Query Intention Acquisition: A Case Study on Automatically Inferring Structured Queries

Leif Azzopardi
Dept. of Computer and Information Sciences
University of Strathclyde, Glasgow G1 1XH
leif@cis.strath.ac.uk

Maarten de Rijke
ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam
mdr@science.uva.nl

ABSTRACT

The process of acquiring the user's intentions is an important phase in the querying process. The identification of their intentions enables the selection of appropriate retrieval strategies. In this paper, we first outline this cross-section work in contextual Information Retrieval. We then focus on one particular type of intention: the syntactic and semantic types associated with a query term. We present a case study using the email search task of the TREC Enterprise Track. We build and analyze a data set of query intentions linked to the email's structure, and then attempt to automatically infer structured queries and study the affect that ambiguity of queries and the difficulty of inferring them has on various retrieval models (structured and unstructured). Our study reveals that predicting the intentions is a hard problem due to the inherent uncertainty within the querying process. We also show that automatically inferred queries do not outperform other types of structured retrieval models, because they are not robust enough to handle the ambiguity nor reliable enough to be accurately inferred.

1. INTRODUCTION

It has been a long recognized problem that the query submitted to an information retrieval (IR) system is a sparse and impoverished representation and expression of a user's actual information need [17]. The problem stems from the series of translations that are undergone when an information need first arises and then is surmised as a two or three keyword query. Much of the meaning or intent of the user's information need is lost. For example, a user wanting the homepage of Vandalay Industries, may submit the query, 'Vandalay Industries' or 'Vandalay Industries homepage'. In the first case, any indication of the user wanting a specific home page is lost. Whilst in the second this intention is present, but will usually be treated as another keyword in the query.

A contextual IR system will attempt to develop a better understanding of the user's underlying information need through what we refer to as, *query intention acquisition* (QIA): the process of acquiring a query and analyzing the query to extract the meaning, semantics and nature of the query. The goal of QIA is to find out what intentions the user has, why they formulated that query, and why they chose those query terms. This represents a shift away from the uniform treatment of queries to approaches that utilize the structure within and of queries in order to capture more of the user's intentions (i.e., a move from key word based queries to more structured queries). This should lead to a better understanding of a user's actual information needs by the system, which if utilized effectively can facilitate improvements in retrieval effectiveness [12]. Thus, QIA is an important component of any contextual IR system.

QIA can be performed either in an explicit questioning of the user by providing them with the capability to more adequately describe their information need [12], or implicitly through inferences and assumptions based on query characteristics [5, 9, 4]. Explicit capturing of the query information can be achieved either through using a formal query language for expressing queries or through application specific interfaces (see Figure 1 for examples). In the former, the user is able to express a more precise information need by constructing a query using a given language (such as Boolean Expressions, XPath, XQuery, SQL, etc). While for the interface approach the same is achieved by having the user populate fields, select tabs, select from drop downs, and so on, to apply filters and constraints and so forth to the search. Both are intuitively appealing for QIA, all be it for different reasons, expressiveness and ease of use, respectively. There *appears* to be a trade off between the two. E.g., a highly expressive language would decrease the ease of use because more training and effort is required in issuing an explicitly structured query. Whilst for an interface to provide more expressiveness it becomes more cumbersome. Further, both of these approaches suffer from two major drawbacks, complexity and lack of usage.

The added complexity involved in creating a structured query is time consuming. This is a significant problem, which in the case of formal languages is compounded by the requirement of formulating valid and syntactically correct queries. At INEX,[1] structured XPath queries proved difficult even for competent users[2] to formulate and construct [14]. Even simple query constructions like Boolean Expressions are error prone and infrequently used despite

DIR '06 Delft, The Netherlands

[1] INEX: Initiative for the Evaluation of XML retrieval, see http://index.is.informatik.uni-duisburg.de/
[2] Researchers from computer science and related disciplines.

Information Need:
Retrieve the email that Makici wrote in October to Maillie that was titled, Multimedia.

Query:
Multimedia in October by Makici

INEX XPath Query:

```
//Email[ subject = "Multimedia" and from = "Makici"
          and date="October"]
```

Interface:

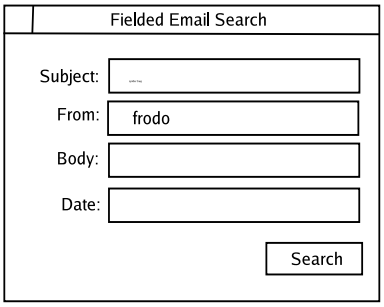

Figure 1: Examples of different ways of expressing an information need.

wide-spread implementation. In terms of usage, when advanced search functionality is provided in digital libraries [9] and on web search engines [3], they are very rarely used. Presumably, this is because the increased time and complexity rarely produces better retrieval performance. The burden and expense in submitting a more precise or better expressed query needs to be mitigated by techniques which are more ubiquitous in the acquisition of query intentions. So instead of performing QIA in such an explicit manner there has been a move towards developing techniques that automatically or implicitly attempt to infer the intents—and this is where this paper's contribution lies. We present a case study on automatically inferring structured queries, thus identifying the semantic types associated with query terms.

The remainder of the paper is organized as follows. In the next section, we outline different types of intentions to provide some background. Section 3 describes the specific query intentions we aim to study and what work has already been done in this area. We then present our case study in the remaining sections: we examine the influence of structure and intentions on the effectiveness of different types of structured retrieval models. We conclude with a discussion of our key findings in Section 7.

2. TYPES OF INTENTIONS

Consider the information needs and queries in Table 1. Each conveys different types of intentions that a user has about their information need. The intentions behind a query vary depending on the information need and stipulate the conditions of relevance. The types of intentions may include, but are not limited to: what type of document format the user wants, what unit of retrieval is sufficient, the type of search required, the pitch of the information desired, and the meaning of a query term in the query. We briefly discuss each of these in turn.

Information Need	Query
I want the homepage of Vandalay Industries	Vandalay Industries homepage
I want to know the syntax for a for loop in c++	for loop in c++
Retrieve the email that Makici wrote in October to Maillie about Multimedia	Multimedia in October by Makici
Find me images of Britney Spears in a School Uniform	Britney Spears jpeg

Table 1: Example Information Needs and Queries.

- **Document type**: Given the variety of documents that are available, users may occasionally be interested in only particular types of document. Common document types often sought after in web, desktop and enterprise search include, emails, minutes, reports particular document formats like PDF, HTML and Microsoft Word file, images, movies, and so forth.
- **Unit of retrieval**: This is related to the document type, but is concerned with the part of a document that should be retrieved. For instance, a user may only want snippets, summaries, sections, passages of documents, citations, the full text, etc. Such intents have been considered as part of INEX.
- **Search task**: The search type (ad hoc, known item, etc) query prediction for retrieval strategy selection; i.e., a web searcher submits a query, where they may be looking for a home page, or many pages relating to a topic. Classifying queries by such intentions has been considered in the late Web track at TREC and more recently in [4].
- **User expertise**: The identification of the user's level of expertise has some influence on what is considered relevant by the user. Identifying and using this information is considered in the HARD track at TREC.
- **The semantics and syntax of the query and query terms**. Examination of the query constituents may provide evidence to suggest many of the above types of intentions. However, it is usually concerned with detecting syntactic and semantic knowledge, such as noun-phrases, term dependencies, what field a query terms refers to, which is used by the retrieval model.

It has been generally posited that knowledge of the above factors can be used to increase the effectiveness of a retrieval system, because the retrieval strategy can be tailored specifically to the retrieval scenario. This area represents a wide cross-section of research performed in contextual Information Retrieval. For the case study we shall examine only the latter type of intention acquisition.

3. INFERRING QUERY TERM INTENTS

There are two main types of intents that are often implicitly inferred by an IR system: ones related to *syntactic* features (like term dependencies), and ones related to *semantic* features (like what field a query term refers to).

One of the initial attempts in structuring queries was by Croft *et al.* [6]. They consider structure to be phrases within

the document. The natural language queries are taken and converted to boolean queries incorporating the extracted phrasal information. They automatically identify phrases by employing three different methods; using a parser based primarily on phrase syntax; a stochastic approach using part of speech information; and, a dictionary of phrases. They show that the retrieval performance using structured queries is more effective than unstructured queries. Further, they showed that automatically structured queries were as effective as structured queries [6].

Several Language Modeling approaches have been proposed over recent years that attempt to exploit dependencies between terms [15, 16, 8]. These focus on the syntactical relationships between query terms defined by co-occurrence data from the collection. Such relationships are usually determined by finding a Maximum Spanning Tree of the query term dependencies and assuming that these terms were dependent accordingly. The probability of producing the query with the specific syntactic structure is used to rank the documents. Such methods have also provided increases in retrieval performance.

The work most relevant to our case study attempts to infer the semantic structure implied by query terms to formulate a structured query automatically. In [5], a set of possible structured queries are inferred from the original natural language query. Their approach assigns the query term to the most likely field. Then a set of structured queries is generated, where the best structured query is the one that maximizes the likelihood of the (query term, field) pairs which is assumed to corresponds to the user's intention. Goncalves *et al.* [9] investigate the effectiveness of automatically structured queries within the context of Digital Libraries containing journal articles. They follow a similar query structuring procedure as in [5] with the constraint of assigning a query term to one and only one field. The selection of the best structured query was vital to their method. Their results show that retrieval effectiveness was as good as or better than a flat query baseline for the majority of queries. However, the comparisons made in these studies do not consider any stronger baselines which utilize document structure in other ways. Nor do they consider what factors influence the structuring and retrieval. In our case study, we investigate different types of structured retrieval models (including ones similar to those used above) and determine how they perform against and with automatically structured queries of varying quality.

Despite the prevalence of structured documents available to users, there has been little other work investigating the benefits and impact that structure plays in the querying process and whether this can be reliably inferred and used effectively. One of the major barriers to such research is that there are no data sets available. In our case study, we build such a data set.

3.1 Benefits

There is a number of reasons why we would like to be able to automatically infer structured queries in a ubiquitous and seamless manner. These include:

- The simplification of the querying interface for the user. There is no requirement for the user to have to use a complicated search interface with multiple boxes or a complicated formal language model expressing structure.
- The formulation of structured queries is often an arduous task—there is limited use of advanced search facilities of my search engines [3].
- The ability to illicit a better understanding of what the user is searching for, so that the retrieval strategy can be tailored to the user's information need.
- This is a move towards bridging the semantic gap between the intent or meaning of the query terms and the information need and by using this knowledge, performance increases could be obtained.

3.2 Evaluation

Invariably, any system that attempts to predict the intent(s) of query terms will need to undergo a more detailed evaluation before being deployed in an operation setting. Specifically, we consider the following criteria:

1. **Reliability:** How accurately can we infer user's intentions from the query?
2. **Robustness:** How robust is the retrieval method with respect to incorrect inferences?
3. **Retrieval Effectiveness:** How effective are the automatically structured queries in terms of retrieval performance?

While other criteria (such as *timeliness*: how quickly can we infer the user's intentions from the query?) are important as well, the three listed above are more fundamental.

4. CASE STUDY: ENTERPRISE TRACK

In our case study we attempt to infer query term intentions in an email forum and consider the difficulty in predicting the intentions of query terms, their ambiguity and the influence this has on different structured retrieval models.

To examine the phenomena of inferring query semantics we have chosen a recent TREC collection, the W3C Public Email Forum from the 2005 Enterprise Track and the task of known-item email searching. We have chosen this collection and task for several reasons. The Email Forum provides a collection which has structure present within the email document (i.e., *subject*, *from*, *to*, etc), which is of a semantic nature and so is suitable for our study. The task is a common search task and the collection provides over 150 example queries from which to build a data set[3] of query intentions. Also, the task is to find a specific email, so reconciling the query terms to the email fields will be possible.

With the collection and task chosen, the case study was broken into the following four steps: (1) The building the data set of query intentions (Section 4.1), (2) An analysis of the query intentions data set (Section 4.2), (3) Automatically structuring queries and classification of query intentions (Section 5), and (4) A study of the influence of ambiguity and the difficulty inferring queries has on various retrieval models, structured and unstructured (Section 6).

4.1 Building a Query Intentions Data Set

The email sub-collection in the W3C corpus (called "lists") contains approximately 170,000 emails posted to the W3C

[3]This data set of query intentions will be in XML and made available from the first author's website.

forums over several years; other non-email documents (such as administrative and navigational pages) in the lists collection were excluded which amounted to the removal of about 30,000 documents. The TREC topics KI1-25 and KI26-KI50 were concatenated to form 150 known item queries.

Each query term for a particular query was manually tagged with the fields in the known email from which the query originates. The possible tags for email fields were: *date*, *from/author*, *subject* and *body*, all other fields were ignored. The assignment of a query term to the email's field was done according to which field was the most salient. By saliency, we refer to how obvious and memorable that field is in an email. The order used was date, author, subject and body, unless surrounding terms indicated otherwise. For instance, if the term 'June' was present in both the date of the email and the author of the email, then it was assigned to the date. Unless some other evidence such as a surname was present or there was an indicator like 'by June' as opposed to 'in June'. If the query term did not occur in the email then it was classed as being "about" one of the possible fields. For instance, referring to persons using a nick name (as shown in Query KI44 in Figure 2, where 'James' is used instead of 'Jim').

Stop words were ignored except for those that seem to indicate topicality ('on', 'in', 'for'), date ('on', 'in'), authorship ('by', 'from'), format of email ('minutes', 'call for papers'); those were tagged as `<T>`, `<A>`, `<B>`, and `<C>`, respectively. Non-text indicators, such as apostrophes, dashes, slashes, and commas were also tagged (`<D>`).

Whilst we have assumed that the query terms have come from a specific field in the email message, this is not necessarily the case in reality. A very frequent term in the email may be in both the body and the subject and chosen by the user because of its overall popularity (or recall-ability) within the email. However, we feel that assigning to the most salient feature of the email is a reasonable approximation under a hard classification. In practice, such hard assignments may not necessarily be employed—this all depends on the retrieval method.

In Figure 2, query KI11 consists of keywords that occur in the fields and their meaning is unambiguous with respect to the known-item email. As there are no other features in the query or extraneous terms there is little loss, except that there is no explicit marking of what fields the query terms refer to. However, the term 'minutes' might possibly suggest an email formatted in such a style or an attachment. The second query KI44 in Figure 2, though, looses several subtle intents in the query through the parsing process. The apostrophe, indicating who wrote the email, the type of email, a question to the forum, the reference to the topic. *Tomcat* with the use of 'in' and indicating what the email is concerned with by the use of 'about' perhaps to denote some vagueness in the description. Further, the query term "Tomcat" is what the known item is about but there is no actual mention of "Tomcat."

4.2 Query Characteristics

The number of times each field occurred and in how many queries is given in Table 2. Interestingly, the majority of the queries (111 out of 150) contained query terms relating to subject, and over half had some form of indicator, whilst the other features occurred somewhat less often. In the queries there were no references to who the email was to. There

Query: KI11
Text: tag minutes 9 june 2003
Parsed:tag minutes 9 june 2003
Marked up in xml:

```
<subject>tag</subject>
<subject><C>minutes</C></subject>
<date>9</date>
<date>june</date>
<date>2003</date>
```

Query: KI44
Text: James' question about the Webdav in Tomcat
Parsed: james question about webdav tomcat
Marked up in xml:

```
<author type="about">James<B>'</B></author>
<body type="about"><C>question</C</body>
<body type="about">about</body>
the
<body>Webdav</body>
<T>in</T>
<body type="about">Tomcat</body>
```

Figure 2: Examples of different ways of expressing an information need.

Table 2: Statistics on the main tags in the query set.

Field	Total Count	Total Queries
Date	19	13
From	41	24
From About	2	1
Subject	323	111
Subject About	28	22
Body	160	62
Body About	61	34
Indicators	105	76
Non word Indicators	28	25

were 91 instances where query terms were "about" a field in an email, which indicates that a considerable amount of noise is present within the queries. We refer to this noise as *ambiguity* as the query being expressed to the system contains uncertain information with respect to the target email. We classified queries according to how much ambiguity was present, using three grades: (0) not, (1) somewhat, or (3) very ambiguous. If all the features occurred in the known-item, then there is little or no ambiguity (i.e., we assumed that the query term was put there with specific reference to some field in the email). However, if more than half the query features are present in the known-item email, then there is some ambiguity in the query. If the majority of query terms do not occur in the email then the query is very ambiguous. To some extent this measure reflects the loss of recall that is experienced by a user when formulating the query; assuming that they are trying to select (remember) the exact words and phrases from the email they have in mind. The more vague the user is about their missing email, presumably the less precise and more ambiguous the query will be as a result.

In this collection of emails, we found that there were 20 very ambiguous queries, 33 somewhat ambiguous queries, and the rest were judged as not ambiguous (97).

Intuitively, we would expect that the less ambiguous a query is, the higher the retrieval performance should be as exact matching techniques will have more accurate information for ranking. For more ambiguous queries, then, the classification of such terms will degrade the accuracy in obtaining the correct intent of the query term. The incorrect structuring of a query could then lead to a serious degradation in retrieval performance, if the retrieval method is not robust enough to handle such ambiguity.

5. INFERRING QUERY STRUCTURE

To automatically create structured queries from unstructured queries we used generative language modeling techniques and decision theory to classify each query term with respect to the fields in the email. This combines and formalizes some of the existing work in a more general framework which can be applied to any type of data collection, independent of the retrieval model.

Within a collection of structured documents, let document d be a structured document which is composed of a set of components $x \in X$. The fields may be any feature (semantic, syntactic, layout) which has been indexed as part of the document representation. We assume that each field is a bag of terms and can be defined as a probability distribution over the vocabulary, such that the probability of a term given a field and document is $p(t|x,d)$. Taking the maximum likelihood estimate:

$$p(t|x,d) = \frac{n(t,x,d)}{\sum_{t',x'} n(t',x',d)},$$

where $n(t,x,d)$ is the number of times the term occurs in the field x of document d. By marginalizing over all documents in the collection, the probability of a term given a field $p(t|x)$ is obtained. This serves as a model of the terms that we expect to be generated from that field.

Now, given an unstructured query q which consists of a series of query terms $\{q_1, \ldots, q_k\}$, the aim is to assign each query term to the corresponding fields within documents and thus form a structured query. A structured query is defined in a manner similar to structured documents. The structured query q^s is a set of sets of query terms q^s_x, one set for each query field $x \in X$. For instance, given the email example where X = {subject, from, to, date, body}, the query q = {'Multimedia', 'Bark', 'Maillie', 'Yurat', 'Makici'} is transformed into the structured query $q^s = \{q^s_{subject} =$ {'Multimedia'}, q^s_{from} = {'Bark', 'Maillie'}, q^s_{to} = {'Yurat', 'Makici'}, $q^s_{date} = \{\}$, $q^s_{body} = \{\}$ $\}$.

5.1 Classification

To classify a given query term, we employ the odds ratio [7] to decide whether the query term q_i was from the field x or not, i.e., $\bar{x}$. Formally, we express this as:

$$O(x,q_i) = \frac{p(x|q_i)}{p(\bar{x}|q_i)},$$

where $p(\bar{x}|q_i) = 1 - p(x|q_i)$.

We wish to determine which field each of the query terms belongs to or is associated with. That is, we wish to infer the structure of the query. We consider the problem from two angles, one where each query term is treated independently and one where we treat the query as a sequence of terms where the dependence between terms is considered.

Independence Model. Here, the query terms are assumed to be independent of each other. We wish to determine the probability of the component (or class) given the query term q_i, i.e., $p(x|q_i)$ for each component x.

This can be estimated by invoking Bayes' theorem:

$$p(x|q_i) = \frac{p(q_i|x)p(x)}{p(q_i)},$$

where $p(q_i) = \sum_x p(q_i|x)p(x)$.

Dependence Model. Here, each query term is dependent of the preceding query term (strict and limited). We wish to determine the probability of the element (or class) given the query terms q_i and q_{i+1}, i.e., $p(e|q_i, q_{i+1})$ for each element e.

This can, again, be estimated by invoking Bayes' theorem:

$$p(x|q_{i+1}, q_i) = \frac{p(q_{i+1}, q_i|x)p(x)}{p(q_{i+1}, q_i)}.$$

Applying the chain rule to $p(q_{i+1}, q_i|x)$ gives

$$p(x|q_{i+1}, q_i) = \frac{p(q_{i+1}|q_i, x)p(q_i|x)p(x)}{p(q_{i+1}, q_i)}$$

and

$$p(x|q_{i+1}, q_i) = \frac{p(q_{i+1}|q_i, x)p(q_i|x)p(x)}{\sum_{x' \in X} p(q_{i+1}|q_i, x')p(q_i|x')p(x')}.$$

Since $p(q_{i+1}|q_i, x)$ is very sparse, we opted to weaken the strict order dependence and compute $p(q_{i+1}|q_i, x)$ proportional to the number of times the terms co-occurs in a window of size of two. (See [2] for further details on computing the conditional probabilities.)

Assigning Query Terms to Fields. Finally, structured queries were created by using two strategies: *strict*—where the query term was assigned to the field x, which maximized $O(x, q_i)$—, and *fuzzy*—where the query term was assigned to the fields, where $O(x, q_i)$ for the field x is greater than some threshold ρ. The threshold enables the assignment of a term to multiple elements.

5.2 Classification Results

The inferred semantic type for each query term was compared to the set of 'ground truth' tags from the manual classification process. After parsing the queries the total number of tagged terms was 685. The break down of each class was: date 19, from 42, subject 323, body 145 and unknown 156. The unknowns immediately resulted in classification failure and so are not reported. However, this meant that 22.8% of the query terms were essentially noise in the query. The classification accuracy performance is reported in Table 3 for statistics on a class by class basis. As a baseline, we assumed a naive model that assigns query terms to the most probable class (i.e., subject). The overall classification accuracy for the baseline was 61.1%. The independence and dependence models performed only marginally better, achieving only 62.0% and 62.4 %, respectively.

We further examined each query and classified the query with respect to the difficulty in predicting correctly the query

Table 3: Classification Accuracy shown as a percentage (%) correct per class on the Independence and Dependence Classification models.

	Classified as :				
	Date	**From**	**Subject**	**Body**	Out of:
Date	**84.2** (**84.2**)	5.3 (5.3)	0.0 (0.0)	10.5 (10.5)	19
From	11.9 (9.5)	**78.6** (**78.6**)	0.0 (2.4)	9.5 (9.5)	42
Subject	2.8 (2.5)	2.5 (2.8)	**72.4** (**73.7**)	22.3 (21.1)	323
Body	0.7 (0.7)	2.1 (2.1)	66.2 (67.6)	**31.0** (**29.7**)	145
Total					529

term intents. Three groups (*easy*, *medium* and *hard*) were obtained by assigning all queries that had at most one incorrectly labeled query term as easy. Queries which had at most half their terms labeled incorrectly were classified as medium, and all other queries were classified as hard. This resulted in: 47 easy, 37 medium, and 65 hard queries.

When we compared the difficulty of inferring the structure of a query against the ambiguity of a query, using a χ-squared test we found that the two groups were actually independent ($p < 0.0001$). This appears to be because the number of unambiguous queries were often difficult to predict (either medium or hard). Hence, these are two independent factors which could impact retrieval performance. Note, that the difficulty is (potentially) only a problem when using inferred structured queries—and not for other retrieval models. However, the difficulty could still be indicative of the performance, regardless of the type of retrieval model.

6. LANGUAGE MODELS AND STRUCTURE

Generative Language Models have been applied successfully in a number of tasks in IR, including structured retrieval [10]. They provide several related models that incorporate structure in various ways. Since the models are related the differences are clear from their formulation and enable a fair discussion and comparison over different experimental factors. Below we give an overview of three different Language Modeling approaches. The first is the standard query likelihood approach to retrieval which does not make any structural assumptions about the query or documents [10]. The latter models incorporate the structure of the query in the ranking of documents in distinctly different ways [13, 11, 1]. The first relies on a combination of evidence to produce a better document model and the second form structured queries with which to query the structured emails.

The *Standard Language Modeling approach* computes the probability of a query q being generated from a document model θ_d on behalf of the document d as follows:

$$p(q|\theta_d) = \prod_{t \in q} \left\{ (1-\lambda) p(t|d) + \lambda p(t) \right\}^{n(t,q)}, \qquad (1)$$

where $p(t|d)$ is the maximum likelihood estimate of term t in document d, $p(t)$ is the unconditional probability of t (also using the maximum likelihood estimate), $n(t,q)$ is the number of times term t occurs in query q, and λ is the smoothing parameter.

The *Combination Language Modeling approach* is an extension of the standard approach [13, 11]. It combines the different fields of a document to form one smoothed document model. The document model becomes a combination over each field within the document, and then the document model is further smoothed by the background collection model:

$$p(q|\theta_d) = \prod_{t \in q} \left((1-\lambda) \sum_{x \in X} \left\{ p(t|x,d) p(x|d) \right\} + \lambda p(t) \right)^{n(t,q)}$$

Each field in the document is weighted by $p(x|d)$, which can be interpreted as an indicator of the importance of that field in representing the document. This model has been highly successful for structured retrieval, despite only using structure on the document side.

The *Fielded Language Modeling approach* is the general solution where the joint probability of the components given the structured document needs to be estimated and a structured query [1]. The simplest approach is to assume that each field is independent of the other. However, depending on the structured document and the task, this assumption may be relaxed to account for the relationship between fields. For example, given an email, the subject is dependent on the body, the body dependent on the author and so forth. In reality it may be infeasible to compute such dependencies between the fields; thus we must resort to the independent Fielded Language Model. It is a simple extension of the standard Language Modeling approach as it treats each field of the email document as an independent source of evidence. From each field, query terms are drawn which generated the structured query. Formally, this can be represented as:

$$\begin{aligned} p(q|d) &= p(q_{x_1}, \ldots, p_{x_n} | \theta_d^x) \\ &= \prod_x p(q_x | \theta_d^x), \end{aligned}$$

where $p(q_x|\theta_d^x)$ is the probability of the query field q_x being generated from the model of the document field θ_d^x. This probability is computed as above for standard documents, but for each of the four fields instead.

Each model utilizes structure differently. Whilst the standard model ignores structure all together the others use structure in different ways. The combination LM ignores any structure in the query and focuses on building a better document representation by marginalizing over all the fields in the document to form a robust statistical estimate of that term occurring in the document. The Fielded LM treats each field independently which provides a natural mechanism for issuing structured queries which are matched against the corresponding fields.

This provides three distinct approaches for dealing with and using structure in the IR process and should enable us to study the impact of ambiguity and difficulty given these retrieval models.

6.1 Experiments

The three Language Models were configured as follows: The Standard LM with the smoothing operator λ set to 0.1

Table 4: The retrieval performance of each of the different retrieval models.

	LM	Setting	MRR					
			Overall	Ambiguity		Difficulty		
				None	Some/Very	Easy	Medium	Hard
a	**Standard**		0.466d	0.537	0.337	0.501	0.409	0.470
b	**Combination**	**uniform**	**0.631**ade	**0.719**	**0.469**	0.701	0.636	**0.578**
c		**automatic**	0.627ade	**0.719**	0.458	**0.705**	**0.638**	0.565
d	**Fielded**	**strict**	0.355	0.436	0.208	0.516	0.455	0.186
e		**fuzzy**	0.546ad	0.667	0.325	0.693	0.574	0.425
f		**explicit**	0.581ad	0.679	0.400	0.687	0.425	0.479

as the baseline model.[4] All other models used the same lambda to try and ensure a fair comparison amongst the different language models. The Combination LM was set to *uniform* or *automatic*. *Uniform* refers to when the prior probability of all fields is uniform i.e., $p(x|d) = 0.25$, and *automatic* refers to when the prior probability of a field is set with respect to the number of the query terms that were classified as a particular field $n(t, x)$, i.e., $p(x|d) = n(t, x)/\sum_{x'} n(t, x')$. This latter assignment uses the inferred intents when estimating each fields importance with the document. The Fielded LM was set to *strict*, *fuzzy* or *explicit*, which refer to the query intent information that was provided. *Strict* and *fuzzy* are with respect to the automatically inferred queries, where *strict* assigns one field label, whereas *fuzzy* assigns multiple field labels (with $\rho = 0.1$). The *explicit* setting refers to when the actual query tags are used (i.e., fully explicit labeling of the query terms and their intents).

6.2 Retrieval Results

In Table 4 we show the retrieval performance for each of the different retrieval models in terms of the Mean Reciprocal Rank (MRR), including a breakdown in performance over Ambiguity and Difficulty. Note that the ambiguity scale was reduced to two levels since the number of queries which were somewhat and very ambiguous were few and so were combined. We compared each model's performance using the Wilcoxon Signed Rank Test ($\alpha = 0.05$). In Table 4, if a model significantly outperformed another then the letter identifying the model being outperformed (in the left-most column) is shown to denote this. We found that the worst performing model (significantly so against all other models) was the strict Fielded LM. On the other hand, the combination LM approaches were significantly better than all other models, except when explicit information was used in the Fielded LM. The fuzzy Fielded LM significantly outperformed the standard model and was on par with the explicit Fielded LM.

7. DISCUSSION AND CONCLUSION

Through the course of our study we have examined two factors related to query intention which impact retrieval performance: *ambiguity* and *difficulty*. Our findings confirm the hypothesis that if the level of ambiguity in a query increases then the retrieval performance will degrade. The magnitude of degradation remains relatively consistent across the different retrieval models, on average 0.26 less in MRR.

[4]Other λ parameters were also tried but the results were similar to those reported here.

In general, our findings also show that as it becomes more difficult to infer the query structure, the retrieval performance degrades. However, there are a few notable exceptions. The standard model appears to be relatively robust to how hard it was to predict structure. Presumably, this is because the standard model does not make use of such information and so can not be affected by any misclassifications. When we compare the differences of the Combination LM against the Fielded LMs we also notice that, because the Fielded LMs rely upon structured queries, retrieval performance suffers more so than for the Combination LM. The most pronounced example of this is when the strict Fielded LM is employed, and there is large drop in MRR when more than half of the query terms were incorrectly classified.

The retrieval results confirm previous findings from [5, 9] that automatically structured queries outperform unstructured queries (Standard LM vs Field LM (fuzzy)). However, our message is more subtle. Compared against the standard model, the combination LMs have the advantage that they account for the structure present in the document by averaging over each field. Compared against the Fielded LMs the structured document model again wins out, for the same reason. Structure is accounted for within the document model of the combination LM, so there is no reliance on query side inferences, and hene the prediction difficulty is not a factor.

Our results show how difficult it is to ascertain the user's actual intent and then how to make good use of these intents. More research is needed to develop retrieval techniques that can handle structured queries which also improve performance. Whenever there is ambiguity and/or difficulty within the queries, this uncertainty needs to be accounted for by the retrieval model.

Further, our results suggest that email search facilities provided with email clients need not be field based, like the example in Figure 1 but could be simplified by employing the combination LM approach.

Our study shows that there are predictable habits within user querying behavior but more research needs to be performed in this area. Whilst the brute force approach would be to develop better classification methods so as to improve the structured queries produced, an alternative approach would be aimed at developing a natural language based querying language, which provides natural cues for the prediction of query terms and their intents. In a domain such as known email searching, this appears to be quite feasible. In some of our examples certain 'stop words' appear indicative of intent, such as 'by' indicating who wrote the email, 'in' or 'on' indicating the date. By using a subset of stop words, more infer-able queries could be submitted to the retrieval

system without significantly increasing the burden to the user because it is simply natural language (i.e., 'Multimedia by Mallie from Makici').

In conclusion, QIA represents an important process in any contextual IR system. However, our study has shown that acquiring the intents of query terms with respect structure is a non-trivial task which requires further research to fully develop and utilize such information. Perhaps controversially, our study has shown that whilst automatically structured queries outperform baseline models and are as good as explicitly structured queries, more sophisticated retrieval models still fair better, again suggesting that more research is required to develop models that can handle structured queries and still provide robust and superior retrieval performance.

Now that a data set has been created, future work can be directed in a number of areas, such as improving the classification accuracy using other techniques, estimates and indicators, improving retrieval effectiveness by considering and developing more robust structured retrieval strategies.

Acknowledgments

Maarten de Rijke was supported by grants from the Netherlands Organization for Scientific Research (NWO) under project numbers 017.001.190, 220-80-001, 264-70-050, 354-20-005, 612-13-001, 612.000.106, 612.000.207, 612.066.302, 612.069.006, 640.001.501, and 640.002.501.

8. REFERENCES

[1] L. Azzopardi, K. Balog, and M. de Rijke. Language Modeling Approaches for Enterprise Tasks. In *Proceedings of the 14th TExt Retrieval Conference (TREC 2005)*, 2006.

[2] L. Azzopardi, M. Girolami, and M. Crowe. Probabilistic hyperspace analogue to language. In *Proceedings of the 28th Annual International ACM SIGIR conference on Research and development in information retrieval*, pages 575–576, New York, NY, USA, 2005. ACM Press.

[3] R. Baeza-Yates. Query usage mining in search engines. In A. Scime, editor, *Web Mining Applications and Techniques*. Idea group, 2004.

[4] M. Bomhoff, T. Huibers, and P. van der Vet. User intentions in information retrieval. In *DIR '05: Proceedings of the 5th Dutch Beligian Workshop in information retrieval*, pages 47–54, 2005.

[5] P. Calado, A. S. da Silva, R. C. Vieira, A. H. F. Laender, and B. A. Ribeiro-Neto. Searching web databases by structuring keyword-based queries. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 26–33, New York, NY, USA, 2002. ACM Press.

[6] W. B. Croft, H. R. Turtle, and D. D. Lewis. The use of phrases and structured queries in information retrieval. In *SIGIR '91: Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 32–45, New York, NY, USA, 1991. ACM Press.

[7] M. DeGroot. *Optimal Statistical Decisions*. McGraw-Hill, New York, 1970.

[8] J. Gao, J. Y. Nie, G. Wu, and G. Cao. Dependence language model for information retrieval. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 170–177. ACM Press, 2004.

[9] M. A. Goncalves, E. A. Fox, A. Krowne, P. Calado, A. H. F. Laender, A. S. da Silva, and B. Ribeiro-Neto. The effectiveness of automatically structured queries in digital libraries. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, pages 98–107, New York, NY, USA, 2004. ACM Press.

[10] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, Enschede, 2001.

[11] J. Kamps, G. Mishne, and M. de Rijke. Language models for searching in web corpora. In *The Thirteenth Text Retrieval Conference (TREC 2004)*, volume SP 500-261 of *NIST Special Publication*, 2005.

[12] D. Kelly, V. D. Dollu, and X. Fu. The loquacious user: a document-independent source of terms for query expansion. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 457–464. ACM Press, 2005.

[13] P. Ogilvie and J. Callan. Combining document representations for known-search. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 143–150, New York, NY, USA, 2003. ACM Press.

[14] R. A. O'Keefe and A. Trotman. The simplest query language that could possibly work. In *Proceedings of the 2nd INEX Workshop*, 2004.

[15] F. Song and W. B. Croft. A general language model for information retrieval. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 279–280. ACM Press, 1999.

[16] M. Srikanth and R. Srihari. Biterm language models for document retrieval. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 425–426. ACM Press, 2002.

[17] R. S. Taylor. *Value-added Processes in Information Systems*. Ablex Publishing, Norwood, NJ, 1968.