IBM in TREC2006 Enterprise Track

Jennifer Chu-Carroll Guillermo Averboch Pablo Duboue David Gondek
J William Murdock John Prager
IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA
{jencc, gaver, duboue, dgondek, murdockj, jprager}@us.ibm.com

Paul Hoffmann Janyce Wiebe
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA, USA
{hoffmanp, wiebe}@cs.pitt.edu

1. Introduction

In 2006, IBM participated for the first time in the Enterprise Track, submitting runs for both the discussion and expert tasks. The Enterprise Track is intended to address information seeking tasks common in corporate settings using information that is readily available on corporate intranets. Because of confidentiality issues, the corpus used for this task is a snapshot of w3c.org as of June 2004, considering the W3C as a "corporation" that conducts its day-to-day business on the web. This corpus consists of a heterogeneous set of data sources, including web pages, mailing lists, code, wiki, etc. [Craswell et al. 2006]. The discussion task seeks e-mail messages that discuss the pro or con of a given subject matter, while the expert task requires that experts for given topics be identified.

The main foci of our Enterprise Track experiments this year were on 1) problem-solving through adoption of multiple discussion and expert finding strategies, 2) combination of structured, semi-structured, and unstructured information for discussion and expert finding, 3) investigation of impact of select NLP techniques, such as multi-document summarization for expert pseudo-document generation and pro/con sentiment identification and retrieval, and 4) use of external resources for discussion and expert finding. The rest of this paper discusses the systems we developed for Enterprise Track participation, focusing on the four aspects outlined above. We will also discuss specific strategies we took to configure the systems for each of the four runs in both tasks, as well as their impact on system performance.

2. Discussion Task

The discussion task utilizes only the lists section of w3c.org, which contains the archives of W3C mailing lists. Given a topic, a ranked list of documents (e-mail messages) is returned in which an e-mail message contains at least one pro or con statement regarding the topic in the "new content" portion of the message (i.e., the relevant pro/con statements do not occur only in quoted text). In the discussion task, we focused our investigation on using multiple problem-solving strategies, adopting NLP techniques for pro/con-based re-ranking of search results, and using external resources for query analysis and expansion.

2.1. Corpus Processing

The lists portion of the w3c corpus was processed to identify the author, subject, and new vs. quoted texts for each e-mail message. A corresponding new document is constructed that contains the author, subject, and new text portion of each e-mail message. These documents form the corpus from which three keyword indices are built for the three search engines we employ in our system. In addition, each document is annotated with our pro/con sentiment analyzer and the annotated documents are stored for runtime use by the IBM pro/con assessor component.

2.2. Runtime System Architecture and Components

Figure 1 shows the architecture of our pro/con discussion retrieval system. Given a topic, which consists of a title, a description, and a narrative, our query analysis component transforms the topic into one or more generic abstract search query representations. The document retrieval module is implemented as a pluggable framework that accepts query generator and search engine pairs to return document hit lists given the abstract query representations. The hit lists returned by the retrieval engines are combined using a linear combination algorithm and additional potentially relevant documents are included by the e-mail-thread-based augmenter. The resulting hit list contains topic-relevant documents. The documents in the hit list are then evaluated for containment of pro/con statements again in a pluggable pro/con re-ranking framework which accepts assessors that rescore the original document list. The rescored lists are again combined to form the final ranked document list. The rest of this section describes select key system components in further detail.

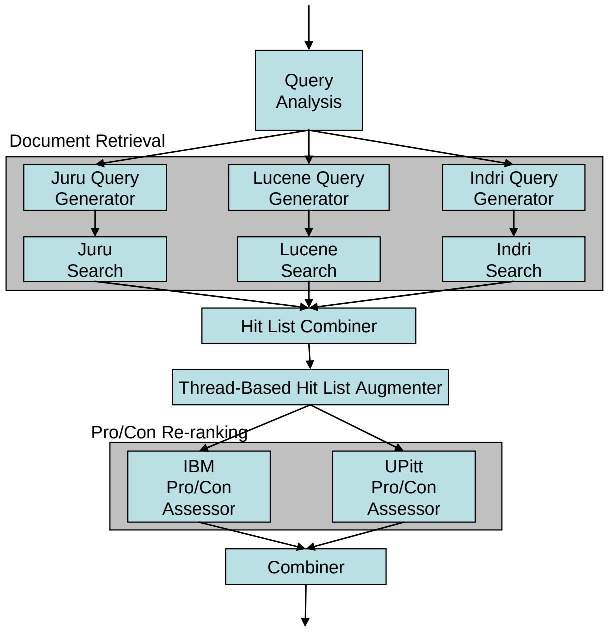


Figure 1 Architecture for Pro/Con Discussion Retrieval System

2.2.1. Query Analysis

A discussion search topic contains a title, a description, and a narrative. Our system makes use of the title and description fields of the topic to generate one or more abstract query representations from which specific search engine queries can be generated. Aside from standard stopword removal, we perform query expansion as well as identify important keywords and their respective salience in an iterative fashion through the use of an external domain-specific dictionary.

Our domain-specific dictionary was developed based primarily on FOLDOC,[1] an online dictionary of computing terms containing more than 14,000 entries. Entries in FOLDOC contain a natural language description of the terms being defined and may also include hyperlinks to other entries in the dictionary. We automatically processed these definitions in FOLDOC and extracted, for each term, its acronym (or expansion if the term is an acronym), if any, and the system's confidence that the acronym and expansion are co-referents of one another. In addition, we extract phrases highly associated with each entry term. We further augment the dictionary with terms of interest that are not present in FOLDOC, in particular, topics addressed by W3C standards.

Our query analysis process employs an iterative procedure that produces one or more abstract query representations based on dictionary lookup and a set of heuristics. An initial abstract query representation contains terms in the title and description fields in the topic, after stopword removal. Based on an n-gram (in our current implementation n=3) dictionary lookup, the abstract query representation may be augmented/duplicated in the following ways. First, all phrases present in the dictionary are marked as "phrase". Second, all dictionary terms/phrases in the query are marked as "required". Third, all high-confidence acronyms/expansions for terms/phrases in the query are returned and a duplicate abstract query representation is constructed in which the acronym/expansion substitutes the original

[1] http://www.foldoc.org. Content downloadable from http://foldoc.org/source.html.

term in the query. The resulting set of abstract query representations forms query representations of high salience. In the second iteration, a set of query representations of medium salience is generated by removing the "required" and "phrase" markers on all query terms from the high-salience set. The third and fourth iterations are identical to the first and second, except that low-confidence acronyms/expansions are returned from the dictionary.

This ranked set of abstract query representations is the output of our query analysis component, and is used by downstream document retrieval components to generate search engine specific queries.

2.2.2. Document Retrieval

As discussed previously, our document retrieval component is a pluggable framework that accepts one or more query generator and search engine pairs. In our TREC system, we employed three publicly available, off-the-shelf search engines: IBM's own Juru search engine[2] [Carmel et al., 2002], the Lucene search engine[3] [Hatcher and Gospodnetic, 2004], and UMass/CMU's Indri search engine[4] [Strohman et al, 2005]. We developed query generation components for each of these search engines in an attempt to generate search queries that best leverage the expressiveness of the query languages of the underlying search engine.

We hypothesized that using multiple search engines and merging their hit lists will result in better performance because the search engines support different query capabilities and employ different ranking algorithms. For instance, Indri supports a window operator (either ordered or unordered) for grouping select query terms as well as weights associated with terms, while Juru and its extension JuruXML support search over semantic types as well as keywords.[5] Furthermore, Indri supports pseudo-relevance feedback, while the other two search engines do not,[6] leading to additional diversity in the documents in the hit lists. Typically, each search engine will use abstract query representations starting from those in the highest salience set. If insufficient documents are retrieved after executing queries at one salience level, then the query representations in the next salience level are used. Once each search engine assembles its hit list, they are combined and augmented as described in the next section.

2.2.3. Hit List Combination and Augmentation

Our hit list combination component adopts a linear combination algorithm trained to optimally combine hit lists from multiple search engines. The weights were trained on the TREC 2005 discussion task dataset using the topic relevance judgments. In order to avoid overfitting, grid search was used to optimize for MAP score and the solution was checked for stability.

Since the indexed portion of each e-mail message contains only the author, subject, and new text section of the message, we developed an e-mail-thread-based augmentation component to ensure that all documents in the same thread as a document in the hit list are also included for pro/con assessment. We made use of the e-mail-threading information made available by William Webber [Webber, 2005] and appended all additional documents found through this mechanism at the end of the hit list.

2.2.4. Pro/Con Assessment

Our pro/con assessment component is again designed as a pluggable framework for experimenting with and combining multiple assessment algorithms. In our TREC system, two assessment components were used, a rule-based pro/con sentiment recognizer developed at IBM, and a statistical topic independent pro/con recognizer developed at the University of Pittsburgh.

The IBM pro/con assessor makes use of a rule-based sentiment recognizer built on top of ESG, a slot-grammar parser [McCord, 1990]. These rules are developed through manual compilation of 700+ sentiment keywords such as

[2] Available as part of the open source UIMA framework, http://sourceforge.net/projects/uima-framework
[3] http://lucene.apache.org
[4] http://www.lemurproject.org/indri/
[5] We did not make use of this semantic search feature of Juru in our TREC system, but we plan to exploit this functionality for discussion search in the future.
[6] It is, of course, possible to implement some pseudo-relevance feedback mechanism for use with the other two engines. However, because of time constraints, we did not explore this option.

"deficient" and "compelling", and development of a rule set that makes sentiment annotations based on these keywords in the context of a parse tree. A sample sentiment annotation is shown in the following:

```
<REL name="Sentiment">
   <RNG type="Obj">HTTP with simple XML</RNG> may be most
   <DMN type="SentimentKeyword">appropriate</DMN>
</REL>
```

Given a document, our assessment component identifies text snippets relevant to the topic and determines the pro/con relevance of the document based on the proximity of the sentiment annotations and the relevant text snippets. Other features such as depth of e-mail message in a thread chain, presence of relevant pro/con statements in the quoted texts portion of a document, as well as the frequency that passages in the current document are quoted by others, are also taken into account in the determining the final ranking of the documents.

The UPitt topic independent pro/con classifier takes as input a document and returns a confidence score on whether or not it contains a pro/con argument. The classifier uses the BoosTexter AdaBoost.MH machine learning algorithm [Schapire and Singer, 2000] with two features. The first feature is a bag-of-words representation of words with the following parts of speech: adverb, verb, noun, pronoun, adjective, modal, existential there, and number. The other feature is a bag-of-words representation of pro/con and non-pro/con extraction patterns generated using AutoSlog-TS from the Sundance/AutoSlog package [Riloff and Phillips, 2004]. AutoSlog-TS takes a set of positive examples (documents judged as pro/con in the TREC 2005 dataset) and negative examples (topic-relevant but not pro/con documents), and generates extraction patterns, ranked in the order of their association with the positive examples (most extraction pattern types used are listed in Figure 1 of [Riloff et al., 2006]). During development, the classifier was cross-validated with the TREC 2005 discussion task dataset. For our TREC 2006 submission, all topic-relevant documents in the 2005 dataset were used to train the classifier.

The ranked hit lists from the assessment components are again combined using a linear combination algorithm to generate the system's final ranked list of documents.

2.3. Evaluation Results

We submitted four runs to the discussion task. For all runs, FOLDOC was used in the query analysis process for query expansion. When the description field is used, only terms found in FOLDOC are included in the query. The key characteristics of the four runs are as follows:

- IBM06JAQ: This run used only the title field of the query. For document retrieval, only the Juru search engine is used, and for pro/con assessment, only the IBM pro/con assessor was used.
- IBM06JAQD: This run is identical to IBM06JAQ except that both the title and description of the query were used.
- IBM06JILAQD: This run used both titles and descriptions. For document retrieval, all three search engines were used, but only the IBM pro/con assessor was used for sentiment analysis.
- IBM06JILAPQD: This run is identical to IBM06JILAQD, except that both pro/con assessors were used to re-rank the documents returned by the search engines.

In order to better determine the contributions of individual components, we ran three additional configurations of the system which were not included in our submission, as follows:

- JQ: This run used only the title field of the query. For document retrieval, only the Juru search engine was used and no pro/con sentiment analysis was performed.
- JILQ: This run is identical to JQ except that for document retrieval, all three search engines were used.
- JILQD: This run is identical to JILQ except that both the title and description were used.

Error! Not a valid bookmark self-reference. shows the results of our runs. The top four rows are the results of our official submitted runs while the last three are additional runs performed post TREC. The columns labeled "topic" measure topic relevance of the hit list, while those labeled "p/c" additionally take into account the presence of pro/con statements in the document. As expected, the run IBM06JILAPQD in which both titles and descriptions were used in query analysis, and where all available search engines and pro/con assessors were employed and their results merged, outperformed other configurations of the system in almost all cases. Our results show that 1) leveraging information provided in the description field of the topic for query analysis results in more effective search queries than using the title alone (JILQD vs. JILQ; rows 7 and 6), 2) effective combining hit lists from multiple search engines can substantially outperform using a one search engine alone (JQ vs. JILQ; rows 5 and 6; as well as IBM06JAQD vs. IBM06JILAQD; rows 2 and 3), 3) leveraging NLP technologies to perform pro/con analysis on the resulting hit list results in further substantial improvement in performance (JILQD vs. IBM06JILAQD; rows 7 and 3), and 4) adopting multiple pro/con assessment components can further improve system performance compared with single strategy pro/con analysis (IBM06JILAPQD vs. IBM06JILAQD; rows 4 and 3). Interestingly, when including the pro/con assessment components in the pipeline, the scores for both topic relevance and pro/con relevance are improved. We hypothesize that this is partially due to the fact that the IBM pro/con assessor performs finer-grained topic relevance analysis at the text snippet level, thus affecting the relative ranked order of topic relevant documents as a side effect of pro/con re-ranking.

Table 1 Discussion Task Results

Run Tag	MAP		R-prec		bpref		P@10		P@100	
	topic	p/c	topic	p/c	topic	p/c	topic	p/c	topic	p/c
IBM06JAQ	0.3146	**0.2030**	0.3527	0.2481	0.3572	**0.2337**	0.5440	**0.3391**	0.3998	0.2487
IBM06JAQD	0.3069	0.1967	0.3493	0.2485	0.3499	0.2336	0.5400	0.3283	0.4054	0.2467
IBM06JILAQD	0.3305	0.2004	0.3709	0.2509	0.3700	0.2294	0.5600	0.3326	0.4184	0.2493
IBM06JILAPQD	**0.3310**	0.2021	**0.3717**	**0.2558**	**0.3709**	0.2323	**0.5640**	**0.3391**	**0.4186**	**0.2515**
JQ	0.2745	0.1654	0.3257	0.2209	0.3218	0.2082	0.4950	0.2800	0.3590	0.2038
JILQ	0.3017	0.1762	0.3524	0.2288	0.3472	0.2083	0.5360	0.2978	0.3844	0.2250
JILQD	0.3095	0.1835	0.3621	0.2422	0.3559	0.2174	0.5360	0.3065	0.3998	0.2359

3. Expert Task

For the expert task, a system is expected to return a ranked list of up to 100 people from a list of 1092 candidates who it considers experts for a given topic. The system may make use of the entire W3C corpus as well as any external resources in determining the ranked list of experts. The expertise of each expert, however, should be justified by up to 20 documents from the W3C corpus. In the expert task, we focused our investigation on using multiple problem-solving strategies, adopting NLP techniques for expertise-driven information extraction and pseudo-document generation, exploiting use of structured, semi-structured, and unstructured information on expert finding, and augmenting strategies that make use of the W3C corpus with those that consult external resources.

Figure 2 shows the architecture of our multi-agent expert search system. At the heart of the system there are six agents each adopting different problem-solving strategies to identify zero or more experts for a given topic. Three of the agents adopt the pseudo-document approach, which is inspired by the top-performing system from TREC 2005 in which a pseudo-document is generated for each candidate expert to represent their expertise [Fu et al., 2006]; although the strategies we adopted for pseudo-document generation differ from theirs substantially. Our three pseudo-document based agents employ different algorithms/components for ranking expertise. The Google Scholar agent makes use of the eponymous external resource to identify experts based on publications. The Expert MetaData agent identifies experts based on expertise information extracted from semi-structured W3C standards documents. Finally, the EKDB agent leverages structured data extracted from texts using task-specific relations for expert identification. Our expert search system leverages the same query analysis component as described in Section 2.2.1, as well as the machine learning based hit list combination component described in Section 2.2.3 to merge the results produced by the six agents. This hit list optionally goes through two heuristic-based post-processors. The first post-processor affects the expert ranked list in which experts in the original hit list may be re-ordered based on certain external criteria. The second post-processor affects the support document ranked list for each expert in that a support document may be removed because it was deemed non-supportive, or that the support documents may be re-ordered based on select heuristics.

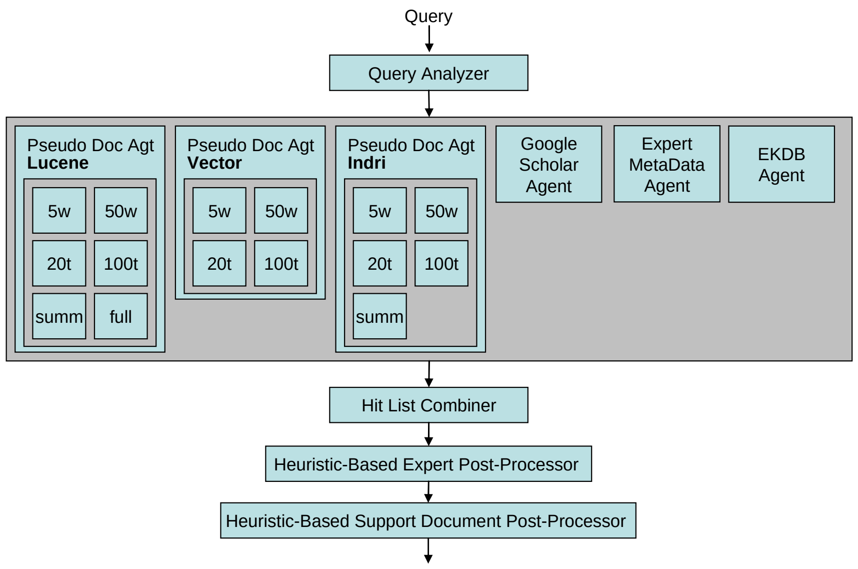


Figure 2 Expert Search System Architecture

3.1. Corpus Processing and Pseudo-Document Generation

Our pseudo-document generation process begins with identifying instances of the candidate experts in documents. To increase the recall of candidate identification, we performed accent normalization on the corpus and developed a set of heuristics for name variation generation, including dropping of middle names and replacing first names with first initials. We adopted four approaches to generate six sets of pseudo-documents in which each set contains one document for each candidate identified in the corpus as follows:

- **Windowing approach**, in which the n sentences before and after each mention of a candidate is selected and appended to the candidate's pseudo-document. We used n=5 and n=50 in our system.
- **Top sentence approach**, in which the first n sentences of a document in which a candidate is mentioned is selected and appended to the candidate's pseudo-document. We used n=20 and n=100 in our system.
- **Whole document approach**, in which all documents in which a candidate is mentioned are concatenated to form the pseudo-document.
- **Summarization approach**, in which MEAD[7] [Radev et al., 2003], a query-based multi-document summarizer was used to generate the pseudo-document for each candidate.

To experiment with different search strategies, we built two search indices for each pseudo-document set, one for Lucene and one for Indri. In addition, we also built a collection of Lucene document vectors for each pseudo-document set.

We also performed additional corpus processing for use by the other agents. The lists and www portions of the corpus were analyzed using a set of named entity and relation recognizers. The named entity recognizers identify generic entity types such as *Person*, *Date*, *Organization*, and *Country*, as well as domain specific types such as *EnterpriseMember* and *ComputingTopic*. The relation recognizers cover two relations of interest for this task, *ExpertIn* and *AuthorOf*. The former identifies text fragments that are indicative of expertise, such as "James Larson, W3C Voice Browser Working Group co-Chair", and associates the person and topic with an *ExpertIn* relation, while the latter pairs all e-mail authors and subjects. The entities and relations in this annotated corpus are stored in our Extracted Knowledge DataBase (EKDB) [Ferrucci et al., 2006] for use by the EKDB agent, and all the keywords and annotations are used to generate a JuruXML semantic search index which was used in our manual submission.

Finally, we automatically identified a set of W3C-specific standards documents which, for the most part, have a semi-structured common format. We processed these documents to extract document title, document ID, editorship,

[7] Available for download at http://www.summarization.com/mead/

and authorship information. We stored this information in as standards document metadata, to be used by our Expert MetaData agent.

3.2. *Runtime Components*

3.2.1. Pseudo-Document Agents

The key idea behind the pseudo-document agents is to construct, for each candidate expert, a document that represents his/her expertise. Given a topic, the agent compares the topic to the pseudo-documents and returns experts whose pseudo-document best matches the topic. We experimented with a number of heuristics for pseudo-document generation as described in Section 3.1. The windowing approach hypothesizes that one's expertise commonly occurs in close proximity to mentions of his/her name. The top sentence approach hypothesizes that one's expertise is indicated by the topic of the document, which is typically established at the beginning of the document. The whole document approach targets a recall-oriented strategy. Finally, the summarization approach experiments with the utility of a multi-document summarization system for capturing the expertise of a given person.

Figure 2 shows the configuration of our three pseudo-document agents, each utilizing a different retrieval strategy, namely Lucene, Indri, and vector similarity, on each of the pseudo-document sets generated.[8] Given a query, a pseudo-document agent sends the query to each pseudo-document index and combines their result to generate its expert hit list. The supporting document list is generated by issuing the same topic to a regular document index and selecting from the most relevant documents those in which the candidate expert occurs.

3.2.2. EKDB Agent

As discussed earlier, we store in our Extracted Knowledge DataBase (EKDB) entities and relations extracted from our automatically annotated corpus. In addition, the EKDB records provenance information which includes documents from which entities and relations were extracted. Of the two relations used for this task, *ExpertIn*, which indicates direct assertions of expertise, is used as high-confidence evidence for expertise, while *AuthorOf,* which indicates authorship of e-mail subjects, is used as low-confidence evidence for expertise.

The EKDB agent generates database queries based on the abstract query representations produced by the query analysis component, and experts are scored based on the system's confidence level as indicated by the *ExpertIn* or *AuthorOf* relation, and the salience level of the abstract query representation used to generate the database queries. Furthermore, documents from which the relation was extracted are used as supporting evidence. If the same expert is found from multiple relations and/or multiple queries, the confidence score is boosted and the confidence values assigned to the supporting documents are also scaled accordingly.

3.2.3. Expert MetaData Agent

The Expert MetaData agent targets high-precision performance by basing expert identification on highly reliable cues for expertise indication. In our TREC system, it made use of the metadata extracted from W3C standards documents in which editorship and authorship of a standards document are considered definite and highly reliable indications of expertise, respectively.

Given a topic, all standards documents potentially relevant to the topic are retrieved. Linguistic heuristics are then applied to determine topic subsumption and thus the degree of match between the query topic and the document topic. The editors and authors of relevant documents are extracted and scored based on the degree of topic match, while the standards document is used as support for the candidate's expertise. Multiple pieces of evidence for the same expert are combined to arrive at a final ranked expert list.

3.2.4. Google Scholar Agent

An external resource we exploited in our expert search system is Google Scholar,[9] which supports search over scholarly publications. For a given query topic (issued in quotes to Google Scholar), we extract the author list and

[8] We were unable to generate some of the indices for larger pseudo-document sets because of memory constraints.
[9] http://scholar.google.com

the number of citations for each of the top 20 publications returned. An author's expertise in the topic is computed based on factors such as the number of publications retrieved, the number of citations as well as the author's order in the author list for each publication. The ranked list is then filtered using the candidate expert list provided to return only eligible experts.

3.2.5. Expert Post-Processor

We developed an affinity-based expert re-ranker to perform post-hoc re-ordering of less confident candidates in the expert list, using an affinity metric based on the hypothesis that experts of the same topic tend to co-occur in documents. The algorithm initially populates a new hit list with the top 5 experts from the original list, and iteratively adds experts to the hit list by selecting those experts whose scores representing their affinity with those already on the hit list exceed a certain threshold.

3.2.6. Supporting Document Post-Processors

We developed two types of supporting document post-processors, one that removes supporting documents from the list, and two that re-ranks the documents. The acknowledgments document filter removes those supporting documents in which the candidate's name appears only in the acknowledgments section of the document. The duplicate document remover increases the diversity of supporting documents by eliminating all duplicates and near-duplicates in the document list using a vector similarity based measure. Finally, the EKDB re-ranker considers documents that support the *ExpertIn* and *AuthorOf* relations in the EKDB to be more reliable supporting documents than others, and uses this information to re-rank supporting documents returned by other agents.

3.3. Evaluation Results

We submitted four runs to the expert task. Three of our runs are automatic, labeled IBM06QO, IBM06PR, and IBM06EXP. All three runs used all of the four agents described above, with different query processing and support document selection strategies. The run labeled IBM06MA was partially manual. The key characteristics of our automatic runs are described below:

- IBM06QO: This run used only the title field of the topic. FOLDOC was used for query expansion.
- IBM06PR: This run used both the title and description fields of the topic in query analysis Select agent parameters were tuned to target higher precision. It also included two additional components: the EKDB document re-ranker and the acknowledgments filter.
- IBM06EXP: This run also used both the title and description fields of the topic. It included two additional components on top of the IBM06QO configuration: the affinity-based expert re-ranker and the duplicate document remover.

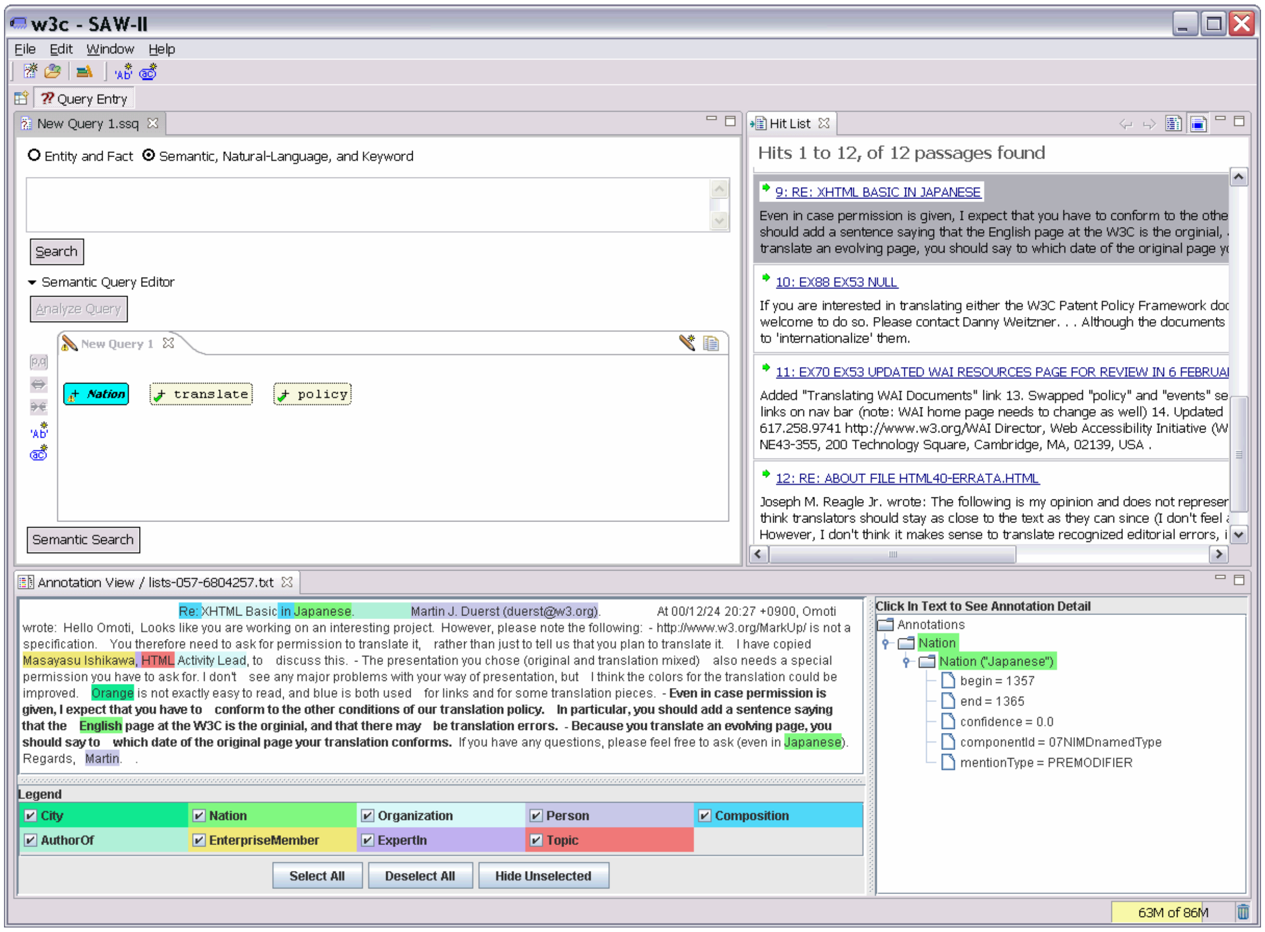
Table 2 Expert Task Results

Run Tag	MAP		R-prec		bpref		P@5		P@10	
	exp	sup	exp	sup	exp	sup	exp	sup	exp	sup
IBM06QO	**0.4536**	**0.2863**	**0.4519**	**0.3303**	**0.4402**	**0.3711**	**0.6653**	**0.4857**	0.5408	**0.4041**
IBM06PR	0.4455	0.2789	0.4442	0.3211	0.4306	0.3522	0.6531	0.4776	0.5265	0.3939
IBM06EXP	0.4357	0.2730	0.4434	0.3201	0.4338	0.3465	0.6531	0.4816	**0.5469**	**0.4041**
IBM06MA	0.5235	0.3346	0.5192	0.3829	0.5180	0.4135	0.7673	0.5878	0.6449	0.4878

Table 2 shows the results of our submitted runs. For each performance measure reported, the score under column heading "exp" represents score when considering only expert correctness, while that under "sup" takes into account whether or not a correct supporting document was found. For all measures, the manual run (IBM06MA) received the highest score. The highest scoring of the three automatic runs for each measure is boldfaced. It is worth noting that our two best scoring runs, IBM06MA and IBM06QO, were our priorities 3 and 4 runs, and therefore were not included in the results pooled for judging.

Our results show that IBM06QO achieved the highest automatic score in almost all cases except for precision@10. The IBM06PR run, which targeted higher precision, did not outperform either of the two systems, even using the P@5 and P@10 measures. We are currently running finer grained experiments to determine the contribution (or lack thereof) of the individual components included in this run. The contributions of components in the IBM06EXP run are easier to determine. The affinity-based expert re-ranker increased precision@10 but fared worse in all other measures. The result for the duplicate document remover is somewhat counter-intuitive. In theory, with the binary supporting document metric adopted, the greater diversity in supporting documents produced by the duplicate document remover should result in an increase performance. However, this does not appear to be the case when comparing the scores for IBM06EXP and IBM06QO. In particular, while there is an increase in expert only score in P@10, the scores with supporting documents are identical for P@10 for the two runs. This is probably due to incomplete judgment for supporting documents.

The goals of our manual run were twofold. First, we wanted to evaluate the potential of incorporating semantic search capabilities into an automatic expert finding system. Since automatic semantic search query construction from descriptions and narratives is a difficult task, our manual run consists of having an expert manually construct queries that leverage our semantic annotations to retrieve more relevant documents. Second, we were interested in evaluating the effectiveness of SAW II, an interactive system that supports semantic query construction, refinement, and results exploration, for this problem-solving task. Figure 3 shows a screenshot of SAW II, illustrating query construction and refinement support (top left panel), result browsing support (top right panel), and detailed document drill down support (bottom panel).



Hits 1 to 12, of 12 passages found

9: RE: XHTML BASIC IN JAPANESE

Even in case permission is given, I expect that you have to conform to the othe
should add a sentence saying that the English page at the W3C is the orginial,
translate an evolving page, you should say to which date of the original page yo

10: EX88 EX53 NULL

If you are interested in translating either the W3C Patent Policy Framework doc
welcome to do so. Please contact Danny Weitzner. . . Although the documents
to 'internationalize' them.

11: EX70 EX53 UPDATED WAI RESOURCES PAGE FOR REVIEW IN 6 FEBRUA

Added "Translating WAI Documents" link 13. Swapped "policy" and "events" se
links on nav bar (note: WAI home page needs to change as well) 14. Updated
617.258.9741 http://www.w3.org/WAI Director, Web Accessibility Initiative (W
NE43-355, 200 Technology Square, Cambridge, MA, 02139, USA .

12: RE: ABOUT FILE HTML40-ERRATA.HTML

Joseph M. Reagle Jr. wrote: The following is my opinion and does not represer
think translators should stay as close to the text as they can since (I don't feel
However, I don't think it makes sense to translate recognized editorial errors, i

Re: XHTML Basic in Japanese. Martin J. Duerst (duerst@w3.org). At 00/12/24 20:27 +0900, Omoti wrote: Hello Omoti, Looks like you are working on an interesting project. However, please note the following: - http://www.w3.org/MarkUp/ is not a specification. You therefore need to ask for permission to translate it, rather than just to tell us that you plan to translate it. I have copied Masayasu Ishikawa, HTML Activity Lead, to discuss this. - The presentation you chose (original and translation mixed) also needs a special permission you have to ask for. I don't see any major problems with your way of presentation, but I think the colors for the translation could be improved. Orange is not exactly easy to read, and blue is both used for links and for some translation pieces. - **Even in case permission is given, I expect that you have to conform to the other conditions of our translation policy. In particular, you should add a sentence saying that the English page at the W3C is the orginial, and that there may be translation errors. - Because you translate an evolving page, you should say to which date of the original page your translation conforms.** If you have any questions, please feel free to ask (even in Japanese). Regards, Martin. .

Click In Text to See Annotation Detail

```
Annotations
  Nation
    Nation ("Japanese")
      begin = 1357
      end = 1365
      confidence = 0.0
      componentId = 07NIMDnamedType
      mentionType = PREMODIFIER
```

Figure 3 SAW II Screenshot

The query shown in Figure 3 demonstrates how we leveraged our semantic annotations to answer topic EX53 on W3C translation policy. By including the *Nation* entity type, which matches both the nominal and adjectival forms of country names, in the query, the returned documents, which must have included a country or language, were much more on topic. In our submitted manual run, we targeted around 5 manually identified experts for each topic. The rest of the hit list was augmented with results from the IBM06PR run.

4. Conclusions

This paper described our efforts in this year's Enterprise Track discussion and expert tasks. Our discussion task results show that our conservative query expansion strategy focusing on FOLDOC-identified domain-specific terms improves performance slightly, while combining the results from multiple search engines substantially outperforms single search engine results. Finally, our NLP based pro/con assessment components again results in substantial performance improvement. On the expert task, our current run results are somewhat less conclusive. We are currently in the process of running further experiments and ablation studies to determine the contributions of individual agents and components.

5. Acknowledgments

We would like to thank Brad Andrews, Eric Brown, David Ferrucci, Sarah Luger, Rajen Subba, and Mikalai Yatskavich for helpful discussions and/or for their help in the development of a heldout test set for the expert task. This work was supposed in part by the Disruptive Technology Office (DTO)'s Advanced Question Answering for Intelligence (AQUAINT) program under contract number H98230-04-C-1577.

6. References

David Carmel, Einat Amitay, Miki Herscovici, Yoelle Maarek, Yael Petruschka, and Aya Soffer. Juru at TREC-10 – Experiments with index pruning. *Proceedings of the 10th Text REtrieval Conference*, 2002.

Nick Craswell, Arjen de Vries, and Ian Soboroff. Overview of the TREC-2005 Enterprise Track. *Proceedings of the 14th Text REtrieval Conference*, 2006.

David Ferrucci, J. William Murdock, and Christopher Welty. Overview of Component Services for Knowledge Integration in UIMA. IBM Research Report RC24074, 2006.

Yupeng Fu, Wei Yu, Yize Li, Yiqun Liu, Min Zhang, and Shaoping Ma. THUIR at TREC 2005: Enterprise Track. *Proceedings of the 14th Text REtrieval Conference*, 2006.

Erik Hatcher and Otis Gospodnetic. *Lucene in Action.* Manning Publications. 2004.

Michael C. McCord. Slot Grammar: A System for Simpler Construction of Practical Natural Language Grammars. In Natural Language and Logic: International Scientific Symposium, Springer Verlag LNCS, 1990.

Dragomir R. Radev, Simone Teufel, Joracio Saggion, Wai Lam, John Blitzer, Jong Qi, Arda Celebi, Danyu Liu, and Elliott Drabek. Evaluation challenges in large-scale multi-document summarization: the MEAD project. *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2003.

Ellen Riloff and William Phillips. An Introduction to the Sundance and AutoSlog Systems. Technical Report UUCS-04-015, School of Computing, University of Utah. 2004.

Ellen Riloff, Siddharth Patwardhan, and Janyce Wiebe. Feature subsumption for opinion analysis. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2006.

Robert E. Schapire and Yoram Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3), 2000.

T. Strohman, D. Metzler, H. Turtle, and W.B. Croft. Indri: A language-model based search engine for complex queries. UMass Amherst CIIR Technical Report IR-407, 2005.

William Webber. Thread structure of w3c lists. http://www.cs.mu.oz.au/~wew/w3c-lists-threads-w3w.tar.gz. 2005.