textual inferences. First, we conducted simulation experiments by expanding each query term with the related concepts from ConceptNet, which demonstrated a considerable upper bound potential of tapping into a knowledge base to overcome the problem of the lack of positive relevance signals in the initial retrieval results for difficult queries. Second, we proposed and experimentally evaluated heuristic and machine learning based methods for selecting a small number of candidate concepts for query expansion. The experimental results on multiple data sets indicate that concept feedback can effectively improve the retrieval performance of difficult queries both when used in isolation as well as in combination with pseudo-relevance feedback.
To My Amazing Parents
Acknowledgements

At the end of this five year long journey through the doctoral program, I would like to remember and thank all the people, who were an essential part of it and made it possible. First, I would like to thank the Department of Computer Science Admissions Committee, who gave me, a graduate from a smaller and much less well-known university, a chance to grow and self-improve as a person and as a researcher by being part of one of the world’s top Computer Science programs. Second, I would like to thank my advisor, Professor ChengXiang Zhai, for his invaluable knowledge, research experience, scientific insight, advice and guidance, while I was making my first, sometimes awkward steps as a researcher. Growing up is not an easy and mistake-free process. Without his patience and support, I would have never been able to reach the end. I would also like to thank other members of my thesis committee for their helpful feedback about this work. Third, I am profoundly grateful for the opportunity to learn from the world-class scholars at UIUC and work with the world-class researchers during my internships at Yahoo! Labs and Microsoft Research: Paul Bennett, Susan Dumais, Ryen White, Jaime Teevan and Pranam Kolari. I am thankful for their mentorship and being great examples to strive for. I would also like to thank the UIUC TIMan group members for their participation in the user studies for parts of this work as well as Yerkin Abdildin and Maria Stone for being good friends and helping me with various things while I was at UIUC. And, finally, nothing would have ever been possible without the unconditional love and emotional support from my wonderful parents, Sergey Kotov and Irina Kotova. I am what I am and where I am only because of them and this work is a dedication to their kindness and wisdom.
# Table of Contents

List of Tables .................................................................................................................. viii

List of Figures .................................................................................................................. ix

Chapter 1 Introduction ........................................................................................................ 1
  1.1 Exploratory queries and question-based feedback ................................................... 3
  1.2 Ambiguous queries and sense feedback ................................................................... 4
  1.3 Under-specified queries, vocabulary gap and concept feedback ............................. 6

Chapter 2 Related work ....................................................................................................... 8
  2.1 Lexical ambiguity in IR ............................................................................................. 8
    2.1.1 Interactive feedback ......................................................................................... 15
    2.1.2 Pseudo-relevance feedback .......................................................................... 20
    2.1.3 External resources ......................................................................................... 23

Chapter 3 Question-based feedback ..................................................................................... 29
  3.1 Introduction .............................................................................................................. 29
  3.2 General idea ............................................................................................................. 30
  3.3 Implementation ....................................................................................................... 31
    3.3.1 Parsing and question generation .................................................................... 31
    3.3.2 Formal definition ........................................................................................... 33
    3.3.3 Indexing ......................................................................................................... 35
    3.3.4 Question ranking ........................................................................................... 35
    3.3.5 Question generation ....................................................................................... 37
    3.3.6 Question-based feedback .............................................................................. 39
  3.4 Experiments .............................................................................................................. 39
    3.4.1 Dataset and queries ....................................................................................... 41
    3.4.2 Judgments ....................................................................................................... 42
    3.4.3 Metrics ........................................................................................................... 42
    3.4.4 Evaluation of ranking .................................................................................... 43
    3.4.5 Evaluation of feedback .................................................................................. 43
    3.4.6 Detailed analysis ............................................................................................ 45
  3.5 Summary .................................................................................................................. 46

Chapter 4 Interactive sense feedback .................................................................................. 47
  4.1 Introduction .............................................................................................................. 47
  4.2 General idea ............................................................................................................. 49
  4.3 Formal definition ..................................................................................................... 50
  4.4 Sense detection ....................................................................................................... 51
    4.4.1 Contextual term similarity matrix construction ......................................... 52
    4.4.2 Sense detection algorithm ............................................................................. 54
  4.5 Sense presentation ................................................................................................. 54
4.6 Experiments ................................................................. 55
   4.6.1 Datasets and experimental setup ............................... 56
   4.6.2 Upper-bound performance ..................................... 57
   4.6.3 User study .......................................................... 60
   4.6.4 Examples of discovered senses ................................ 62
   4.6.5 Error analysis ..................................................... 63
4.7 Summary ................................................................. 64

Chapter 5 Concept feedback ............................................... 65
   5.1 Introduction .......................................................... 65
   5.2 Heuristic expansion ................................................ 67
      5.2.1 Path finding .................................................... 69
      5.2.2 Random walk .................................................. 69
   5.3 Learning-based expansion ........................................ 69
      5.3.1 Model ............................................................ 69
      5.3.2 Features ......................................................... 69
   5.4 Experiments .......................................................... 71
      5.4.1 Experimental setup and datasets ............................ 71
      5.4.2 Upper-bound performance .................................. 71
      5.4.3 Edge weighting ................................................ 72
      5.4.4 Learning-based expansion .................................. 74
      5.4.5 Comparison of methods ..................................... 75
   5.5 Summary ............................................................... 76

Chapter 6 Conclusions .................................................... 77
References ................................................................. 79
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Examples of syntactic patterns and sentences matching them</td>
<td>34</td>
</tr>
<tr>
<td>3.2</td>
<td>Matching documents, instances and generated questions for a sample query</td>
<td>37</td>
</tr>
<tr>
<td>3.3</td>
<td>Non-parametric ranking of questions for a sample query</td>
<td>37</td>
</tr>
<tr>
<td>3.4</td>
<td>User judgments for different query types</td>
<td>42</td>
</tr>
<tr>
<td>3.5</td>
<td>Performance of different ranking functions</td>
<td>43</td>
</tr>
<tr>
<td>3.6</td>
<td>User behavior with respect to different question types</td>
<td>45</td>
</tr>
<tr>
<td>4.1</td>
<td>HAL space for the sentence “the effects of pollution on the population”</td>
<td>53</td>
</tr>
<tr>
<td>4.2</td>
<td>Statistics of the experimental datasets</td>
<td>56</td>
</tr>
<tr>
<td>4.3</td>
<td>Comparison of the upper-bound performance of sense feedback with the baselines on all topics and collections</td>
<td>59</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparison of the upper-bound performance of sense feedback with KL-divergence retrieval model (KL) and model-based pseudo-feedback (KL-PF) on difficult topics. * indicates statistically significant difference relative to KL (95% confidence level), according to the Wilcoxon signed-rank test. † indicates statistically significant difference relative to KL-PF (95% confidence level), according to the Wilcoxon signed-rank test.</td>
<td>59</td>
</tr>
<tr>
<td>4.5</td>
<td>Number of difficult (D) and normal (N) topics improved by pseudo-feedback (KL-PF) and sense feedback (SF) in different datasets</td>
<td>60</td>
</tr>
<tr>
<td>4.6</td>
<td>Fraction of the queries (in percentages), for which the users selected the optimal sense of the optimal term (in boldface) and the optimal term, but not necessarily the optimal sense (in parenthesis).</td>
<td>60</td>
</tr>
<tr>
<td>4.7</td>
<td>Retrieval performance of user sense selections on difficult topics with respect to MAP, depending on the sense presentation method. Performance of the baselines is shown in the first two rows of the table.</td>
<td>61</td>
</tr>
<tr>
<td>4.8</td>
<td>Automatically discovered senses for the term “stealth” in the query “stealth aircraft”</td>
<td>62</td>
</tr>
<tr>
<td>4.9</td>
<td>Automatically discovered senses for the term “cancer” in the query “radio waves and brain cancer”</td>
<td>63</td>
</tr>
<tr>
<td>5.1</td>
<td>Features for ranking the expansion terms. Baseline feature set is designated in the BL column with .</td>
<td>70</td>
</tr>
<tr>
<td>5.2</td>
<td>Statistics of the datasets used for experiments</td>
<td>71</td>
</tr>
<tr>
<td>5.3</td>
<td>Comparison of the upper-bound performance of concept feedback (CF) by varying the the size of the expansion context with KL-divergence retrieval model (KL) and model-based pseudo-relevance feedback (KL-PF) on difficult topics</td>
<td>72</td>
</tr>
<tr>
<td>5.4</td>
<td>Number of times the best expansion term was connected to the expanded query term with the relation of each type</td>
<td>73</td>
</tr>
</tbody>
</table>
List of Figures

3.1 Example of a dependency tree ................................................................. 32
3.2 Compressed dependency tree (syntactic pattern) ......................................... 32
3.3 Dependency tree of an example sentence .................................................... 33
3.4 Fragment of a question-answers list for the query “bill clinton” ......................... 40
3.5 Precision@n for all feedback and non-feedback questions ................................. 44
3.6 nDCG@n for all feedback and non-feedback questions ..................................... 44
3.7 Distribution of clicked, interesting, and relevant questions over question lengths . 45
4.1 Performance of sense detection methods by varying the size of the HAL sliding window ................................................................. 58
4.2 Performance of sense detection methods by varying the interpolation parameter α (the name of the sense detection method is before the hyphen and the similarity measure is after the hyphen). ................................................................. 58
4.3 Retrieval performance of user sense selections for all the queries in AQUAINT, depending on the value of interpolation parameter α ................................................................. 62
5.1 Comparison of performance of different learning-based expansion methods on the AQUAINT dataset ........................................................................ 74
5.2 Comparison of performance of different learning-based expansion methods on the ROBUST04 dataset ........................................................................ 75
5.3 Comparison of performance of heuristic and learning-based expansion and pseudo-feedback methods on the AQUAINT dataset ........................................................................ 75
5.4 Comparison of performance of heuristic and learning-based expansion and pseudo-feedback methods on the ROBUST04 dataset ........................................................................ 76
Chapter 1

Introduction

Since search systems are mediators between humans and information resources, effective user interaction is a critical component of information retrieval (IR) process. The model and the role of user interaction in the IR process has been evolving in parallel with the development of retrieval models and broadening of our understanding of users’ search behavior. The focused and specific nature of the information needs of early search systems users led to the adoption of the textual query-driven search interface and the query-results interaction model, which remain a standard up to these days. Although, on one hand, such interaction model and search interface can save user efforts, on the other hand, it implicitly encourages the searchers to provide less information about their search requests in the queries. In case of the majority of simple information needs, omitting certain aspects of them may not exert any negative influence on search results. As a result, the assumption that the users are able to formulate good initial keyword queries (i.e. queries that return at least some relevant results) largely shaped the development of additional interaction mechanisms on top of the traditional query-results interaction model of IR, such as explicit and implicit feedback. Explicit relevance feedback is based on the idea of asking the searchers to mark the relevant information objects in the initially retrieved results, in order to update the search system’s model of their information need and retrieve more relevant documents. Implicit relevance feedback takes into account indirect evidence of users recognizing positive relevance information in search results, such as gaze position, mouse movements and clicks. Therefore, the success of the traditional interaction model as well as explicit or implicit relevance feedback relies on the searchers’ ability to form effective initial queries, which requires the searchers to:

- have simple or well-defined information needs and specific search goals;
- know the properties of the collection being searched and its domain coverage;
- know the vocabulary of the domain of the search problem.

Although these preconditions are generally valid in the collection-based IR, the advent of the World Wide Web has to some extent violated each one of them.
In particular, since the World Wide Web allows to address a virtually unlimited spectrum of information needs, Web searchers not always have specific search goals and often pose exploratory queries, aiming at researching a certain broad topic. Although such queries may return many relevant documents, they are typically scattered across millions of non-relevant results matching the broad query terms. The cognitive burden of scanning the long lists of the retrieved information units renders problematic employing the traditional query-results interaction model as well as explicit and implicit relevance feedback. In addition to that, neither the traditional interaction model nor any other existing supplementary interaction mechanism on top of it offer any support to the users in trying to understand the dimensions of a query topic and formulate the new queries to focus on any specific dimension.

Even when the users have specific information needs and pose relatively focused queries, the quality of search results may still be negatively affected by the two fundamental natural language phenomena: polysemy (the capacity of a word to have multiple meanings) and synonymy (multiple words designating the same concept). The effect of polysemous (or ambiguous) queries on retrieval results is somewhat comparable to that of exploratory queries, since the search results for ambiguous queries are often dominated by the non-relevant documents, which match the polysemous query terms used in the sense that was not intended by the searcher. Although the problem of ambiguity is less acute when the collection being searched covers a limited number of domains and the possibility of surface matching of the wrong senses is naturally eliminated, the World Wide Web, however, is the ultimate information resource, which covers all existing domains of human knowledge.

Synonymy has to do with the problem of vocabulary mismatch, which occurs when the authors of the potentially relevant documents and the searchers use different terms to refer to the same concepts. In particular, this problem often arises when non-professional users perform domain-specific searches and are not closely familiar with the vocabulary of the domain of the search problem. The most common examples of such domains are legal and medical searches. In general, a query is called difficult for a particular retrieval model, if all or most of the top-ranked documents retrieved with this model are non-relevant. The users are usually unaware of the underlying problem that makes a query difficult and existing search systems offer no support to the users in trying to improve the search results, which either include too many potentially relevant documents or no relevant documents at all. Although many users are aware that the quality of search results can be improved by reformulating a query, finding the right query formulation can be a fairly difficult and time consuming process. There also exist information needs, which are inherently difficult to formulate as keyword queries.
How the query should be refined varies, depending on the reason why it was not successful. Methods to establish the cause of difficult queries and predict the query performance have been extensively studied in previous work [14] [108] [22] [18]. The present thesis proposes three novel interactive feedback methods aiming to help the users improve the quality of retrieval results when the traditional interaction model fails: question feedback [49] for exploratory queries, sense feedback [50] for ambiguous queries and concept feedback [51] for vocabulary mismatch. All three proposed methods share the common idea of leveraging user interaction to find the optimal query reformulation and are based on the following interaction model. After a user poses an exploratory or difficult query, a search system steps in and generates a set of natural language refinement questions and presents them along with the initially retrieved results. If a user clicks on a refinement question, the system generates a reformulated query corresponding to the clicked question and retrieves a new set of results. We believe that such interaction model takes the best of both worlds. On one hand, a search system can leverage the collection in the absence of positive relevant documents in the top search results to determine the optimal (from the retrieval perspective) candidates for query reformulations and present those candidates in a user-friendly way as natural language questions. On the other hand, the users can leverage their intelligence and world knowledge to recognize and select the reformulation, which best matches their information need. This new interactive model can be envisioned as a problem solving process, in which a search system actively collaborates with the users by helping them to explore their information needs, correct problematic queries and ultimately achieve their search goals. At the same time, the question-based interaction model is intended to be complimentary to the traditional ranked list presentation of search results and the users can always decide when to use the one or the other.

In the following sections, we introduce the general ideas behind each of the proposed interactive feedback methods.

1.1 Exploratory queries and question-based feedback

Users, who don’t have a specific information need and just want to explore a particular topic, typically formulate a broad query consisting of one or several key aspects of the topic and then browse the initial search results to determine further directions for exploring it. For example, if a user is researching the biography of John Kennedy, the easiest and most straightforward way to do so for a person, who is used to keyword-based search, would be to pose a query, such as “john kennedy”, “kennedy” or “jfk”. However, such a query is likely to return a large number of potentially relevant documents, including those that cover the facts about JFK, in which the user is not interested. In addition to that, the users have to browse, select and
read the documents in the result set in order to find out what specific facts about the JFK’s biography draw their attention. Although employing search results diversification and filtering strategies may potentially reduce this burden, searchers posing exploratory queries would presumably prefer the search system to also inform them about the dimensions of the query topic, which are not present in search results, and prompt them for exploring those directions. In order to effectively address the exploratory information needs, we propose question feedback, a novel retrieval framework, in which the retrieval system automatically generates clarification questions in response to exploratory queries. Since asking questions is the fastest and the most natural way to obtain information for human beings, almost all queries posed to search systems typically correspond to some underlying questions. For instance, in the above example, the underlying question that has caused a user to search, can be as broad as “Who is John Kennedy?” or as specific as “When was Kennedy sworn as the President of the United States?”. Accurate determination of the questions underlying the information need may substantially improve the quality of search results and the usability of search interfaces. The question-based feedback strategy has the following benefits:

- instead of having the searchers explicitly reformulate their queries, clarification questions save user efforts by placing the query terms into specific contexts and, thus, allowing to narrow down the scope of the queries and improve the quality of search results. This way, clarification questions can be considered as a mechanism to correct imprecise queries;

- automatically generated questions can also suggest the new directions for exploring the search results. From this point of view, clarification questions can be considered as an interactive mechanism for exploring a particular domain;

- questions provide references to specific facts and, since they are generated based on the system’s internal information repository, can always be answered precisely. From this point of view, the clarification questions can be considered as a way to organize search results by shortcuts to specific answers.

The retrieval framework for implementing the question-guided feedback is presented in more detail in Chapter 3.

### 1.2 Ambiguous queries and sense feedback

Lexical ambiguity is a fundamental property of natural language can be of two types: syntactic and semantic. Syntactic ambiguity is caused by the differences in syntactic categories of words (e.g., the term “stop” can be used both as a noun and as a verb). Semantic ambiguity has to do with differences in meaning and is caused
by polysemous words or words having multiple senses. Lexical ambiguity negatively affects the quality of retrieval by decreasing precision. The difficulty of lexical ambiguity resolution (or sense disambiguation) varies greatly depending on several factors. When a query is sufficiently long, other terms in the query may serve as effective disambiguation clues due to the collocation effects [52]. This problem may be partially resolved by asking the users to put their queries in context (i.e. use phrases, rather than single query terms). Krovetz et al. [52] argue, however, that it might not be always possible to provide phrases, in which the word occurs only with the desired sense and this requirement might also place a significant cognitive burden on the users. Automatic disambiguation, however, proved to be very challenging, since even humans cannot perform it with perfect accuracy and mixed results have been reported.

One may also argue that the initial retrieval results for ambiguous queries may include some relevant documents and, thus, the query ambiguity can be resolved indirectly by using two strategies. The first strategy consists in employing a sophisticated interface or a search result diversification mechanism to represent the initial retrieval results in such a way that all possible interpretations for the query topic are easily identifiable. The second strategy involves engaging the users in some form of relevance feedback by asking them to indicate the relevant documents among those that are already retrieved and performing query expansion with terms extracted from the selected documents. Since both approaches rely on the assumption that the relevant documents are among the top-ranked retrieved results, they are likely to be ineffective when the senses of a polysemous query term correspond to substantially different numbers of relevant documents and the sense that a user has in mind is in fact a minor sense in the collection being searched. In this case, the top-ranked retrieved results will likely correspond to the major sense of a query term and may not contain any relevant documents corresponding to its minor sense. Therefore, the users may not be able to provide any positive signals, which can be interpreted by explicit or implicit relevance feedback approaches. Consequently, designing a feedback method that will be effective for the queries that are both ambiguous and difficult is a theoretically and practically important problem, particularly in the domains, where short and ambiguous queries prevail, such as Web search. To address this issue, we propose a novel concept of interactive sense feedback. Sense feedback is aimed at improving the poor initial search results that were caused by lexical ambiguity, by interactively clarifying the intended sense of an ambiguous query term. Senses are clarified with automatically generated clarification questions in the form of “Did you mean {ambiguous query term} as {sense label}?”, where a sense label can be a single term or multiple terms representing a sense. Sense feedback is presented in more detail in Chapter 4.
1.3 Under-specified queries, vocabulary gap and concept feedback

The complexity of an information need is essentially determined by the number and nature of concepts and aspects that constitute it. For example, a search goal of finding a list of cardiovascular diseases has one aspect, whereas searching for doctors in Chicago, who specialize in cardiovascular diseases, involves three different aspects. When formulating a query, users need to translate their original information need into a small set of keywords and attempt to capture all aspects of their information need by providing at least one keyword for each aspect. Therefore, the performance of a particular query depends on how all aspects of the information need are represented in the query. No matter how effectively all aspects are represented, certain amount of information is inevitably lost during the process of translation from the information need to the actual query.

In addition to being ambiguous, natural language is also inherently redundant. Multiple synonyms can be used to refer to the same concept and the terms that a searcher uses to describe an aspect of the information need may be different from the terms that were used by the authors of relevant documents. This problem is known as a vocabulary mismatch problem (or vocabulary gap) and it negatively affects the quality of retrieval by decreasing recall. It is because of the vocabulary mismatch problem that simple matching between the query and documents in the collection is unlikely to produce the acceptable retrieval results. Query expansion is a standard recall-enhancing technique designed to overcome the problems of differing vocabularies and partially specified information needs by selecting the query terms to expand and adding the new terms or phrases, which are associated with the terms being expanded, to the initial query. Typical sources of term associations for query expansion can be static and already existing at the time of query, such as query logs, external lexico-semantic resources and statistical thesauri constructed from the corpus, or dynamic, such as the top-ranked documents from the initial retrieval, which can be either selected automatically (pseudo-relevance feedback) or interactively by the users (explicit relevance feedback). All approaches using dynamic sources of expansion terms share the common problem that they are relying on the assumption that the initial retrieval results include some relevant documents, which can be used as a source of expansion terms. Therefore these approaches are not applicable to the case of difficult queries. While using query logs or statistical co-occurrence thesauri constructed through global analysis of the document collection allows to avoid dependence on the initial retrieval results, the vocabularies of these resources are limited by the collection and may simply not contain effective expansion terms for a particular query.

The main difficulty in effective application of automatic query expansion lies in the correct identification
of underrepresented aspects of the information need and selecting the right number of candidate expansion
terms. If only a few automatically identified expansion terms are added to the query, there is a possibility
that some effective expansion terms will be missed and the retrieval output is unlikely to be substantially
improved. On the other hand, when the query vocabulary is substantially altered, the advantages gained
from some useful added terms might be lost because of the noisy terms and topic drift.

The goal of concept feedback is to automatically select a small number of highly effective candidate ex-
pansion terms, which are conceptually related to the query terms. Concept feedback, utilizes ConceptNet, an
on-line commonsense knowledge base and natural language processing toolkit. As opposed to ontologies and
other knowledge bases, such as WordNet and Wikipedia, ConceptNet is not limited to hyponym/hypernym
relations and features a more diverse relational ontology as well as a graph-based knowledge representation
model, which allows to make more complex textual inferences. First, we conducted simulation experiments
by expanding each query term with the related concepts from ConceptNet, which demonstrated a consider-
able upper bound potential of tapping into a knowledge base to overcome the problem of the lack of positive
relevance signals in the initial retrieval results for difficult queries. Second, we proposed and experimentally
evaluated heuristic and machine learning based methods for selecting a small number of candidate concepts
for query expansion. The experimental results on multiple data sets indicate that concept feedback can
effectively improve the retrieval performance of difficult queries both when used in isolation as well as in
combination with pseudo-relevance feedback. Concept feedback is presented in more detail in Chapter 5.

In the following chapters, we provide an overview of the related work and then examine each of the
proposed feedback methods in detail.
Chapter 2

Related work

Lexical ambiguity and vocabulary mismatch are two fundamental problems in information retrieval. While all of the previously proposed methods to address lexical ambiguity did not involve any interaction with the user, researchers have actively experimented with interactive methods to overcome the problem of vocabulary mismatch. Query expansion through relevance feedback is a traditionally used method to overcome the problem of vocabulary mismatch and improve recall of information retrieval systems. Relevance feedback is the process of modifying the original query after the initial retrieval results have been generated. Information retrieval research has explored two major directions for performing relevance feedback: automatic (or pseudo-relevance) feedback and interactive (or explicit) feedback. There is no consent in the research community about which type of feedback is more effective. The main argument in favor of automatic feedback is that systems can process much more data than users can potentially examine and, thus, can make more informed decisions about which expansion terms to select. The main argument in favor of interactive feedback is that users have direct control over the criteria for relevance and, thus, should be able to make better decisions about which terms are more effective for expansion. Both automatic and interactive feedback have been traditionally approached by capturing lexico-semantic relations between the vocabulary terms and using them to select related terms for expansion. Lexico-semantic relations can be extracted either from the entire collection being searched (or some subset of it, such as the top-ranked documents from the initial retrieval results) or from external resources. In this chapter, we overview major lines of previous related work on addressing the problem of lexical ambiguity and improving the quality of search results through relevance feedback: interactive feedback, pseudo-relevance feedback and using external resources for query expansion.

2.1 Lexical ambiguity in IR

Methods to improve the quality of retrieval results by reducing the negative impact of lexical ambiguity have been studied for many years. Mixed results have been reported and this research direction proved to be very challenging. There are two major lines of work along this direction.
Understanding ambiguity

The first line is aimed at understanding the nature of lexical ambiguity in IR. This direction has been started by the work of Krovetz and Croft [52], who conducted a series of experiments to examine the quantitative aspects of lexical ambiguity in information retrieval test collections and determine its influence on retrieval performance. They made several interesting observations, regarding the nature of ambiguity. First, they tried to gain a better understanding of the relationship between word frequency and ambiguity. Zipf [113] first pointed out that the number of senses of a word is strongly correlated with the square root its frequency. Krovetz and Croft [52] attempted to incorporate sense weighting into a retrieval function and observed a relatively small improvement in retrieval effectiveness. They explained this result by the fact that, in general, most of the query terms appear in a relatively small number of documents, since the queries in the standard retrieval collections tend to be very specific. At the time of their study, it was largely assumed that queries posed to retrieval systems are typically sentence-like statements, expressing the information need in details. Secondly, they found out that there is a very strong correlation between the meaning of terms in a query, the meaning of the same terms in a document and relevance judgments. In addition to that, highly ambiguous terms occur relatively infrequently in IR test collections and sense disambiguation failures are more likely to happen in non-relevant documents, than in relevant ones. In other words, word senses provide a clear distinction between relevant and non-relevant documents. They attributed this phenomenon to the effects of word collocations and the distribution of senses in the corpus. Since highly ranked documents tend to match on a number of words in the query, ambiguous query words generally match only on the correct sense in such situations. For example, although the word “bat” is ambiguous, the query “bat echolocation” is unlikely to retrieve the top ranked documents, referring to sports equipment. Another important point is that not all words are equally worth of being disambiguated. Some words can be very ambiguous, but because some senses occur much more frequently than others, those words can be considered “relatively unambiguous” in practice. Krovetz and Croft also pointed out that query terms that are worth of disambiguation are either uniformly ambiguous (i.e., words, whose sense distribution is not skewed) or terms with a skewed sense distribution, but used in their minority sense in the query. As a result, they concluded that achieving benefits from applying disambiguation methods may be dependent on how successful a sense-aware IR system is in discriminatingly applying them, rather than on the accuracy of disambiguation methods per se.

Analyzing the results of Voorhees [96], Sanderson [85] introduced artificially created pseudo-words into standard retrieval collections to simulate ambiguity. A pseudo-word is created by assigning two or more word senses, for example “banana” and “door”, to one, “banana/door”. Having introduced artificially ambiguous terms into a particular collection, he measured the retrieval performance using the modified collection and
compared the results against the baseline of using the original collection. He found that retrieval based on very short queries (one or two terms) was more affected by ambiguity than based on longer queries, essentially confirming the existence of collocation effects pointed out in [52]. His experimental results also indicated strong negative impact of disambiguation errors on retrieval effectiveness, which he suggested might be the main cause of negative results in [96]. Sanderson [82] later summarized three key factors that affect the effectiveness of using WSD methods in IR. Firstly, skewed sense distributions and collocation effects are the main reasons why ambiguity has a limited impact on IR performance. Secondly, in order to benefit from disambiguation, a particular WSD method should be highly accurate. In particular, in [81], Sanderson concluded that improvements in IR effectiveness from using automatic disambiguation methods can be observed only if those methods can deliver the disambiguation accuracy close to that of humans (i.e., above 90%). Finally, dictionary and thesaurus based methods have not been shown to offer substantial improvements to IR effectiveness, when used in isolation. Therefore, broader semantic groupings might be required to achieve higher accuracy. Later Sanderson [83] examined the ambiguity of Wikipedia with respect to queries containing proper nouns, such as titles, names etc. He pointed out that studies of ambiguity in information retrieval have been hampered by the lack of standard test collections, containing ambiguous queries. Across virtually all test collections used in IR research, topics have a single interpretation, which is explicitly defined in the topic descriptions and/or narratives and implicitly defined in relevance judgments. In order to overcome this problem, Sanderson proposed a method [83] to automatically create collections for exploration of lexical ambiguity. By analyzing the works of Sanderson, Schütze et al. [86] identified that successful application of disambiguation to information retrieval is problematic in the following cases:

- A query is not a specific statement of the information need (broad and generic);
- A query contains terms that are optimal discriminators between relevant and non-relevant documents;
- Query terms that are disambiguated are either low-frequency terms or medium and high frequency terms with a majority sense that is much more frequent than the other senses.

**Automatic sense disambiguation**

The second line aims at developing methods to perform automatic sense disambiguation during retrieval. Despite years of research, there is still no consensus within the IR research community about what kind of information is most useful for automatic sense disambiguation. Depending on how the authors define the notion of a sense, there are two major views of the role of disambiguation methods in IR. Within the first view, the sense of a word is defined as its intrinsic semantic property, corresponding to the high-level
concepts, denoted by the word lexeme. This view assumes that the correct and comprehensive specification
of the word sense requires full knowledge about the world and can only be provided in the form of a manually
created dictionary or thesaurus. Early disambiguation approaches, such as Krovetz and Croft [52] along with
other disambiguation research of the early 1990s, largely adopted this approach and mainly relied on these
type of external resources. The basic idea behind thesaurus based approaches is to compute the distribution
of thesaurus classes for each document, by incrementing the counts of thesaurus classes, to which each word
in a document belongs. The most frequent thesaurus classes are then used as a bias for disambiguation.
Lesk [57] proposed a simple algorithm for disambiguation that uses a machine-readable dictionary to count
the overlap between the words used in the definitions of senses. For example, the word “pine” can have two
senses: a tree, or sadness (as in “pine away”) and the word “cone” may be a geometric structure, or a fruit
of a tree. Lesk’s algorithm computes the overlap between different senses of “pine” and “cone”, and finds
that the senses corresponding to “tree” and “fruit of a tree” have the most words in common. Wilks [100]
performed similar experiments, but rather than counting the overlap between definitions, all the words in
the definition of a particular sense of some word were grouped into a vector. In order to determine the sense
of a particular word in a sentence, a vector of words from the sentence has been compared to the vectors
constructed from sense definitions. The word is assigned the sense, which corresponds to the most similar
vector. The idea that senses of a word can be discriminated by analyzing the contexts, in which the word
occurs was first proposed in the work of Weiss [99] and received a more thorough examination later on.
Weiss’s approach is based on the idea that disambiguation can be performed using a set of syntactic rules
or patterns, which take into account word co-occurrences within windows of varying sizes. Since the early
works on the subject, word collocations have been shown to be an effective source for ambiguity resolution.
Collocations can be of two types: general word co-occurrences and co-occurrences within a phrase. Despite
being potentially ambiguous when considered in isolation, words usually have a very specific meaning when
they are used as parts of phrases or co-occur within a certain context. For example, the words “bat”,
“ball”, “pitcher” and “base” are all ambiguous and can be used in a variety of contexts, but collectively
they indicate a single context and have particular meanings. Voorhees [96] conducted the first large scale
study of a word sense disambiguation system applied to the topics and documents of 5 test collections.
Words were disambiguated and the retrieval effectiveness of applying an IR system to those collections
was compared to the effectiveness of the system searching on the collection without disambiguation. The
result of the indexing procedure is a vector, in which some of the terms represent word senses, instead of
original words. WordNet synsets are first transformed into broader semantic groupings by mapping from
synsets to one or more classes (called “hoods”), representing senses. If many words from a particular hood
co-occur with an ambiguous word, then this word is disambiguated as belonging to the sense, associated with the hood. If a word has multiple senses, but its contextual words do not belong to any of the senses, then no disambiguation decision can be made. Quite unexpectedly, Voorhees concluded that the use of disambiguation methods reduced retrieval effectiveness. Kim [46] proposed a root sense tagging approach for coarse-grained disambiguation. A root sense is one of the 25 unique terms associated with the root synsets of each of the WordNet’s noun hierarchies. In their approach, each noun in both documents and queries is classified into one of the candidate root senses by considering the neighboring context word, having the highest mutual information with the word being classified. Liu et al. [62] proposed several heuristics for disambiguating query terms that used adjacent query terms and WordNet. Sussna [92] tried a variety of approaches, based on minimizing the objective function, which was defined based on semantic distance between the nodes in WordNet. The distance function takes into consideration several heuristics, associated with the relative location of the nodes in the hierarchy. The relatedness between the topics, represented by the nodes, is captured by the weights of the edges forming the shortest path, connecting the nodes, so that shorter distance corresponds to greater relatedness. The disambiguation hypothesis is that, given a set of terms, occurring near each other in the text, each of which might have multiple meanings, correct senses are those that minimize the overall distance between the nodes in a network. Gonzalo et al. [30] converted the manually sense tagged Semcor 1.6 corpus into an IR test collection to evaluate retrieval, compared to a gold standard disambiguated corpus. Their results demonstrate an 11% increase in performance using the sense data in Semcor over a purely term based model. All of the research described in this section makes a common assumption that at the time an ambiguous query is submitted to a search engine, it will be known somehow, which sense of each ambiguous word was intended by the person, who issued the query. Such information might be worked out from the query itself (if the query is detailed enough), from a profile of the user held by the search engine, from information on the context of the search, click data from past searches, or simply by asking the user to clarify their information need. Based on the advances in understanding the role of ambiguity in retrieval made by previous work, Stokoe et al. [91] later performed retrieval experiments on the TREC WT10G Web corpus with an idea that the benefit from using WSD is most likely to be observed in domains, where short queries prevail. Their learning-based disambiguation algorithm was trained and evaluated using Semcor 1.6 [53] and used statistical co-occurrence, collocation information and sense counts from WordNet as disambiguation features. They experimented with a number of disambiguation strategies, but were unable to find a more effective technique than applying each of the knowledge sources (co-occurrence, co-occurrence and sense frequency) in a stepwise fashion. First, they identified all possible senses of a target word, by using the sentence where the target word occurred as a context window. If the context window
contains any collocates, indicative of a particular sense, which are learned from Semcor, the target word is labeled with that sense. If no collocates are observed in the context, an attempt to classify the target word based on co-occurrences is made. If the target word can’t be classified by using neither collocates, nor co-occurrence data, it is tagged based on the frequency statistics in WordNet. The reliance on sense frequency as a fall-back strategy makes their algorithm behave as traditional tf-idf weighting in cases of assigning sense based on weak assumptions. In general, they presented their disambiguation strategy as a trade-off of high precision techniques, when there is sufficient training data to perform reliable sense assignment, and traditional sense frequency statistics, when the training data is sparse and there is a risk of serious negative effects of inaccurate disambiguation on performance of the IR system. To compare the performance of traditional and sense-aware retrieval, they produced four retrieval runs: traditional tf-idf ranking, sense frequency based ranking (sf-idf) without disambiguation, tf-idf ranking using stemming and sf-idf ranking based on disambiguating all terms in the corpus, which have the same stem as a query term.

All thesaurus and dictionary based approaches, however, share the problem of coverage, since specific domains usually exhibit rare and specialized meanings, which may not be covered by generic lexical resources. In addition to that, the cost of manually constructing and maintaining a dictionary can be prohibitively high. The second view assumes that the sense of a word is determined by various contextual clues, such as its syntactic role and nearby context, rather than being its predefined property. A sense of a word, in this case, can be considered as a group of similar contextual usages of the word. Schütze et al. [86] proposed a method to derive the knowledge needed for disambiguation directly from the text collection in the form of an automatically generated thesaurus, which is essentially a term co-occurrence matrix, where each word is represented by a vector formed from the counts of its neighboring words in the corpus. In other words, two semantically related words are likely to have similar contextual vectors. Since the co-occurrence matrix can be high dimensional and sparse, dimensionality reduction techniques (such as singular value decomposition) can be applied to make computation more tractable. In [86], a context vector for a word is produced by summing the thesaurus vectors of the words that co-occur with it. The context vector characterizes the local topic of an individual word occurrence. Schütze et al. [86] proposed to consider senses as directions in the vector space, which can be found by partitioning the set of context vectors into regions of high density using a clustering algorithm. They achieved their best experimental results by allowing a word to be tagged with up to three senses and combining word and sense ranking. Their experimental results were one of the few at the time that demonstrated the positive effect of applying WSD methods to IR. It is important to note, however, that the context of information retrieval places several important constraints on disambiguation methods. The first constraint is that the consistency of disambiguation of both the document and query terms is more
important than accuracy, which is the main goal of general purpose disambiguation methods, developed in the Natural Language Processing community. The second constraint is that flexible disambiguation, which assigns several possible senses to a given word, is more preferable for information retrieval than traditional strict disambiguation, assigning only the most probable sense to any given term. The third constraint is that in the information retrieval context, a complete, dictionary-quality disambiguation is generally not required. Dictionaries often make very fine-grained distinctions between senses, which might not necessarily be important for correct disambiguation. For example, the word “stock” has 17 different senses in the WordNet. Given the fact that even humans cannot determine the correct sense of the word with 100% accuracy in different contexts, fine-grained automatic disambiguation is even more likely to cause errors, which could potentially result in disastrous effects on the overall quality of retrieval.

As has been pointed out in previous work, one of the key problems to successful integration of sense disambiguation into information retrieval is defining the appropriate sense representation. Such representation should be broad enough to abstract away the variability of raw text and sufficiently fine-grained to enable correct sense disambiguation. Method using existing thesauri and lexical hierarchies such as WordNet can be significantly limited in their coverage. Therefore, ideally, concept representations should be learned from the document collections. Concept hierarchy is an example of the sense representation structure that has been successfully used in document summarization. The majority of methods to construct concept hierarchies are based on using heuristic techniques. In particular, Sanderson and Croft [84] proposed to construct subsumption hierarchies, which establish term dependencies by calculating conditional probabilities of pairs of terms. The term \( t_1 \) is said to subsume the term \( t_2 \) if the textual windows in which \( t_2 \) occurs are a subset, or nearly a subset of the windows, in which \( t_1 \) occurs. A window could be an entire document or a smaller unit of text. Since subsumption is a pairwise relationship, its complete specification requires calculating conditional probabilities for all pairs of candidate terms. Once all individual subsumption relationships are found, the hierarchy is constructed in a bottom-up fashion. Lawrie et al. [56] proposed a formal method to construct multi-document summaries by finding topic terms and organizing them into a hierarchical structure. In the context of summarization task, topic hierarchies are typically constructed in such a way that a summary consists of the terms that are strongly predictive of the rest of the vocabulary. There are several approaches to estimating the strength of relationships between the terms for constructing topic hierarchies. The first one uses measures of co-occurrence, such as mutual information. The second one takes into account how some of the terms in the vocabulary can predict other terms, as measured by conditional probability. Topic terms are the terms, which are good at predicting other terms. Each level of the topic hierarchy consists of the topic terms that cover the same vocabulary as the terms in the previous levels and further specify
the higher level topics. The method proposed in [56] first recasts the language model as a graph, in which the vertices represent the terms and edges are weighted by the conditional probabilities in the language model and uses a greedy approximation to the dominating set problem to find the set of terms that have the maximal predictive power and coverage of the vocabulary.

Sense feedback, proposed in the present thesis, addresses the problem of ambiguity in IR from a conceptually new perspective. Instead of attempting to automatically disambiguate the ambiguous query terms, it analyzes the collection to determine the collection-specific senses of each query term and presents the discovered senses to the searchers, who can select the right sense, if necessary. To the best of our knowledge, the proposed method is the first method for interactive sense disambiguation.

2.1.1 Interactive feedback

Interactive feedback methods typically generate and present to the users the potential query reformulations in some form along with the retrieved documents and allow the users to choose a particular reformulation. If the user decides to reformulate a query, new results and new suggestions are presented and the process continues until the user is satisfied with the results or gives up searching. Most of the previously developed retrieval systems, offering interactive feedback functionality, present potential expansion terms as a simple list. In general, choosing an appropriate set of terms to present to a user as suggestions for query refinement remains a difficult problem, since there are typically hundreds of terms that are potentially relevant to an information need and without some structure imposed upon the terms, it might be hard for a user to interpret this information. Much work has been conducted examining the effectiveness of various techniques for selecting the terms to display for term relevance feedback since the potential benefit of interactive term relevance feedback is still related to the quality of the terms suggested by the system. No matter how many terms a user chooses for expansion, if all the terms are of poor quality, then such feedback is unlikely to increase the quality of retrieval. During the three HARD TREC participants have experimented with a variety of techniques and interface features for providing and eliciting term relevance feedback from users. In general, results have been mixed, and no participant achieved exceptional performance with any technique or interface. Below we provide a brief overview of methods and techniques for selecting and presenting terms for interactive relevance feedback.

Term selection

Fowkes and Beaulieu [26] showed that searchers prefer interactive query expansion when dealing with complex queries. Magennis and Van Rijsbergen [64] conducted an experiment aimed at measuring the effectiveness
of users in interactive query expansion by comparing the best possible query expansion decisions to those of the real users. Their conclusion was that most real users tend to make sub-optimal judgments about the potential utility of query terms. Ruthven [78] attempted to determine how good a user’s query term selection would have to be to surpass the retrieval effectiveness of automatic strategies for query expansion. Comparing the best interactive query expansion decisions with automatic expansion, he concluded that interactive query expansion has a potential to be a more stable technique, as it improves more queries, and a more effective technique, as it results in the highest average precision. Ruthven also proposed three potential guidelines that could be given to the users in order to help them maximize the utility of their interactive query expansion: “select more terms”, “trust the system” and “use semantics”. The first guideline has to do with the fact that the average number of terms in the best query expansion decisions is similar to the average number of terms in the good and poor decisions. Intuitively, the more expansion terms the users select, the more evidence the retrieval system can have regarding their information need. The second guideline advises the users to give priority to those terms that are most strongly suggested by the system, based on the ranking in [74]. The third guideline has to do with experimentally determined fact that users tend to ignore those expansion terms, which appear to have been suggested for purely statistical reasons. Moreover, a user study in [78] indicates that users cannot always easily identify semantic relationships between their information need and expansion terms. Even in cases when users can identify several potential semantic relationships, they cannot easily determine which semantic relationships are going to retrieve more relevant documents. Users also tend to select the terms that are semantically related to the topic description of queries, rather than to the retrieved relevant documents, if semantic connection of those terms to query topic is not entirely clear. Spink [90] found that the users’ written question statements about the query topic are the most effective sources of terms for query expansion. In particular, 38% of expansion terms came from the user question statements and these terms on average retrieved about 82% of the relevant documents. Based on these results, Spink suggested that IR interfaces should encourage users to use their own knowledge as a source of terms for query expansion. Harman [34] provided preliminary experimental results aimed at discovering a method to filter a large number of terms provided by relevance feedback to several small subsets of terms, that could be presented to the users in an interactive feedback session. The experimental interactive interface consisted of three windows: one with relevance feedback terms, one with variants of the original query terms and one with the terms, derived from a thesaurus. Tan et al. [93] proposed a method for interactive term feedback in the language modeling framework. As opposed to the interactive feedback methods, which ask the users to judge the relevance of entire documents in the result set, the idea behind term feedback is to present a reasonable number of individual terms to the users and ask
them to judge the relevance of each term or directly specify their probabilities in the query model. Judging a set of terms of reasonable size is easier and less time consuming than judging the entire documents. They proposed to cluster pseudo-feedback documents in order to ensure sufficient representation of all topics by feedback terms and use multinomial distributions associated with clusters to update the query language model. In their experiments, they presented all feedback terms in a single batch, but pointed out that a better strategy would involve presenting a small number of terms first and updating a set of terms as users make their judgments at each step of the interactive feedback cycle.

**Term presentation**

Several researchers experimented with alternative presentations of expansion terms. Koenemann and Belkin [47] showed that giving the users more control over how expansion terms are added to the query can increase retrieval effectiveness and user satisfaction. Anick [3] analyzed the search log sessions for two groups of users interacting with variants of a web search engine: a baseline group that was given no terminological feedback and a feedback group, to whom twelve refinement terms have been presented along with the search results. Although the analysis showed no difference between the two user groups based on the overall measure of retrieval effectiveness, Anick found out that there exists a subset of users who can make an effective use of terminological feedback on a continuous basis. Joho et al. [44] conducted user studies to evaluate a method to present the expansion terms as a hierarchy of nested menus, in which each lower level menu specifies a particular expansion term in the higher level menu. They did not observe any significant differences in retrieval performance between the list and menu hierarchy presentations of expansion terms, although they pointed out that users on average tend to select more terms from the menu hierarchy. Yang et al. [106] studied the usability of several relevance feedback interfaces that allow users to mark terms, phrases, and documents and submit passages from documents as relevance feedback. Wu et al. [101] explored a cluster-based interface for relevance feedback and found that, while users preferred this type of interface over a list display, there were no difference in retrieval performance. Belkin [6] argued that users can understand and learn about their information needs only through interaction with the documents. Specifically, Belkin states that “interaction with texts implies at least the possibility of an unpredictable, and therefore unspecifiable change in the condition which led to the interaction in the first place (e.g., the information need)”. With respect to term relevance feedback, this indicates that terms may no longer be useful, since the query, on which they are based, may no longer be appropriate. This also suggests that as users interact with the retrieved documents, they may identify, recognize or realize potentially useful query expansion terms. Kelly and Fu [45] suggested that observance of contextual occurrences of query expansion terms can stimulate
users to think about their information needs and this can help users identify additional terms to add to their queries. They experimented with three different term relevance feedback interfaces, each one of which utilized a different method for displaying the contexts, in which the query expansion terms occurred. The first interface displayed only a list of twenty feedback terms, suggested by the system based on the criteria, outlined in [75], without any context and users were asked to mark the check-boxes next to the terms they wanted to add to their queries. The second interface displayed a list of the twenty suggested feedback terms plus sentences, in which these terms appeared, and users could add feedback terms to the query by marking the check-boxes next to them. The third interface displayed only sentences, containing contextual occurrences of suggested expansion terms and a text box, in which users could enter their own query expansion terms by examining the presented sentences. Although they observed large differences in the length of the queries, expanded by adding feedback terms through each of the three interfaces, rather unexpectedly all queries performed similarly, regardless of their length. Overall, users tend to add more terms to the query by using the text box, rather than by marking the suggested terms and those terms generally improve the precision of search results, and specifically precision at the top of the retrieved document list. Based on these results, they concluded that without the appropriate context, it can be difficult for users to understand why the particular expansion terms were suggested and how those terms may be used to improve the retrieval results.

Faceted search is one of the prevailing search mechanisms in e-commerce web cites. It has recently attracted interest from the human-computer interaction community as a method for interactive navigation in complex information spaces. In a faceted search system, a query is a list of facet-value pairs that jointly specify the required properties of a matching document. Instead of waiting for the user to create structured queries from scratch, a faceted search interface allows the user to progressively narrow down the choices by selecting suggested query refinements presented as list of key-value pairs. The main problem in making the concept of faceted search applicable to general purpose retrieval is the development of an automatic mechanism to generate semantically differentiated facets and facet-values for presenting to a user at any given time. In faceted search, the search expression is divided into primary “search topic” and a “focus”. The initial query of a user defines a “search topic”, for which a faceted search system computes and presents a set of facets. After a user has examined search results and a corresponding list of associated facets and associated query suggestions, they can either click on an associated query and, thus, completely change the “search topic” and the list of facets or “focus” on existing topic, while keeping the same facet list. The “search topic” option allows the users to shift contexts completely when they encounter a topic of interest within the terminological feedback. Each time the user clicks on a facet value, the system constructs a new query by concatenating the original topic terms and a full facet phrase and retrieves a new set of results.
The facets, however, are not recomputed, leaving the user with a stable context, from which to choose other query refinements, so that users can continue exploring the repository by iterating through facet phrases one at a time, reshuffling the ranked result list with each click on a facet. Thus, in faceted search, information seeking is conducted as a sequence of small focused searches, each one building upon the knowledge, gained from previous results. Koren et al. [48] examined the possibility of several statistical modeling approaches for automatic generation of facet-value pairs. Their approach did not attempt to capture semantic relationships among facets and they also did not model the effects of changes in user interests over time. Anick et al. [4] proposed a linguistic approach for faceted interactive query refinement based on the observation that the key domain concepts usually participate in semantically related lexical compounds. The key idea behind their approach is that new concepts are often expressed not as new single words, but rather as concatenations of existing nouns and adjectives, which are called lexical compounds. They showed that statistical analysis of the terms, appearing within noun compounds in a document collection may expose some of the main topical threads running through the collection. Specifically, they defined the lexical dispersion hypothesis, stating that the number of different compounds that a word appears in within a given document set can be used as a diagnostic for automatically identifying the key concepts, or “facets” of the document set. They also described a web application, called Paraphrase Search Assistant to support interactive information seeking dialog. The method of Anick and Tipireni [4] was aimed at creating lexical hierarchies by identifying all phrases in a document set and finding the most frequent single words that occur in those phrases. They introduced the lexical dispersion hypothesis, which states that “the number of different compounds that a word appears in within a given document set can be used as a diagnostic for automatically identifying the key concepts in that document set.

Question feedback, proposed in the present thesis, is a conceptually new approach to interactive feedback. Instead of presenting the users with a set of terms, question feedback generates natural language questions that contextualize the query terms and allow the users to refine their information need. The advantages of question feedback are twofold. On one hand, the number of potential refinement questions for a query is typically much less than the number of potential feedback terms, which reduces the cognitive load on the user to select the refinement. On the other hand, question feedback uses linguistic analysis of the entire collection and, unlike term-based interactive feedback, does not depend on the initial retrieval results. In retrieval situations where users pose very short, broad and potentially ambiguous queries, this distinction is particularly important, since it is very likely that most of the documents retrieved in response to such queries will be irrelevant and, thus, the expansion terms extracted from those documents will be ineffective.
2.1.2 Pseudo-relevance feedback

Pseudo-relevance feedback (PRF) has been a widely used technique in IR. There are two general categories of methods for query expansion through pseudo-relevance feedback.

Local analysis

The first category involves extracting and adding to the original query a subset of terms that occur in the top ranked documents from the initial retrieval results. Such approaches are commonly referred to as local document analysis. The basic assumption behind the local document analysis is that the top-ranked documents in the initial retrieval results contain some useful terms, which can help discriminate relevant documents from non-relevant ones. One potential problem with this assumption is that a large fraction of the top ranked documents may be non-relevant, which can introduce noise into the feedback results. In general, the expansion terms can be extracted either by comparing the term distribution in the initially retrieved documents with the term distribution in the entire document collection (extracting most specific terms in the feedback documents) or by using the term distribution in the initially retrieved documents only. PRF based on local document analysis has been implemented in different retrieval models. Rocchio [76] proposed a PRF method based on the vector space model, in which the weights of the terms in the original query vector are adjusted by moving it closer to the centroid vector of the relevant documents and further away from the irrelevant centroid. Local context analysis (LCA) proposed by Xu and Croft in [103] combined passage-level retrieval with concept expansion, where concepts are single terms and phrases. However, in this work the weights of concepts were estimated in a heuristic manner and it is unclear how much the phrases helped over the single terms alone. In a language modeling framework [72], Lavrenko and Croft [55] proposed the concept of relevance models and Zhai and Lafferty [112] proposed two different approaches to update the query language model based on the feedback. One approach is based on minimizing the KL-divergence between the feedback language model and the query language model. The other approach assumes that the feedback language model to be extracted is the most distinctive from the language model of the entire document collection. Each feedback document is assumed to be generated by the topic model and the collection model and an EM algorithm is used to extract the topic model by maximizing the likelihood of the feedback documents. Both the relevance model and the feedback language model are then combined with the original query language model through linear interpolation. Although model-based feedback and relevance models have been shown to be effective pseudo-feedback techniques, they both tend to ignore important issues such as term dependence, proximity, and document structure.

Most of the early approaches for pseudo-relevance feedback utilized a simple bag-of-words document rep-
resentation and were based on the assumption that document terms and query terms are independent of each other and of the expansion terms. More recent approaches extended the language models by incorporating term relations or dependencies. Term relations can be considered from the two perspectives. On one hand, one may assume that relations exist only between the document terms or only between the query terms. In such a way, a sentence (either in a query or a document) is interpreted not only as a set of words, but also as a set of relations between the words. Under such interpretation, in order for a document to be considered retrieved for a query, it has not only to match the terms in a query, as in the classical language models, but also the relations between the query terms. Adopting this view, Metzler and Croft [68] proposed the idea of Latent Concept Expansion (LCE), which uses Markov Random Fields to model the dependencies between the query terms in order to extract sets of conceptually related terms for automatic query expansion. LCE is based on the assumption that when users formulate their original queries, they have a set of concepts in mind, but are only able to express a small number of them in the form of a query. The concepts that the users had in mind, but did not explicitly express in the query, are called latent concepts. LCE attempts to recover these latent concepts from the documents retrieved with the original query based on their co-occurrence with the concepts explicitly expressed by the terms in the original query. LCE has a limitation that it does not consider the dependencies between the query terms and expansion terms and assumes that a query term is equally associated with all the terms in a document (i.e. query terms and expansion terms are conditionally independent given a document). On the other hand, term relations can also be considered between query terms and expansion terms, so that indirect correspondence between the documents and a query can be inferred during query evaluation. Lang et al. [54] addressed this limitation of existing PRF methods by proposing to use Hierarchical Markov Random Fields to model various types of dependencies that may exist between the original query terms and expansion terms.

In addition to making the assumption about term independence, early pseudo-feedback methods considered all the top retrieved documents as relevant and used all the terms in those documents for expansion. Usually, however, not all of the top retrieved documents are relevant and not all the terms in the relevant documents are beneficial for expansion. He at al. [35] attempted to improve the expansion effectiveness at the document level by detecting good feedback documents. They classified all feedback documents using a variety of features such as the distribution of query terms in the feedback documents and proximity between the expansion terms and the original query terms in the feedback documents. Cao et al. [12] addressed this problem at the term level by using classification methods to select effective expansion terms from pseudo-relevant documents. They defined a set of features based on distribution of the expansion terms in both the retrieved documents and the entire collection, as well as co-occurrence and proximity of the expansion terms.
with the original query terms. In [105] a Bayesian logistic regression model was used to actively select the relevant documents. Another potential limitation of using all feedback documents for extracting expansion terms is that retrieval results typically consist of several topics and some of them may not correspond to any aspect of the initial query. Using the terms from those topics for expansion may cause the query drift. Several approaches which are based on clustering feedback documents have been proposed to overcome this problem. Liu and Croft [63] proposed to cluster the initially retrieved documents and used the discovered clusters to smooth the document language model. The potential of query-specific clustering performed on the initially retrieved results has been examined in several works [36] [94] [58] [95]. It was experimentally demonstrated that there exists an optimal cluster (i.e. a cluster, which if used for query expansion will always improve performance), but automatically finding such cluster is a difficult problem, which has not been yet addressed.

While existing automatic query expansion methods successfully increase recall for easy queries with high precision, they provide little benefit for hard queries with low precision. One relatively new research area is estimating query difficulty [108]. One suggested benefit of estimating query difficulty is that it could improve automatic query expansion by identifying easy queries. However, easy queries are less likely to need refinement. Wang et al. [98] proposed pseudo-feedback strategies for improving difficult queries, however, to the best of our knowledge, there has been no prior work on improving the difficult queries through interactive feedback. Therefore, the primary motivation behind interactive sense feedback is to overcome the limitations of existing pseudo-relevance feedback methods that make them ineffective for difficult and ambiguous queries.

Global analysis

The second category involves analyzing the collection to determine term associations within the collection on the basis of term co-occurrence. Such approaches are commonly referred to as global document analysis. In global document analysis a query is expanded with the terms that are strongly associated with the initial query terms [29] [110]. More complicated models, utilizing information other than simple term co-occurrence, can further improve the results. The query expansion method proposed by Qiu and Frei [73] for generalized vector-space retrieval models uses global term-term co-occurrence information to select the best expansion terms by ranking them according to the vector-space based similarity score of a term and the entire query. Carmel et al. [13] proposed to use lexical affinities extracted from the collection to automatically select expansion terms in such a way that the information gain of the retrieved document set is maximized. In the context of language modeling approach to retrieval, Bruza and Song [9] proposed a
method for augmenting the query language model with HAL-based information flows. Unlike traditional
pair-wise global term relationships, information flow is context-dependent. It is computed between a set
of terms and another term (for example, between “java, computer” and programming). Xu and Croft
[102] compared the performances of utilizing local and global document analysis for pseudo-feedback. Their
experiments concluded that local analysis is more effective than global analysis. In their approach, candidate
expansion terms are ranked by their co-occurrence correlations with the query and weighted by a constant
according to their rank. A potential problem with using global document analysis is a possibility of query
drift, which decreases precision. Although sense feedback uses global analysis for sense detection, it avoids
the problem of query drift since users select semantically coherent clusters of terms for expansion.

2.1.3 External resources

Early attempts to utilize lexico-semantic relationships include introduction of the concept of associative
retrieval. Associative retrieval is based on the association hypothesis, which was first formulated by van
Rijsbergen and states that “if an index term is good at discriminating relevant from irrelevant documents,
then any closely associated index term is also likely to be good at this”. In associative retrieval, knowledge
about associations among information items (terms, concepts or documents) is represented as a network, in
which information items correspond to the nodes and associations between them to the links connecting the
nodes. Constrained spreading activation [21] is a typical processing paradigm used in associative retrieval.
Spreading activation was first introduced into information retrieval by Salton [80]. Cohen [17] proposed a
method for assigning grants to scientific proposals that used constrained spreading activation on a network
of research topics. The main difficulty in applying early approaches to associative retrieval was the labor-
intensive process of manual construction of the association network between the documents and concepts.
Berger and Lafferty [7] later proposed a similar view of a query as a potential translation of the document
and proposed techniques for estimation of the translation probabilities using synthetic training data.

Automatic thesauri and WordNet

Researchers have also actively experimented with using various external resources for query expansion.
Most of the early approaches attempted to address the problem of synonymy by expanding a query with
the related terms from manually or automatically constructed thesauri. All thesaurus-based approaches for
query expansion can be classified into three major categories:

1. Based on manually constructed thesauri [97] [62];
2. Based on automatically constructed thesauri, which use statistical co-occurrence relations [15] [23] [73] [86];

3. Based on automatically constructed thesauri, which use head-modifier relations [31] [32] [37] [43] [77].

Both manually and automatically constructed thesauri are typically composed of a set of thesaurus classes, each of which corresponds to a set of closely related terms and represents a particular semantic category. The relationships contained in manually constructed thesauri, such as WordNet [69], are really between word senses rather than individual words, where each sense is represented by a set of synonyms. Therefore, in order to correctly expand the query with the terms from a manually constructed thesaurus, the retrieval system should not only correctly infer the sense of the query term being expanded, but also the senses of the terms that are used for expansion.

Several researchers proposed expansion methods based on using WordNet. Voorhees [97] experimentally determined the upper bound for query expansion using different strategies for selecting the related terms from WordNet. She annotated each query topic with WordNet synsets and manually selected the query terms for expansion. The original query terms and expansion terms were assigned different weights. The results for this experiment indicated that query expansion makes little difference to retrieval effectiveness, even with manual selection of the query terms to be expanded, if the original queries are relatively complete descriptions of the information need. On the other hand, lexico-semantic relations have the potential to significantly improve less well formulated initial queries, although an expanded query is unlikely to be as effective as an interactively supplied user query reformulation. She concluded that designing an automatic procedure for choosing the correct synonyms for expansion is a difficult task, since synonyms need to correspond to the correct sense of important concepts in the query and a poor choice is worse than not expanding at all. Liu et al. [62] proposed several heuristic methods for disambiguating and selecting candidate expansion terms using adjacent query terms and WordNet. Only the candidate terms that are globally correlated with the query terms were used for expansion. Shah and Croft [87] proposed heuristic methods for query term re-weighting and locating query terms to expand with WordNet synonyms with the goal of improving precision in the top document ranks. In order to improve the quality of expansion terms it is necessary to identify various query aspects. This process is usually referred to as query splitting. A trivial approach is to split a query into single words, although it does not sufficiently capture the aspects as many words change substantially from their individual meanings, once placed into a sequence. A more sophisticated approach uses clustering to identify multi-word aspects [109]. Intuitively, expansion terms that affect multiple aspects of the original query are more effective. Crabtree et al. [20] proposed a method for automatic query expansion of hard queries by identifying the query aspects which are underrepresented in the initial search results and expanding those
aspects with synonyms from WordNet.

In addition to demonstrating that using lexical and semantic relations separately can improve the quality of retrieval, researchers have also experimented with how different types of term relations can be used in combination, in order to overcome the problem of data sparsity that inevitably arises, no matter how large a source for each individual type of relations may be. Methods to combine different types of term relations have been proposed in the context of both vector-space and language modeling approaches to retrieval. In the context of vector-space models, Mandala et al. [65] proposed a method to combine three different thesaurus types for query expansion: manually constructed (WordNet), automatically constructed based on co-occurrence relations and automatically constructed based on head-modifier relations. The key assumption behind their approach is that each thesaurus type has different characteristics and hence their combination can provide a valuable resource for query expansion. Experiments have shown that the combined use of all three thesaurus type results better retrieval performance than using only one particular type of thesaurus. Bodner et al. [8] conducted similar experiments by combining WordNet and co-occurrence based thesauri for query expansion. In the context of language modeling approach, Bai et al. [5] proposed a method for query expansion by integrating term relationships explicitly into the query language model. They used document co-occurrence, HAL space co-occurrence, globally and locally computed HAL-space information flows as sources of term relationships. In Cao et al. [11] term relationships from co-occurrence statistics and WordNet were used to smooth the document language model, so that the probabilities of the related terms in the document model are increased. Collins-Thompson and Callan [19] proposed a Markov chain framework for query expansion, combining multiple sources of term associations, such as synonyms from WordNet, terms that share the same prefix when stemmed to the same root, terms co-occurring in a large Web corpus and terms co-occurring in the top retrieved documents. Given a small set of initial query terms, they constructed a network of related terms from different sources and used a random walk to estimate the likelihood of relevance for potential expansion terms and select the most related terms for query expansion.

Approaches based on manually constructed thesauri have several major limitations. First, due to the limited coverage, manually constructed general-purpose thesauri may not contain the related terms for certain domain-specific terms. Second, due to the possibility of query drift, thesaurus-based expansion methods can only succeed if the domain of a thesaurus closely corresponds to the domain of a collection. Co-occurrence based thesauri are free from the above limitations, since they are constructed based on the document collection. However, co-occurrence based thesauri may contain significant amount of noise and it is generally difficult to determine the appropriate size of the word window, within which to consider co-occurrence. Another drawback of automatically constructed thesauri is that any two words are considered
similar only if they appear together in the same document a certain number of times, which for some semantically related pairs of words, such as “astronaut” and “cosmonaut”, may not always be possible. Although head-modifier based thesauri derived from the corpus do not have such problems, words with similar heads and modifiers are not always good candidates for expansion.

**Knowledge bases**

Recently, the emergence of the Web and collaboratively edited general purpose (e.g. Wikipedia) and domain specific (e.g. MeSH) knowledge bases enabled access to the new sources of term associations for query expansion. Global analysis has been applied at the level of the entire Web and was shown to significantly improve performance over the algorithms using only local document analysis. In particular, Diaz and Metzler [24] demonstrated that using a high quality external corpus that is comparable to the target corpus can be as, if not more, effective than using the web for pseudo-relevance feedback in the context of relevance models. Yin et al. [107] proposed an expansion method, based on using a random walk on the query-URL graph generated from the web query logs and snippets provided by an external search engine. Their main assumption is that users submit various queries to express the same information need and, therefore, the query can be expanded using related query formulations (i.e. “the wisdom of crowd”). Fonseca et al. [25] proposed a concept-based query expansion technique for disambiguating web queries using search logs. Their disambiguation method is based on using association rules mined from the search logs to construct a query relations graph and identify cliques in the constructed graph. Their main assumption is that such cliques correspond to the high-level concepts, which can be used to determine the sense of an ambiguous query. The identified query concepts are then presented to the users, who can select the concept that they believe is the most relevant to the query. They did not propose a method to automatically label the presented concepts, rather a concept is composed of a set of past queries. In addition to selecting the relevant query concepts, for each selected concept users can indicate whether it is a synonym, specialization, generalization or association of their original query. Depending on the type of an expansion concept, different boolean operators are used to attach the expansion terms to the original query. They experimentally demonstrated that expanding ambiguous queries based on the feedback provided by the users through selection of a single most related concept can improve precision of web search results. Li et al. [59] used articles retrieved from Wikipedia by using the initial query to perform pseudo-relevance feedback. Their key finding was that although pseudo-feedback outperforms Wikipedia based expansion in terms of MAP, Wikipedia-based expansion performs better than pseudo-feedback according to the measures favoring difficult queries. They found that the queries which were hurt by PRF did not perform well in the initial retrieval and those queries
were improved by Wikipedia-based expansion. Xu et. al. [104] proposed a method to use Wikipedia for query-dependent expansion. They used a simple method to classify TREC queries into three categories, based on Wikipedia: queries about specific entities, which can be mapped directly to the entity page with the same title; ambiguous queries, which can be mapped to a Wikipedia disambiguation page and broader queries. Depending on the query type, they proposed to use several different strategies to generate pseudo-relevant documents. For ambiguous queries, their method first clustered the top 100 documents retrieved for the original query and considered the top-ranked cluster as the dominant sense for the query. The top ranked cluster is then compared to all the entity pages extracted from the Wikipedia disambiguation page, associated with an ambiguous query and the top matching entity page is chosen for query expansion. The terms in the expansion page are then ranked according to their TF/IDF scores and the top scoring terms are chosen for expansion. Han et al. [33] proposed a method, which uses Wikipedia to first detect the sub-topics of a query by finding coherent Wikipedia concept groups from search results and then organize search results using topic-driven clustering algorithm. Meij et al. [66] showed that discriminative semantic annotations of documents using domain-specific ontologies, such as MeSH, can be effectively used to improve retrieval. They proposed a two step process that extends pseudo-relevance feedback and uses a pivotal conceptual language. In the first step, the documents from an initial retrieval run are used to translate the textual query into a conceptual query model, which unambiguously represents the user’s information need at a different, higher conceptual level than the original query. This explicit conceptual representation can be used to suggest relevant concepts to the user or for matching a conceptual representation of the documents. In the second step, the conceptual model is translated back into the textual query model, since the textual representation of documents is more detailed than its conceptual representation. They used local analysis to obtain conceptual representation of a query and global analysis to translate the conceptual representation back into textual form.

WordNet and Wikipedia are the two most actively explored external resources for query expansion. However, the emergence of ConceptNet [61], a conceptually new knowledge base remained relatively unnoticed by the IR community. ConceptNet is presently the largest commonsense knowledge base, consisting of more than 1.6 million assertions about the real world, which, similar to Wikipedia, were gathered as simple sentences from a large number of on-line collaborators. Its framework is a semantic network. Nodes in ConceptNet correspond semi-structured natural language fragments (e.g., “food”, “grocery store”, “buy food”, “at home”), which represent concepts in the real world. An edge between two nodes represents a relationship between two concepts. As opposed to other ontologies, such as WordNet, ConceptNet is not limited to hyponym/hypernym relations and features a more diverse relational ontology of twenty relationship types,
such as causal, spatial and functional, which allows to make richer and more complex textual inferences. Although using other external lexico-semantic resources, such as WordNet or Wikipedia, can help to address the issue of vocabulary divergence between the queries and documents, a query can sometimes conceptually diverge from its relevant documents, rather than only in vocabulary. For example, a query “etude and polonaise” may be relevant to a document containing the word “piano”, since etude and polonaise are both common piano pieces. Establishing a relationship between such terms may require several inference steps, which is a relatively easy procedure in ConceptNet, since its knowledge is structured as a semantic network. The hierarchy of synsets in WordNet and vector space based knowledge representation in Wikipedia [28], on the other hand, are not well suited for multi-step inference. Hsu and Chen [38] investigated the usefulness of commonsense knowledge in ConceptNet for image retrieval by focusing on finding concepts related through spatial relationships. They used simple spreading activation constrained by following only spatial relationships and found that commonsense knowledge is deeply context-sensitive and suitable for precision-oriented tasks. Hsu et al. [39] compared the utilization of both WordNet and ConceptNet and for query expansion. They performed a simple expansion using spreading activation and compared the retrieved results in terms of discrimination ability and concept diversity. The goal of this study was to compare the two lexico-semantic resources, rather than optimize the retrieval performance by experimenting with different expansion strategies. The experimental results demonstrated that WordNet and ConceptNet can complement each other for the task of ad hoc retrieval. Queries expanded with WordNet have higher discrimination ability (i.e., expansion concepts from WordNet are usually more specific than those from ConceptNet), whereas queries expanded with ConceptNet have higher concept diversity (i.e., expansion concepts from ConceptNet usually co-occur with topical terms in relevant documents). They also demonstrated that the retrieval performance improves when expansion concepts are manually filtered to remove noise, but did not proposed any algorithms for automatic query expansion. Hsu and Chen [40] selected all immediate neighbors of query terms from both ConceptNet and WordNet as expansion candidates and used classification methods to select the best expansion terms. However, to the best of our knowledge, an extensive and systematic study of the feasibility of using ConceptNet for query expansion has not yet been conducted.

*Concept feedback* is an interactive query expansion method aimed to improve the precision of hard queries by finding terms that are conceptually related to the query aspects, which are underrepresented in the result set. Concept feedback leverages ConceptNet, which, as opposed to WordNet and Wikipedia, allows to make complex multi-step inferences to establish relations between the query terms and expansion terms at the conceptual level.

In the following chapters we discuss each of the proposed feedback methods in detail.
Chapter 3

Question-based feedback

3.1 Introduction

Presenting a ranked list of URL anchor texts and their associated snippets in return for a user query has become a standard interface for all major commercial search engines. While for most simple queries the ranked list-based presentation of search results is sufficient to easily find relevant documents, for more complicated queries it would take a user significantly more time to peruse a long list of returned documents and, potentially, reformulate the query multiple times. In order to understand why existing search interfaces often fail on certain types of queries and propose an improvement, we need a closer look at the very nature of search queries.

The search paradigm based on keyword queries assumes that search engine users have sufficient knowledge about the query domain and are able to find good differentiator terms to make their queries specific and precise. In reality, however, there is still a large number of queries, which are over- or under-specified, and it is often the case that the users are unable to find anything useful as a result of their first search, sometimes even after tedious perusal of document titles and snippets. This has to do with the fact that in their daily life people naturally tend to use verbose or imprecise statements to express their requirements and, thus, are not used to formulating artificial short string requests. According to [71], formulating natural language questions is the most natural way for search engine users to express their information needs. Unfortunately, state-of-the-art question answering systems cannot yet accurately answer arbitrary natural language questions posed by users. Moreover, in case of exploratory queries, users often do not have a clear search goal and want to simply explore a particular topic. Hence users cannot provide a clear question reflecting their information need, even if a search system were able to answer it. In this chapter, we propose a novel question-based feedback technique, in which a search system helps the users improve search accuracy for exploratory queries by generating clarification questions.

Ideally, questions should refine the query topic from multiple perspectives. For example, presented with the query “john kennedy”, an interactive question-based retrieval system can generate the following
questions: “Who is John Kennedy?”, “When was John Kennedy born?”, “What number president was John F. Kennedy?”, “Who killed President Kennedy?”. Each of the above questions can be considered as a clarification question, which puts the general query terms in a specific context. Our intuition is that by automatically generating clarification questions, an information retrieval system would enable the users to interactively specify their information need. Since the questions are generated based on the system’s internal information repository, they can always be answered precisely, which is not always the case with ordinary question answering systems. In addition to providing answers, which are guaranteed to be correct, this model of interaction also has the benefit of helping the users to quickly navigate to the information they are looking for, effectively eliminating the need to read the documents to locate it.

3.2 General idea

The idea of question-guided search comes naturally from the fact that a search for information is often motivated by the need for answering a question. Asking a well-formulated question is the fastest and the most natural way to express the search goal. However, the current search technologies cannot fully support a search interface, which is based entirely on free natural language question queries. Moreover, search engine users have already got used to the keyword-based search paradigm. In this work, we propose a method to augment the standard ranked list presentation of search results with a question based interface to refine initially imprecise queries.

A typical scenario for question-guided search is as follows. After a user types in initial keyword query, the automatically generated clarification questions can be presented next to the traditional ranked list of documents or any other search result presentation interface, should the system decide that a query requires further specification. Alternatively, users may press a button (e.g., “Guide Me”) and see the list of questions any time they want. In general, we envision that question-guided query refinement is likely to be very useful for exploratory search, especially for imprecise or ambiguous queries.

Clarification questions can be short (more general) or long (more specific) and should ideally be about different aspects of the query topic. Similar to documents in the classic relevance feedback scenario, questions place the query terms in a specific context, which may help the users find relevant information or initiate exploration of other topics. However, unlike the full-sized documents, questions are much shorter and hence require less time and effort from the users for reading and relevance judgment. In addition to questions, users may also be presented with short answers to them, when they point to a particular question. Users can also click on the question and be redirected to the document, containing the answer, for further information.
In this sense, questions can be considered as shortcuts to specific answers.

We also believe that questions can more naturally engage the users into a relevance feedback cycle. By clicking on the questions, users indicate their interest in a particular aspect of the query topic. Therefore, based on that signal, a search system can present the next set of questions and search results, by adding the terms in the clicked question to the current query to improve results. Although question-guided search can be used to supplement the results of any query, it may not be equally effective for all types of queries. Short, under-specified queries are the best candidates for refinement through questions. Since question generation algorithm is based on capturing syntactic relations between the terms, queries, containing named entities are well-suited for refinement through questions as well, since refining questions will allow to explore potential relations of the named entities in a query with other named entities in a corpus. Overall, question-guided search is a novel way of applying natural language processing methods to improve the usability of search. It seamlessly integrates lightweight search results navigation and contextual interactive relevance feedback into one retrieval framework.

3.3 Implementation

In this section, we demonstrate how the idea of natural language question-guided retrieval process can be implemented in a search engine. In order to experimentally evaluate the proposed idea, we have built a prototype of a QUestion-guided Search Engine, which we called QUSE. In the following sections, we consecutively focus on each individual component of the retrieval process: indexing, retrieval, ranking and feedback.

3.3.1 Parsing and question generation

Due to the fact that information contained in a sentence is represented not only by its basic lexical units (words), but also by syntactic relations between them, any natural language sentence can be phrased in multiple ways, even if the meaning conveyed by all the variants is identical. According to the linguistic theory of dependency grammars [67], any sentence can be represented as a set of dependency relations, which form a tree structure, usually called a dependency tree. A dependency relationship is an asymmetric binary relationship between a word, called the head (or governor, parent), and another word called the modifier (or dependent, daughter). Each term in a sentence can have several modifiers, but can modify at most one other term. The root of a dependency tree does not modify any other words. Verbs cannot modify any other constituents and, thus, are always the roots of dependency trees. For example, the dependency
structure of the sentence “John found a solution to the problem” is shown in Figure 3.1.

In the example sentence in Figure 3.1, there are six pairs of dependency relationships, depicted by the arrows from heads to modifiers. Each edge is labeled by the syntactic role of a modifier. For example, the label “subj” means that the modifier in this relation is the subject of a sentence.

In order to convert the sentences in a document collection into dependency trees, we used Minipar [60], a broad coverage dependency parser. Given an input sentence, Minipar returns its dependency tree, in which the nodes correspond to the terms in the sentence along with the syntactic and semantic labels assigned to them, and the edges represent the dependency relationships between the terms. Minipar also classifies proper nouns into semantic categories (names of people, organizations, geographical locations, titles, currencies), based on its internal dictionary.

If we consider only syntactic and semantic labels of the nodes in a dependency tree, disregarding the specific terms corresponding to the nodes, we will get a generalized dependency tree or syntactic pattern. Obviously, a syntactic pattern is a compressed representation of all dependency trees with the same structure. We will refer to the nodes of a syntactic pattern as slots. During indexing, slots are filled with the actual words from a matching sentence. When the semantic role of a constituent is important, it is specified after the syntactic label of a node. For example, node 1 of the generalized tree in Figure 3.2 has the label “subj:person”, which means that a parse tree or subtree can match this particular pattern, only if there is a node at that specific position, which is syntactically labeled as the subject of a sentence and semantically labeled as a proper name, designating a person.

![Figure 3.1: Example of a dependency tree](image)

![Figure 3.2: Compressed dependency tree (syntactic pattern)](image)
Dependency trees can be used to convert any nominative sentence (or part of it) into a question. The transformation of a nominative sentence into a question involves changes only to its syntactic structure, without any significant changes to its lexical content. The general idea behind the question generation algorithm is that we can index the instances of syntactic patterns in a document collection along with the terms filling the slots of these patterns and convert those instances into questions, according to the question generation templates. The algorithm to convert sentences into questions is illustrated with the following example sentence: “John went to school in Massachusetts”, the dependency tree of which is shown in Figure 3.3.

In particular, we can manually define the following question templates for the syntactic pattern in Figure 3.2:

- Where did \{1:stem\} \{0:stem\} \{2:term\} \{3:term\}?
- Who \{0:term\} \{2:term\} \{3:term\} \{4:term\} \{5:stem\}?

“Term” in the slot description of a question template means that when the actual question is generated from this template, the original form of the word from the corresponding slot of a syntactic pattern instance is used. “Stem” means that a morphologically normalized version of a word is used. Given our example sentence “John went to school in Massachusetts”, which matches the pattern in Figure 3.2, the following questions can be generated from the question templates above:

- Where did John go to school?
- Who went to school in Massachusetts?

Examples of other patterns, used in QUSE, along with the sample sentences, matching each of them, are shown in Table 3.1. Terms, filling the slots of pattern instances, are highlighted with numbered under-braces. Efficient algorithms for recognition of syntactic patterns are discussed in detail in [27].

### 3.3.2 Formal definition

Let \( D = \{d_1, d_2, \ldots, d_n\} \) be a collection of \( n \) documents, composed from a set of \( k \) terms \( T = \{t_1, \ldots, t_k\} \) and their stems \( T' = \{t_1', \ldots, t_k'\} \).
### Definitions

**Definition 1** Slot: given a set of syntactic labels $L$ and a set of semantic roles $R$, a set of slots $S$ is a subset of $L \times R$. A slot of a syntactic pattern is a relation $(l, r) \in S$, where $l \in L$ and $r \in R$.

Slots are parts of both the patterns and their instances. In the patterns, slots specify what kind of lexemes can match the pattern. In the instances, slots store the actual constituents of matching sentences and their stems.

**Definition 2** Syntactico-semantic pattern $P$ defines a structure on a subset of a set of slots $S$, given the relation of syntactic dependency. In other words, a syntactic pattern is a set of the ordered pairs of slots:

$$P = \{(s_i, s_j), \ldots, (s_k, s_m)\}$$

such that in each pair $(s_i, s_j)$, $s_i$ is a head of syntactic dependency relationship and $s_j$ is a modifier.

Let $\mathcal{P} = \{P_1, P_2, \ldots, P_M\}$ be a collection of $M$ syntactic patterns.

**Definition 3** An Instance of a Syntactic Pattern $I$ is a mapping $T \times T \rightarrow S$, where $S$ is a set of slots belonging to some pattern $P \in \mathcal{P}$.

An instance of a syntactic pattern occurs when a sentence in the corpus matches one of the syntactic patterns. An instance is stored and represented by pairs of words and their stems, which are filling the slots.

### Examples

<table>
<thead>
<tr>
<th>Syntactic Patterns</th>
<th>Matching Sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $(1:s(person)) \rightarrow (0:i) \rightarrow (2:mod) \rightarrow (3:pcomp-n(location)) \rightarrow (4:mod) \rightarrow (5:pcomp-n(date))$</td>
<td>Wilson lived in Columbia, South Carolina, the state capital, from 1870 – 1874, where his father was professor at the Columbia Theological Seminary.</td>
</tr>
<tr>
<td>2. $(1:s(person)) \rightarrow (0:i) \rightarrow (2:obj(person)) \rightarrow (3:mod) \rightarrow (4:pcomp-n(location)) \rightarrow (5:mod) \rightarrow (6:pcomp-n(date))$</td>
<td>In 1764, Adams married Abigail Smith at Weymouth, Massachusetts.</td>
</tr>
<tr>
<td>3. $(1:s(person)) \rightarrow (0:i) \rightarrow (2:obj) \rightarrow (3:mod) \rightarrow (4:pcomp-n(location))$</td>
<td>Kennedy had near-legendary status in Ireland, as the first person of Irish heritage to have a position of world power.</td>
</tr>
<tr>
<td>4. $(1:s(person)) \rightarrow (0:i) \rightarrow (2:pred) \rightarrow (3:mod) \rightarrow (4:pcomp-n(person))$</td>
<td>Voight is the father of actress Angelina Jolie (Angelina Jolie Voight is her birthname) and actor James Haven.</td>
</tr>
<tr>
<td>5. $(1:s(person)) \rightarrow (0:i) \rightarrow (2:pred) \rightarrow (3:mod) \rightarrow (4:pcomp-n(location))$</td>
<td>President Kennedy was assassinated in Dallas, Texas at 12:30 p.m.</td>
</tr>
<tr>
<td>6. $(1:s(location)) \rightarrow (0:i) \rightarrow (2:pred) \rightarrow (3:mod) \rightarrow (4:pcomp-n(location))$</td>
<td>Washington, D.C., formally the District of Columbia and commonly referred to as Washington, the District, or simply D.C., is the capital of the United States, founded on July 16, 1790.</td>
</tr>
</tbody>
</table>
of a matching pattern.

Definition 4 Context of a pattern instance includes the sentence, containing a pattern instance, and the sentences immediately before and immediately after it. The context of a sentence is saved to be later shown as an answer to the question generated from an instance.

The purpose of the context is to provide a short answer to the automatically generated question.

Definition 5 Question template is a subset of the set of ordered slots \( S \) of a syntactic pattern \( P \in \mathcal{P} \), perturbed and mixed with other terms in such a way that, when instantiated from an instance of a pattern, it conveys the semantics of a question.

3.3.3 Indexing

In question-guided search, the purpose of the index is to store the instances of syntactic patterns. The nature of syntactic patterns allows to use relational tables for storing them in the index. The most important parts of the index, used for question generation are the following relations:

- Dictionary of terms and stems \( V(id, term) \): \( id \) - the ID of a term or a stem; \( term \) - term or stem itself;
- Documents in the repository \( D(id, wcount) \): \( id \) - the ID of a document; \( wcount \) - number of words in a document
- Instances of syntactic patterns:

\[ I(iid, did, sid, pid, slid, tid, stid) \]

where \( iid \) is the ID of an instance; \( did \) is the ID of the document, where an instance occurred; \( sid \) is the ID of a sentence in the document, where an instance occurred; \( pid \) is the ID of the pattern, corresponding to an instance; \( slid \) is the number of the slot, which the term and its stem are filling; \( tid \) is the ID of the term, filling the slot of a pattern instance; \( stid \) is the ID of the stem, filling the slot of a pattern instance.

3.3.4 Question ranking

Similar to the traditional document-based retrieval model, the goal of question ranking methods is to determine and use as many useful heuristics (features) as possible to bring potentially interesting and relevant questions up to the top of the list of clarification questions, returned for a keyword query. Our approach to
question ranking is based on determining the position of a newly added question in the ranked list, according to several heuristics, numerically expressing the relative interestingness and relevance of questions.

Formally, given a set \( H = \{h_1, h_2, \ldots, h_n\} \) of \( n \) ranking heuristics (features), where each heuristic is a function \( h : \Theta \to \mathbb{R} \), mapping questions in the set \( \Theta \) into the real numbers (feature values), and the two questions \( \delta_1 = (h_1(\delta_1), \ldots, h_n(\delta_1)) \) and \( \delta_2 = (h_1(\delta_2), \ldots, h_n(\delta_2)) \), represented as \( n \)-tuples of feature values, a non-parametric question ranking function \( r \) is a binary function: \( \Theta \times \Theta \to \{0, 1\} \) on question pairs, such that, if \( r(\delta_1, \delta_2) = 1 \), then the question \( \delta_1 \) should be ranked above \( \delta_2 \) or, i.e., question \( \delta_1 \) is more relevant to the query than the question \( \delta_2 \), or \( \delta_1 \succ \delta_2 \).

Therefore, the ranking procedure is similar to the insertion sorting algorithm, where each new question is compared with the questions that are already in the list until a less relevant question is found or the end of the list has been reached. When such a question is found, a new question is inserted before it. It is important to note that, in such a setting, the order, in which the heuristics are applied, determines their relative importance for ranking. We applied the following ranking heuristics in the order, in which they are presented below:

**QT**: \( qt(\delta, q) \), the number of query terms that occur both in the query \( q \) and the question \( \delta \), generated from it. The motivation behind this heuristic is that the questions matching more query terms are potentially more relevant to the information need.

**PM**: \( pm(\delta, q, I) \), the number of query terms that occur both in the query \( q \) and the slots of the pattern instance \( I \), from which the question \( \delta \) was generated. The intuition behind this heuristic is that questions generated from instances that match more query terms are more specific, and, thus, are more aggressively guiding the users towards their search goals.

**DS**: \( ds(\delta, q, d) \), the retrieval score of the query \( q \) with respect to the document \( d \) that contains an instance of the pattern, from which the question \( \delta \) was generated. This heuristic allows to use the scores of traditional retrieval models (vector space, probabilistic or language modeling based) for question ranking. In our implementation, we used the popular Okapi/BM25 retrieval formula [88]:

\[
\begin{align*}
    s(q, d) &= \sum_{t \in Q, D} \ln \frac{N - df + 0.5}{df + 0.5} \times \frac{(k_1 + 1)tf}{k_3 + qtf} \times \frac{(k_3 + 1)qf}{k_3 + qtf} \\
    &\times \frac{d}{d + 0.5} \times \frac{q}{q + 0.5} \\
    &\times \frac{d}{d + 0.5} \times \frac{q}{q + 0.5}
\end{align*}
\]

where \( N \) is the total number of documents in the collection; \( df \) is the number of documents that contain a query term; \( tf \) is the term’s frequency in a document; \( qtf \) is the term’s frequency in a query; \( dl \) is the document’s length; \( avdl \) is the average length of a document in the collection.

We will illustrate our non-parametric approach to question ranking with the following example. Suppose
a user submits a query \( q = \{t_1, t_2, t_3\} \), which matches three pattern instances (query terms that are matching the slots of an instance are given in brackets for each instance) in two documents, such that the instances \( I_1 \) and \( I_2 \) occur in the document \( d_1 \) and the instance \( I_3 \) occurs in the document \( d_2 \). The retrieval score of the document \( d_2 \) with respect to the query \( q \) is greater than the score of the document \( d_1 \), \( ds(\delta, q, d_2) > ds(\delta, q, d_1) \). Six questions, which are summarized in Table 3.2, were generated from the instances \( I_1, I_2 \) and \( I_3 \). The query terms, contained in each question, are given in braces after each question.

<table>
<thead>
<tr>
<th>documents</th>
<th>instances</th>
<th>questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>( d_1 )</td>
<td>( I_1(t_1, t_2, t_3) )</td>
<td>( \delta_1(t_1, t_2, t_3) )</td>
</tr>
<tr>
<td>( I_2(t_1, t_3) )</td>
<td>( \delta_2(t_2) )</td>
<td></td>
</tr>
<tr>
<td>( I_3(t_2) )</td>
<td>( \delta_3(t_2, t_3) )</td>
<td></td>
</tr>
<tr>
<td>( d_2 )</td>
<td>( I_3(t_2) )</td>
<td>( \delta_4(t_2) )</td>
</tr>
<tr>
<td>( I_3(t_2) )</td>
<td>( \delta_5(t_1, t_3) )</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.2: Matching documents, instances and generated questions for a sample query

The final ranking of the sample questions in Table 3.2 by applying the non-parametric ranking heuristics

\[
H = \{qt(\delta, q), pm(\delta, q, I), ds(\delta, q, d)\}
\]

is shown in Table 3.3.

<table>
<thead>
<tr>
<th></th>
<th>qt</th>
<th>pm</th>
<th>ds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>( \delta_1 )</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2.</td>
<td>( \delta_3 )</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3.</td>
<td>( \delta_5 )</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>4.</td>
<td>( \delta_6 )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5.</td>
<td>( \delta_2 )</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>6.</td>
<td>( \delta_4 )</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 3.3: Non-parametric ranking of questions for a sample query

### 3.3.5 Question generation

In this section, we present an algorithm for generating a ranked list of clarification questions for keyword queries. Let \( I \) be a set of instances:

\[
I = \{(t_{11}, t_{11}\ell), \ldots, (t_{1i}, t_{1i}\ell)\}, \ldots, [(t_{m1}, t_{m1}\ell), \ldots, (t_{m\ell}, t_{m\ell})]\}
\]

of \( m \) syntactic patterns \( P \):

\[
P = \{(s_{11}, s_{12}, \ldots, s_{1i}), \ldots, (s_{m1}, s_{m2}, \ldots, s_{mi})\}
\]
obtained after indexing a document collection. Suppose a user poses an \( n \)-term keyword query \( q = \{t_1, t_2, \ldots, t_n\} \). Let \( r(\delta_i, \delta_j) \) be a ranking function defined on a set of ranking heuristics:

\[
H = \{qt(\delta, q), pm(\delta, q, I), ds(\delta, q, d)\}.
\]

The algorithm to generate a list of clarification questions \( \Theta \) ranked according to the ranking function \( r(\delta_i, \delta_j) \) is shown in Algorithm 1.

**Algorithm 1** Algorithm to generate a ranked list of clarification questions \( \Theta \) for a keyword query \( q \)

**Require:** Keyword query, \( q = \{t_1, t_2, \ldots, t_n\} \)

**Require:** Set of \( m \) syntactic patterns, \( \mathcal{P} \)

**Require:** Set of \( l \) instances of syntactic patterns, \( \mathcal{I} \)

**Require:** Ranking function \( r(\delta_i, \delta_j) \)

1: \( \mathcal{I}_0 \leftarrow \{I : \exists \delta, t \in Q \text{ and } t \in I\} \)
2: for all \( I, I \in \mathcal{I}_0 \) do
3: \( P \leftarrow \text{pattern}(I) \)
4: \( d \leftarrow \text{document}(I) \)
5: \( T \leftarrow \text{template}(q, P, I) \)
6: for \( i = 0 \) to \( |T| \) do
7: \( \delta[i] \leftarrow I[i] \)
8: end for
9: \( qt(\delta, q) = |\delta \cap q| \)
10: \( pm(\delta, q, I) = |q \cap I| \)
11: \( ds(\delta, q, d) = BM25(d, q) \)
12: for \( i = 0 \) to \( |\Theta| \) do
13: \( \Theta[i] \leftarrow \delta_i \)
14: if \( r(\delta_i, \delta_i) = 1 \) then
15: \( \text{insert}(\Theta, \delta_i) \)
16: end if
17: end for
18: end for

Algorithm 1 operates as follows. First, a set of pattern instances \( \mathcal{I}_0 \) with at least one query term is obtained by querying the index (line 1). Next, for each instance in \( \mathcal{I}_0 \), the corresponding pattern and the document, where the pattern instance occurred, are obtained (lines 3 and 4, respectively). Templates of the questions, which are focused on the query terms and include other slots of the instance, are obtained in line 5. Next, the slots of the question templates are filled with the terms from the corresponding slots of the pattern instance (lines 6 and 7). Once a question is generated from the template, the values of the ranking features are calculated in lines 9-11: the number of query terms, occurring in the generated question, is obtained in line 9; the number of query terms occurring in the slots of the pattern instance, from which the question \( \delta \) was generated, is obtained in line 10; the score of a document containing the pattern instance \( I \), from which the question \( \delta \) was generated, is obtained in line 11. Finally, the current list of questions is
being searched (lines 12-17) for the question, which should be ranked below the question \( \delta \), according to the ranking function (line 14). If such a question is found at position \( i \), the newly generated question is inserted at this position (line 15), pushing other questions towards the end of the list.

### 3.3.6 Question-based feedback

Our method for automatic question generation provides a natural way for implicit relevance feedback. Indeed, when a question is clicked, it can be assumed that a user is interested in this question. Suppose a user submits a query: \( q = \{t_i, \ldots, t_j, t_k, \ldots, t_n\} \) and, after viewing the ranked list of questions \( \Theta \), clicks on the question \( q(t_j, t_k, t_l, t_m) \), which was generated from the instance \( I = \{t_p, \ldots, t_j, t_k, t_l, t_m, \ldots, t_q\}, I \in \mathcal{I} \). The key idea for question-based relevance feedback is that when a user clicks on the question, containing non-query terms, a system can interpret this action as an indication of the direction of interest, and all the non-query terms in the question can then be added to the original query to enrich the representation of information need. Specifically, the original query can be augmented with the terms from other slots of the same instance of a syntactic pattern that was matched with the original query. Formally, a new query is \( q' = q \cup f \), where \( f = I \setminus q \); for the example above, \( q' = \{t_i, \ldots, t_j, t_k, \ldots, t_n\} \cup \{t_p, \ldots, t_q\} \).

For example, suppose a user submits a query containing a person’s name and clicks on the question, generated from the pattern instance, involving a location and a date. Both the location and the date can now be added to the original query. The new query can then be re-submitted to the search system to generate an updated question list and search results, achieving the effect of feedback.

### 3.4 Experiments

In this section, we present the results of an experimental evaluation of a prototype search system with the question-guided functionality (i.e., QUSE) by a group of users. The evaluation is aimed at demonstrating the added value of the question-guided search process from the two major perspectives: easier and faster navigation in the search results and interactive feedback. Within the first perspective, the focus is on the quality of question generation (automatically generated questions should be grammatically correct) and ranking (relevant and interesting question should be presented first). The second perspective is related to how natural, interesting and interactive the question feedback is for the users (generated questions should encourage further exploration of the query topic).
3.4.1 Dataset and queries

We crawled, preprocessed and indexed a subset of Wikipedia, consisting of 3000 most viewed articles in 2009, combined with the biographic articles about the famous Americans. Such composition of the test collection allows the users to pose a variety of interesting exploratory queries. The test collection includes 19547 articles and its total size is around 300 megabytes. The indexer was configured to recognize and index the occurrences of 32 different syntactic patterns, some of which are presented in Table 3.1.

We designed a special evaluation interface for the system and opened it to the users for a week. The users, who participated in the evaluation, were a group of 20 engineering graduate students. We allowed the users to select the queries from a list of predefined queries or type their own queries directly into the search box. After submitting their own query or clicking on a link for a predefined one, the users were forwarded to a page with search results, which were organized into question-answer pairs. For each query, a maximum of 30 top-ranked questions, along with the answers, have been presented for evaluation. A snapshot of the sample result page is shown in Figure 3.4.

<table>
<thead>
<tr>
<th>W</th>
<th>I</th>
<th>R</th>
<th>Did you mean?</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>What did bill clinton do to h i l l a r y c l i n t o n ?</td>
<td>She was the first First Lady to hold a post graduate degree and to have her own professional career up to the time of entering the White House. In January 1993, Bill Clinton appointed Hillary Clinton to head and be the chairwoman of the Task Force on National Health Care Reform, hoping to replicate the success she had in leading the effort for Arkansas education reform. The recommendation of the task force became known as the Clinton health care plan, a comprehensive proposal that would require employers to provide health coverage to their employees through individual health maintenance organizations.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>What place was bill clinton in born?</td>
<td>His policies, on issues such as the North American Free Trade Agreement and welfare reform, have been described as centrist. Bill Clinton was born William Jefferson Blythe III in Hope, Arkansas. His father, William Jefferson Blythe Jr., was a traveling salesman who died in an automobile accident three months before Bill was born.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>What did clinton do to brady bill?</td>
<td>In his inaugural address he declared Clinton signed the Brady Bill into law on November 30, 1993, which imposed a five-day waiting period on handgun purchases. He also expanded the Earned Income Tax Credit, a subsidy for low income workers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Where was bill clinton inducted?</td>
<td>He has since recovered On May 1, 1998, Bill Clinton was inducted into the DeNoyes International Hall of Fame. On September 9, 2008, Bill Clinton was named as the next chairman of the National Constitution Center in Philadelphia, Pennsylvania.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>What was bill clinton named as?</td>
<td>On September 9, 2008, Bill Clinton was named as the next chairman of the National Constitution Center in Philadelphia, Pennsylvania. On September 9, 2008, Bill Clinton was named as the next chairman of the National Constitution Center in Philadelphia, Pennsylvania. His term began January 1, 2009.</td>
</tr>
</tbody>
</table>

Figure 3.4: Fragment of a question-answers list for the query “bill clinton”

Users were asked to provide their judgments regarding the well-formedness (column ‘W’ in Figure 3.4), interestingness (column ‘I’ in Figure 3.4) and relevance (column ‘R’ in Figure 3.4) of each question, by putting a check mark into the corresponding check box. We defined a well-formed question as a question, which is grammatically correct and meaningful to the user; an interesting question as a question, which is either unexpected or about some fact not previously known by the user, or if it generates interest in further

---

exploration of the question topic; and a relevant question as a question relevant to the topic of a query. We also explicitly clarified that some questions may be interesting, but not necessarily relevant, as well as some relevant questions may not necessarily be interesting. For example, if a user submits the query “clinton” and is willing to find some information about Bill Clinton, questions about Hillary Clinton are not relevant. However, among the questions about Hillary Clinton, there can still be questions interesting to the user.

The ’Answer’ column in Figure 3.4 was intended to help the users judge the interestingness and, especially, the relevance of questions. Well-formedness of a question is not related to its interestingness or relevance. A question can be well-formed, even if it is not interesting or relevant. Note that the questions in Figure 3.4 are presented as hyperlinks, which may be clicked on, should the user be interested in exploring the topic of the clicked question. After clicking on a question, the user is presented with another ranked list of feedback question-answer pairs, generated by issuing a reformulated (feedback) query. A maximum of 10 feedback questions have been presented for evaluation during each feedback cycle.

3.4.2 Judgments

After running the system for a week, we collected the user judgments of 2895 questions generated for 184 queries (63 non-feedback queries and 121 feedback ones). In order to get a more detailed picture of how the proposed retrieval framework performs on different types of information needs, we manually classified the collected queries into the three groups, which are listed below along with some sample real queries:

- **SQ** (short queries): short (one term only), underspecified and potentially ambiguous queries: e.g., “ford”, “paris”, “illinois”;
- **NQ** (normal queries): well-formed, generally unambiguous, exploratory queries: “michael jackson”, “bill gates”;
- **LQ** (long queries): long (three or more terms), very specific queries: “barry bonds babe ruth record”, “bush gulf war”;
- **FB** (feedback queries): queries, generated by the system, when one of the questions was clicked: “cher david letterman return”, “diagnose disease reagan ronald”.

The aggregated statistics of user judgments with respect to the absolute number (upper half of each cell) and the relative percentage (lower part of each cell) of clicked (C), well-formed (W), interesting (I), and relevant (R) questions to the total number (T) of questions, generated for the queries of each type, are shown in Table 3.4. All queries, regardless of the type, are designated as ALL.
<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>W</th>
<th>I</th>
<th>R</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQ</td>
<td>19</td>
<td>232</td>
<td>128</td>
<td>135</td>
<td>310</td>
</tr>
<tr>
<td></td>
<td>6.13%</td>
<td>74.84%</td>
<td>41.29%</td>
<td>43.55%</td>
<td>100%</td>
</tr>
<tr>
<td>NQ</td>
<td>99</td>
<td>940</td>
<td>421</td>
<td>606</td>
<td>1105</td>
</tr>
<tr>
<td></td>
<td>8.96%</td>
<td>85.06%</td>
<td>38.1%</td>
<td>54.84%</td>
<td>100%</td>
</tr>
<tr>
<td>LQ</td>
<td>11</td>
<td>216</td>
<td>122</td>
<td>85</td>
<td>270</td>
</tr>
<tr>
<td></td>
<td>4.07%</td>
<td>80.0%</td>
<td>45.19%</td>
<td>31.48%</td>
<td>100%</td>
</tr>
<tr>
<td>FB</td>
<td>0</td>
<td>987</td>
<td>709</td>
<td>463</td>
<td>1210</td>
</tr>
<tr>
<td></td>
<td>0.0%</td>
<td>81.57%</td>
<td>58.6%</td>
<td>38.26%</td>
<td>100%</td>
</tr>
<tr>
<td>ALL</td>
<td>129</td>
<td>2375</td>
<td>1380</td>
<td>1289</td>
<td>2895</td>
</tr>
<tr>
<td></td>
<td>4.45%</td>
<td>82.03%</td>
<td>47.67%</td>
<td>44.52%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Table 3.4: User judgments for different query types

There are several important conclusions, which could be made based on the analysis of Table 3.4. First, questions corresponding to the feedback queries have the largest proportion of interesting questions. This clearly shows the benefit of the question-based feedback strategy. Second, the overall question click-through rate greater than 3.33% indicates that the users clicked on at least one of the 30 questions presented for each non-feedback query. Third, relevance of the questions varies across different query types and is the highest for normal queries. Therefore, unambiguous queries generate relatively more relevant questions. The low precision of questions, generated by the long and feedback queries, can be explained by the more specific information need corresponding to those types of queries, and hence a smaller subset of potentially relevant questions/answers. Ambiguous queries naturally result in questions with lower precision. Finally, the well-formedness of questions is independent of the query type and is about 80% across all query types.

After taking a high-level look at the initial user judgments, we are now ready to move on to a more detailed analysis of all components of the question-guided retrieval process.

### 3.4.3 Metrics

Due to the fact that a set of questions, which can be potentially returned for a query, can be much larger than a set of documents, accurate ranking of questions in the question-guided search framework is very important. Since there may be many relevant questions and their usefulness to the users may vary, we distinguish different levels of usefulness of a question and use the Normalized Discounted Cumulative Gain or nDCG [42] to measure the quality of a ranked list of questions. The DCG at the $i$-th question is computed as:

$$DCG(i) = \begin{cases} 
G(i), & \text{if } i = 1 \\
DCG(i - 1) + \frac{G(i)}{\log_2(i + 1)}, & \text{otherwise}
\end{cases}$$
where \( G(i) \) is the grade of the \( i \)-th question \( \delta_i \) in the ranked list, which is computed as follows:

\[
G(i) =
\begin{cases} 
3 & \text{if } \delta_i \text{ is both interesting and relevant} \\
2 & \text{if } \delta_i \text{ is just relevant} \\
1 & \text{if } \delta_i \text{ is just interesting} \\
0 & \text{if } \delta_i \text{ is neither interesting, nor relevant}
\end{cases}
\]

Given a DCG vector \( V = (v_1, v_2, \ldots, v_k) \) computed for a list of \( k \) questions \( \Theta \) that are generated by some ranking method and the DCG vector \( I = (i_1, i_2, \ldots, i_k) \), which corresponds to the ideal ranking of the same question list \( \Theta \), a normalized DCG vector is \( nDCG = (v_1/i_1, v_2/i_2, \ldots, v_k/i_k) \).

### 3.4.4 Evaluation of ranking

In this section, we present the results of an experimental evaluation of different question ranking strategies described in Section 3.3.4 to determine the best performing non-parametric ranking function. First, we started with the ranking functions that include only one ranking heuristic \( qt, pm, ds \) at a time. Then, we kept adding additional heuristics to the best performing ranking function at each step to determine the best performing combination of ranking heuristics. The relative performance of different ranking functions is summarized in Table 3.5

<table>
<thead>
<tr>
<th></th>
<th>( r(pm) )</th>
<th>( r(qt) )</th>
<th>( r(ds) )</th>
<th>( r(pm, ds) )</th>
<th>( r(ds, pm) )</th>
<th>( r(pm, ds, qt) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP</td>
<td>0.8962</td>
<td>0.8823</td>
<td>0.8889</td>
<td>0.9080</td>
<td>0.8920</td>
<td>0.9083</td>
</tr>
<tr>
<td>MRR</td>
<td>0.2927</td>
<td>0.2895</td>
<td>0.2932</td>
<td>0.2968</td>
<td>0.2936</td>
<td>0.2969</td>
</tr>
<tr>
<td>Avg. NDCG</td>
<td>0.8765</td>
<td>0.8651</td>
<td>0.8841</td>
<td>0.8907</td>
<td>0.8850</td>
<td>0.8911</td>
</tr>
<tr>
<td>Prec@5</td>
<td>0.6435</td>
<td>0.6402</td>
<td>0.6543</td>
<td>0.6620</td>
<td>0.6533</td>
<td>0.6625</td>
</tr>
<tr>
<td>Prec@10</td>
<td>0.4755</td>
<td>0.4717</td>
<td>0.4761</td>
<td>0.4804</td>
<td>0.4761</td>
<td>0.4821</td>
</tr>
</tbody>
</table>

Table 3.5: Performance of different ranking functions

As follows from Table 3.5, the best performing non-parametric question ranking function is \( r(pm, ds, qt) \). This indicates that all three ranking heuristics are useful. The sequence of application of ranking heuristics in the best-performing ranking function also suggests that the questions, generated from the more specific patterns (those that match more query terms), should be ranked higher. This can be explained by fact that the users prefer more specific questions to the broader ones.

### 3.4.5 Evaluation of feedback

One of the key benefits of the question-based retrieval process is the possibility of contextual query expansion. We evaluated the effectiveness of question-based feedback by comparing precision@n (Figure 3.5) and
nDCG@n (Figure 3.6) across all feedback and non-feedback questions. Non-feedback questions were presented after the users submitted their initial queries or clicked on the predefined query. Feedback questions were generated and presented after the users clicked on one of the initial questions and the updated initial query has been re-submitted to the system. Since the updated query includes the original query terms, the clicked question may appear in the feedback questions, however it may not necessarily be ranked high enough to be presented to the users, since the updated query also generates other questions, which could be ranked higher than the clicked one.

![Figure 3.5: Precision@n for all feedback and non-feedback questions](image)

The steep slope of the precision curve for the feedback questions in Figure 3.5 indicates that the question-based feedback aggressively refines the information need by bringing up a small number of both highly relevant and interesting questions to the top of the question list.

![Figure 3.6: nDCG@n for all feedback and non-feedback questions](image)

Figure 3.6 further confirms our conclusion that the question-based feedback effectively improves question ranking by bringing the highly relevant and interesting questions to the first three positions of the ranked list.
3.4.6 Detailed analysis

The proposed novel retrieval framework opens up many interesting opportunities for exploration of user search behavior. In this section, we aim to analyze user preferences regarding different types of questions. In particular, we focus on the two specific questions:

- is there any relationship between the head word of a question and user judgments/click-through rate?
- is there any relationship between the length of a question and user judgments/click-through rate?

In order to answer the first question, we calculated the breakdown of clicked, interesting, and relevant questions across the different question types, which is shown in Table 3.6. From Table 3.6, it follows that the users find factual questions (i.e. the “what” questions) and questions about a person (i.e. the “who” questions) to be more interesting than questions about time or location. The same applies to clickthroughs, although the difference is less pronounced, which could be partially explained by the low absolute number of clickthroughs compared to the judgments. In order to answer the second question, in Figure 3.7 we plotted the distribution of clicked, interesting, and relevant questions across the questions of different length. From

<table>
<thead>
<tr>
<th>Head</th>
<th>Click</th>
<th>Inter</th>
<th>Relev</th>
</tr>
</thead>
<tbody>
<tr>
<td>how</td>
<td>21</td>
<td>215</td>
<td>212</td>
</tr>
<tr>
<td>what</td>
<td>43</td>
<td>472</td>
<td>461</td>
</tr>
<tr>
<td>who</td>
<td>32</td>
<td>454</td>
<td>390</td>
</tr>
<tr>
<td>when</td>
<td>26</td>
<td>180</td>
<td>168</td>
</tr>
<tr>
<td>where</td>
<td>7</td>
<td>59</td>
<td>58</td>
</tr>
</tbody>
</table>

Table 3.6: User behavior with respect to different question types

Figure 3.7: Distribution of clicked, interesting, and relevant questions over question lengths

Figure 3.7, it follows that the users mostly click on the medium length 3,4,5-word questions. Users also find such medium-length questions to be more interesting and relevant than others.
3.5 Summary

In this chapter, we presented question feedback, a novel interactive strategy of refining exploratory queries with automatically generated natural language questions. The generated questions can naturally supplement the standard search result presentation methods to improve the utility of search engines in two ways. First, questions enable the users to navigate directly into the answers, contained in search results, without needing to read the documents, when the generated questions are relevant to their information need. Second, in case of imprecise or ambiguous queries, the automatically generated questions can naturally engage the users into a feedback cycle to refine their information need and guide them towards their search goals as well as stimulate new interests for exploratory search. We proposed a suite of methods for implementing the question-guided search strategy, including the methods for indexing the instance of syntactic patterns, generating questions from pattern instances with question templates and ranking questions with multiple heuristics. We implemented these methods in a prototype and evaluated it on a subset of Wikipedia. The experimental results demonstrated that the proposed method for question-based query refinement allows the users to more easily navigate in search results and effectively explore the results space in an interactive and natural way.

We believe that question-guided search is a very promising novel paradigm of interactive search. Our work is only a first step to show its feasibility; there are many interesting directions for future research. First, it would be interesting to further explore alternative methods for question presentation and ranking; in particular, applying learning to rank methods to optimize the ranking of questions would be very interesting. Second, we have only explored question generation based on manually created templates; it would be interesting to develop techniques for automatic induction of interesting syntactic patterns and question generation templates. Finally, a question-guided search engine would generate rich user history, including sequences of questions clicked by the users; such search log data offers interesting opportunities for user intent analysis and massive implicit feedback.
Chapter 4

Interactive sense feedback

4.1 Introduction

Ambiguity is a fundamental property of natural language, which negatively affects the quality of retrieval results by decreasing precision. Generally, an ambiguous query can be defined as any query which contains one or several polysemous terms. The difficulty of lexical ambiguity resolution (or sense disambiguation) varies greatly depending on several factors. When a query is sufficiently long, other terms in the query may serve as effective disambiguation clues due to the collocation effects [52]. In such cases, a search system may attempt to resolve ambiguity in an unsupervised way or by leveraging external resources, such as on-line dictionaries [57] or thesauri (e.g., WordNet [96] [92] [62]). Automatic disambiguation, however, proved to be very challenging, particularly because queries are usually very short and even humans cannot perform it with perfect accuracy.

The problem of ambiguity is exacerbated when a user’s information need corresponds to a minority (non-popular) sense of an ambiguous query term in the collection. In such a case, the initial retrieval results would most likely be dominated by a large number of non-relevant documents covering the popular, but distracting senses of an ambiguous query term, while the relevant documents covering the non-popular sense that the user is interested in may be ranked so far down in the ranked list that even diversification of search results would not be very helpful. Clearly, for such difficult queries, any feedback techniques that rely on the assumption that there is some relevant information in the top ranked results (e.g., pseudo feedback, document-level relevance feedback, top results-based term feedback) would not work well either. Consequently, designing an effective feedback method for such difficult queries is a theoretically and practically important problem, particularly in those domains, where short and ambiguous queries prevail, such as Web search.

In this work, we propose interactive sense feedback (ISF), a new method for interactive query disambiguation and reformulation, which, unlike the previously proposed methods for interactive relevance feedback [79], such as explicit [41] and term feedback [45] [93], does not rely on the assumption that the initial retrieval results contain relevant documents. Because of its independence of the initial retrieval results, ISF
can leverage user interaction both during the early stages of the search process or after it is complete.

At the high level, the proposed ISF is similar to query spelling correction, a popular and widely used feature of all major search engines. When a user submits a misspelled query, she may not be aware (at least immediately) of the reason the search results are of poor quality. A search system, however, can detect the problem, step in and try to improve the results by asking a user if she accidentally misspelled the query. Similarly, when users submit ambiguous queries, they are likely to spend some time and effort perusing search results, not realizing that the sense of a polysemous query term that they had in mind is not the most common sense in the collection being searched. Similar to spelling correction, along with presenting the initial search results, a search system can provide sense suggestions to narrow down the scope of the query. Ideally, sense suggestions can be presented as clarification questions (e.g., “Did you mean ambiguous query term as sense label?“), where the sense label can be either a single term or multiple terms.

Our approach is aiming to only signal and reveal the ambiguity of one or several query terms, leaving the final decision whether to disambiguate the query or not to the user. In some sense, our approach takes the best of both worlds: search systems can leverage the vastness of the data and their processing capabilities to infer the collection-specific senses of query terms and signal potential problems early on, while the users can leverage their intelligence and world knowledge to interpret the signals from the system and make the final decision. If the users are satisfied with search results, they may simply disregard sense suggestions. However, if the quality of search results is poor and a user can easily identify the desired sense of an ambiguous query term, she may indicate that sense and rely on the search system to update the results, according to the provided feedback.

We illustrate the idea of interactive sense feedback with the following example scenario. Suppose a user submits an ambiguous short query like “piracy” and is looking for documents about instances of copyright law violations as opposed to armed ship hijackings. In a collection of recent news documents, the intended sense of “piracy” corresponds to a minority sense, and one would expect the top-ranked retrieved documents to be non-relevant. Instead of having a user go through the search results and locate the relevant documents, a search system can instead find all the contexts, in which the query term occurred in the collection, indicate that the query term likely has two distinct collection-specific senses and ask the user “Did you mean piracy as copyright infringement?” or “Did you mean piracy as ship hijacking?”.

From the above discussion, it follows that interactive sense feedback needs to address the following two major problems. The first problem is designing an efficient algorithm for automatic off-line identification of discriminative senses of query terms through the global analysis of document collection. We emphasize the global analysis because a local analysis method such as pseudo-feedback cannot discover minority senses
when the initial search results are poor, a scenario which we focus on in this work. The second problem is how to generate representations of the discovered senses in such a way that each sense is easily interpretable and the best sense (i.e. the sense that results in the best retrieval performance) is easily identifiable by the users.

To solve the first problem, we propose and study several different algorithms for discovering the query term senses based on the global analysis of the collection. We compare these algorithms based on their upper bound retrieval performance and select the best performing one.

To solve the second problem, we propose several alternative methods for concise representation of the discovered senses and conducted a user study to evaluate the effectiveness of each method with the actual retrieval performance of user sense selections.

4.2 General idea

Despite years of research, there is still no consensus within the AI and IR research communities about what kind of information is most useful for sense disambiguation. Depending on the definition of a word sense, there are two major ways to approach sense disambiguation. Within the first view, the sense of a word is defined as its intrinsic property and corresponds to the high-level concepts denoted by the word lexeme. This view assumes that correct and comprehensive specification of the word sense requires complete knowledge about the world and can only be provided in the form of a manually created dictionary. The second view assumes that the senses of a word, rather than being its predefined property, can be differentiated by various contextual clues, such as its syntactic role and the nearby context.

This work adopts the latter view and is based on the assumption that the senses of a query term can be differentiated by grouping and analyzing all the contexts, in which it appears in the collection. Consequently, a sense-aware retrieval model should consider not only individual query terms, but also all the contextual (or neighboring) terms, with which those terms appear in the collection. We distinguish two types of contexts of a query term: local context, which corresponds to an individual co-occurrence of a query term with other terms within a certain unit of text (such as a window of certain size or the entire document) and the global context, which aggregates all local contexts associated with a term. Such aggregation allows to eliminate noise and identify strong, collection-wide semantic relations of a given query term with all other terms in the vocabulary of a collection. The global context of a particular query term can then be analyzed to identify the subsets of terms, which appear in the global contexts of each other. We consider such subsets of terms as the collection-specific senses of a query term.
Algorithm-wise, sense feedback works as follows:

1. First, a document collection is preprocessed to construct the contextual term similarity matrix, which includes all the terms in the vocabulary of a collection using one of the methods in Section 4.4.1; the contextual term similarity matrix is a sparse matrix, in which the rows correspond to the global contexts of each term in the vocabulary of a collection.

2. Given a query, the retrieval system first constructs the term similarity graph for each query term, which includes all the terms appearing in the global context of the given query term and the contextual co-occurrence relations between them. Next the system identifies clusters of terms in the term similarity graph. Each of those clusters is then converted into a language model, which takes into account the strength of semantic relations between the terms in the contextual term similarity matrix and represents a collection-specific sense of a query term.

3. For each of the identified senses, the system generates a concise representation using one of the methods in Section 4.5, which is presented to a user. If a user recognizes the intended sense of an ambiguous query term among those presented by the system, the language model of the original query is updated with the language model of the selected sense. The updated query language model can then be used to retrieve a new set of documents reflecting user feedback and focused on the specific sense of the initially ambiguous query term.

The interactive sense feedback approach has several advantages over the existing feedback methods. Firstly, sense feedback does not rely on the initial retrieval results and can be used either on-line or off-line. Secondly, only those senses that actually occur in the collection would be presented to the users. Finally, sense feedback does not rely on any external resources, and hence is completely general.

4.3 Formal definition

We study interactive sense feedback with the language modeling approach to IR, specifically the KL-divergence retrieval model [111], according to which the retrieval task involves estimating a query language model, $\Theta_q$ for a given term-based query $q$ and document language models $\Theta_D$, for each document $D_i$ in the document collection $C = \{D_1, \ldots, D_m\}$. The documents in the collection are scored and ranked according to the Kullback-Leibler divergence:

$$KL(\Theta_q || \Theta_D) = \sum_{w \in V} p(w|\Theta_q) \log \frac{p(w|\Theta_q)}{p(w|\Theta_D)}$$
Within the KL-divergence retrieval model, relevance feedback is considered as the process of updating the query language model $\Theta_q$, given the feedback obtained after the initial retrieval results are presented to the users. Such feedback may be explicitly provided by the user or implicitly derived from the retrieved results. According to this view, sense feedback can be treated as the process of updating $\Theta_q$ with the sense of an ambiguous query term identified by the user as relevant to her information need.

By following the language modeling approach, given a term-based query $q = \{q_1, \ldots, q_n\}$, a particular sense $s$ of the query term $q_i$ is represented as a sense language model $\hat{\Theta}_{q_i}^s$.

**Definition 6** Sense Language Model. $\hat{\Theta}_{q_i}^s$ for a particular sense $s$ of term $t \in V$ is a probability distribution $p(w|\hat{\Theta}_{q_i}^s)$ over a subset of words $S \subseteq V$, where $V$ is a vocabulary of a particular document collection $C$.

Given that a user selects a particular sense $s$ for the query term $q_i$, the language model $\hat{\Theta}_{q_i}^s$, associated with the selected sense can be naturally used for updating the original query language model $\Theta_q$ through linear interpolation:

$$p(w|\hat{\Theta}_q) = \alpha p(w|\Theta_q) + (1 - \alpha)p(w|\hat{\Theta}_{q_i}^s)$$

where $\alpha$ is the interpolation coefficient between the sense language model and the original query model.

**Definition 7** Contextual Term Similarity Matrix. is a sparse matrix $S$ of size $n \times n$ where $n = |V|$. Each row $S_i$ corresponds to a word $w_i \in V$ and represents a probability distribution over all other words $w$ in the vocabulary $V$, such that the probability mass would be concentrated on the terms, which are strongly semantically related to $w_i$. Each element $S_{ij}$ of the matrix corresponds to a probability $p(w_j|w_i)$, which indicates the strength of semantic relatedness of the words $w_i$ and $w_j$ in a document collection $C$.

**Definition 8** Term Similarity Graph $G_{w_i} = (V_{w_i}, E_{w_i})$ for a term $w_i$ is a graph, in which $\forall j \in V_{w_i}, S_{ij} \neq 0$ and $\forall u, v$, such that $(u, v) \in E_{w_i}, S_{uv} \neq 0$.

Having formally defined the concept of a sense, in the following sections we discuss the proposed approaches to sense detection and presentation in more detail.

### 4.4 Sense detection

In this section, we focus on the two components of the sense detection method, introduced in Section 4.2: constructing the contextual similarity matrix and clustering the query term similarity graph.
4.4.1 Contextual term similarity matrix construction

Constructing the contextual term similarity matrix for a document collection requires a method to calculate the strength of semantic relations between the terms in the vocabulary. In this work, we experiment with two such methods: mutual information (MI) and hyperspace analog to language (HAL).

**Mutual Information**

Given two words \(w\) and \(v\), the mutual information between them is calculated by comparing the probability of observing \(w\) and \(v\) together with the probabilities of observing them independently, according to the following formula:

\[
MI(w, v) = \sum_{X_w = 0,1} \sum_{X_v = 0,1} p(X_w, X_v) \log \frac{p(X_w, X_v)}{p(X_w)p(X_v)}
\]

where \(X_w\) and \(X_v\) are binary variables indicating whether \(w\) or \(v\) are present or absent in a document. The probabilities are estimated as follows:

\[
p(X_w = 1) = \frac{c(X_w = 1)}{N}
\]

\[
p(X_w = 0) = 1 - p(X_w = 1)
\]

\[
p(X_v = 1) = \frac{c(X_v = 1)}{N}
\]

\[
p(X_v = 0) = 1 - p(X_v = 1)
\]

\[
p(X_w = 1, X_v = 1) = \frac{c(X_w = 1, X_v = 1)}{N}
\]

\[
p(X_w = 1, X_v = 0) = \frac{c(X_w = 1) - c(X_w = 1, X_v = 1)}{N}
\]

\[
p(X_w = 0, X_v = 1) = \frac{c(X_v = 1) - c(X_w = 1, X_v = 1)}{N}
\]

\[
p(X_w = 0, X_v = 0) = 1 - p(X_w = 1, X_v = 1) - 
\]

\[
p(X_w = 0, X_v = 1) - p(X_w = 1, X_v = 0)
\]

where \(c(X_w = 1)\) and \(c(X_v = 1)\) are the numbers of documents containing the words \(w\) and \(v\), respectively, and \(c(X_w = 1, X_v = 1)\) is the number of documents that contain both \(w\) and \(v\). Mutual information measures the strength of association between the two words and can be considered as a measure of their semantic relatedness. The higher the mutual information between the two words, the more often they tend to occur in the same documents, and hence, the more semantically related they are. For each term \(t\) in the vocabulary of a collection, we identify the top \(k\) terms that have the highest mutual information with \(t\) and use those
terms as the global context of \( t \) in the contextual term similarity matrix of a collection.

**Hyperspace Analog to Language**

HAL \[10\] is a representational model of high dimensional concept spaces, which was created based on the studies of human cognition. Previous work \[89\] has demonstrated that HAL can be effectively applied to IR. Constructing the HAL space for an \( n \)-term vocabulary involves traversing a sliding window of width \( w \) over each term in the corpus, ignoring punctuation, as well as sentence and paragraph boundaries. All terms within a sliding window are considered as part of the local context for the term, over which the sliding window is centered. Each word in the local context receives a certain weight according to its distance from the center of the sliding window (words that are closer to the center receive higher weight). After traversing the entire corpus, an \( n \times n \) HAL matrix \( H \), which aggregates the local contexts for all the terms in the vocabulary, is produced. In this matrix, the row vectors encode the preceding word order and the column vectors encode the posterior word order. An example of the HAL space for the sentence “the effects of pollution on the population” constructed using the sliding window of size 10 (5 words before and after the center word) is shown in Table 4.1.

In the HAL-based approach, the global co-occurrence matrix is first produced by merging the row and column corresponding to each term in the HAL space matrix. Each term \( t \) corresponds to a row in the global co-occurrence matrix \( H_t = \{(t_1, c_1), \ldots, (t_m, c_m)\} \), where \( c_1, \ldots, c_m \) are the number of co-occurrences of the term \( t \) with all other terms in the vocabulary. After the merge, each row \( H_t \) in the global co-occurrence matrix is normalized to obtain the contextual term similarity matrix for the collection:

\[
S_{ti} = \frac{c_i}{\sum_{j=1}^{m} c_j}
\]

Unlike mutual information, HAL uses the contextual windows of sizes smaller than the entire document to create the local contexts, which should presumably result in less noisy global contexts.
4.4.2 Sense detection algorithm

Algorithm 2 is a high-level representation of a method to detect the senses of a given query term $q_i$.

**Algorithm 2** Sense detection for a query term $q_i$

1. forall $j : S_{ij} \neq 0$
   
   $V_{q_i} \leftarrow V_{q_i} \cup j$

2. forall $(u, v) : (u, v) \in V_{q_i} \times V_{q_i}$
   
   if $S_{uv} \neq 0$
   
   $E_{q_i} \leftarrow E_{q_i} \cup ((u, v); S_{uv})$

3. $G_{q_i} \leftarrow G(V_{q_i}, E_{q_i})$

4. $C \leftarrow \text{cluster}(G_{q_i})$

   for $k = 1$ to $|C|
   
   forall $t : t \in V_{C_k}$

   $p(t|\hat{\Theta}^k_{q_i}) = \frac{\sum_{u : (t, u) \in E_{C_k}} S_{tu}}{\sum_{u : (u, u) \in E_{C_k}} S_{uu}}$

The algorithm works as follows:

1. Given a query term $q_i$, a set of terms related to $q_i$ from the contextual term similarity matrix $S$ forms a set of vertices of the term similarity graph $G_{q_i}$;

2. For each pair of vertices in $G_{q_i}$, check if there exists a relation in $S$ with non-zero weight between the terms corresponding to those vertices. If so, the strength of relation becomes the weight of the edge between those terms in $G_{q_i}$;

3. The dynamically constructed query term similarity graph $G_q$ is clustered into a set of subgraphs using one of the graph clustering algorithms;

4. Each cluster (subgraph) $C_k$ is converted into a sense language model $\hat{\Theta}^k_{q_i}$, by normalizing the sum of the weights of all edges adjacent to each node in the cluster with the sum of the weights of all edges in the cluster.

Note that query term similarity graphs are typical small world graphs (i.e. graphs, in which most pairs of nodes are connected with very short paths), which are known to contain inherent community or cluster structure. In this work, we experiment with two methods for finding this structure: Clauset-Newman-Moore community clustering algorithm [16] and clustering by committee [70].

4.5 Sense presentation

In the proposed sense feedback approach, a sense is represented as a sense language model. Although such representation is effective for retrieval, it may not be suitable for presenting the discovered senses to the
users, since interpreting language models may place a significant cognitive burden on them. Therefore, a retrieval system needs to generate a concise and interpretable representation for each sense. In this work, we explore two sense presentation methods: using the top $k$ terms with the highest probability in the sense language model and selecting a small number of the most representative terms from the sense language model as a sense label. The latter approach uses a subgraph of the query term similarity graph, from which the sense language model was created to find a subset of terms that cover the subgraph in such a way that the sum of the weights of the vertices in the cover is maximized. This is known as the Dominating Set Problem, which is NP-complete.

**Algorithm 3** Generate a set of labels $L$ for a sense language model $\hat{\Theta}_q^s$

$$
\begin{align*}
L & \leftarrow \emptyset \\
C & \leftarrow \emptyset \\
W & \leftarrow \emptyset \\
\text{forall } t : t \in \hat{\Theta}_q^s & \\
1. & \quad W_t \leftarrow W_t \cup \sum_{v: (t,v) \in E_{C_t}} S_{tv} \\
 & \quad W \leftarrow \text{sort}(W) \\
2. & \quad \text{forall } t : t \in W_t \\
 & \quad \quad \text{if } t \notin C \\
 & \quad \quad \quad L \leftarrow L \cup t \\
3. & \quad \text{forall } v : (t,v) \in E_{C_t} \\
 & \quad \quad C \leftarrow C \cup v
\end{align*}
$$

Therefore, we employ a greedy Algorithm 3, which works as follows:

1. Sort the vertices according to their weights;

2. Traverse the sorted set of vertices $W_t$, each time selecting the remaining uncovered vertex with the highest weight and adding the selected vertex to the set of sense labels $L$;

3. Add the selected vertex and all the vertices adjacent to it in the cluster subgraph to the set of covered vertices and select the next label, until all the vertices of the subgraph, which corresponds to the sense being labeled, are covered.

### 4.6 Experiments

In this section, we present the results for an experimental evaluation of sense feedback. First, we describe our experimental setup and two experimental settings used to study the upper-bound and actual retrieval effectiveness of sense feedback. In the first setting, in order to determine the upper bound for the potential retrieval effectiveness of sense feedback on several standard TREC datasets, we simulated the optimal user
behavior by measuring the retrieval performance of all the senses discovered by each sense detection method and saving only the retrieval results of the optimal (best performing) sense. We also determined the optimal parameter settings for each sense detection method through simulation experiments and compared the upper-bound effectiveness of each method with the baselines. In the second setting, in order to find out whether the users can recognize the query term senses discovered by the best sense detection method and effectively use them to improve the quality of retrieval results, we conducted a user study by asking the users to pick one sense for each query based on different sense presentation methods. We then determined the best method for sense presentation and the actual performance of sense feedback based on user sense selections.

4.6.1 Datasets and experimental setup

All experiments in this work were conducted on three standard TREC collections: AP88-89, which was used for various Ad Hoc tracks; ROBUST04, which was used for the 2004 Robust track [1] and AQUAINT, which was used for the 2005 HARD [2] and Robust tracks. Various statistics for the experimental datasets are summarized in Table 4.2.

<table>
<thead>
<tr>
<th>Corpus</th>
<th>#Docs</th>
<th>Size(Mb)</th>
<th>#Topics</th>
<th>Avg. top.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP88-89</td>
<td>164,597</td>
<td>507</td>
<td>100</td>
<td>3.5</td>
</tr>
<tr>
<td>ROBUST04</td>
<td>528,155</td>
<td>1910</td>
<td>250</td>
<td>2.65</td>
</tr>
<tr>
<td>AQUAINT</td>
<td>1,033,461</td>
<td>3042</td>
<td>50</td>
<td>2.56</td>
</tr>
</tbody>
</table>

Table 4.2: Statistics of the experimental datasets

The TREC topics 51-150 for the AP88-89 collection are long, sentence-like queries, which include on average more than 3 query terms. The TREC topics 301-450 and 601-700 for the ROBUST04 collection are mostly 2-3 term queries with a small number of highly ambiguous one term queries (e.g., metabolism, robotics, tourism, creativity). The 50 AQUAINT topics include the hard (i.e. resulting in the very low retrieval performance) topics from ROBUST tracks. All documents and queries have been preprocessed by stemming with the Porter stemmer and removing the stop words. For each of the test collections, we precomputed the contextual term similarity matrices using both the mutual information and HAL. We did not include very rare terms (the ones that occur less than 5 times in the entire collection) or very popular ones (the ones that occur in more than 10% of documents) in the contextual term similarity matrices. A maximum of 100 most contextually similar terms according to a particular similarity measure have been stored for each term in the contextual term similarity matrix. For construction of the query term similarity graphs we used only those terms, the similarity weight between which and the given query term is greater than 0.001.
4.6.2 Upper-bound performance

In this section, we experimentally determine the upper bound for retrieval performance of sense feedback and compare it with the baseline feedback method on all three test collections. In the first set of experiments, we determined the upper bound for the retrieval performance of sense feedback and compared it with the baseline feedback method on all three test collections. We chose model-based feedback method proposed in [112] as a baseline, since it is based on the same KL-divergence retrieval framework as sense feedback. We used the suggested parameter settings for model-based feedback: mixture noise coefficient was set to 0.95 and the feedback coefficient to 0.9. Note that since the proposed sense feedback method is meant to be a complementary, rather than a competing method to pseudo-feedback (any pseudo-feedback method can be easily combined with sense feedback), we only included pseudo-feedback as a reference baseline and were not aiming to compare sense feedback with all existing pseudo-feedback methods.

The upper bound for the retrieval performance of sense feedback is determined by simulating a user, who is always able to select the optimal sense for each query. Specifically, we first identified all possible senses for each query term and then used each sense to expand the initial query model and estimate the retrieval effectiveness of the expanded query using relevance judgments. The sense that maximizes the average precision of the retrieved results is chosen as the best sense for a given query. For model-based pseudo-relevance feedback we used the top 10 retrieved documents. For initial retrieval, we used the KL divergence retrieval method with a Dirichlet smoothing prior set to 2000. Before comparing different sense detection methods to the baseline, we determined the optimal parameter setting for for each of them on the held-out dataset (AP88-89).

Parameter setting

In the first experiment, we set the interpolation coefficient $\alpha$ to 0.9 and empirically determined the optimal size of the sliding window used for construction of the HAL-based contextual term similarity matrix. Figure 4.1 shows the performance of Community Clustering (CC) and Clustering By Committee (CBC) in conjunction with the HAL-based contextual term similarity matrix construction method with respect to MAP by varying the size of the sliding window used for its construction.

Two important conclusions can be drawn from Figure 4.1. First, community clustering consistently outperforms clustering by committee for all sizes of the HAL window. Second, the optimal size of the HAL window for both sense detection methods is 20 (10 words before and after the center word). Next, we determined the optimal value of the interpolation coefficient $\alpha$ for different combinations of methods for construction of the contextual term similarity matrix and sense detection. In these experiments, we set the
Figure 4.1: Performance of sense detection methods by varying the size of the HAL sliding window size of the HAL window to its optimal value of 20.

Figure 4.2: Performance of sense detection methods by varying the interpolation parameter $\alpha$ (the name of the sense detection method is before the hyphen and the similarity measure is after the hyphen).

From Figure 4.2, it follows that the combination of community clustering and HAL-based term similarity weights outperforms all other sense detection methods. The best configuration for each sense detection method is as follows: $w = 20$ and $\alpha = 0.5$ for CC-HAL; $w = 20$ and $\alpha = 0.7$ for CBC-HAL; $\alpha = 0.6$ for CC-MI and $\alpha = 0.7$ for CBC-MI. Having determined the optimal parameter setting for each sense detection method, in the next set of experiments we determined the best sense feedback method with respect to the upper-bound retrieval performance and compared it with the baselines.

Upper-bound comparison of sense feedback

The upper-bound performance of different combinations of methods for construction of the contextual term similarity matrix and sense detection on all three experimental datasets is summarized and compared with the baselines in Table 4.3. For these experiments, we used the best configuration for each sense detection method empirically determined in the previous section. All feedback methods are evaluated based on their ranking of the top 1000 documents with respect to the mean average (non-interpolated) precision (MAP), precision at top 5 and 20 documents (Pr@5 and Pr@20) and the total number of relevant documents retrieved.
We also report the retrieval performance of the initial KL-divergence based retrieval run (KL), which is used for model-based pseudo-feedback (KL-PF). As explained earlier, we include pseudo feedback only as a reference baseline, since sense feedback can be easily combined with pseudo feedback.

<table>
<thead>
<tr>
<th>Method</th>
<th>AP88-89</th>
<th>ROBUST04</th>
<th>AQUAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>KL</td>
<td>0.2492</td>
<td>0.2462</td>
<td>0.1942</td>
</tr>
<tr>
<td>KL-PF</td>
<td>0.3066</td>
<td>0.2569</td>
<td>0.2189</td>
</tr>
<tr>
<td>CC-MI</td>
<td>0.2955</td>
<td>0.2538</td>
<td>0.2237</td>
</tr>
<tr>
<td>CC-HAL</td>
<td>0.3323</td>
<td>0.3002</td>
<td>0.2286</td>
</tr>
<tr>
<td>CBC-MI</td>
<td>0.2786</td>
<td>0.2477</td>
<td>0.2206</td>
</tr>
<tr>
<td>CBC-HAL</td>
<td>0.2588</td>
<td>0.2571</td>
<td>0.2004</td>
</tr>
</tbody>
</table>

Table 4.3: Comparison of the upper-bound performance of sense feedback with the baselines on all topics and collections.

The following conclusions can be drawn from the analysis of Table 4.3:

1. The combination of community clustering and HAL-based construction of contextual term similarity matrix outperforms all other methods and the baselines both in terms of MAP and Pr@N, indicating the potential of using the automatically identified senses of query terms to improve retrieval;

2. Community clustering generally outperforms clustering by committee both in combination with mutual information and HAL-based term similarity weighting;

3. Sense feedback is equally effective for both short (AQUAINT and ROBUST04) queries and longer (AP88-89) queries.

Table 4.4 compares the upper-bound effectiveness of sense feedback with the baselines in case of difficult queries. As follows from Table 4.4, sense feedback effectively improves the performance of difficult queries and outperforms both baselines, particularly improving the ranking of the top results, as indicated by significant improvements in Pr@5 and Pr@10. Pseudo-feedback, on the other hand, decreased the retrieval performance on the AQUAINT dataset.

The absolute numbers of difficult and normal topics improved by pseudo-feedback and sense feedback in different datasets are shown in Table 4.5.

As follows from Table 4.5, sense feedback improved the retrieval performance of a significantly larger number of both difficult and normal queries than pseudo-feedback in each dataset.
Table 4.4: Comparison of the upper-bound performance of sense feedback with KL-divergence retrieval model (KL) and model-based pseudo-feedback (KL-PF) on difficult topics. * indicates statistically significant difference relative to KL (95% confidence level), according to the Wilcoxon signed-rank test. † indicates statistically significant difference relative to KL-PF (95% confidence level), according to the Wilcoxon signed-rank test.

<table>
<thead>
<tr>
<th></th>
<th>KL</th>
<th>KL-PF</th>
<th>SF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AP88-89</td>
<td>0.0346</td>
<td>0.0744</td>
<td>0.0876*</td>
</tr>
<tr>
<td>Pr@5</td>
<td>0.1118</td>
<td>0.1529</td>
<td>0.25</td>
</tr>
<tr>
<td>Pr@10</td>
<td>0.0824</td>
<td>0.1412</td>
<td>0.2031</td>
</tr>
<tr>
<td>ROBUST04</td>
<td>MAP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.04</td>
<td>0.067</td>
<td>0.073*†</td>
</tr>
<tr>
<td>Pr@5</td>
<td>0.1567</td>
<td>0.1675</td>
<td>0.3054</td>
</tr>
<tr>
<td>Pr@10</td>
<td>0.1527</td>
<td>0.1554</td>
<td>0.2608</td>
</tr>
<tr>
<td>AQUAINT</td>
<td>MAP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.0473</td>
<td>0.0371</td>
<td>0.0888*†</td>
</tr>
<tr>
<td>Pr@5</td>
<td>0.125</td>
<td>0.075</td>
<td>0.2875</td>
</tr>
<tr>
<td>Pr@10</td>
<td>0.1188</td>
<td>0.0813</td>
<td>0.2375</td>
</tr>
</tbody>
</table>

Table 4.5: Number of difficult (D) and normal (N) topics improved by pseudo-feedback (KL-PF) and sense feedback (SF) in different datasets.

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>D</th>
<th>N</th>
<th>KL-PF</th>
<th>SF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D+</td>
<td>N+</td>
</tr>
<tr>
<td>AP88-89</td>
<td>99</td>
<td>34</td>
<td>65</td>
<td>19</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
<td>37</td>
</tr>
<tr>
<td>ROBUST04</td>
<td>249</td>
<td>74</td>
<td>175</td>
<td>37</td>
<td>89</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>68</td>
<td>153</td>
</tr>
<tr>
<td>AQUAINT</td>
<td>50</td>
<td>16</td>
<td>34</td>
<td>4</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12</td>
<td>29</td>
</tr>
</tbody>
</table>

4.6.3 User study

Although it is clear from the simulation experiments that automatically identified senses have the potential to improve the quality of retrieval, the next important question to answer is whether the users can recognize and select the optimal sense from retrieval perspective. In order to answer this question, we conducted a user study, for which we selected the AQUAINT topics. The reason for this is that those topics were used in 2005 TREC HARD track, which was created to explore the methods for improving the accuracy of retrieval systems through “highly focused, short-duration interaction with the searcher”. In the study, we asked the six participants to assume that they are typing the provided TREC queries into the search engine box and the search engine asks to clarify the meaning of a query by first selecting a query term and one of its senses that best fits the description of the query and makes the entire query less ambiguous.

We used the best performing combination of community clustering and HAL scores to generate the candidate senses of the query terms for the user study and presented the discovered senses using one-term labels, two-term labels, three-term labels, the top 3 terms from the sense language model and the top 10 terms from the sense language model. We then compared the query term and sense selections made by the users with the query term and sense selections resulting in the best upper-bound retrieval performance determined through simulation. Table 4.6 shows the accuracy of sense selection by the users as the fraction
(in percentages) of all the queries, for which the users selected both the optimal term and the optimal sense (in boldface) and the optimal term only (in parenthesis), regardless of whether the selected sense of that term is optimal or not.

<table>
<thead>
<tr>
<th>User</th>
<th>LAB1</th>
<th>LAB2</th>
<th>LAB3</th>
<th>SLM3</th>
<th>SLM10</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>18(56)%</td>
<td>18(60)%</td>
<td>20(64)%</td>
<td>36(62)%</td>
<td>30(60)%</td>
</tr>
<tr>
<td>User 2</td>
<td>24(54)%</td>
<td>18(50)%</td>
<td>12(46)%</td>
<td>20(42)%</td>
<td>24(54)%</td>
</tr>
<tr>
<td>User 3</td>
<td>28(58)%</td>
<td>20(50)%</td>
<td>22(46)%</td>
<td>26(48)%</td>
<td>22(50)%</td>
</tr>
<tr>
<td>User 4</td>
<td>18(48)%</td>
<td>18(50)%</td>
<td>18(52)%</td>
<td>20(48)%</td>
<td>28(54)%</td>
</tr>
<tr>
<td>User 5</td>
<td>26(64)%</td>
<td>22(60)%</td>
<td>24(58)%</td>
<td>24(56)%</td>
<td>16(50)%</td>
</tr>
<tr>
<td>User 6</td>
<td>22(62)%</td>
<td>26(64)%</td>
<td>26(60)%</td>
<td>28(64)%</td>
<td>30(62)%</td>
</tr>
</tbody>
</table>

Table 4.6: Fraction of the queries (in percentages), for which the users selected the optimal sense of the optimal term (in boldface) and the optimal term, but not necessarily the optimal sense (in parenthesis).

As follows from Table 4.6, for most labeling methods the users, in general, were able to select the best term for sense feedback for at least half of the queries in the study, which indicates that the users, in general, have the ability to identify the potentially ambiguous query terms that can benefit most from sense feedback. The fraction of the queries, for which the users could select both the best term for sense feedback and the best sense of that term is less, achieving the maximum of 36%. The following interesting conclusions can also be made from the analysis of Table 4.6:

1. Users do not tend to select the best sense more often when they observe more terms both in the label and the sense language model. One-term label is often sufficient to recognize the best sense and adding more terms to the label may mislead and confuse the users. The best result of 36% correctly identified optimal senses for one of the users is achieved when the top-3 terms in the sense language model are presented as a sense label;

2. 3-term labeling and choosing the top 3 terms from the sense language model perform comparably, which suggests that the terms with the highest probability are generally the most representative for a sense and vertices corresponding to them cover most of the sense subgraph.

In order to determine the practical utility of interactive sense feedback, we generated and evaluated the retrieval results based on the actual user sense selections. First we tuned $\alpha$, the parameter for interpolating the sense language model into the original language model. Using sense selections of users for the best sense representation method (we used top 10 terms with the highest weights in the sense language model for parameter tuning and evaluation, since it is the best sense representation method, according to Table 4.6), we varied the value of the interpolation coefficient $\alpha$ and plotted the resulting performance on all AQUAINT queries with respect to MAP in Figure 4.3.
Figure 4.3: Retrieval performance of user sense selections for all the queries in AQUAINT, depending on the value of interpolation parameter $\alpha$.

From Figure 4.3, it follows that sense feedback is consistently most effective for all the users when $\alpha = 0.8$.

Setting $\alpha$ to its optimal value, we determined the retrieval performance of actual user sense selections on difficult topics for different sense presentation methods. The results are presented in Table 4.7.

<table>
<thead>
<tr>
<th>KL MAP=0.0473</th>
<th>KL-PF MAP=0.0371</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAB1</td>
<td>LAB2</td>
</tr>
<tr>
<td>User 1</td>
<td>0.0543</td>
</tr>
<tr>
<td>User 2</td>
<td>0.0516</td>
</tr>
<tr>
<td>User 3</td>
<td>0.0533</td>
</tr>
<tr>
<td>User 4</td>
<td>0.0506</td>
</tr>
<tr>
<td>User 5</td>
<td>0.0519</td>
</tr>
<tr>
<td>User 6</td>
<td>0.0526</td>
</tr>
</tbody>
</table>

Table 4.7: Retrieval performance of user sense selections on difficult topics with respect to MAP, depending on the sense presentation method. Performance of the baselines is shown in the first two rows of the table.

As follows from Table 4.7, although the user sense selections do not achieve the upper bound performance, we can conclude that interactive sense feedback can effectively improve the retrieval performance of difficult queries.

4.6.4 Examples of discovered senses

To gain some insight at how the automatically identified collection-specific senses may look like, in Tables 4.8 and 4.9, we show some sample senses discovered by using the community clustering algorithm in combination with the HAL-based weighting for the query term “stealth” of the AP88-89 topic #132 “stealth aircraft” and for the query term “cancer” of the AQUAINT topic # 310 “radio waves and brain cancer”. Inferring the meaning behind each sense from the top representative terms is not hard, but sometimes requires certain background knowledge. For example, Sense 2 of the query term “stealth” clearly corresponds to the aircrafts
with low radar visibility.

![Table 4.8: Automatically discovered senses for the term “stealth” in the query “stealth aircraft”](image)

<table>
<thead>
<tr>
<th>Sense 1</th>
<th>Sense 2</th>
<th>Sense 3</th>
<th>Sense 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>p(w</td>
<td>s)</td>
<td>w</td>
</tr>
<tr>
<td>budget</td>
<td>0.05</td>
<td>1.17a</td>
<td>0.068</td>
</tr>
<tr>
<td>senator</td>
<td>0.05</td>
<td>us</td>
<td>0.05</td>
</tr>
<tr>
<td>fiscal</td>
<td>0.045</td>
<td>plane</td>
<td>0.047</td>
</tr>
<tr>
<td>cut</td>
<td>0.0421</td>
<td>fighter</td>
<td>0.0463</td>
</tr>
<tr>
<td>chenni</td>
<td>0.0391</td>
<td>f</td>
<td>0.0461</td>
</tr>
</tbody>
</table>

In case of the term “cancer”, senses are less distinguishable, but nevertheless correspond to semantically coherent aspects of the query topic. For example, sense 1 most likely corresponds to cancer research, sense 2 is about different types of cancer, sense 3 is about cancer treatment and sense 4 is likely to correspond to cancer statistics in the US.

![Table 4.9: Automatically discovered senses for the term “cancer” in the query “radio waves and brain cancer”](image)

<table>
<thead>
<tr>
<th>Sense 1</th>
<th>Sense 2</th>
<th>Sense 3</th>
<th>Sense 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>p(w</td>
<td>s)</td>
<td>w</td>
</tr>
<tr>
<td>research</td>
<td>0.065</td>
<td>disease</td>
<td>0.076</td>
</tr>
<tr>
<td>new</td>
<td>0.057</td>
<td>caus</td>
<td>0.058</td>
</tr>
<tr>
<td>study</td>
<td>0.050</td>
<td>liver</td>
<td>0.051</td>
</tr>
<tr>
<td>scientist</td>
<td>0.048</td>
<td>lung</td>
<td>0.049</td>
</tr>
<tr>
<td>dr</td>
<td>0.0448</td>
<td>drug</td>
<td>0.049</td>
</tr>
</tbody>
</table>

It is important to note that most TREC queries consist of at least 2-3 terms and are generally not highly ambiguous. Therefore, several collection-based senses of a query term may have comparable retrieval performance to the best sense and users often select these senses instead of the best performing sense. For example, for the query #625 “arrests bombing WTC” the best sense is the sense labeled as “police” for the query term “bombing”. However, all the users who participated in the study selected the sense labeled as “arrest” for the query term “WTC”. Similarly, for the query #639 “consumer on-line shopping” most users selected the sense labeled as “web” for the query term “consumer”, whereas the best sense is the sense labeled “online” for the query term “shopping”.

### 4.6.5 Error analysis

It is important to note that most TREC queries consist of at least 2-3 terms and are generally not highly ambiguous. Therefore, several collection-based senses of a query term may have comparable retrieval performance to the best sense and users often select these senses instead of the best performing sense. For example, for the query #625 “arrests bombing WTC” the best sense is the sense labeled as “police” for
the query term “bombing”. However, all the users who participated in the study selected the sense labeled as “arrest” for the query term “WTC”. Similarly, for the query #639 “consumer on-line shopping” most users selected the sense labeled as “web” for the query term “consumer”, whereas the best sense is the sense labeled “online” for the query term “shopping”.

4.7 Summary

In this chapter, we presented interactive sense feedback, a set of methods to automatically discover collection specific senses of query terms, present them to the users and update the initial query based on user sense selections. Because the senses are discovered from the entire collection, such feedback strategy is not biased towards the majority senses in the top-ranked results, and thus is especially useful for improving the performance of difficult queries.

We experimentally determined the upper bound for the retrieval performance of all possible combinations of several different methods for automatic sense discovery and measuring the strength of semantic relatedness between the terms. Experimental results indicate that the combination of community clustering and hyperspace analog to language (HAL) has the best overall retrieval performance and can also significantly improve the retrieval accuracy for difficult queries. We also proposed different presentation methods for the discovered senses and evaluated the effectiveness of user sense selections when the senses are concisely represented. According to the results of our user study, users in most cases are able to select the optimal sense for feedback, which results in the improvement of average retrieval accuracy for difficult queries. Therefore, sense feedback has all the potential to be used as an alternative or supplemental technique to the existing interactive feedback methods, such as term, relevance and pseudo-feedback, particularly for difficult queries.

Interactive sense feedback can be extended in several ways. First, we can explore other methods for automatic sense detection and compare them with the ones proposed in this work. Second, we can investigate alternative ways of effectively presenting senses to the users. Finally, it would be very interesting to experiment with sense feedback for real ambiguous Web-style queries and incorporate sense feedback into search engine infrastructure as a complimentary strategy to search results diversification. We envision that sense feedback will show its full real potential in this case.
Chapter 5

Concept feedback

5.1 Introduction

The information needs that searchers express as keyword queries vary greatly in complexity, which is determined by the number of concepts that constitute the need. For this reason, the quality of search results largely depends on how completely and effectively all concepts constituting the information need are represented in the query. It is often the case that the users of Web search systems tend to minimize their effort by intentionally posing very short queries. As a result, many documents representing some aspects of the information need will be missing in the search results. In addition to that, since natural language allows to use different terms to refer to the same concept, it is often the case that the searchers and the authors of relevant documents use different terms to designate the same concepts. This problem is known as term mismatch (or vocabulary gap) problem and it negatively affects the quality of retrieval results by decreasing recall. It particularly often arises when ordinary users perform searches in a specialized domain (e.g. medical and legal searches). Underspecified queries and term mismatch are one of the main reasons behind poor search results.

Query expansion is a standard recall-enhancing technique allowing to mitigate the problems of differing vocabularies and partially specified information needs by selecting and adding the related terms and phrases to the initial query. The main difficulty in effective application of automatic query expansion lies in correct identification of underrepresented aspects of the information need and selecting the right expansion terms with the right weights. Typical sources of term associations for query expansion can be either static and exist at the time of query (search logs, ontologies, encyclopedias, manual or statistical thesauri constructed from the corpus) or dynamic, such as some of the top-ranked initially retrieved documents, which can be selected either automatically by the system (pseudo-relevance feedback) or interactively by the users (relevance feedback). All approaches using dynamic sources of expansion terms share the common problem that they are relying on the assumption that the initial retrieval results include some relevant documents, which can be used as a source of expansion terms. It is often the case that the top-ranked search results
for a query include very few or no relevant documents, so the users cannot recognize the positive signals and communicate them back to the system through relevance feedback. Such queries are typically referred to as difficult. While the search logs and statistical co-occurrence thesauri constructed through the global analysis of document collections allow to avoid dependence on the initial retrieval results, the coverage of these resources is limited and they may simply not contain effective expansion terms for a particular query.

In this chapter, we systematically and comprehensively explore the potential of concept feedback, a set of different strategies for leveraging ConceptNet [61] as a source of expansion terms for difficult queries. ConceptNet is presently the largest commonsense knowledge base, consisting of more than 1.6 million assertions about the world. Similar to Wikipedia, ConceptNet reflects the “wisdom of the crowds” and was constructed by gathering a large number of sentence-like assertions about the real world from on-line collaborators. It uses semantic network as a knowledge representation framework. Nodes in the semantic network of ConceptNet correspond to semi-structured natural language fragments (e.g., “food”, “grocery store”, “buy food”, “at home”) and represent the real world concepts. An edge between the two nodes represents semantic relationship between the two concepts. As opposed to other ontologies, such as WordNet, ConceptNet is not limited to hyponym/hypernym relations and features a more diverse relational ontology of twenty relationship types, such as causal, spatial and functional. As opposed to on-line encyclopedias, such as Wikipedia, the network structure of ConceptNet does not require additional processing to establish relations between the concepts. We believe that network-based structure of ConceptNet in combination with its rich relational ontology opens up possibilities for making more complex, multi-step textual inferences for expanding difficult queries. Although WordNet has been shown to help address the issue of vocabulary divergence between the queries and relevant documents, a potential expansion term may have much broader conceptual relation to the expanded query term, than the tight semantic coherence of a WordNet synset may allow. For example, we empirically determined that the term “mission” is an effective expansion term for the query “hubble telescope achievements”, since Hubble telescope is a space telescope, which requires space shuttle missions to maintain it. Establishing such complex semantic relations between the query and the expansion terms requires several inference steps, for which the semantic network structure of ConceptNet is well-suited. On the other hand, the hierarchical structure of WordNet and vector-space knowledge representation models for Wikipedia [28] present certain difficulties for making complex, multi-step inferences for query expansion. Concept feedback is based on the idea of leveraging the semantic network structure of ConceptNet to perform complex multi-step inferences for selecting a small number of concepts that can be used for effective expansion of the original query.

1http:\\www.conceptnet.org
In this section, we address the following two research questions. The first question is whether ConceptNet can be potentially leveraged to improve the accuracy of retrieval results for difficult queries? To answer this question, we conducted simulation experiments, in which we tried all neighboring concepts to each query term for expansion and used the best expansion terms to determine the upper bound for the potential retrieval effectiveness of ConceptNet concepts. The results for the upper-bound experiments are presented in Section 5.4.2. The second question is how to design the methods to automatically select a small number of expansion concepts from ConceptNet? To answer this question, in Sections 5.2 and 5.3 we propose heuristic and machine learning methods for selecting effective expansion terms from ConceptNet and provide the results of an experimental evaluation of the proposed methods in Section 5.4.

The main contribution of the present work is in systematic and comprehensive exploration of the heuristic and machine learning methods for selecting expansion terms from ConceptNet and comparing the effectiveness of query expansion methods leveraging ConceptNet with the effectiveness of automatic query expansion based on pseudo-relevance feedback (PRF).

5.2 Heuristic expansion

In this work, we generally adopt the language modeling approach to retrieval, specifically the KL-divergence retrieval model [111], according to which the retrieval task involves estimating a query language model, \( \Theta_q \), for a given keyword-based query \( q \) and the document language models \( \Theta_{D_i} \) for each document \( D_i \) in the collection \( C = \{D_1, \ldots, D_m\} \). The documents in the collection are scored and ranked according to the Kullback-Leibler divergence:

\[
\text{KL}(\Theta_q || \Theta_D) = \sum_{w \in V} p(w|\Theta_q) \log \frac{p(w|\Theta_q)}{p(w|\Theta_D)}
\]

Within the KL-divergence retrieval model, relevance feedback [112] is considered as the process of updating the query language model \( \Theta_q \), given the feedback obtained after the initial retrieval results are presented to the users. Such feedback may be explicitly provided by the users or implicitly derived from the top-ranked retrieval results. Following this approach, a concept expansion language model, \( \hat{\Theta}_q \), derived for a given query \( q \) from ConceptNet can be used for updating the original query language model \( \Theta_q \) through linear interpolation:

\[
p(w|\hat{\Theta}_q) = \alpha p(w|\Theta_q) + (1 - \alpha) p(w|\Theta_q)
\]

where \( \alpha \) is the interpolation coefficient between the concept expansion language model and the original query model.
The main challenge in using external resources for query expansion is to select the right number of effective concepts and avoid introducing noise. If a limited number of automatically identified expansion terms are added to the query, there is a possibility that effective expansion terms will be missed and the retrieval output is unlikely to be substantially improved. On the other hand, when the query vocabulary is substantially altered, the advantages gained from some useful added terms might be lost because of noisy terms and query drift. In language modeling context, selecting the right number of terms is less important than the right allocation of weights. In this section, we propose two heuristic methods for query expansion: method for finding expansion terms along the paths between the query terms and the method for random walk on the query concept graph. Before introducing the methods, we provide several important definitions.

**Definition 9** Concept graph \( G_c = (V, E) \) is a weighted graph, in which a set of vertices corresponds to a subset of concepts and a set of edges corresponds to the relations between those concepts extracted from ConceptNet.

When a concept graph is constructed from ConceptNet, all the concepts designated by a phrase are split into individual terms. For example, given a pair of concepts ‘telescope’ and ‘astronomical tool’ and a relation ‘IsA’ between them from ConceptNet, the node ‘telescope’ in the resulting concept graph will be connected with an edge to two separate nodes ‘astronomical’ and ‘tool’.

**Definition 10** Query term context \( C^d_q \) of size \( d \) for a given query term \( q \) includes all the concept terms \( c \) from ConceptNet that are within a certain distance \( d \) from \( q \).

For example, query term context of size 2 includes all the concept terms that are connected to the given query term in the concept graph (query term neighbors) and the concept terms that are connected to the query term neighbors. When constructing the query terms context, for each concept term we used 100 neighboring concept terms with the highest IDF \( \text{IDF}(t) = N \times \log(c(t, d)) \), where \( N \) is the total number of documents in the collection and \( c(t, d) \) is the number of documents, containing term \( t \) in the collection) and excluded very common terms (terms that occur in more than 10% of the documents in the collection).

**Definition 11** Query Concept Graph \( G^d_Q = (V, E) \) of size \( d \) for a query \( Q = \{q_1, q_2, \ldots, q_n\} \) is a weighted graph, in which \( V = \bigcup_{i=1}^n C_q \) and the set of weighted edges \( E = \{(c_1, c_k, w_{1k}), \ldots, (c_m, c_n, w_{mn})\} \) corresponds to the relations between the concept terms.

Since the relations between the concepts in ConceptNet do not have explicit weights, we designed an empirical procedure to assign them (details are in Section 5.4.3).
5.2.1 Path finding

**Definition 12** Path \( p(c_1) \rightarrow p(c_2) \) in the concept graph \( G^d_Q \) between the two concepts \( c_1 \) and \( c_2 \) is sequence of edges and concepts connecting those two concepts.

Given a query \( Q = \{q_1, q_2, \ldots, q_n\} \) and a query concept graph \( G^d_Q \), the method finds all the paths between the query terms and uses the concept along those paths as the expansion concepts.

5.2.2 Random walk

The random walk algorithm proceeds as follows. Given the query concept graph, \( G^d_Q \), we first construct the concept matrix \( C \) and perform a \( k \)-step random walk on that matrix. The weight of the expansion concept \( c \) for a query term \( q_i \) is determined as follows:

\[
p(c|q_i) = (1 - \alpha)\alpha^k C^k_{c,q_i}
\]

where \( \alpha \) is the probability of continuing the random walk.

5.3 Learning-based expansion

The learning-based expansion method is based on training a regression model, where the dependent variable is the measure of performance of an expanded query and the independent variables are the features of the expansion concept.

5.3.1 Model

We chose generalized linear regression model (GLM) as the learning algorithm. Given a vector of features \( \hat{x} \), the model estimates a vector of weights \( \hat{w} \) during training, and generates the output as a linear combination of the feature and weight vectors, \( f(\hat{x}) = \hat{x}\hat{w} \), during testing. One of the advantages of GLM over other models is that feature weights are easily interpretable and allow to identify the important properties of expansion terms.

5.3.2 Features

The set of features used in the model is presented in Table 5.1.

The feature set in Table 5.1 reflects the properties of individual queries, expansion concepts and expansion concepts with respect to queries. It extends the set of features used in [40] (designated by bullets in the BL
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Features of the query</strong></td>
<td></td>
</tr>
<tr>
<td>NumQryTerms</td>
<td>number of query terms, $</td>
</tr>
<tr>
<td>TopDocScore</td>
<td>retrieval score of the top-ranked document for the initial query</td>
</tr>
<tr>
<td><strong>Features of the expansion concept</strong></td>
<td></td>
</tr>
<tr>
<td>ExpTDocScore</td>
<td>retrieval score of the top-ranked document for the query expanded with the concept</td>
</tr>
<tr>
<td>TopTermFrac</td>
<td>ratio of the number of occurrences of the expansion concept over all the terms in the top 10 retrieved documents</td>
</tr>
<tr>
<td>NumCanDocs</td>
<td>number of the top 10 documents, containing the expansion concept</td>
</tr>
<tr>
<td>AvgCDocScore</td>
<td>average retrieval score of the documents, containing the expansion concept</td>
</tr>
<tr>
<td>MaxCDocScore</td>
<td>maximum retrieval score of the documents, containing the expansion concept</td>
</tr>
<tr>
<td>ConIDF</td>
<td>IDF of the expansion concept</td>
</tr>
<tr>
<td>ConFanOut</td>
<td>fan out of the expansion concept node in the query concept graph</td>
</tr>
<tr>
<td>SpActScore</td>
<td>spreading activation score of the expansion concept in the query concept graph</td>
</tr>
<tr>
<td>SpActRank</td>
<td>rank of the expansion concept after spreading activation on the query concept graph</td>
</tr>
<tr>
<td>RndWalkScore</td>
<td>score of the expansion concept by using the Finite Random Walk method</td>
</tr>
<tr>
<td>PathFindScore</td>
<td>score of the expansion concept by using the Path Finding method</td>
</tr>
<tr>
<td><strong>Features of expansion concept with respect to query terms</strong></td>
<td></td>
</tr>
<tr>
<td>AvgColCor</td>
<td>average co-occurrence of the expansion concept with the query terms in the collection</td>
</tr>
<tr>
<td>MaxColCor</td>
<td>maximum co-occurrence of the expansion concept with the query terms in the collection</td>
</tr>
<tr>
<td>AvgTopCor</td>
<td>average co-occurrence of the expansion concept with the query terms in the top 10 retrieved documents</td>
</tr>
<tr>
<td>MaxTopCor</td>
<td>maximum co-occurrence of the expansion concept with the query terms in the top 10 retrieved documents</td>
</tr>
<tr>
<td>AvgTopPCor</td>
<td>average co-occurrence of the expansion concept with pairs of query terms in the top 10 retrieved documents</td>
</tr>
<tr>
<td>MaxTopPCor</td>
<td>maximum co-occurrence of the expansion concept with pairs of query terms in the top 10 retrieved documents</td>
</tr>
<tr>
<td>AvgQDist</td>
<td>average distance of the expansion concept to the query terms in the query concept graph</td>
</tr>
<tr>
<td>MaxQDist</td>
<td>maximum distance of the expansion concept to the query terms in the query concept graph</td>
</tr>
<tr>
<td>AvgPWeight</td>
<td>average weight of the paths to the expansion concept from the query terms in the query concept graph</td>
</tr>
<tr>
<td>MaxPWeight</td>
<td>maximum weight of the paths to the expansion concept from the query terms in the query concept graph</td>
</tr>
</tbody>
</table>

Table 5.1: Features for ranking the expansion terms. Baseline feature set is designated in the BL column with *. 

and includes 7 new features, focused on the structural properties of the expansion concepts with respect to query terms in the query concept graph: ConFanOut, RndWalkScore, PathFindScore, AvgQDist, MaxQDist, AvgPWeight, MaxPWeight. Since PathFindScore and RndWalkScore correspond to the score of the expansion terms using the heuristic methods presented in Section 5.2.1 and
5.2.2 respectively, learning based approach unifies the heuristic based approaches.

5.4 Experiments

In this section, we present the results for experimental evaluation of unsupervised and supervised query expansion with concepts from ConceptNet. We first discuss our experimental setup and two experimental datasets.

5.4.1 Experimental setup and datasets

All experiments in this work were conducted on two standard TREC collections: ROBUST04, which was used in TREC 2004 Robust Track [1] and AQUAINT, which was used in TREC 2005 HARD [2] and Robust Tracks. Both collections are composed of English newswire documents. Various statistics for these document collections are summarized in Table 5.2. For the AQUAINT dataset, we used TREC topics 303-689 and for ROBUST04 TREC topics 301-700.

<table>
<thead>
<tr>
<th>Corpus</th>
<th>#Docs</th>
<th>Size(Mb)</th>
<th>#Topics</th>
<th>Avg. top.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AQUAINT</td>
<td>1,033,461</td>
<td>3042</td>
<td>50</td>
<td>2.56</td>
</tr>
<tr>
<td>ROBUST04</td>
<td>528,155</td>
<td>1910</td>
<td>250</td>
<td>2.65</td>
</tr>
</tbody>
</table>

Table 5.2: Statistics of the datasets used for experiments

In this work, we focus on studying the effectiveness of expansion using ConceptNet with respect to difficult queries. We define a difficult query as a query, for which either the average precision is less than 0.1 or the top 10 retrieved results are non-relevant (i.e. $Pr@10 = 0$).

5.4.2 Upper-bound performance

In order to determine the upper bound for the potential effectiveness of using ConceptNet for query expansion, we conducted a simulation experiment, in which for each query term it was first checked if there exists a concept node in ConceptNet that matches it. If such node was found, then all the concepts in its context of certain size were identified. We experimented with the contexts of size one, two and three. We then expanded each query with each concept term in the context by simply adding it to the query with the weight $1/|Q|$, where $|Q|$ is the length of an expanded query and evaluated the effectiveness of such expansion for difficult queries in each collection with respect to mean average precision (MAP), geometric mean average precision (GMAP), total number of relevant documents retrieved (RR) and precision at top 10 retrieved documents ($P@10$). The results of this comparison are reported in Table 5.3.
Table 5.3: Comparison of the upper-bound performance of concept feedback (CF) by varying the size of the expansion context with KL-divergence retrieval model (KL) and model-based pseudo-relevance feedback (KL-PF) on difficult topics

As follows from Table 5.3, in the upper bound Concept Feedback (CF) significantly improves the performance of KL-divergence retrieval and outperforms the baseline (model-based pseudo-relevance feedback), even when the context of size 1 (column CF-1) is used for expansion. In other words, for each query term there exists a single ConceptNet expansion concept, which on average doubles the performance of the baseline KL-divergence retrieval on difficult queries from both datasets. Using contexts of larger size improves the performance even more, with the context of size three (column CF-3) having triple the performance of KL-divergence retrieval without expansion. This simulation experiment clearly illustrates that using concept terms from ConceptNet for query expansion has a tremendous potential for improving the performance of difficult queries. However, how to automatically identify those few highly effective expansion concepts is unclear. In the rest of this work, we propose and study heuristic and learning-based methods for selecting a limited number of expansion concepts.

5.4.3 Edge weighting

Since our automatic query expansion procedure selects multiple expansion terms, we need to design a method to allocate weights to them. In particular, for this task we can use several properties of the expansion terms, such as the length of the paths, as well as the types and weights of ConceptNet relations connecting them to the query terms. Given a concept graph constructed from ConceptNet for a particular query, we used the following empirical procedure to assign the weights to its edges:

1. First, before query processing we used the results of simulation experiments described in Section 5.4.2 on the dataset with the most number of queries (ROBUST04) to count the number of times the best expansion concept was connected to the expanded query term with the relation of each type.

We then sorted the relations according to those counts and divided the relations into three groups of the same size, which are presented in Table 5.4.
Table 5.4: Number of times the best expansion term was connected to the expanded query term with the relation of each type

<table>
<thead>
<tr>
<th>Relation</th>
<th>Count</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA</td>
<td>132</td>
<td>1</td>
</tr>
<tr>
<td>HasProperty</td>
<td>72</td>
<td>1</td>
</tr>
<tr>
<td>CapableOf</td>
<td>65</td>
<td>1</td>
</tr>
<tr>
<td>AtLocation</td>
<td>40</td>
<td>1</td>
</tr>
<tr>
<td>ConceptuallyRelatedTo</td>
<td>35</td>
<td>1</td>
</tr>
<tr>
<td>UsedFor</td>
<td>35</td>
<td>1</td>
</tr>
<tr>
<td>HasA</td>
<td>27</td>
<td>2</td>
</tr>
<tr>
<td>DefinedAs</td>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td>ReceivesAction</td>
<td>21</td>
<td>2</td>
</tr>
<tr>
<td>PartOf</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>CausesDesire</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>LocatedNear</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Causes</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>HasPrerequisite</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Desires</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>InstanceOf</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>MadeOf</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>MotivatedByGoal</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>HasFirstSubevent</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>SimilarSize</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

2. Second, we constructed a term relationship graph for all the terms in the vocabulary of the collection. Term relationship graph is a weighted graph, in which the set of vertices corresponds to the terms in the collection and the edges correspond to the semantic relationships between them. The weight of an edge represents the degree of semantic relatedness of two terms. We used Hyperspace Analog to Language (HAL) scores [10] as a measure of the strength of semantic relationship between the terms. Unlike mutual information, in which the entire document is used as the context to calculate the number of co-occurrences between the terms, HAL uses narrower contextual windows (we used the sliding window of size 20) and has been shown in previous work [50] to produce less noisy term relationship graphs.

3. Third, for each query we construct the concept graph, in which the nodes and edges correspond to concepts and relations from ConceptNet, and perform two passes over its edges. In the first pass, if an edge in the concept graph also exists in the term relationship graph, its weight is used to calculate the average weight of edges belonging to the same group, according to Table 5.4. In the second pass, if an edge in the concept graph also exists in the term relationship graph, its final weight is equal to the product of the weight of that edge in the term relationship graph and the IDF of the target concept, otherwise it weight is equal to the product of the average weight of relations in the same relation group and the IDF of the target concept.
5.4.4 Learning-based expansion

We used 5-fold cross validation to train and test the proposed regression model. During testing we selected 100 top-scoring concepts and used them for expansion. In order to determine the optimal setting for learning-based concept expansion and learning-based pseudo-feedback, we experimented with different feature sets and contexts of different size (2 and 3). Performance of different learning-based ranking settings for AQUAINT and ROBUST04 datasets is presented in Figures 5.1 and 5.2, respectively.

![Figure 5.1: Comparison of performance of different learning-based expansion methods on the AQUAINT dataset](image)

Several interesting observations can be made based on the analysis of the AQUAINT data in Figure 5.1. First, for both the learning-based concept expansion and learning-based pseudo-feedback, using extended feature set (FULL) generally results in better performance than using the baseline (BASE) feature set, which empirically demonstrates the benefits of exploiting the concept graph-based structural properties of expansion concepts with respect to query terms. Second, using larger expansion context improves the performance for both feature sets and for most values of interpolation coefficient $\alpha$, which is consistent with the results of simulation experiment. However, for the baseline feature set, the best possible performance of both the learning-based expansion and learning-based pseudo-feedback using the expansion context of size 2 is greater than the best possible performance using the extended feature set, which is not the case with the context of size 3.

Similar observations can also be made by analyzing the behavior of learning-base expansion and pseudo-feedback methods on the ROBUST04 dataset in Figure 5.2, although the expansion and, in particular, pseudo-feedback methods behave more similar to each other. Consequently, we can conclude that using extended feature set (FULL) along with larger expansion context of size 3, results in the optimal performance.
Figure 5.2: Comparison of performance of different learning-based expansion methods on the ROBUST04 dataset

5.4.5 Comparison of methods

Having determined the best performing configuration of the learning-based methods, we then compare them with the heuristic methods.

Figure 5.3: Comparison of performance of heuristic and learning-based expansion and pseudo-feedback methods on the AQUAINT dataset

Figure 5.3 shows the performance of different expansion methods on the AQUAINT dataset. As follows from this figure, learning-based methods outperform the heuristic methods and a combination of learning-based expansion and model-based pseudo-feedback performs better than using learning-based expansion alone. Moreover, as the weight of the original query language model in the mixture decreases, the performance of the combined learning-based and pseudo-feedback method sharply improves, which indicated the effectiveness of expansion concepts.

Similar conclusions can be made from Figure 5.4, which illustrates the comparison of different expansion methods on the ROBUST04 dataset.
Figure 5.4: Comparison of performance of heuristic and learning-based expansion and pseudo-feedback methods on the ROBUST04 dataset

5.5 Summary

In this chapter, we presented the results of the first systematic exploration of the potential for applying semantic knowledge in ConceptNet to improve the retrieval results of difficult queries and overcome the problem of the lack of relevant documents for such queries in the initial search results. In particular, we conducted a simulation experiment to determine the upper bound for the effectiveness of query expansion with the related concepts from ConceptNet, which demonstrated that there exists a small number of highly effective expansion concepts. We also proposed several heuristic and learning-based methods for automatically selecting such terms and empirically compared the proposed methods on two standard datasets. Our results indicate that learning-based method can effectively leverage the common sense knowledge in ConceptNet to improve difficult queries both through query expansion alone and in combination with traditional pseudo-feedback methods. Designing methods to improve the search results of difficult queries is a challenging and very important practical and theoretical problem in information retrieval and our results have significant implications for improving the experience of Web search engine users in those cases, when they need it the most.
Chapter 6

Conclusions

Improving the users’ search experience when the initial retrieval results are of poor quality is one of the most theoretically and practically important problems in IR research. In this thesis, we proposed and experimentally evaluated the utility of three novel feedback methods, which address the main scenarios leading to poor search results: question feedback, sense feedback and concept feedback.

In the first scenario, a searcher does not have a specific information need and would like to explore a certain broad topic. In such cases, an existing query-response interaction model requires a user to issue several queries and spend considerable effort examining search results. Question feedback is a conceptually new type of interactive feedback aimed at the refinement of exploratory queries by automatically generating and presenting to the users a list of natural language clarification questions.

In the second scenario, a searcher has a specific information need, but due to the inherent ambiguity of natural language, the sense of a query term corresponding to his information need is a minority sense. Sense feedback enables the users to interactively improve the quality of retrieval results by selecting the intended sense from a list of automatically generated collection-specific senses of ambiguous query terms presented as a set of questions. Unlike the previously proposed approaches to relevance feedback, sense feedback is independent of the initial retrieval results and, thus, can be effectively applied to improve the performance of difficult queries.

In the third scenario, a searcher has a specific information need, but due to the differing vocabulary of relevant documents, the majority of relevant search results are missed. To address this scenario, we propose concept feedback, a machine learning based method to select highly effective expansion concepts from ConceptNet. Unlike all the previously proposed methods for query expansion using other external resources, such as Wikipedia or WordNet, concept feedback leverages the possibility of multi-step inference on the semantic network of ConceptNet and can identify broadly related expansion terms.

Since the proposed feedback strategies are complementary to each other and to the traditional query-results interaction model, they can be all potentially combined to provide better support to the users when their queries do not perform well. Therefore, a major direction for future work is to evaluate the real-
world utility of the proposed feedback methods by implementing them in a search engine infrastructure and conducting a large-scale evaluation involving the real users.
References


