Towards Natural Question-Guided Search

Alexander Kotov
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
akotov2@illinois.edu

ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
czhai@cs.uiuc.edu

ABSTRACT

Web search is generally motivated by an information need. Since asking well-formulated questions is the fastest and the most natural way to obtain information for human beings, almost all the queries posed to search engines correspond to some underlying questions, which represent the information need. Accurate determination of these questions may substantially improve the quality of search results and usability of search interfaces. In this paper, we propose a new framework for question-guided search, in which a retrieval system would automatically generate potentially interesting questions to the users. Since the answers to such questions are known to exist in search results, these questions can potentially guide the users directly to the answers they are looking for, eliminating the need to scan the documents in the results list. Moreover, in case of imprecise or ambiguous queries, automatically generated questions can naturally engage the users into feedback cycles to refine their information need and guide them towards their search goals. Implementation of the proposed strategy raises new challenges in content indexing, question generation, ranking and feedback. We proposed new methods to address these challenges and evaluated them with a prototype system on a subset of Wikipedia. The evaluation results show the promise of this new question-guided search strategy.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]: Linguistic Processing; H.3.3 [**Information Search and Retrieval**]: Relevance feedback, Search process

General Terms

Algorithms, Experimentation

Keywords

Interactive search, query processing, search interface

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 the 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.

Almost all the queries are motivated by some underlying question. For example, if a user is searching for information about John Fitzgerald Kennedy, the easiest and the 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"* or *"kennedy."* However, such a query does not fully specify the user's information need, which often includes particular aspects of information about JFK that a user is most interested in. For example, in this case, the underlying question that has caused a user to search, might be as broad as *"Who is John Kennedy?"* or as specific as *"When was Kennedy sworn as the President of the United States?"*.

The query-based search paradigm 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 [17], formulating natural language questions is the most natural way for most 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.

Another widely recognized deficiency of modern search systems is the lack of interactivity. Bookstein [2] points out that information retrieval should be envisioned as a process, in which the users are examining the retrieved documents in sequence and the system can and should actively gather the user feedback to adjust retrieval. For this reason, the interface of search engines should reflect the fact that Web search is a process, rather than an event. This work is an attempt to emphasize the exploratory nature of Web search by providing a search interface that helps the users refine their queries with automatically generated natural language questions, which can be answered precisely.

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.

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, an interactive question-based retrieval system must be able to locate and index the content, which can be used for question generation, generate well-formed and meaningful questions in response to user's queries and rank those questions in such a way that the most interesting and relevant questions appear at the top of the question list. In this paper, we propose solutions to all of the above problems and evaluate our approach with a prototype system.

The rest of this paper is organized as follows. In Section 2, we provide a brief overview of existing work. Section 3 introduces the general concept of question-guided search. Implementation details of the question-guided search framework are presented in detail in Section 4. Section 5 presents the results of an experimental evaluation of our question-guided search engine prototype. Finally, in Section 6 we give concluding remarks and identify directions for future work.

2. RELATED WORK

In this section, we provide an overview of research efforts in two major research areas that are the closest to our work: application of Natural Language Processing (NLP) methods to Information Retrieval (IR) and presentation of search results. The brittleness of NLP methods have traditionally been one of the major concerns and obstacles to using them in IR. However, it has been experimentally shown in [20] that "the speed and robustness of natural language processing has improved to the point where it can be applied to the real IR problems". One of the research directions that has recently gained particular attention is focused on using lexical and semantic relations to improve the accuracy and completeness of search results over the traditional "bag-of-words" approaches. The importance of lexico-syntactic relations for information retrieval was demonstrated by Wang et al. [21]. Zhai et al. [23] experimentally showed that indexing lexical atoms and syntactic phrases improves both the precision and recall. Hearst [7] used the semantics of lexico-syntactic patterns to automatically extract hyponym relations between the lexical items in large corpora. Syntactic pattern matching is also a popular technique in Question Answering (QA) to improve the performance on factoid [11] and definitional [4] questions. Katz et al. [12] extracted syntactic relations between the words through dependency parsing and used these relations to retrieve potential answers to a question. Jijkoun et al. [11] experimentally proved that a linguistically deeper method for factoid QA, based on a small number of patterns, including syntactic relations, outperforms the traditional surface text pattern-based methods. Our work can be viewed as a new way of using NLP techniques to support question answering in an IR system that is more robust than the regular question answering methods in the sense that automatically generated questions can always be answered.

An important aspect of any information access system is effective presentation of retrieval results. Hearst [8] provides an extensive overview of research efforts over the past several decades that aimed at improving the usability of search. The various alternatives to traditional presentation of search results can be classified into three major categories: clustering [9] [22] [1], summarization [13] and visualization. The information retrieval community has for a long time explored a number of graphical visualization techniques as an alternative to the ranked list presentation of search results. Fowler et al. [5] combined visual representations of queries, retrieved documents, associative thesaurus and a network of inter-document similarities into one graphical retrieval environment. Chalmers et al. [3] proposed to represent bibliography articles as particles in a 3-dimensional space and apply the potential fields modeling equations from theoretical physics to represent the relationships between the articles by their relative spatial position. InfoCrystal [19] is a visual query language that allows the users to graphically formulate arbitrarily complex boolean and vector-space queries by organizing the graphical query building blocks into hierarchies. Leuski et al. [14] integrated the ranked list representation and clustering of the retrieved documents into a visual environment for interactive relevance feedback. As a new way to present search results, our work enables the users to navigate directly into the answers they are searching for without needing to read the documents.

In the following sections, we present the general idea behind the question-guided retrieval process and examine its each individual component in detail.

3. QUESTION-GUIDED SEARCH

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 difficult, 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.

4. 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 **QU**estion-guided **S**earch **E**ngine, which we called QUSE. In the following sections, we consecutively focus on each individual component of the retrieval process: indexing, retrieval, ranking and feedback. In Section 4.1, we begin with a description of how dependency parsers can be used to index the occurrences of syntactic patterns, which can be later transformed into questions. In Section 4.3, we describe the structure of the index, used in QUSE. Question ranking is discussed in Section 4.4. And, finally, we overview the question-based relevance feedback in Section 4.6.

4.1 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 [16], 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 1.

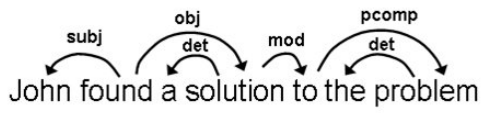


Figure 1: Example of a dependency tree

In the example sentence in Figure 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 [15], 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 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.

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*

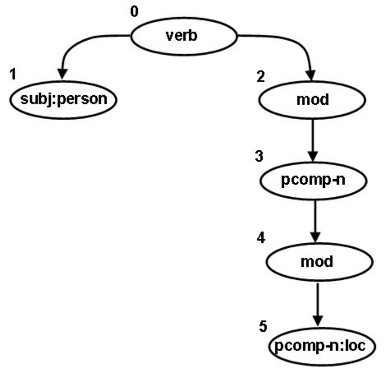


Figure 2: Compressed dependency tree (syntactic pattern)

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.

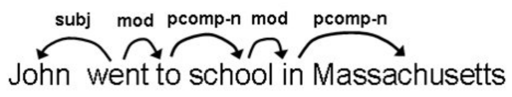


Figure 3: Dependency tree of an example sentence

In particular, we can manually define the following question templates for the syntactic pattern in Figure 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 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 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 [6].

4.2 Formal definition

Let $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$ be a collection of n documents, composed from a set of k terms $\mathcal{T} = \{t_1, \ldots, t_k\}$ and their stems $\mathcal{T}\prime = \{t_1\prime, \ldots, t_k\prime\}$.

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 $T \times T\prime \to S$, 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 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.*

4.3 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.

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

Table 1: Examples of syntactic patterns and sentences matching them

4.4 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 $H = \{h_1, h_2, \dots, h_n\}$ of n ranking heuristics (features), where each heuristic is a function $h : \Theta \to \mathbb{R}$, mapping questions in the set Θ into the real numbers (feature values), and the two questions $\delta_1 = (h_1(\delta_1), \dots, h_n(\delta_1))$ and $\delta_2 = (h_1(\delta_2), \dots, h_n(\delta_2))$, 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 δ_1 should be ranked above δ_2 or, i.e., question δ_1 is more relevant to the query than the question δ_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 δ, 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 δ 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 δ 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 [18]:

$$\begin{aligned} s(q,d) &= \sum_{t \in Q,D} \ln \frac{N-df+0.5}{df+0.5} \times \frac{(k_1+1)tf}{k_1((1-b)+b\frac{dl}{avdl})+tf} \\ &\times \frac{(k_3+1)qtf}{k_3+qtf} \end{aligned}$$

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 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.

The final ranking of the sample questions in Table 2 by

documents	instances	questions
d_1	$I_1[t_1, t_2, t_3]$	$\delta_1(t_1, t_2, t_3)$
		$\delta_2(t_2)$
		$\delta_3(t_2, t_3)$
	$I_2[t_1, t_3]$	$\delta_4(t_3)$
		$\delta_5(t_1, t_3)$
d_2	$I_3[t_2]$	$\delta_6(t_2)$

Table 2: Matching documents, instances and generated questions for a sample query

applying the non-parametric ranking heuristics

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

is shown in Table 3.

	qt	pm	ds
1. δ_1	3	3	$s(d_1)$
2. δ_3	2	3	$s(d_1)$
3. δ_5	2	2	$s(d_1)$
5. δ_2	1	3	$s(d_1)$
6. δ_4	1	2	$s(d_1)$
4. δ_6	1	1	$s(d_2)$

Table 3: Non-parametric ranking of questions for a sample query

4.5 Question generation

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

$$\mathcal{I} = \{\{(t_{11}, t_{11'}), \ldots, (t_{1i}, t_{1i'})\}, \ldots, \{(t_{m1}, t_{m1'}), \ldots, (t_{ml}, t_{ml'})\}\}$$

of m syntactic patterns $\mathcal{P}$:

$$\mathcal{P} = \{(s_{11}, s_{12}, \ldots, s_{1i}), \ldots, (s_{m1}, s_{m2}, \ldots, s_{ml})\}$$

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 Θ ranked according to the ranking function $r(\delta_i, \delta_j)$ is shown in Algorithm 1.

Algorithm 1 operates as follows. First, a set of pattern instances $\mathcal{I}\prime$ with at least one query term is obtained by querying the index (line 1). Next, for each instance in $\mathcal{I}\prime$, 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 δ was generated, is obtained in line 10; the score of a document containing the pattern instance I, from which the question δ 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 δ, 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.

Algorithm 1 Algorithm to generate a ranked list of clarification questions Θ 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: $\mathcal{I}\prime \leftarrow \{\forall I : \exists t, t \in Q \, and \, t \in I\}$
 2: for all $I, I \in \mathcal{I}\prime$ do
 3:   $P \leftarrow pattern(I)$
 4:   $d \leftarrow document(I)$
 5:   $T \leftarrow template(q, P, I)$
 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:     $\delta_i \leftarrow \Theta[i]$
14:     if $r(\delta, \delta_i) = 1$ then
15:       $insert(\Theta, \delta, i)$
16:     end if
17:   end for
18: end for
```

4.6 Interactive 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 Θ, 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 $q\prime = q \cup f$, where $f = I - I \cap q$; for the example above, $q\prime = \{t_i, \ldots, t_j, t_k, \ldots, t_n\} \cup \{t_p, \ldots, t_q\}$.

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.

5. 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).

5.1 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 [1]. 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 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 4.

Users were asked to provide their judgments regarding the well-formedness (column 'W' in Figure 4), interestingness (column 'I' in Figure 4) and relevance (column 'R' in Figure 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 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 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.

[1] `http://en.wikipedia.org/wiki/Category:Lists_of_people_by_U.S._state`

5.2 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:

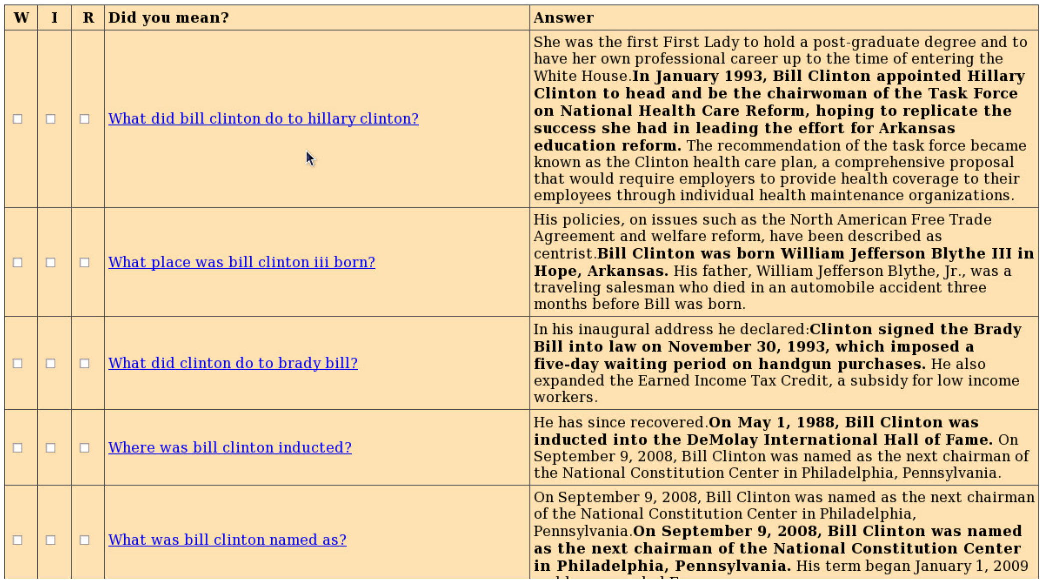
- **SQ** (short queries): short (one term only), underspecified and potentially ambiguous queries: e.g., "ford", "paris", "illinois";
- **NQ** (normal queries): well-formed, generally unambiguous, exploratory queries: "michael jackson", "bill gates";
- **LQ** (long queries): long (three or more terms), 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 4. All queries, regardless of the type, are designated as ALL.

	C	**W**	**I**	**R**	**T**
SQ	19	232	128	135	**310**
	6.13%	74.84%	41.29%	43.55%	100%
NQ	99	940	421	606	**1105**
	8.96%	85.06%	38.1%	**54.84%**	100%
LQ	11	216	122	85	**270**
	4.07%	80.0%	45.19%	31.48%	100%
FB	0	987	709	463	**1210**
	0.0%	81.57%	**58.6%**	38.26%	100%
ALL	**129**	**2375**	**1380**	**1289**	**2895**
	4.45%	82.03%	47.67%	44.52%	100%

Table 4: User judgments for different query types

There are several important conclusions, which could be made based on the analysis of Table 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,



W	I	R	Did you mean?	Answer
☐	☐	☐	What did bill clinton do to hillary clinton?	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.
☐	☐	☐	What place was bill clinton iii born?	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.
☐	☐	☐	What did clinton do to brady bill?	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.
☐	☐	☐	Where was bill clinton inducted?	He has since recovered.**On May 1, 1988, Bill Clinton was inducted into the DeMolay International Hall of Fame.** On September 9, 2008, Bill Clinton was named as the next chairman of the National Constitution Center in Philadelphia, Pennsylvania.
☐	☐	☐	What was bill clinton named as?	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

Figure 4: Fragment of a question-answers list for the query "bill clinton"

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.

5.3 Ranking and feedback

5.3.1 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 [10] 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 δ_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 $V = \langle v_1, v_2, \ldots, v_k \rangle$ computed for a list of k questions Θ that are generated by some ranking method and the DCG vector $I = \langle i_1, i_2, \ldots, i_k \rangle$, which corresponds to the ideal ranking of the same question list Θ, a normalized DCG vector is $nDCG = \langle v_1/i_1, v_2/i_2, \ldots, v_k/i_k \rangle$.

5.3.2 Evaluation of ranking

In this section, we present the results of an experimental evaluation of different question ranking strategies described in Section 4.4 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 5

As follows from Table 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.

5.3.3 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 5) and nDCG@n (Figure 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.

The steep slope of the precision curve for the feedback

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

Table 5: Performance of different ranking functions

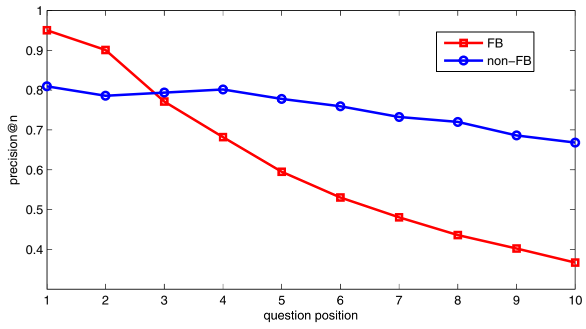


Figure 5: Precision@n for all feedback and non-feedback questions

questions in Figure 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.

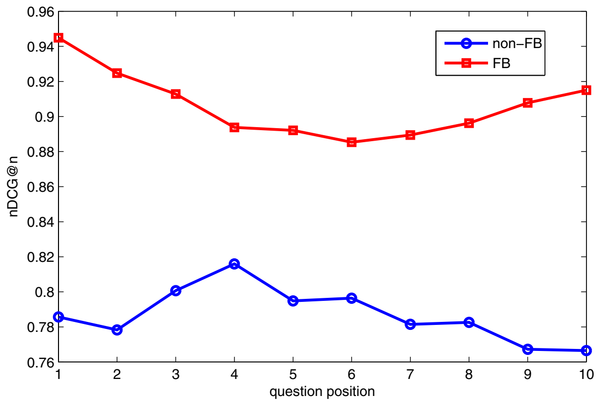


Figure 6: nDCG@n for all feedback and non-feedback questions

Figure 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.

5.4 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?

Head	Click	Inter	Relev
how	21	215	212
what	43	472	461
who	32	454	390
when	26	180	168
where	7	59	58

Table 6: User behavior with respect to different question types

- 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 6. From Table 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 click-throughs, although the difference is less pronounced, which could be partially explained by the low absolute number of click-throughs compared to the judgments. In order to answer the second question, in Figure 7 we plotted the distribution of clicked, interesting, and relevant questions across the questions of different length. From Figure 7, it follows

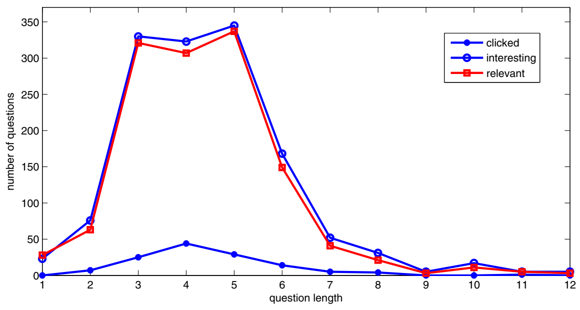


Figure 7: Distribution of clicked, interesting, and relevant questions over question lengths

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.

6. CONCLUSIONS AND FUTURE WORK

This work proposed a novel idea of question-guided search process, in which a retrieval system would automatically generate and present to the users a set of potentially interesting questions, based on the search results of a query. The generated questions can naturally supplement the standard search result presentation methods to improve the utility of search engines in two ways. First, it enables 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 of syntactic pattern instances, 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.

7. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant Numbers IIS-0347933, IIS-0713581, IIS-0713571, and CNS-0834709. We would like to thank everyone, who participated in the evaluation of QUSE, as well as the anonymous reviewers for their comments, which helped to improve the quality of this paper.

8. REFERENCES

[1] R. B. Allen, P. Obry, and M. Littman. An interface for navigating clustered document sets returned by queries. In *Proceedings of the ACM COOCS*, pages 166–171, 1993.

[2] A. Bookstein. Information retrieval: A sequential learning process. *Journal of the American Society for Information Science*, 34(5):331–342, 1983.

[3] M. Chalmers and P. Chitson. Bead: Exploration in informaion visualization. In *Proceedings of the ACM SIGIR*, pages 330–337, 1992.

[4] H. Cui, M.-Y. Kan, and T.-S. Chua. Generic soft pattern models for definitional question answering. In *Proceedings of the ACM SIGIR*, pages 384–391, 2005.

[5] R. H. Fowler, W. A. Fowler, and B. A. Wilson. Integrating query, thesaurus, and documents through a common visual representation. In *Proceedings of the ACM SIGIR*, pages 142–151, 1991.

[6] K.-S. Fu and B. K. Bhargava. Tree systems for syntactic pattern recognition. *IEEE Transactions on Computers*, C-23(12):1087–1098, 1973.

[7] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Lingustics*, pages 539–545, 1992.

[8] M. A. Hearst. *Search User Interfaces.* Cambridge University Press, 2009.

[9] M. A. Hearst and J. O. Pedersen. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *Proceedings of the ACM SIGIR*, pages 76–84, 1996.

[10] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[11] V. Jijkoun, M. de Rijke, and J. Mur. Information extraction for question answering: Improving recall through syntactic patterns. In *Proceedings of the 20th International Conference on Computational Linguistics*, 2004.

[12] B. Katz and J. Lin. Selectively using relations to improve precision in question answering. In *Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering*, pages 43–50, 2003.

[13] D. Lawrie, W. B. Croft, and A. Rosenberg. Finding topic words for hierarchical summarization. In *Proceedings of the ACM SIGIR*, pages 349–357, 2001.

[14] A. Leuski and J. Allan. Improving interactive retrieval by combining ranked lists and clustering. In *Proceedings of the RIAO*, pages 665–681, 2000.

[15] D. Lin. Dependency-based evaluation of minipar. In *Proceedings of the Workshop on Evaluation of Parsing Systems at LREC*, 1998.

[16] I. Melčuk. *Dependency syntax: theory and practice.* State University of New York Press, 1987.

[17] A. Pollock and A. Hockley. What's wrong with internet searching. In *Proceedings of the "Designing for the Web: Empirical Studies"*, 1996.

[18] A. Singhal. Modern information retrieval: A brief overview. *IEEE Data Engineering Bulletin*, 24(4):35–43, 2001.

[19] A. Spoerri. Infocrystal: A visual tool for information retrieval & management. In *Proceedings of CIKM*, pages 11–20, 1993.

[20] T. Strzalkowski, J. P. Carballo, and M. Marinescu. Natural language information retrieval. In *Proceedings of the 3rd Text Retrieval Conference (TREC-3)*, 1994.

[21] Y.-C. Wang, J. Vandendorpe, and M. Evans. Relational thesauri in information retrieval. *Journal of the American Society for Information Science*, 36(1):15–27, 1985.

[22] O. Zamir and O. Etzioni. Grouper: A dynamic clustering interface to web search results. In *Proceedings of the WWW*, pages 46–54, 1999.

[23] C. Zhai, X. Tong, N. Milić-Frayling, and D. A. Evans. Evaluation of syntactic phrase indexing - clarit nlp track report. In *Proceedings of the 5th Text Retrieval Conference (TREC-5)*, 1997.