COVER SHEET

Woodley, Alan and Geva, Shlomo (2005) NLPX at INEX 2004. In Fuhr, Norbert and Lalmas, Mounia and Saadia, Malik and Szlavik, Zoltan, Eds. *Proceedings Proceedings Third International Workshop of the Initiative for the Evaluation of XML Retrieval*, pages pp. 382-395, Dagstuhl Castle, Germany.

NLPX at INEX 2004

Alan Woodley, Dr. Shlomo Geva

Centre for Information Technology Innovation
Faculty of Information Technology
Queensland University of Technology
GPO Box 2434 Brisbane Q 4001 Australia

{ap.woodley@student.qut.edu, s.geva@qut.edu.au}

Abstract. Users of information retrieval (IR) systems require an interface that is powerful and easy-to-use in order to fulfill their information requirement. In XML-IR systems this is a non-trivial task since users expect these systems to fulfill both their structural and content requirements. Most existing XML-IR systems accept queries formatted in formal query languages, however, these languages are difficult to use. This paper presents NLPX – an XML-IR system with a natural language interface that is user friendly enough so it can be used intuitively, but sophisticated enough to be able to handle complex structured queries. NLPX accepts English queries that contain both users' content and structural requirements. It uses a set of grammar templates to derive the structural and content requirements and translates them into a formal language (NEXI). The formal language queries can then be processed by many existing XML-IR systems. The system was developed for participation in the NLP Track of the INEX 2004 Workshop, and results indicated that natural language interfaces are able to capture users' structural and content requirements, but not as accurately as some formal language interfaces.

1 Introduction

The widespread use of Extensible Markup Language (XML) documents in digital libraries has led to development of information retrieval (IR) methods specifically designed for XML collections. While traditional IR systems are limited to whole document retrieval, XML-IR systems can retrieve very precise and very relevant material, thereby satisfying both the content and structural needs of user. However, XML-IR systems must understand users' content and structural requirements before they can satisfy them. Thus, XML-IR systems require an interface that is powerful enough to thoroughly express users' information need, but user-friendly enough that it can be used intuitively.

Historically, two types of queries have been supported by the INEX Workshop: Content Only (CO) and Content and Structure (CAS), each with their own interface. CO queries only express users' content requirements so their interface consists of a list of keywords. In comparison, CAS queries express both the structural and content requirements of users, and therefore require a more sophisticated interface. To meet

this requirement CAS queries have been formatted using complex query languages (XPath [2] in 2003, NEXI [12] in 2004). Unfortunately, in a structured IR system neither interface optimally addresses users' needs. Keyword based systems are too simplistic, since they do not allow users to expresses their structural requirements. Alternatively formal query languages are too difficult to use, and require users to have an intimate knowledge of a documents' structure.

In this paper we present an alternative interface for XML- IR systems, NLPX, that allows users to express their need in natural language. This type of interface is very applicable to the INEX collection since each topic already contains a description element that expresses users' content and structural needs in natural language. While there already exists an extensive body of research into natural language processing in traditional IR, largely thanks to The Text Retrieval Conference (TREC) [11] and the Special Interest Group for Information Retrieval (ACM-SIGIR) [10], work on an XML-IR interface is still largely undocumented and many challenges remain unsolved.

This paper identifies some of these challenges and presents some solutions. We begin by outlining some of the motivating factors behind the development of a natural language interface for XML-IR systems. We then present our own NLPX system, which participated in the INEX 2004 NLP Track, including the methodology used to process Natural Language Queries (NLQs), how we tested the system and finally our results from the INEX 2004 Workshop. We conclude with a short discussion on where our system is headed and how the number of participants in the NLP track can increase.

2 Motivation

This section outlines several motivating factors behind the development of a natural language interface for XML-IR systems. As these factors are specific to when users specify both structural and content requirements they are more closely related to the CAS than CO task. Motivating factors that are more closely related to the CO task, or other keyword based system are already covered in publications of The Text Retrieval Conference (TREC) [11] and the Special Interest Group for Information Retrieval (ACM-SIGIR) [10].

The main motivation is that formal query languages are too difficult for users to accurately express their information need. A very good example of this occurred at the INEX 2003 Workshop: More than two-thirds of the proposed queries had major semantic or syntactic errors [6]. Furthermore, the erroneous queries were difficult to fix, requiring 12 rounds of corrections. Therefore, if experts in the field of structured information retrieval are unable to correctly use complex query languages, one cannot expect an inexperienced user to do so. However, we feel that users would be able to intuitively express their information need in a natural language.

The second motivation is that users require an intimate knowledge of a document's structure in order to properly express their structural requirements. It is likely that most users will understand the conceptual document model of a particular collection.

For instance they will know that most journal articles start with elements such as titles and authors, have a body with sections and paragraphs, and finish with a list of references. Therefore, it is likely that users would request elements such as sections and paragraphs from articles. However, they are not able to express these requests in a formal language without knowing the physical document structure. While this information may be obtained from a document's DTD or Schema there are situations where the proprietor of the collection does not wish users to have access to those files. However, in a natural language interface the underlying document structure can be completely hidden from users, who only require a conceptual model of a document when formulating queries. So instead of requesting sec and p tags, users will be able to explicitly request sections or paragraphs. Then the system can map the users semantic request to a specific tag,

The final motivation is that formal queries do not scale well across multiple or heterogeneous collections, even if the collection falls within a single domain. Again we use the domain of journal articles as an example where multiple collections are conceptually equivalent. However, it is unlikely that multiple collections will have exactly the same DTD or Schema. For instance, one collection may use the tag <*p*> to denote paragraphs while another collection may use the tag <*para*>. A similar situation exists in a heterogeneous system where multiple DTDs or Schemas are used in the same collection. However, this problem can be resolved via natural language since as noted in the previous motivation, users will express their information needs conceptually and will have no need to know the physical underlying document structure.

3 NLPX System

Figures 1 and 2 show examples of CO and CAS topics.

```
<inex_topic topic_id="XX"
query_type="CO">
<title>
 "multi layer perception"
 "radial basis functions"
 comparison
</title>
<description>
 The relationship and compari-
 sons between radial basis
 functions and multi layer per-
 ceptions
</description>
</inex_topic>
```

Fig 1. A CO Topic

```
<inex_topic topic_id="XX"
query_type="CAS">
<title>
 //article[about(.,information
 retrieval)]//sec[about(., com-
 pression)]
</title>
<description>
 Find sections about compress-
 ion in articles about infor
 mation retrieval.
</description>
</inex_topic>
```

Fig. 2. A CAS Topic

Both the description and title elements express the users' information need. The description expresses users' need in a natural language (e.g. English). The title expresses

users' information need in either a list of keywords/phrases (CO) or as a formal XPath-like language (CAS) called Narrowed Extended XPath I (NEXI) [12].

We developed our natural language interface to accept the description element from an INEX topic. This was the obvious choice, since the description is defined as a faithful 1:1 natural language translation of the title element (although, as we will explain later in this paper, this is not always the case). We had already developed a system for participation in the Ad-hoc track. Therefore, instead of developing a completely new system for participation in the NLP track, we developed a natural language query to NEXI translator. We did this for three main reasons:

First, by developing a NLQ-to-NEXI translator we were able to use our Ad-hoc system as a backend retrieval engine.

Secondly, since the description element is a 1:1 Translation of the title element, NLP systems are able to use the existing set of Ad-hoc topics and assessments for internal testing. Furthermore, future NLP tracks can be officially evaluated using the same topics and assessments as future Ad-hoc tracks, resulting in very little extra work for future INEX organisers.

Finally, we can output the translated queries and compare them with the original NEXI queries to evaluate the successfulness of the translator.

The syntax of NEXI is similar to XPath, however, it only uses XPath's descendant axis step and extends XPath by incorporating an 'about' clause to provide an IR-like query. NEXI's syntax is //A[about(//B,C)] where A is the context path, B is the relative path and C is the content requirement.

Note that a single query may contain more than one information request. For example, the query in Figure 2 contains two requests, one for sections about compression, and a second one for articles about information retrieval. NEXI handles this scenario by using multiple about clauses. Also note that the user only wishes to retrieve elements matching the fist request, that is, sections about compression rather than entire articles. We refer to these requests and their elements as 'return requests' and 'return elements', which corresponded with NEXI's leaf (that is rightmost) about clause. In contrast, elements matching the second request, that is, articles about information retrieval, are used to support the return elements in ranking. We refer to these requests and their elements as 'support requests' and 'support elements'.

The following subsections describe how a natural language query is first translated to NEXI format and how a NEXI query is then handled by the backend system.

3.1 Natural Language Queries to NEXI Translator

Suppose that the natural language queries (NLQ) in Figure 3 are input into the system.

NLQ 1: The relationship and comparisons between radial basis functions and multi layer perceptions

NLQ 2: Find sections about compression in articles about information retrieval

Fig. 3. CO and CAS Natural Language Query

3.1.1 Lexical and Semantic Tagging. Suppose that the contents of Figure 3 are input into the system as natural language queries (NLQ). Translating the NLQs into NEXI takes several steps. First each word is tagged either as a special connotation or by its part of speech. Special connotations are words of implied semantic significance within the system. The special connotations were developed by inspection of previous INEX queries and are stored in a system dictionary. Our system uses three types of special connotations: structural words that indicate users' structural requirements (e.g. article, section, paragraph, etc.), boundary words that separate the users' structural and content requirements (e.g. about, containing) and instruction words that indicate whether we have a return or support request. All other words are tagged by their part of speech. In theory, any part of speech tagger could perform this task; however, our system uses the Brill Tagger [1]. The Brill Tagger is a trainable rule-based tagger that has a success rate comparable to state of the art stochastic taggers (>95%). The Brill Tagger defines tags as specified by the Penn Treebank [8]. Figure 4 presents some of the tags used in the Penn Treebank and Figure 5 presents the tags that denote a special connotation. Figure 6 shows examples of the NLQs after tagging.

CC	Coordinating Conjunction
DT	Determiner
IN	Preposition / Subordinating Conjunction
JJ	Adjective
NN	Noun, Singular or Mass
NNS	Noun Plural

Fig. 4. Some Penn Treebank Tags

XIN	Instruction Word
XST	Structural Word
XBD	Boundary Word

Fig. 5. Special Connotation Tags

NLQ 1: The/DT relationship/NN and/CC comparisons/NNS between/IN radial/JJ basis/NN functions/NNS and/CC multi/NNS layer/NN perceptions/NNS

NLQ 2: Find/XIN sections/XST about/XBD compression/NN in/IN articles/XST about/XBD information/NN retrieval/NN

Fig. 6. Tagged CO and CAS Natural Language Queries

3.1.2 Template Matching. The second step of the translator is to derive information requests from tagged NLQs. This is performed by matching the tagged NLQs to a predefined set of grammar templates. The grammar templates were developed by inspection of previous years' INEX queries. Initially it may seem that a large number of templates would be required to fully capture the semantics of natural language. However, NLQs share the same narrow context, that is, information requests, thus, comprehending NLQs is a subset of classical natural language understanding. Therefore, a system that interprets natural language queries requires fewer rules than a system that attempts to understand natural language in its entirety. This theory was verified by the inspection of previous INEX queries and recognising that the format of most queries corresponded to a small set of patterns. By extracting these patterns we

were able to formulate a small set of grammar templates that match the majority of queries. Figure 7 shows an example of some of the grammar templates.

Query: Request+
Request : CO_Request | CAS_Request
CO_Request: NounPhrase+
CAS_Request: SupportRequest | ReturnRequest
SupportRequest: Structure [Bound] NounPhrase+
ReturnRequest: Instruction Structure [Bound] NounPhrase+

Fig. 7. Grammar Templates

Conceptually, each grammar template corresponds to an individual information request. Once the semantically tagged text was matched to a grammar template, we derived information requests from the query. Each information request contains three separate attributes. **Content**: A list of terms or phrases that express the content requirements of the user. This is derived from the noun phrases from the matched grammar template. **Structure**: A logical XPath expression that describes the structural constraints of the request. This value is derived via a function that maps structural words (e.g. section) to the XML tags (e.g. /article/sec) as specified in the document's DTD. **Instruction**: "R" if we have a return request or "S" if we have a support request. Figure 8 shows examples of the derived information requests.

NLQ 1:
Structure: /*
Content: relationship, comparisons, radial basis functions, multi layer perceptions
Instruction: R
NLQ 2:

	Request 1	**Request 2**
Structural:	/article/sec	/article
Content:	compression	information retrieval
Instruction:	R	S

Fig. 8. Derived Information Requests

3.1.3 NEXI Query Production. The final step in the translator is to merge the information request into a single NEXI query. However, if a NLQ has multiple information requests, then those requests are merged into a single request and output. Return requests are formatted in the form A[about(.,C)] where A is the request's structural attribute and C is the request's content attribute. Support requests are then added to the output in two stages. First the system must locate the correct position within the output to place the support request. This will be the largest shared ancestor between the support and return requests. For example if the internal representation is X/Y/Z[about(,.C)] and the support request has a structural attribute of X/Y/A, then the structural request should be placed in position Y. Using a string matching function, a comparison is made between the internal representation and the support request structural attribute to determine the correct position of the support request. Then the re-

quest is added to the internal representation in the form A[about(B,C)] where A is the longest matching string, B is the remainder of the support request's structural attribute and C is the support request's content attribute. Figure 9 shows how the NEXI queries would appear after the information requests for each NLQ have been merged.

NLQ 1:
//*[about(.,relationship, comparisons, radial basis functions, multi layer perceptions)]
NLQ 2:
//article[about(.,information retrieval)]//sec[about(.,compression)]

Fig. 9. NLQ-to-NEXI Queries

After the NLQs have been translated into NEXI they can be input in existing INEX systems. The following section describes how the NEXI query was processed, after being input into our existing XML-IR system.

3.2 GP-XOR Backend

3.2.1 NEXI Interface. Once NEXI queries are input into the system they are converted into an intermediate language called the RS query language. The RS query language converts NEXI queries to a set of information requests. The format of RS queries is

Request: Instruction '|' Retrieve_Filter '|' Search_Filter '|' Content.

The Instruction and Content attributes are the same as they were in the previous section; however, the Structural attribute has been divided into a **Retrieve** and **Search** Filter. While both are logical XPath expressions the Retrieve Filter describes which elements should be retrieved by the system, whereas the Search Filter describes which elements should be searched by the system. These filters correspond to an information request's context and relative path. So for the NEXI query //A[about(//B,C)], the retrieve filter or context path is //A, while its search filter or relative path is //A//B. Figure 10 presents an example of the queries introduced earlier converted to RS queries.

RS Query 1:
R|//*|//*| relationship, comparisons, radial basis functions, multi layer perceptions
RS Query 2:
R|//article//sec|//article//sec|compression
S|//article|//article| information retrieval

Fig. 10. Example of an RS Query

3.2.2 System Structure and Ranking Scheme. We index the XML collection using an inverted list. Given a query term we can derive the filename, physical XPath and

the ordinal position within the XPath that it occurred in. From there we construct a partial XML tree containing every relevant leaf element for each document that contains a query term. Further information on our structure can be found in [4].

Elements are ranked according to their relevance. Data in an XML tree is mostly stored in leaf elements. So first we calculate the score of relevant leaf elements, then we propagate their scores to their ancestor branch elements.

The relevance score of leaf elements is computed from term frequencies within the leaf elements normalised by their global collection frequency. The scoring scheme rewards elements with more query terms. However, it penalises elements with terms that occur frequently in the collection, and rewards elements that contain more distinct terms. This is because terms that occur frequently in a collection are less likely to be significant than terms that only occur a few times in a collection.

The relevance score of a non-leaf is the sum of the children scores. However, leaf element scores are moderated by a slight decay factor as they propagate up the tree. Branch elements with multiple relevant children are likely to be ranked higher than their descendents – as they are more comprehensive – while branch elements with a single relevant child will be ranked lower than the child element as they are less specific. Further information on our ranking scheme can be found in [5]

4 Testing

4.1 Testing Methodology

Our initial experiments were conducted using the INEX 2003 set of topics and evaluation metrics. Our NLP system accepted the topics' description tag, translated it into NEXI and processed it using our backend GPX engine. GPX produced an INEX submission file that was input into the official INEX evaluation program (inex_eval) to calculate the recall/precision graphs, just as if it was an Ad-hoc submission. As specified by the INEX guidelines [3] the precision value was calculated over two dimensions: exhaustiveness, which measures the extent to which a component discusses the information request; and specificity, which measures the extent to which a component is focused on the information request.

Initially, we executed two runs for both the CO and CAS topic sets. The first run accepted NEXI queries, that is the topic's title tag, as input, whereas the second run accepted natural language queries, that is the topic's description tag, as input. These runs corresponded with INEX's Ad-hoc and NLP tracks and allowed us to compare how well our system performed with and without our NLP-to-NEXI frontend. It was hoped that the results of these two runs would be fairly similar, however, we identified one major obstacle that hindered our progress; the fact that the title and description were not always equivalent.

4.2 Testing Obstacles

INEX guidelines specifically state that the description tag should be as close as possible to a 1:1 natural language translation of the title [3]. However, during testing it became clear that many of the descriptions are not faithful 1:1 title translations. Therefore, it would be very difficult for a system with a NEXI interface to produce similar results as a system with a NLQ, even if they used the same backend. From our observations we have identified that many of the INEX topics had inconsistencies between the title and description elements, that is, the title and description had different content or structural requirements. Examples of these inconstancies are topic 76, 81, 93 and 104. Given these inconsistencies**,** it is unlikely that an NLP system and Ad-hoc system would produce the same results. To overcome these obstacles we modified the description elements so that they were a faithful 1:1 translation of the title elements. It must be stressed that in this modification we were very careful not to generate descriptions that favour our natural language processing algorithms. Modifications mostly ensured that the same keywords and structural tag names appeared in both the NEXI formatted topic and the natural language description. Figures 11 – 14 are examples of some of the topics with the modified descriptions.

Topic: `//article[(./fm//yr = '2000' OR ./fm//yr = '1999') AND about(., 'intelligent transportation system')] //sec[about(.,'automation +vehicle')]`
Original Description: Automated vehicle applications in articles from 1999 or 2000 about intelligent transportation systems.
Modified Description: Retrieve sections about automation vehicle in articles from 1999 or 2000 about intelligent transportation systems.

Fig. 11. Modified Topic 76

Topic: `//article[about(.//p, 'multi concurrency control') AND about(.//p, 'algorithm') AND about(./fm//atl, 'databases')]`
Original Description: We are interested in articles that can provide information or reference to information on algorithms for multiversion concurrency control in databases.
Modified Description: We are interested in articles with a paragraph about multi concurrency control or a paragraph about algorithm and a frontmatter title about databases

Fig. 12. Modified Topic 81

Topic: `"Charles Babbage" -institute -inst.`
Original Description: The life and work of Charles Babbage.
Modified Description: Charles Babbage not institute nor inst

Fig. 13. Modified Topic 93

Topic: `Toy Story`
Original Description: Find information on the making of the animated movie Toy Story, discussing the used techniques, software, or hardware platforms.
Modified Description: Find information on Toy Story

Fig. 14. Modified Topic 104

4.3 Testing Results

Overall we tested our system using three runs for both the CO and CAS topics sets: one with NEXI queries, one with the original NLQ description elements and one with the altered NLQ description elements. The plots for these three runs are displayed in Figures 15-18. A fourth plot is the recall/precision line of the University of Amsterdam's systems that achieved the best results at INEX 2003 in the CO and SCAS tracks [9]. This allowed us to compare the quality of our system with the best official INEX alternative. Two metrics were used to formulate the recall-precision values, the strict metric that evaluates highly relevant and highly precise results, and the generalized metric that evaluates results based on a graded measure (or degree) of relevancy and/or precision. Further details on the metrics are available in [3].

The plots for each task and quantisation are listed in Figures 15-18. The solid line is the Amsterdam submission, the dotted line is the Ad-hoc submission, the dashed line is the NLP submission and the dash-dotted line is the altered-NLP submission.

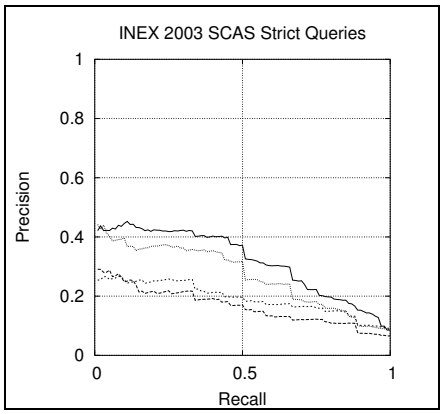

Fig. 15. INEX 2003 SCAS Strict R/P Curve

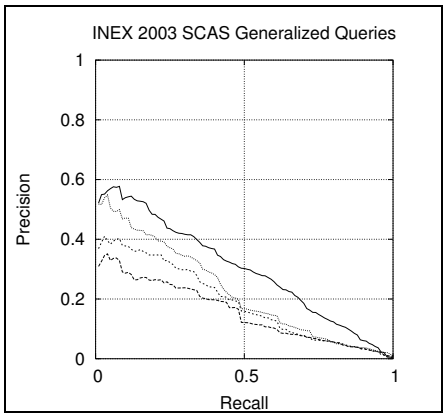

Fig. 16. INEX 2003 SCAS Generalized R/P Curve

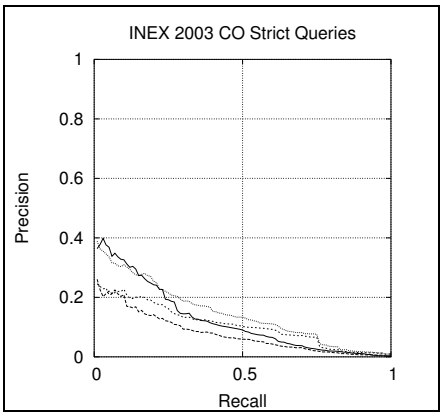

Fig. 17. INEX 2003 CO Strict R/P Curve

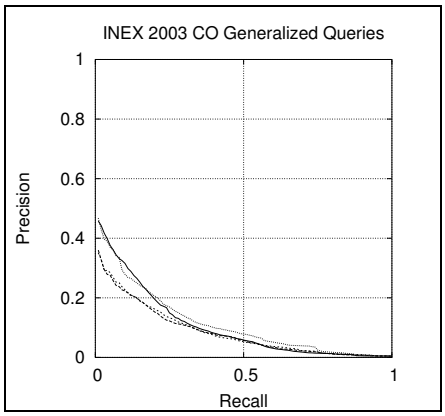

Fig. 18. INEX 2003 CO Generalized R/P Curve

5.1 INEX 2004 Examples

Here we present some examples of how well our system was able to translate NLQs into NEXI. Since none of the INEX 2004 topics were used to 'train' the system, this section is a valid example of how well the system is able to translate unseen queries. For each topic we present the original NLQ (description), the original NEXI query (title) and the translated NLQ to NEXI query. Figures 19 and 20 are example of topics that the system was able to translate successfully (topic 130 and 159). In contrast Figure 21 and 22 are examples of topics that the system had difficulty translating (topics 145 and 158).

NLQ: We are searching paragraphs dealing with version management in articles containing a paragraph about object databases.
NLQtoNEXI:
//article[about(//sec//p,object databases) //bdy//sec//p [about(.,version management)]
Original NEXI: //article[about(.//p,object database)] //p[about(.,version management)]

Fig. 19. Topic 130 Including NLQtoNEXI

NLQ: Articles about bayesian networks find sections that are about learning the structure of the network
NLQtoNEXI: //articles[about(.,bayesian networks]//sec[about(.,learning structure network)]
Original NEXI:
//article[about(.,bayesian networks)] //sec[about(.,learning structure)]

Fig. 20. Topic 159 Including NLQtoNEXI

NLQ: We are looking for paragraphs in articles about information retrieval dealing with relevance feedback.
NLQtoNEXI:
//article//bdy//sec//p[about(.,information, retrieval, relevance, feedback)]
Original NEXI:
//article[about(.,information retrieval)] //p[about(., relevance feedback)]

Fig. 21. Topic 145 Including NLQtoNEXI

NLQ: I am looking for articles where the main theme is the Turing test or "imitation game" where machines imitate human intelligence and consciousness.
NLQtoNEXI: //article[about(.,main, theme, test, imitation, game, machines, human, intelligence, consciousness)]
Original NEXI: article[about(.//fm, turing test) or about(.//abs, turing test)]//bdy[about(.,turning test consciousness intelligence imitation game)]

Fig. 22. Topic 158 Including NLQtoNEXI

5.2 INEX 2004 Results

In the Ad-hoc track our system was ranked 1st from 52 submitted runs in the VCAS task and 6th from 70 submitted runs in the CO task. In the NLP track the system was ranked 1st in the VCAS task and 2nd in the CO task. While the NLP track was limited

to 9 participants initially of which only 4 made official submissions, the most encouraging outcome was that our NLP system outperformed several Ad-Hoc systems. In fact, if the NLP submission was entered in the Ad-hoc track it would have ranked 12th from 52 in VCAS and 13th from 70 in CO. This seems to suggest that in structured IR, natural language queries have the potential to be a viable alternative, albeit not as precise as a formal query language such as NEXI.

The Recall/Precision Curves for the Ad-hoc track, along with the R/P curve for our NLP runs are presented in Figures 23 and 24. The top bold curve is the Ad-hoc curve, the lower is the NLP curve, and the background curves are of all the official Ad-hoc runs at INEX 2004.

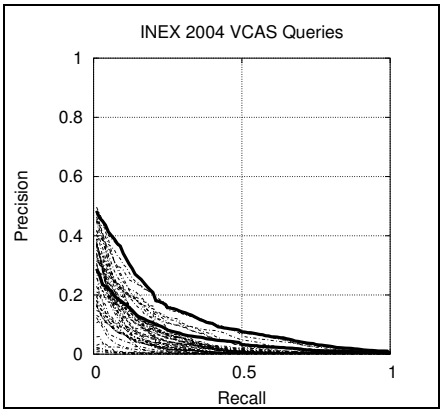

Fig. 23. INEX 2004 VCAS R/P Curve

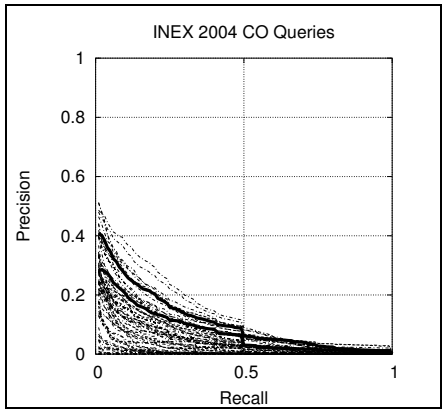

Fig. 24. 2004 INEX CO R/P Curve.

6 Future Outlook

The most promising aspect of our participation in this year's NLP track was that our system was able to outperform the majority of Ad-hoc systems, thereby verifying that natural language queries have the potential to be a viable alternative to complex formal query languages such as NEXI. However, there is still progress to be made, and we shall strive to improve the performance of our NLP system.

The most disappointing aspect of this year's track was the lack of submissions. INEX should encourage more participation in the NLP track in order to broaden the knowledge in the domain of natural language interfaces to XML IR and to strengthen INEX as a whole. Future participants will most likely come from two areas. The first will be existing INEX participants in the Ad-hoc and other tracks who will develop a natural language interface to their existing system, similar to the approach we took. The second will be from participants in workshops such as TREC [11] and SIGIR [10], which are traditionally strong in the domain of natural language IR systems. Both types of competitors are likely to bring different perspectives on the problem and their participation should be welcomed.

Also the INEX organisers should strive to ensure that the title and description elements are faithful 1:1 translations, both in terms of structural and content requirements.

7 Conclusion

This paper presents a natural language interface to XML-IR system. The interface uses template matching to derive users' content and structural requests from natural language queries (NLQs). The interface then translates the NLQs into an existing standard formal query language, allowing them to be processed by many existing systems. Our backend system responds to user queries with relevant and appropriately sized results in a timely manner and our ranking scheme is comparable with the INEX best alternatives. While the NLP interface requires further development, initial results are promising.

Bibliography

[1] Brill, E.: A Simple Rule-Based Part of Speech Tagger. In Proceedings of the Third Conference on Applied Computational Linguistics (ACL), Trento, Italy. (1992)

[2] Clark, J. and DeRose, S.: "XML Path Language XPath version 1.0.", Technical report, W3C (1999) W3C Recommendation available at http://www.w3.org/TR/xpath.

[3] Fuhr, N. and Malik, S.: Overview of the Initiative for the Evaluation of XML Retrieval (INEX) 2003. In INEX 2003 Workshop Proceedings, Dagstuhl, Germany, December 15-17 2003 (2004) 1-11

[4] Geva, S. and Spork, M.: XPath Inverted File for Information Retrieval, In INEX 2003 Workshop Proceedings, Dagstuhl, Germany, December 15-17 2003 (2004) 110-117

[5] Geva, S.: GPX at INEX 2004. In INEX 2004 Workshop Proceedings, Dagstuhl, Germany, December 6-8 2004 (2005)

[6] O'Keefe, R. and Trotman, A.: "The Simplest Query Language That Could Possibly Work", In INEX 2003 Workshop Proceedings, Dagstuhl, Germany, December 15-17 2003 (2004) 167-174

[7] Manning, C. D. and Schutze, D. "Foundations of Statistical Natural Language Processing", MIT Press, Cambridge (1999)

[8] Marcus, M., Santorini, N. and Marcinkiewicz, M. Building a large annotated corpus of English: The Penn Treebank, In. Computational Linguistics (1993)

[9] Sigurbjornsson, B., Kamps, J. de Rijke, M. An Element-based Approach to XML Retrieval, In INEX 2003 Workshop Proceedings, Schloss Dagstuhl, Germany, December 15-17 2003 (2004) 19-26

[10] Special Interest Group on Information Retrieval (SIGIR) Homepage, http://www.acm.org/sigir/.

[11] Text REtreival Conference (TREC) Homepage, http://trec.nist.gov/.

[12] Trotman, A., & Sigurbjörnsson, B. Narrowed Extended XPath I (NEXI). In INEX 2004 Workshop Proceedings, Dagstuhl, Germany, Decemeber 8 -10 2004 (2005)

[13] Van Rijsbergen, R. J. Information Retrieval, Butterworths, Second Edition (1979)