In Proceedings of the Information Retrieval for Question Answering (IR4QA) Workshop at SIGIR 2004, July 2004, Sheffield, England

What Works Better for Question Answering: Stemming or Morphological Query Expansion?

Matthew W. Bilotti, Boris Katz, and Jimmy Lin
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, Massachusetts, USA
{mbilotti, boris, jimmylin}@csail.mit.edu

ABSTRACT

How do different information retrieval techniques affect the performance of document retrieval in the context of question answering? An exploration of this question is our overall research goal. In this paper, we specifically examine strategies for coping with morphological variation. This work quantitatively compares two different approaches to handling term variation: applying a stemming algorithm at indexing time, and performing morphological query expansion at retrieval time. We discovered that, compared to the no-stemming baseline, stemming results in lower recall, and morphological expansion yields higher recall. By separately weighting different term variants, we were able to achieve even higher recall, which opens the door to interesting question analysis algorithms for sophisticated query generation. Another significant contribution of our work is the development of a reusable question answering test collection to support our experiments.

Categories and Subject Descriptors

H.3.1 [**Information Systems**]: Content Analysis and Indexing; H.3.3 [**Information Systems**]: Information Storage and Retrieval—*Query formulation*; H.3.4 [**Information Systems**]: Systems and Software—*Question-answering (fact retrieval) systems*

General Terms

Stemming, morphology, query expansion, indexing

Keywords

Question answering

1. INTRODUCTION

The task of a question answering (QA) system is to provide direct, succinct responses to natural language questions posed by a user. Current research focuses on so-called "factoid" questions such as "How many floors are in the Empire State Building?", which can typically be answered by a short noun phrase. Unlike document retrieval systems, which return ranked lists of potentially relevant documents, question answering systems seek to pinpoint answers directly.

Over the past few years, the question answering tracks at the Text Retrieval Conferences (TRECs) [13, 10, 11, 12] have brought formal and rigorous evaluation methodologies to bear on the question answering task. Every year, participants are given a blind testset of natural language questions and a corpus of approximately one million news articles (the current task uses the AQUAINT corpus). Systems must automatically extract answers from the corpus within a fixed timespan to complete the task. These evaluations provide a shared forum for comparing different question answering techniques, and serve as an effective vehicle for the dissemination of results.

Although factoid question answering is distinct from the task of retrieving relevant documents in response to a user query (so-called *ad hoc* retrieval), document retrieval systems nevertheless play a central role in the question answering process. Because natural language processing techniques are relatively slow, a question answering system typically relies on traditional document retrieval techniques to first produce a set of candidate documents, thereby reducing the amount of text that must be analyzed.

Functionally, most question answering systems today can be decomposed into four major components (see Figure 1): question analysis, document retrieval, passage retrieval, and answer extraction (cf. [3, 10]). The question analysis component classifies user questions by the expected semantic type of the answer, e.g., the expected answer type of "Where was Kennedy assassinated?" is *location*. In addition, it is responsible for formulating one or more queries targeted at a particular document retriever; these queries are used to find a set of potentially relevant documents from the corpus. From these documents, the passage retrieval component selects a handful of paragraph-sized fragments. Most often, passage retrieval algorithms perform a density-based weighting of query terms, i.e., they favor query terms that appear close together (see [9] for a survey). In some systems, however, document and passage retrieval are performed simultaneously. Finally, the answer extraction component searches the passages for the answer to the question, for example, by finding named-entities that match the expected answer type.

In a pipelined question answering architecture, recall is more important than precision at the document retrieval stage. Irrelevant documents can be filtered by downstream modules, which may have access to more linguistic knowledge and better reasoning capabilities. Relevant documents that are not returned by a document retriever, however, pose serious problems. If a document containing the answer is not retrieved in the first place, then no amount of intelli-

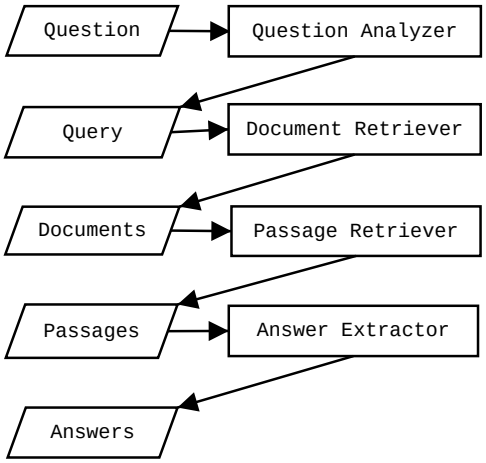

Figure 1: A typical pipeline question answering architecture.

gent processing by subsequent modules will matter. Given that document retrieval is a component in a larger question answering architecture, it is unclear how different retrieval techniques affect the overall end-to-end performance of the system.

The question of how document retrieval affects question answering performance has been previously explored by some authors. For example, Tellex *et al.* [9] performed a quantitative evaluation of passage retrieval algorithms for question answering. They discovered the importance of density-based metrics for passage scoring, and revealed some interesting cross-coupling effects between document and passage retrieval. Monz [6] has similarly experimented with a variety of document retrieval techniques in the context of question answering: blind relevance feedback, passage retrieval, stemming, structured queries, term selection, and term expansion. He showed conclusively that techniques geared toward *ad hoc* retrieval cannot be directly applied to document retrieval for question answering.

It is our belief that *ad hoc* retrieval and document retrieval for question answering represent different sets of needs, requirements, and tradeoffs. An in-depth exploration of these differences is one of our major research goals. In addition to the obvious difference in output behavior, question answering distinguishes itself from *ad hoc* retrieval in not working with full-length topics, but rather with natural language questions, which generally contain far fewer query terms.

In this paper, we specifically examine two different strategies for coping with morphological variation: stemming and query expansion. Does either technique improve document retrieval performance for the purpose of question answering? Should the corpus be indexed with word stems, or should morphologically-related terms be added to the query at retrieval time?

2. PREVIOUS WORK

Morphological variation poses a challenge to all types of information retrieval systems. Ideally, a system should be able to retrieve documents containing closely-related variants of keywords found in the query, e.g., the query term *love* should not only match the term *love* present in documents, but also the terms *loving*, *loved*, and *loves*.

There are, in principle, two different ways for coping with morphological variation. The most popular strategy is to apply a stemming algorithm at indexing time and store only the resulting word stems; this naturally requires user queries to be similarly analyzed. The effects of this morphological normalization process have been well studied in the context of document retrieval, but it is still unclear whether or not stemming is effective. Since morphologically related words are believed to denote the same semantic concept, this technique should boost recall without negatively affecting precision. Experimental results, however, point to a more complex picture.

An alternative strategy is indexing the original word forms, as they are, and expanding query terms with their morphological variants at retrieval time. Performing query expansion at retrieval time, however, requires the ability for structured querying, a capability that may not be present in all document retrieval systems. For convenience, we will refer to the first approach as stemming, and the second approach as morphological query expansion. Although we compare the effectiveness of both strategies, previous studies focus for the most part on stemming.

Harman [2] has shown that the application of a stemming algorithm does not improve document retrieval performance in general; stemming hurts as many queries as it helps, leading to little overall performance change. When stemming does help, however, she found the improvements to be minor—at most changing the ranks of relevant documents by a couple of positions. The downside of stemming is that it often promotes the scores of irrelevant and relevant documents equally. Depending on the degree to which the stemmer conflates word variants, irrelevant documents can often outrank relevant documents.

Other researchers have presented findings that are different from those of Harman. Popovic and Willett [7] report that indexing stemmed word forms clearly improves document retrieval performance for morphologically-rich languages. Even for English, which does not exhibit particularly complex word morphology, Krovetz [5] demonstrated that linguistically-motivated stemming does indeed contribute to better retrieval performance. Work by Hull [4] showed that some form of stemming almost always results in at least a small improvement in retrieval performance, but can make a huge difference for certain individual queries. More recently, Monz [6] has specifically examined the issue of stemming in the context of question answering. He reported increases in both precision and recall when employing a stemmer at indexing time.

3. EXPERIMENTAL DESIGN

The primary purpose of our experiments is to study the effects of document retrieval techniques in the context of question answering. In particular, we set out to determine which strategy for handling morphological variation results in higher performance: applying a stemmer at indexing time, or performing query expansion at retrieval time. We believe

that there are deficiencies in the test collection currently being used by the question answering community; therefore, to support our experiments, we have manually crafted a question answering test collection that specifically addresses these issues.

When we discuss document retrieval in the context of question answering, we are referring to the top two rows in Figure 1. Given a natural language question, our system generates a set of queries to a document retrieval system. The system is evaluated on what proportion of the returned documents are relevant to the question asked, where a "relevant document" is defined as not only containing the answer to the question, but also supporting it. It is assumed that subsequent processing modules in the pipeline will pinpoint the location of the answer within these documents. Fetching relevant documents is an important first step in the question answering process, and is a sub-task that can be isolated and independently evaluated.

Our experiments are situated in the context of a question answering testbed called Pauchok [9, 1]. Its architecture supports the rapid construction and evaluation of different question answering components. Pauchok is built on top of Lucene[1], an open-source document retrieval system that implements *tf-idf* weighting and supports boolean queries.

3.1 Test Collection

Natural language questions from previous TREC evaluations serve as the basis for our test collection. Each year, NIST compiles a list of known relevant documents by pooling the responses of all participants. Since the average performance of factoid question answering systems at the evaluations is still somewhat poor (see [12] for a summary of last year's results), the number of known relevant documents for each question is exceedingly small, averaging 1.95 relevant documents per question on the TREC 2002 testset. In the same testset, no single question had more than four known relevant documents. Even a casual examination of the AQUAINT corpus reveals the existence of many more relevant documents, demonstrating that the judgments are not, in fact, exhaustive. Moreover, careful inspection of the documents reveals duplicates and errors. To be fair, NIST merely provides this list every year for convenience; they were never meant to serve as a complete test collection for document retrieval experiments. For lack of any better resources, these partial judgments have been employed by many researchers in question answering experiments.

To understand the danger of evaluating systems with the current set of judgments, one must first understand how they are presently being used. In the standard setup, the output of a question answering system is a pair consisting of an answer string and a supporting document. To automatically score an answer, the answer string is matched against NIST-supplied answer patterns, and the supporting document is matched against the list of known relevant documents. Because this list is far from exhaustive, new retrieval techniques may not be properly rewarded—a perfectly acceptable answer may be judged as incorrect simply because its supporting document does not appear on the list of known relevant documents. In the context of question answering, it is entirely possible that with sophisticated linguistic processing, a system can extract answers from documents that share few or no keywords with the question. As a result, systems employing advanced techniques are likely to return documents that have never been assessed before, but which are assumed to be irrelevant by default. The current set of judgments serves as a poor basis for evaluating new systems; without an exhaustive set of relevant documents, one would never know if a system's performance is actually improving.

The creation of a truly reusable test collection for factoid question answering remains an open research problem, and the issues become more challenging as the community moves toward the evaluation more difficult question types, e.g., definition and relationship questions, or those that involve inferencing. We decided to tackle this problem by confronting it head on, and over the past few months, we have manually built a reusable question answering test collection [1] consisting of 120 questions selected from the TREC 2002 testset. This testset is based on the AQUAINT corpus.

Working from known answers to the TREC 2002 questions, we constructed queries with certain terms selected from each question and its answer, terms which we believed a relevant document would have to contain. Although it is entirely possible that a relevant document may contain none of the words from the question and the answer, we assumed that this happens very rarely. Using these queries, we retrieved tens of thousands of documents that we manually examined, judging each to be either relevant, irrelevant, or unsupported for a particular question.

To give a concrete example, consider question 1396, "What is the name of the volcano that destroyed the ancient city of Pompeii?", whose answer is "Vesuvius". Any relevant document containing the answer to this question must necessarily contain the keywords "Pompeii" and "Vesuvius"; therefore, we manually examined all documents with those two keywords. For this question, we noted fifteen relevant, three unsupported, and ten irrelevant documents. All other documents not explicitly marked are presumed to be irrelevant. An example of a clearly relevant document is `APW19990823.0165`[2] which says that "In A.D. 79, long-dormant Mount Vesuvius erupted, burying the Roman cities of Pompeii and Herculaneum in volcanic ash." An unsupported document is one that contains the answer and discusses it in the correct sense and context, but does not completely and clearly answer the question. An example is `NYT20000405.0216`, which states that, "Pompeii was pagan in A.D. 79, when Vesuvius erupted." The document also addresses speculations that "the people of Pompeii were justly punished by the volcano eruption," but does not explicitly mention or imply that the city was destroyed by Vesuvius. An irrelevant document containing the terms "Pompeii" and "Vesuvius" is `NYT20000704.0049`, which discusses winemaking in Campania, the region of Italy containing both Pompeii and Vesuvius. The document talks about vineyards near the ruins of Pompeii, and about a species of grape that grows in the volcanic soil at the foot of Mt. Vesuvius.

[1] `http://jakarta.apache.org/lucene`

[2] Incidentally, this document is not present in the NIST-supplied list of relevant documents.

For each question in the test collection, we have compiled three lists of documents, one each for those known to be relevant, unsupported, and irrelevant. Due to our methodology of working backwards from the answer, we are fairly confident that the lists of relevant documents are exhaustive. There are an average of 15.84 relevant documents for each question, which is an increase of approximately an order of magnitude from the NIST-supplied relevant documents list for TREC 2002. We hope that this test collection will provide more accurate results for our document retrieval experiments. In the near future, we plan on releasing this test collection for use by the research community.

3.2 Experimental Conditions

Using Lucene, we constructed two separate indexes of the AQUAINT document collection, which contains approximately one million articles totalling around three gigabytes. The first index (unstemmed) simply stores the original word forms. The second index (stemmed) employed the Porter stemming algorithm [8] to build a stemmed index (which conflates term variants that the Porter algorithm considers to have the same root).

For querying, we used a boolean "conjuncts of disjuncts" paradigm. It has been shown previously that, for the question answering task, boolean queries result in comparable performance to state-of-the-art ranked retrieval systems, but have the added advantage of allowing finer-grained manipulation of structured queries [9].

The basic query to the Lucene document retriever is a conjunction of clauses, where each clause is a disjunction of the original query term and its morphological variants (if used). We used a simple dropping strategy that gradually relaxes the query by successively dropping the disjunct associated with the most common query term, i.e., the one with the lowest inverse document frequency. Our system sequentially executes each query, concatenating the results (removing duplicate documents) until a preset limit has been achieved.

With this general querying scheme, we performed the following experiments:

1. **Baseline**: The baseline experiment contains no stemming or morphological expansions of any sort. Query terms simply consist of unmodified non-stopwords from the natural language question, and queries are issued on the non-stemmed Lucene index.
2. **Stemming**: This experiment tests the stemming technique. Query terms consist of non-stopwords from the natural language question that have been stemmed using the Porter algorithm, and queries are issued on the Porter-stemmed Lucene index.
3. **Unweighted Inflectional Expansion**: In this experiment, the inflectional variants of every non-stopword term are added to the query. Inflectional variants for verbs include different conjugations, inflectional variants for nouns include different pluralizations, and inflectional variants for adjectives include the comparative and superlative forms. As previously described, the final query is a conjunction of disjuncts, where each of the conjoined clauses contains the original query term and its variants joined together by a boolean *or*.
4. **Weighted Inflectional Expansion**: This experimental condition is exactly the same as the previous one, with the exception that morphological variants are assigned a discount factor, i.e., matching a variant produces a lower score than matching the original query term. This weight has been previously tuned for the corpus and for our evaluation framework [1].

For each of the four experimental conditions, we performed runs for five different values of the document limit (i.e., maximum number of documents to return): 100, 250, 500, 750 and 1000. Although most question answering systems are relatively limited in the number of documents that can be analyzed by linguistically-sophisticated techniques, we still report figures with larger limits in anticipation of faster and more efficient language processing algorithms.

As a concrete example, consider question 1410, "What lays blue eggs?" The following shows the base queries generated for this question in each of our four experimental conditions:

- **Baseline:** $blue \wedge eggs \wedge lays$
- **Stemming:** $blue \wedge egg \wedge lai$
- **Unweighted Inflectional Expansion:** $blue \wedge (eggs \vee egg) \wedge (lays \vee laying \vee lay \vee laid)$
- **Weighted Inflectional Expansion:** $blue \wedge (eggs \vee egg^{\alpha}) \wedge (lays \vee laying^{\alpha} \vee lay^{\alpha} \vee laid^{\alpha})$

In each query, the terms are arranged in order of increasing *idf*, and subsequent (backoff) queries are formulated by successively dropping the first conjunct. The only difference between the unweighted inflectional expansion and weighted inflectional expansion condition is the parameter α, the discount factor assigned to morphological variants of query terms. As previously mentioned, this weight was tuned independently [1].

3.3 Evaluation Metrics

Because document retrieval techniques are typically employed to produce a smaller set of candidate documents that are subsequently processed by other modules, we believe that the performance tradeoff should favor recall over precision. As previously mentioned, it is possible for downstream modules to filter out wrong answers, but it is impossible for additional processing layers to cope with relevant documents that the document retriever failed to return.

For our experiments, we collected two metrics: recall at n and total document reciprocal rank (TDRR). Recall at n is simply the fraction of known relevant documents that our system has fetched up to a particular cutoff. The explanation of total document reciprocal rank is a bit more complicated: the reciprocal rank of a relevant document is the inverse of the rank at which it is retrieved, i.e., a relevant

		Recall				TDRR			
Limit	Experiment	relevant	Δ	both	Δ	relevant	Δ	both	Δ
100	unstemmed	0.2720		0.2595		0.6403		0.6673	
	stemmed	0.2589	−4.82%	0.2460	−5.20%	0.5869	−8.33%	0.5987	−10.28%
	expanded	0.2748	+1.03%	0.2612	+0.66%	0.5752	−10.16%	0.5968	−10.56%
	w. expanded	0.2944	+8.24%	0.2798	+7.82%	0.6094	−4.82%	0.6305	−5.52%
250	unstemmed	0.3738		0.3584		0.6509		0.6790	
	stemmed	0.3626	−3.00%	0.3474	−3.07%	0.5995	−7.90%	0.6122	−9.84%
	expanded	0.3682	−1.50%	0.3533	−1.42%	0.5863	−9.93%	0.6090	−10.31%
	w. expanded	0.3776	+1.02%	0.3618	+0.95%	0.6185	−4.98%	0.6406	−5.67%
500	unstemmed	0.5393		0.5123		0.6596		0.6879	
	stemmed	0.5364	−0.54%	0.5097	−0.51%	0.6086	−7.74%	0.6216	−9.65%
	expanded	0.5467	+1.37%	0.5182	+1.15%	0.5957	−9.69%	0.6186	−10.08%
	w. expanded	0.5551	+2.93%	0.5258	+2.64%	0.6279	−4.81%	0.6501	−5.50%
750	unstemmed	0.5981		0.5689		0.6614		0.6899	
	stemmed	0.5934	−0.79%	0.5638	−0.90%	0.6103	−7.72%	0.6234	−9.63%
	expanded	0.6093	+1.87%	0.5799	+1.93%	0.5976	−9.65%	0.6207	−10.03%
	w. expanded	0.6112	+2.19%	0.5816	+2.23%	0.6296	−4.81%	0.6520	−5.49%
1000	unstemmed	0.6196		0.5917		0.6618		0.6904	
	stemmed	0.6131	−1.05%	0.5824	−1.57%	0.6111	−7.67%	0.6238	−9.64%
	expanded	0.6290	+1.52%	0.5993	+1.28%	0.5980	−9.65%	0.6211	−10.03%
	w. expanded	0.6290	+1.52%	0.5993	+1.28%	0.5980	−9.65%	0.6211	−10.03%

Table 1: Performance of different document retrieval algorithms.

document at the first position receives a score of 1, at rank 2, 1/2, at rank 3, 1/3, etc. The total document reciprocal rank is simply the sum of all reciprocal ranks of all relevant documents per question (averaged over all questions). With a cutoff of 100 documents, for example, the maximum TDRR, corresponding to the situation where all the retrieved documents are relevant, is $1 + 1/2 + 1/3 + 1/4 + \ldots + 1/100$. The TDRR can thus be viewed as a weighted recall, where higher-ranking documents are favored.

In all our experiments, we measure both recall and TDRR against the list of relevant documents (the "relevant" condition) and the union of relevant and unsupported documents (the "both" condition).

4. RESULTS

Our experimental results are shown in Table 1. In addition to absolute recall and TDRR values for each of the experimental conditions, we have provided performance differences against the baseline (which employed only the original word forms).

In terms of recall, we discovered that indexing and retrieving based on Porter word stems negatively affects performance at all document limits. Expanding inflectional variants results in consistently higher recall, and separately weighting the score contributions of the inflectional variants further increases recall at all document limits. Not surprisingly, these effects were most prominent at lower document cutoff limits.

A very interesting picture emerges when performance is measured in terms of total document reciprocal rank (TDRR). Both stemming and expansion strategies result in consistently lower scores compared to the baseline, at all document limits. This means that although our system is fetching more relevant documents in the case of the expansion strategy, relevant documents appear at lower ranks; in other words, both stemming and expansion techniques are promoting the scores of some irrelevant documents more than they are promoting the scores of relevant documents.

5. DISCUSSION

Our experimental results can be summarized as follows: compared to an unstemmed baseline, indexing-time stemming with the Porter stemmer results in lower recall, while retrieval-time expansion with inflectional variants produces higher recall. It appears, however, that both strategies for coping with morphological variation lower the total document reciprocal rank, i.e., relevant documents are returned at lower ranked positions. Note that this result is inconsistent with the findings of Monz [6], who reported both increased precision and recall when employing the Porter stemmer at indexing time (compared to an unstemmed baseline). These differences may be attributed to the usage of our manually-crafted test collection or different retrieval strategies (boolean vs. vector space).

Our findings can be explained in a number of ways. The most obvious culprit that contributes to the recall decline in indexing-time stemming is the stemming algorithm itself. Because the Porter stemmer uses a purely orthographic algorithm that does not understand the semantics of the word forms it is altering, it is prone to large classes of errors in conflating totally unrelated words. Since most stemmers do not use a lexicon, many spurious forms are created, and some reasonable stemmings are not. For example, it is well-documented that the Porter algorithm stems "organization" to "organ", and conflates "police" with "policy". On the other hand, the Porter stemmer fails to make appropriate transformations such as stemming "european" to "europe". In query expansion, we have taken care to generate only sensical term variants, ensuring that all generated word forms

are conceptually related and relevant. The difference in recall between the "stemmed" run and the "unweighted inflectional expansion" run is a gauge of how much conflating unrelated query terms (by the Porter stemmer) affects performance.

Another distinct advantage of morphological query expansion is the possibility of assigning different weights to different term variants, reflecting the relative importance of each variant and its "semantic distance" to the concept represented by the original term. We have shown that by simply weighting the expanded variants less than the original term from the question, we can further boost the recall of our system (this is shown by the "weighted inflectional expansions" runs). Although term variants are uniformly weighted under our current scheme, there is no reason why these weights cannot be individually assigned after performing linguistic analysis on the question. This opens the door for question analysis techniques that are able to detect situations where the selective addition of certain query terms would be beneficial. Sophisticated processing techniques can lead to custom weighting schemes optimized for particular classes of natural language questions, potentially increasing recall even further. The ability to manipulate the relative importance of different morphological forms on a per-query basis is a key advantage offered by the general paradigm of query expansion; such techniques are impossible if a system indexes stemmed word forms.

As with many document retrieval techniques, increased recall comes at a cost. In this case, total document reciprocal rank suffers with both stemming and expansion techniques. This means that although more relevant documents are being retrieved, some irrelevant documents with morphological variants are being promoted above relevant documents. With the stemming approach, this is unsurprising because the Porter stemmer incorrectly conflates unrelated terms. In the expansion approach, the presence of additional terms retrieves documents that use the variants in different contexts. It is unclear, though, what impact lower TDRR scores will have on question answering performance, especially since the metric is very sensitive to position swaps at the higher document ranks. For example, a document retrieved at rank one in one experiment and at rank two in another experiment will have an absolute TDRR score difference of 0.5, but to downstream modules, the difference might be inconsequential. From our own experience, we have argued that recall is of utmost importance in end-to-end question answering performance, especially in a pipeline-style architecture. We have not, however, performed any quantitative experiments to support this claim. Future work will be needed to explore the system-wide performance effects of the recall–TDRR tradeoff.

6. FUTURE DIRECTIONS

In this study, we have discovered that handling morphological variation using query expansion not only results in higher recall, but allows better control over the query generation process. This improved flexibility of querying opens new doors for question analysis, immediately suggesting avenues for future research. Now that we have the ability to do selective query expansion, how can we develop methods of question analysis to take advantage and fine-tune this process? How can we automatically decide which variants to use, and how much to weight them? These are important questions that will guide our future research in this area. We believe that document retrieval performance can be improved by tailoring query expansion strategies to specific classes of questions, and we are currently developing and evaluating such techniques.

7. CONCLUSION

The effectiveness of different techniques for handling morphology varies with the nature of the queries, the test collection, the judgments, and the metrics. For document retrieval in the service of answering short, fact-based, natural language questions, we have discovered that indexing stemmed word forms results in decreased recall, while retrieval-time query expansion increases recall. Higher recall, however, comes at the tradeoff of lower TDRR. These results are consistent with our own intuitions, and are confirmed by quantitative experiments performed against a comprehensive, reusable question answering test collection that we meticulously constructed by hand. These findings contribute to our higher-level goal of understanding how document retrieval relates to question answering.

8. ACKNOWLEDGMENTS

This work was supported in part by the Advanced Research and Development Activity (ARDA)'s Advanced Question Answering for Intelligence (AQUAINT) Program.

9. REFERENCES

[1] M. W. Bilotti. Query expansion techniques for question answering. Master's thesis, Massachusetts Institute of Technology, 2004.

[2] D. Harman. How effective is suffixing? *Journal of the American Society for Information Science*, 42(1):7–15, 1991.

[3] L. Hirschman and R. Gaizauskas. Natural language question answering: The view from here. *Journal of Natural Language Engineering, Special Issue on Question Answering*, Fall–Winter 2001.

[4] D. A. Hull. Stemming algorithms: a case study for detailed evaluation. *Journal of the American Society for Information Science*, 47(1):70–84, 1996.

[5] R. Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–202. ACM Press, 1993.

[6] C. Monz. *From Document Retrieval to Question Answering*. Ph.D. dissertation, Institute for Logic, Language, and Computation, University of Amsterdam, 2003.

[7] M. Popovic and P. Willett. The effectiveness of stemming for natural-language access to Slovene textual data. *Journal of the American Society for Information Science*, 43(5):384–390, 1992.

[8] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[9] S. Tellex, B. Katz, J. Lin, G. Marton, and A. Fernandes. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, 2003.

[10] E. M. Voorhees. Overview of the TREC 2001 question answering track. In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001)*, 2001.

[11] E. M. Voorhees. Overview of the TREC 2002 question answering track. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, 2002.

[12] E. M. Voorhees. Overview of the TREC 2003 question answering track. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, 2003.

[13] E. M. Voorhees and D. M. Tice. Overview of the TREC-9 question answering track. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, 2000.