Structured Retrieval for Question Answering

Matthew W. Bilotti, Paul Ogilvie, Jamie Callan and Eric Nyberg
Language Technologies Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
{ mbilotti, pto, callan, ehn }@cs.cmu.edu

ABSTRACT

Bag-of-words retrieval is popular among Question Answering (QA) system developers, but it does not support constraint checking and ranking on the linguistic and semantic information of interest to the QA system. We present an approach to retrieval for QA, applying structured retrieval techniques to the types of text annotations that QA systems use. We demonstrate that the structured approach can retrieve more relevant results, more highly ranked, compared with bag-of-words, on a sentence retrieval task. We also characterize the extent to which structured retrieval effectiveness depends on the quality of the annotations.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; H.3.3 [**Information Storage and Retrieval**]: Information Storage and Retrieval

General Terms

Algorithms

Keywords

Structured retrieval, question answering

1. INTRODUCTION

Modern Question Answering (QA) systems often represent the information required to answer a question using linguistic and semantic constraints. For example, Lin and Hovy [8] and Harabagiu et. al. [6] have developed topic representations (or *topic signatures*) which encode not only question keywords, but also relations among them, including the syntactic relations such as *subject* and *object* that hold between verbs and their arguments. The use of these relations can help a system to recognize that a passage does not answer a given question, even if it contains the correct keywords in a minimal window (e.g., *Ruby killed Oswald* is not an answer to the question *Who did Oswald kill?*).

Bag-of-words retrieval does not support checking such relational constraints at retrieval time, so many QA systems simply query using the key terms from the question, on the assumption that any document relevant to the question must necessarily contain them. Slightly more advanced systems also index and provide query operators that match named entities, such as *Person* and *Organization* [14]. When the corpus includes many documents containing the keywords, however, the ranked list of results can contain many irrelevant documents. For example, question 1496 from TREC 2002 asks, *What country is Berlin in?* A typical query for this question might be just the keyword *Berlin*, or the keyword *Berlin* in proximity to a *Country* named entity. A query on the keyword alone may retrieve many documents, most of which may not contain the answer. A query that includes the named entity constraint is more accurate, but still matches sentences such as *Berlin is near Poland*.

Many QA systems adopt the approach of retrieving large sets of documents using bag-of-words retrieval with question keywords, then exhaustively post-processing to check the higher-level constraints that indicate a likely answer. One criticism of this approach is that it is difficult to know *a priori* how many documents to retrieve; even a very large set of results may not contain an answer if the key terms are very common. A second criticism is that it can be slow; though they aim for a 'large enough' set of documents, systems may retrieve and process more text than is really required to answer a question. Interactive systems are particularly vulnerable, as they must balance the amount of processing against the number of results processed while the user waits.

In situations where complex corpus analysis can be done off-line, another potential bottleneck arises. In a QA system, downstream processing after the retrieval stage can include information extraction, answer merging and justification, and presentation. The processing cost scales with the number of retrieved results. Bag-of-words may retrieve a large number of documents that do not contain answers. It may therefore be necessary to process many results before a satisfactory answer is found. If documents could be ranked based on the linguistic and semantic constraints of interest to the QA system, fewer results would need to be processed downstream, potentially improving system efficiency.

This paper explores the application of structured retrieval to the problem of retrieval for QA. Structured retrieval techniques found success in systems using Boolean logic or doc-

SIGIR'07, July 23–27, 2007, Amsterdam, The Netherlands.

ument structure, or some combination of the two. Examples include web document retrieval [12] and XML element retrieval [3]. This work is a logical extension of these ideas. In our approach, linguistic and semantic content of interest to the QA system is modeled as text annotations, which are represented as fields in the index. With a retrieval model that supports constraint-checking and ranking with respect to document structure and annotations as well as keywords, a QA system can directly express higher-level constraints on answer structures that can be checked at retrieval time.

We hypothesize that structured retrieval can improve retrieval performance for QA, compared to the bag-of-words approach, by retrieving more relevant documents that are more highly ranked, subject to certain environmental conditions: a) for particular types of questions; b) for particular corpora; and c) for particular sets of annotations. Structured retrieval may also boost overall system efficiency by retrieving fewer irrelevant results that must be processed.

In this paper, we present an approach to retrieval for QA based on an application of structured retrieval techniques, along with an empirical evaluation that supports our hypotheses. Section 2 gives a brief overview of related work. Section 3 discusses the retrieval needs of QA systems in more detail, and poses interesting questions about the nature of the problem of retrieval for QA. Section 4 describes a series of experiments designed to yield a better understanding of the issues raised by these questions. The results are presented and discussed in Sections 5 and 6. A summary of the contributions of the work thus far can be found in Section 7.

2. RELATED WORK

Considerable past research investigated whether concept-based, controlled vocabulary, or logical-form document representations provide better retrieval accuracy than full-text representations. Although some communities continue to study this issue, our interpretation of past research is that, in general, a combination of representations is more effective than any single representation alone [12, 15].

For many QA applications, concept-based and controlled vocabulary representations are not available. Most QA systems seek to gain some of the advantages of concept-based and logical-form representations by extending a full-text representation with text annotations that represent higher-level analysis, e.g. named entities, as in [14], or syntactic structure. Such a representation provides somewhat stronger semantics than a representation based on stemmed words.

An alternate approach is the two-step retrieval process favored by many QA systems, which involves using a sliding window passage scoring algorithm to choose the best passages from the top-n documents matching a bag-of-words query. Recent work has moved from density-based scoring, in which scores are boosted for proximate keyword occurrences [19], to scoring based on term dependency relations [4]. This work, however, still relies on an initial retrieval step; the dependency relations are used only to rerank the top 200 documents. In contrast, the structured retrieval approach scores the entire index at query time.

3. RETRIEVAL APPROACHES FOR QUESTION ANSWERING

In many QA systems, retrieval is used as a coarse, first-pass filter to narrow the search space for answers to the question. It has been argued in the QA literature that retrieval for QA and *ad-hoc* retrieval are different tasks with different requirements; while many *ad-hoc* retrieval evaluations place roughly equal emphasis on both precision and recall, retrieval for QA should be optimized for maximum recall of relevant documents [2, 9]. This gives imperfect answer extraction technology and redundancy-based answer validation the best chance of selecting the correct answer.

In this work, we consider two QA systems based on different retrieval approaches. Each system is built on a pipelined architecture that begins with a question analysis module that analyzes input questions and formulates queries, which are later executed by the retrieval system. Retrieved text goes through downstream processing, which includes constraint checking and answer extraction. The unit of retrieval for the remainder of this paper will be individual sentences.

3.1 System A: Bag-of-Words Approach

System A uses a bag-of-words retrieval approach, similar to [14], in which the queries contain question keywords and an operator that matches named entities corresponding to the expected answer type. This approach assumes only that the corpus has been annotated with named entity occurrences, which have been indexed along with the keywords for use at query time. System A must process each retrieved sentence with a semantic parser so that it has the information required for constraint checking and answer extraction. Any sentence for which the semantic parser fails is discarded, as the system can not extract answers from it. A more interactive variant of the System A architecture uses a pre-annotated corpus to avoid on-the-fly semantic parsing, but these analyses are not used at retrieval time; constraints are still checked as a part of the downstream processing.

3.2 System B: Structured Approach

The structured retrieval approach used by System B requires that the corpus has been pre-processed with a semantic parser and a named entity tagger. The semantic analyses are represented as text annotations, which are indexed along with the named entities and the keywords. System B's question analysis module analyzes the question using the semantic parser, mapping the resulting structure into one or more posited answer-bearing structures. These structures are expressed as sets of constraints on semantic annotations and keywords, and translate directly into structured queries. System B executes separate queries for each structure and merges the lists of results. Constraint checking is performed using the retrieved sentences' pre-cached semantic analyses.

3.3 Research Questions

When thinking about how to compare Systems A and B, several interesting questions come to mind. How does the effectiveness of the structured approach compare to that of bag-of-words? Does structured retrieval effectiveness vary with question complexity? To what degree is the effectiveness of structured retrieval dependent on the quality of the annotations? The following sections present a series of experiments designed to help address these questions.

4. EXPERIMENTAL METHODOLOGY

This section describes the retrieval system, and the methodology behind the two experiments discussed in this paper.

4.1 Retrieval System

Both experiments use the Indri [18] retrieval system, a part of the open-source Lemur toolkit[1]. Indri uses a variant of the Inference Network Model [20] to rank extents of text, in which belief estimates are based on statistical language models [22] instead of the Okapi ranking function. The index structures were extended to support query-time constraint-checking of hierarchical relationships among arbitrary overlapping fields, but the retrieval model is the same as in the release version.

4.2 Answer-Bearing Sentence Retrieval

We hypothesize that structured retrieval is capable of retrieving more relevant sentences at higher ranks, compared with bag-of-words. To test this hypothesis, we set up a comparison between Systems A and B, running the same set of questions over the same corpus and measuring recall of relevant sentences after the retrieval step. Note that, because the bag-of-words approach is ranking sentences, it will behave very similarly to a proximity-based retrieval function, which is a strong baseline for QA applications.

The relevant sentences in this evaluation are known as *answer-bearing sentences*, which are defined as completely containing the answer without requiring inference over the document. For example, consider the question, *Who killed Kennedy?* Even in a document describing the assassination, the sentence *Oswald killed him* can not be considered answer-bearing, because it requires anaphora resolution to understand that *him* refers to *Kennedy*. A sentence such as *Oswald assassinated Kennedy* is answer-bearing, because we do allow for synonymy. Similarly, *Everest is the tallest mountain* would be considered an answer bearing sentence for the question, *What is the highest point on Earth?*

For a given QA system, it is reasonable to assume that there are some answer-bearing sentences that the answer extraction component can not use, perhaps because the structure is too complex. As a result, this experiment examines two extreme conditions. The *single structure* case holds when only one type of answer-bearing sentence is usable. In the *every structure* case, all answer-bearing sentences are usable. We expect that, in practice, a given system will handle some but not all types of answer-bearing sentences, and so recall will fall somewhere between the extremes reported.

4.2.1 Corpus Preparation

This experiment uses the AQUAINT corpus, which has been used in several past TREC QA track evaluations. The AQUAINT corpus contains 375 million words of newswire text in English published by the Associated Press, the New York Times and the Xinhua News Agency from 1996 through 2000 [5]. The corpus was annotated with MXTerminator [16] for sentence segmentation, BBN Identifinder [1] for named entity recognition and ASSERT [13] version 0.11c, a shallow semantic parser. ASSERT identifies verb predicate-argument structures and labels the arguments with semantic roles in the style of PropBank [7]. When indexing verb predicate-argument structures in Indri, the verb, labeled *target* by ASSERT, is indexed as the parent of the arguments.

4.2.2 Topics and Judgments

For this experiment, we use a set of 109 factoid questions from TREC 2002 for which exhaustive, human relevance judgments over the AQUAINT corpus are available [2, 9]. From these document-level judgments, we derived sentence-level judgments by manually identifying the answer-bearing sentences within the relevant documents. For each question, every sentence in a relevant document that matched the question's TREC-provided answer pattern[2] was manually judged answer-bearing or not according to the definition. The 109 questions were randomly split into roughly equal sized training (55) and testing (54) sets, keeping the distribution of answer type constant across the two sets.

Question 1402
What year did Wilt Chamberlain score 100 points?
`#combine[sentence]( #any:date wilt chamberlain score 100 point )`

Figure 1: Bag-of-words query formulation example.

4.2.3 Query Formulation

System A's question analysis module constructs bag-of-words queries by filtering stopwords, stemming the remaining terms and adding them to a `#combine[sentence]` query clause. The clause also contains an `#any:`*type* operator that matches any occurrence of a named entity representing the expected answer type, when this can be determined. Figure 1 shows an example of bag-of-words query formulation.

System B's question analysis module maps the input question into a set of structured queries likely to retrieve answer-bearing sentences for that question. This mapping is a three-step operation. First, the verb predicate-argument structure of the question is analyzed. The question structure is then mapped into a set of posited answer-bearing structures. This mapping can be trained using hand-labeled answer-bearing structures, as shown in Figure 2, step 1. Finally, a query is formulated from each of these posited structures.

The process for creating a structured query from a posited structure is illustrated in Figure 2, steps 2 and 3. Named entities corresponding to question keywords are converted into clauses of the form `#max( #combine[`*type*`](` *keywords* `))`. The named entity expected answer type is converted into an `#any:`*type* operator. The query components are assembled recursively from the bottom-up. Arguments become clauses of the form `#max( #combine[./`*arg*`](` *contained clauses* `))`, containing keywords, named entity clauses, and nested targets, if any. The `./`*arg* extension to the query language checks that the *arg* annotation is a child of the annotation in the `#combine` that contains this query clause. Targets become clauses of the form `#max( #combine[target](` *keywords, clauses of child arguments* `))`.

4.2.4 Procedure

In the *single structure* case, the QA system is searching for a specific answer-bearing structure known to be usable for answer extraction, and formulates an explicit query to retrieve it. To model this situation, each answer-bearing structure posited by the question analysis module is considered a unique information need, where only sentences that match that structure are treated as relevant. There are 369 such structures in the training set, and 250 in the test set.

[1]See: http://www.lemurproject.org

[2]See: http://trec.nist.gov/data/qa/2002_qadata/main_task_QAdata/patterns.txt

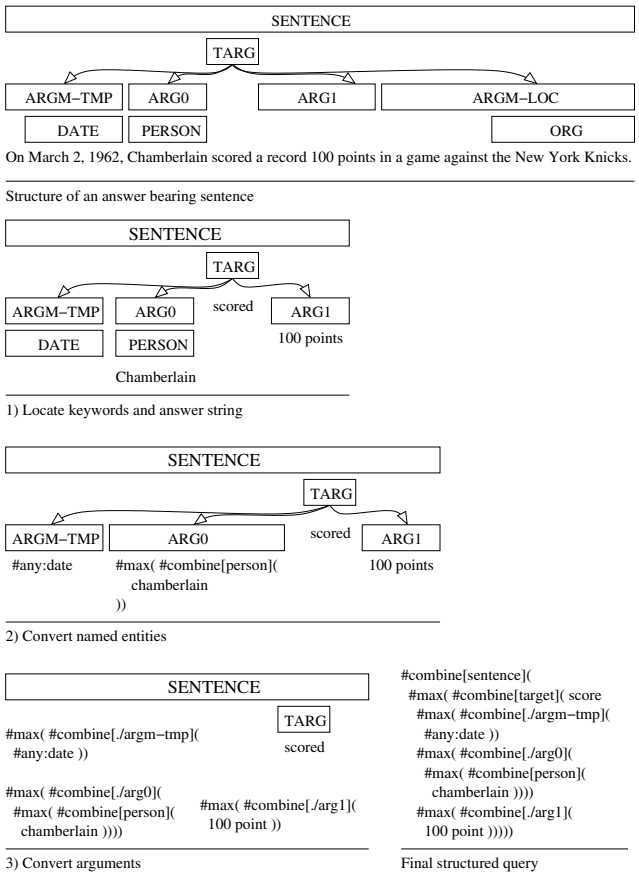

Figure 2: Identification of relevant structures (step 1), and the construction of a structured query (steps 2 and 3) from a relevant structure. Example is for Question 1402: *What year did Wilt Chamberlain score 100 points?*

In the *every structure* case, the QA system considers every answer-bearing structure useful for a particular question. In this case, either the results or the structured queries must be merged. We tried several standard meta-search techniques including combMNZ and combSUM [17], Reciprocal Rank [12] and Round Robin [21], as well as methods of query combination using an outer `#combine[sentence]` or a Boolean `OR` operator. Round Robin was found to be the best, even when compared with combSUM and combMNZ using common score normalization approaches [11]. The cause may be that scores of results retrieved by very different queries are not always directly comparable. The experiments described in Section 5.1 will use only the Round Robin meta-search technique. To be fair, the merging technique applied to the structured queries is also applied to the keyword queries. This preserves any advantages the bag-of-words approach might receive even though, in practice, it would likely only use one query to retrieve the results.

While it is less realistic than the *single structure* case, the *every structure* experiment is valuable, as it helps characterize the performance of the structured approach in the limiting case in which question analysis will produce every possible structure prior to retrieval. These two cases form the end-points of a space of possibilities. Most QA systems will likely fall somewhere between the two extremes in terms of the number of structures generated by question analysis.

4.3 The Effect of Annotation Quality

To measure the effect of annotation quality on structured retrieval performance, we ran two separate comparisons between Systems A and B running the same task on the same corpus, with the second run using degraded, or less accurate, annotations. We measure the difference between bag-of-words and structured retrieval, with and without the degraded annotations, in terms of recall of relevant sentences.

4.3.1 Corpus Preparation

This experiment uses a portion of the Penn Treebank containing one million words of Wall Street Journal text published in 1989 [10]. The text is hand-labeled for sentence boundaries, parts-of-speech and syntactic structure. We prepared two different versions of this corpus with varying quality of the annotations. The version we refer to as WSJ_GOLD contains gold-standard verb predicate-argument structures and semantic role labels for verb arguments from PropBank [7]. The other version is referred to as WSJ_DEGRADED and has semantic role annotations provided by ASSERT. Although ASSERT was trained on the WSJ_GOLD corpus, we have found that it is only 88.8% accurate in terms of number of arguments correctly identified and labeled. Gold-standard syntactic analysis was not made available to ASSERT as it annotated the WSJ_DEGRADED corpus.

4.3.2 Topics and Judgments

There are no sets of questions with associated relevance judgments readily available for use with the WSJ corpus, but the corpus is small enough that we were able to use the automatic procedure described in this section to generate an exhaustive list of questions answerable over the corpus, with their associated sentence-level relevance judgments.

For each sentence in the WSJ corpus, PropBank [7] contains zero or more hand-annotated predicate-argument structures, each of which contains zero or more arguments. These structures were grouped first by the predicate verb, and then by combinations of arguments shared in common. Each group is defined by a particular verb, and a particular set of arguments. A sentence containing s predicate-argument structures, where the i-th structure has a_i arguments, will appear in g groups:

$$g = \sum_{i=1}^{s} \sum_{j=1}^{a_i} \left(\begin{array}{c} a_i \\ j \end{array} \right)$$

Consider the following predicate-argument structure from document WSJ_0427, expressed in ASSERT's notation:

[ARG0 *Dow Jones*] [TARGET *publishes*] [ARG1 *The Wall Street Journal, Barron's magazine, other periodicals and community newspapers*]

This sentence's two-argument structure puts it into three groups corresponding to the questions it can answer:

1. [ARG1 *What*] *does* [ARG0 *Dow Jones*] [TARGET *publish*]*?*
2. [ARG0 *Who*] [TARGET *publishes*] [ARG1 *The Wall Street Journal, Barron's magazine, other periodicals and community newspapers*]*?*

3. *Does* [ARG0 *Dow Jones*] [TARGET *publish*] [ARG1 *The Wall Street Journal, Barron's magazine, other periodicals and community newspapers*]*?*

These groups also contain other sentences that have similar structures. For example, Group 1 contains two structures also having the TARGET *publish* and the ARG0 *Dow Jones*.

Once the grouping is complete, each group contains only and all of the sentences in the corpus containing predicate-argument structures that answer a particular question. Each group, then, constitutes exhaustive sentence-level relevance judgments for its corresponding question. Of the questions generated by this method, 96.3% had only one relevant sentence. These questions were discarded, as they would have skewed the averages, leaving 10,690 questions with associated judgments. This set of topics and judgments is known as the *original* judgments.

The experiment also uses a *reduced* judgment set, created by removing relevant sentences for which the degraded annotations do not match the gold-standard annotations from PropBank. This occurs because ASSERT can omit, mislabel or incorrectly identify the boundaries of an annotation. As an example, consider the following gold-standard PropBank structure from WSJ_0003:

[ARG1 *A form of asbestos*] [ARGM-TMP *once*] [TARGET *used*] [ARG2-PNC *to make Kent cigarette filters*]

Using the degraded annotations, the structure is:

[ARG0 *A form of asbestos*] [ARGM-TMP *once*] [TARGET *used*] *to make Kent cigarette filters*

ASSERT has fundamentally altered the meaning of the annotation by considering *a form of asbestos* to be the ARG0, or agent, of the verb *used* as opposed to the ARG1, or patient. ASSERT has also missed the ARG2-PNC (purpose, not cause). When ASSERT omits or mis-labels an argument required by a particular question, the sentence that contains the error is removed from the reduced set of judgments. Errors in bounding are tolerated if the gold-standard and degraded versions of the annotation overlap. Otherwise, it is considered an omission. In the reduced judgments, 802 of the 10,690 topics have no relevant sentences at all.

The reduced judgments model the reality of QA system, in which the relevance of a sentence can not be determined if the annotations are missing or incorrect. The QA system can not use a sentence for which the analysis fails. Even if it contains an answer, the system can not find it, because constraint checking and answer extraction both depend on the analysis.

4.3.3 Query Formulation

Query formulation for both bag-of-words and structured retrieval follows the procedure described in Section 4.2.3.

4.3.4 Procedure

Systems A and B are each run twice on the set of 10,690 questions. The first run uses the WSJ_GOLD corpus and the original judgments, while the second run uses the WSJ_DEGRADED corpus and the reduced judgments. The accuracy of the bag-of-words retrieval approach is not affected by degrading the annotations, since it completely ignores them when ranking results. System A, however, has no use for sentences that can not be analyzed with ASSERT, as answers can not be extracted from them. The reduced judgments model this situation by not assigning relevance to sentences for which the ASSERT annotations are missing or malformed in the WSJ_DEGRADED corpus.

5. EXPERIMENTAL RESULTS

In this section, the experimental results comparing bag-of-words and structured retrieval, and measuring the effect of degraded annotations on structured retrieval, are presented.

5.1 Answer-Bearing Sentence Retrieval

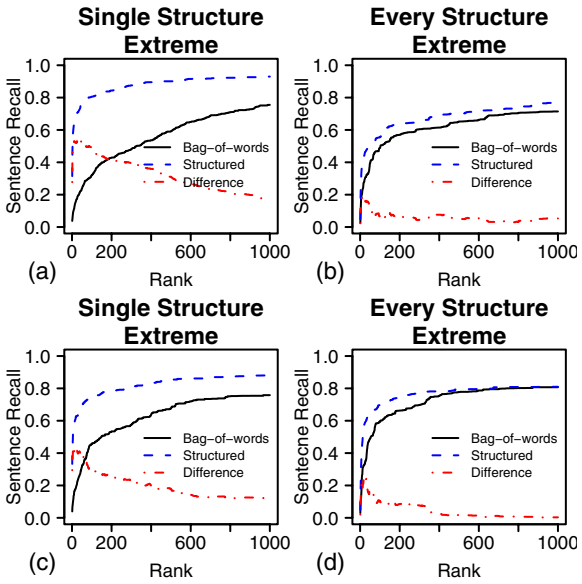

Figure 3: Recall of answer-bearing sentences for both bag-of-words and structured retrieval approaches on the AQUAINT corpus. Training topics are above and test topics are below.

This section describes the experimental results comparing the bag-of-words and structured retrieval approaches, in terms of recall of answer-bearing sentences, for both the *single structure* and *every structure* experimental conditions. All of the experimental runs described in this section use the Jelinek-Mercer smoothing method [22] in Indri, where the weights on the collection, document and sentence language models are 0.2, 0.2 and 0.6, respectively. These weights were chosen to optimize recall at rank 1000 for both approaches on the training topics. Smoothing using Dirichlet priors [22] was considered, and found to be sub-optimal.

Figure 3 shows the recall of the two approaches on the AQUAINT corpus. We see from Figures 3a and 3c that, for the *single structure* case, the structured retrieval approach has much higher recall than bag-of-words. For example, at rank 200, structured retrieval provides a 96.9% and 46.6% improvement over the bag-of-words approach on the training and test topics, respectively. Figures 3b and 3d compare the two approaches for the *every structure* case. The advantage that structured retrieval enjoys over bag-of-words is less pronounced, but the improvement at rank 200 is still 12.8% and 11.4% for the training and test topics, respectively.

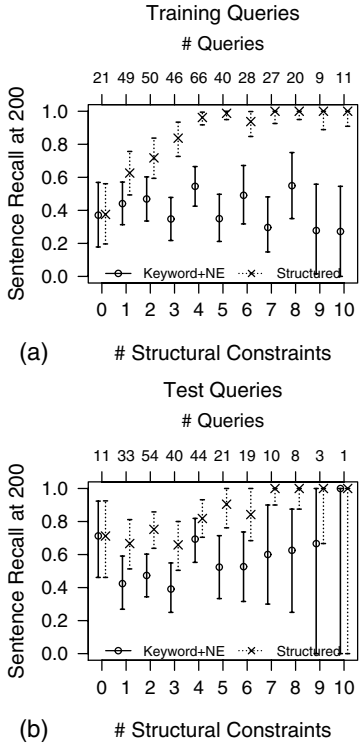

Figure 4: Recall at rank 200, varying structural complexity of answer-bearing sentences. Results are shown for the *single structure* case, where a separate query is formulated for each answer-bearing structure for each question. Structural complexity is estimated by counting the number of `#combine` operators in the structured query. Points show estimated average recall, with error bars covering a 95% confidence interval. The number of queries used for an estimate are listed along the top of the plot.

Figure 3 shows that structured retrieval has superior recall of answer-bearing sentences on average, compared to the bag-of-words approach, but tells little about the types of queries for which structured retrieval is most helpful. In an attempt to isolate the conditions where structure is important, Figure 4 shows the average recall at rank 200 for queries used in the *single structure* experiment having between zero and ten structural constraints. The number of structural constraints is defined as the number of `#combine` operators in the query, not counting the outer `#combine [sentence]` operator. The confidence intervals were constructed by simulating the distribution of average recall by averaging samples from a beta distribution fitted to the observed recall values using the method of moments estimator.

For both the training and testing sets, it appears that the more complex the structure sought, the more useful knowledge of that structure is in ranking answer-bearing sentences. Figure 4a also suggests that the performance of the bag-of-words approach is largely unaffected by the complexity of structure containing the keywords in the answer-bearing sentences. The results are less clear on the test topics, shown in Figure 4b. One factor is that there are fewer queries in the test set, especially for queries where the structure of the answer-bearing sentences is quite complex. This necessarily widens the confidence intervals of our estimates on the right side of the plot. Despite the difficulties of the test set, the structured run consistently performs better than the bag-of-words run; in many cases there is little or no overlap of the confidence intervals on our estimates.

5.2 The Effect of Annotation Quality

This section describes the results of the annotation quality experiment, comparing the structured and bag-of-words retrieval approaches, with and without degraded annotations. All experimental runs described in this section used the optimized smoothing parameters detailed in Section 5.1.

Figures 5a and 5b demonstrate that the structured queries are quite adept at retrieving most of the relevant items at early ranks, and that the bag-of-words queries take much longer to achieve a similar level of recall, at ranks beyond the horizon of the plot. For a QA system that needs to achieve a recall level of, say, 0.90, perhaps for an answer validation scheme based on redundancy, the system could process the top few results of a structured query, or process all the way down to rank 50 for a bag-of-words query. For situations in which post-processing of results is expensive, it may be more advantageous to choose structured retrieval.

The difference between Figures 5a and 5b is subtle and lies in the scale of the vertical axes. The curves in Figure 5b are similar to those in Figure 5a, but are shifted down slightly. Despite this shift, the relative difference between the structured and bag-of-words approaches remains almost the same. This trend is most easily seen in Figure 5c, which plots the difference curves from both Figures 5a and 5b on a common set of axes. This shows that the performance advantage that structured retrieval holds over bag-of-words is stable when the annotations are degraded, within the context of a QA system that considers a sentence relevant if and only if it is both answer-bearing and analyzable.

6. DISCUSSION

In the previous sections, we performed experiments that compare System A's bag-of-words retrieval approach and the structured retrieval approach used by System B. We wanted to understand if retrieval accuracy can be improved by using structured retrieval and, if so, how the benefit depends on the complexity of the query structures and the quality of the annotations. Structured retrieval clearly performs well at its intended purpose; leveraging document annotations, where available, to retrieve sentences satisfying certain constraints, and backing off gracefully to a keyword match where no annotations are available. Results in Section 5.1 have demonstrated that structured retrieval yields an improvement in recall of answer-bearing sentences, compared with bag-of-words retrieval. Structured retrieval performs best when query structures anticipate answer-bearing structures, and when these structures are complex.

One might have expected structured retrieval to be much better than bag-of-words for certain questions such as, *What country is Berlin in?* or *Who did Oswald kill?*. Questions such as these contain very common keywords that co-occur frequently. As a result, the keyword query should retrieve large numbers of documents that contain the correct terms, but not the structure that answers the question. We did not observe this in our experiments. When we examined the reasons why, we discovered a hidden bias in our question set; it

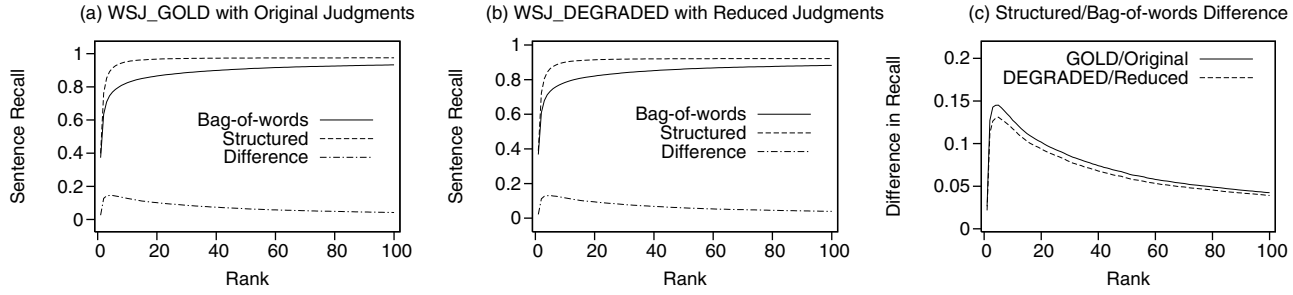

Figure 5: Average recall through rank 100, showing the difference between the structured and bag-of-words retrieval approaches on (a) WSJ_GOLD with original judgments and (b) WSJ_DEGRADED with reduced judgments. Both difference curves are shown in (c).

contains none of this type of question. To make the human assessment task tractable, questions such as *What country is Berlin in?* were intentionally excluded from consideration. This unfortunately biases the results in a way that advantages keywords. This bias notwithstanding, structured retrieval has demonstrated superior performance on the retrieval for QA task, compared to bag-of-words. In a more realistic sampling of questions, it stands to reason that the improvement afforded by structured retrieval could be even more pronounced.

Clearly, the quality of the annotations affects the performance of structured retrieval. Figure 5 shows that degrading the quality of the annotations has an adverse effect on recall. A real QA system, however, can only work with the annotations it has. If the system's text analysis components can not analyze a particular sentence, it is worthless because an answer can never be extracted from it. This is true regardless of the approach used to retrieve it. Figure 5b shows the more realistic picture. This comparison shows that, in a realistic situation without the benefit of gold standard-annotations, structured retrieval can still improve over bag-of-words, using the annotations that do exist.

Evaluation of structured queries is slower than keyword queries because there are many more operators in the inference network. When there is only one answer-bearing structure for a question, structured retrieval is 1.97 times as slow as bag-of-words. For a question with the average number of 6.22 structures, structured retrieval is 12.28 times as slow as bag-of-words. The structured approach, however, can offer an improvement in the sum total of retrieval time and answer extraction time. Extraction time is reduced because question analysis only posits answer-bearing structures that are easy to extract answers from. For the structures described in Section 4.2.3, answer extraction is a constant-time operation that involves plucking the value of the appropriate argument or named entity field from the structure.

The developer of a QA system could analyze the costs of the two retrieval approaches to decide which would achieve better run-time performance. The following example analysis uses estimates from the retrieval approaches, queries, and system used for the experiments in Section 5.1.

In this analysis we assume that the experimenter sets a desired average recall level l to be achieved by the retrieval system. Let $C_s(l)$ be the cost of retrieval and answer extraction of the structured retrieval approach and $C_b(l)$ be the cost for the bag-of-words approach. We would then prefer to use the structured retrieval approach if

$$C_s(l) < C_b(l) \tag{1}$$

The cost for both approaches can be expanded using the retrieval costs R_s and R_b and the time it takes to perform answer extraction on a sentence E:

$$R_s + n_s(l) \cdot E < R_b + n_b(l) \cdot E \tag{2}$$

where $n_s(l)$ is the number of sentences that must be retrieved for each query to achieve an average recall level of l for the structured approach and $n_b(l)$ is the number of sentences for the bag-of-words approach. In the above, we assume that the cost of retrieval is fairly constant with respect to the desired recall level. We also assume that the corpus has been pre-annotated for both approaches, so the cost of extraction from a sentence retrieved by bag-of-words is the same as the cost of extraction from a sentence retrieved by the structured approach. Further algebraic manipulation yields that it is more efficient to use the structured retrieval approach when

$$\frac{R_s - R_b}{n_b(l) - n_s(l)} < E \tag{3}$$

provided that $n_b(l) > n_s(l)$. The variables R_s, $n_b(l)$, and $n_s(l)$ are dependent on the nature of the question answering system, the corpus, and the questions. In this section, we assume that R_b does not depend on the number of candidate answer structures, even though we allowed multiple query variants in Section 5.1. This more accurately models the costs of bag-of-words retrieval, which would likely use only one query. We now consider specific observed experimental values for a desired recall level of $l = 0.75$.

Every structure case: On the test topics, assuming all answer-bearing sentences for a question are relevant, $n_b(l) = 393$, $n_s(l) = 209$, $R_s = 174$ seconds and $R_b = 15$ seconds, on average. It would therefore be more efficient to use the structured retrieval approach when the cost of extraction E exceeds 0.87 seconds per retrieved sentence, on average.

Single structure case: On the test topics, assuming only sentences matching a specific answer-bearing structure are relevant, $n_b(l) = 787$, $n_s(l) = 107$, $R_s = 30$ seconds and $R_b = 15$ seconds, on average. As a result, it would be more efficient to use the structured retrieval approach when $E > 0.022$ seconds per retrieved sentence, on average.

This analysis is simplified for the sake of this paper, and other researchers would have to consider the costs using their own systems before making a decision. Factors affecting E include any processing the QA system may do downstream of retrieval. If the corpus has not been pre-annotated, the time required to analyze results on-the-fly will be added to E. Additional downstream processing that may affect E includes answer merging, validation and presentation.

7. CONTRIBUTIONS

This paper presents an approach to retrieval for Question Answering that directly supports indexing and retrieval on the kind of linguistic and semantic constraints that a QA system needs to determine relevance of a retrieved document to a particular natural language input question. Our approach to structured retrieval for QA works by encoding this linguistic and semantic content as annotations on text, and by using a retrieval model that directly supports constraint-checking and ranking with respect to document structure and annotations in addition to keywords.

We present an evaluation that shows that, when a QA system is able to posit the likely answer-bearing structures it is looking for within the text collection, structured queries retrieve more relevant results, more highly-ranked, when compared to traditional bag-of-words retrieval. In addition to providing a higher quality ranked list of results to pass to downstream modules, structured retrieval can also reduce the sum total retrieval and extraction time. Although a structured query is slower to execute than a bag-of-words query, the system does not have to parse the retrieved results or process as far down the ranked list as it would have to if it were working with less relevant results obtained from a bag-of-words query. Interactive systems can respond more quickly using structured retrieval, as they would not have to parse texts at retrieval time while the user waits.

We observe that structured retrieval can be more effective than bag-of-words, but we have not spent much effort optimizing data structures and algorithms. With current tools, a benefit is realized at 200 sentences, which is reasonable for today's QA systems. Current QA systems do not process thousands of results, preferring instead to find a "sweet spot" of enough results to achieve sufficient recall, but not so much that they can not be post-processed. With more efficient retrieval algorithms and indexing structures, we may see a benefit for a broader range of conditions than currently observed in this paper.

We characterize the extent to which structured retrieval performance depends on the quality of the predicate-argument annotations, leaving named entity annotations for future work. Although accuracy degrades when the annotation quality degrades, the relative performance edge that structured retrieval enjoys over bag-of-words retrieval is maintained. The reason for this is that the QA system can neither determine relevance nor extract answers from text that can not be properly annotated by the available tools. This is true whether the text is annotated ahead of time, as in the structured retrieval approach, or on-the-fly, after having been retrieved by a bag-of-words query.

If the structure of a sentence impacts the relevance of that sentence, it is crucial to index and query on that structure. Within the context of a QA system, it is reasonable to imagine that the process of question analysis could yield a small set of candidate answer-bearing structures that would match common or likely ways of expressing the answer in the corpus. By choosing the structured retrieval approach instead of bag-of-words, a QA system can improve recall of relevant sentences, which can translate to improved end-to-end QA system accuracy and efficiency.

8. REFERENCES

[1] D. Bikel, et al. An algorithm that learns what's in a name. *ML*, 34(1-3):211–231, 1999.

[2] M. Bilotti, et al. What works better for question answering: Stemming or morphological query expansion? In *Proc. of IR4QA at SIGIR*, 2004.

[3] D. Carmel, et al. Searching XML documents via XML fragments. In *Proc. of SIGIR*, 2003.

[4] H. Cui, et al. Question answering passage retrieval using dependency relations. In *Proc. of SIGIR*, 2005.

[5] D. Graff. *The AQUAINT Corpus of English News Text*. LDC, 2002. Cat. No. LDC2002T31.

[6] S. Harabagiu, et al. Employing two question answering systems in TREC-2005. In *Proc. of TREC-14*, 2005.

[7] P. Kingsbury, et al. Adding semantic annotation to the penn treebank. In *Proc. of HLT*, 2002.

[8] C. Lin and E. Hovy. The automated acquisition of topic signatures for text summarization. In *Proc. of COLING*, 2000.

[9] J. Lin and B. Katz. Building a reusable test collection for question answering. *JASIST*, 57(7):851–861, 2006.

[10] M. Marcus, et al. Building a large annotated corpus of english: the penn treebank. *CL*, 19(2):313–330, 1993.

[11] M. Montague and J. Aslam. Relevance score normalization for metasearch. In *Proc. of CIKM*, 2001.

[12] P. Ogilvie and J. Callan. Combining document representations for known-item search. In *Proc. of SIGIR*, 2003.

[13] S. Pradhan, et al. Shallow semantic parsing using support vector machines. In *Proc. of HLT*, 2004.

[14] J. Prager, et al. Question-answering by predictive annotation. In *Proc. of SIGIR*, 2000.

[15] T. Rajashekar and W. B. Croft. Combining automatic and manual index representations in probabilistic retrieval. *JASIS*, 46(4):272–283, 1995.

[16] J. Reynar and A. Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proc. of ANLP*, 1997.

[17] J. Shaw and E. Fox. Combination of multiple searches. In *Proc. of TREC-3*, 1994.

[18] T. Strohman, et al. Indri: A language model-based search engine for complex queries. In *Proc. of ICIA*, 2005.

[19] S. Tellex, et al. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proc. of SIGIR*, 2003.

[20] H. Turtle and W. Croft. Evaluation of an inference network-based retrieval model. *ACM TOIS*, 9(3):187–222, 1991.

[21] E. Voorhees, et al. The collection fusion problem. In *Proc. of TREC-3*, 1994.

[22] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proc. of SIGIR*, 2001.