

# Answer Passage Retrieval for Question Answering

Andres Corrada-Emmanuel, W. Bruce Croft, Vanessa Murdock

Center for Intelligent Information Retrieval  
Department of Computer Science  
University of Massachusetts Amherst  
Amherst, MA 01003-9264

{corrada, croft, vanessa}@cs.umass.edu

## ABSTRACT

Document or passage retrieval is typically used as the first step in current question answering systems. The accuracy of the answer that is extracted from the passages and the efficiency of the question answering process will depend to some extent on the quality of this initial ranking. We show how language model approaches can be used to improve answer passage ranking. In particular, our results show that information that is often used in answer extraction, such as question type and Wordnet synonyms, can be incorporated into the retrieval step, and can have a significant impact on effectiveness.

## Keywords

Question Answering Systems, Relevance Models, Answer Language Models, Answer Passage Retrieval

## 1. INTRODUCTION

Current question answering systems typically include an initial document (or passage) retrieval step to simplify the task of identifying good answer passages [1,2]. The answer passages are then analyzed by a variety of techniques to extract the final answers. The accuracy of the final answer will depend to some extent on the quality of the passages retrieved. This will be especially true in environments where computational resources are bounded (i.e. real systems rather than laboratory experiments) and the number of passages that will be examined is limited. If the answer passage retrieval process can be relied on to deliver high-quality results, then the question answering system will only need to process a small number of top-ranked passages.

In this paper, we show how the language model approaches used recently for document retrieval can be applied to answer passage retrieval. We discuss both the query-likelihood and relevance model approaches [3,4,5] and evaluate their performance on the task of finding 250 byte answer passages specified in the TREC-9 question answering track. [6].

In many question answering systems, a variety of additional features, such as the question type, are used in an *ad-hoc* way to filter out passages unlikely to contain answers. We believe that many of these features could be incorporated directly into the passage retrieval model. To demonstrate this, we investigate two classes of information. The first is an *answer model* corresponding to a particular query type. The answer model specifies likely text patterns that would be found in answers of a given query type. For example, answers to “location” questions will typically contain entities of type *location*. The second class of information used is Wordnet synonyms for query words [7]. We

show how these classes can be incorporated into both the query-likelihood and relevance models and compare the results.

The following section describes the language modeling frameworks that are the basis of the answer passage retrieval algorithms. We also show how answer models can be incorporated. Section 3 discusses the overall system that was used to retrieve passages, including a discussion about different types of passages. Section 4 describes the document, query collections, and evaluation measures used for the experiments. Section 5 presents the results of the experiments. Section 6 gives a brief overview of related work. The final section summarizes the paper and describes future work

## 2. LANGUAGE MODEL FRAMEWORKS

A statistical language model is a probability distribution over all possible sentences (or other linguistic units) in a language [8]. It can also be viewed as a statistical model for generating text. In most applications of language modeling, such as speech recognition and information retrieval, the probability of a sentence is decomposed into a product of *n-gram* probabilities. A bigram model is estimated using information about the co-occurrence of pairs of words, whereas a unigram model uses only estimates of the probabilities of individual words. For applications such as speech recognition or machine translation, word order is important and higher-order (usually trigram) models are used. In information retrieval, the role of word order is less clear and unigram models have been used extensively.

The basic approach for using language models for IR assumes that the user generates a query as text that is representative of the “ideal” document. The task of the system is then to estimate, for each of the documents in the database, which is most likely to be the ideal document. That is, we calculate:

$$\arg \max_d P(d | q) = \arg \max_d P(q | d)P(d) \quad (1)$$

where the prior  $P(d)$  is usually assumed to be uniform and a language model  $P_d(q)$  is estimated for every document. In other words, we estimate a probability distribution over words for each document and calculate the probability that the query is a sample from that distribution. This *query-likelihood* retrieval model was first proposed by Ponte and Croft [3] and described in terms of a “noisy channel” model by Berger and Lafferty [4]. This approach to retrieval, although very simple, has produced results that are at least comparable to the best retrieval techniques previously available.

The classical probabilistic model of retrieval [9] is described in terms of a Bayesian classification of documents into the classes

relevant ( $R$ ) and non-relevant ( $N$ ) for each query. In this case, we rank documents by the ratio  $P(d|R)/P(d|N)$ . A generative approach to this *document-likelihood* model requires that we estimate the language models  $P_R(d)$  and  $P_N(d)$ . This means that we must estimate probabilities for words in the relevant (and non-relevant) classes of documents. Given that the only information available initially about relevance is the query, this is a challenging task. Lavrenko and Croft [5] show that if the *relevance model* is estimated by:

$$P(w|R) \approx P(w|q) \quad (2)$$

where  $w$  is a word, then the estimation process involves a version of query expansion. This results in better effectiveness than the simple query-likelihood model. Lavrenko and Croft introduce two ways of estimating  $P(w/q)$ . In this work, we use

$$P(w, q_1, \dots, q_m) = \sum_D P(d) P(w/d) \prod_{i=1}^m P(q_i/d) \quad (3)$$

It seems reasonable, given these two models, to propose that the documents whose models are most similar to the relevance model should be retrieved. Relative entropy (also known as Kullback-Leibler divergence) is a standard metric for comparing distributions that has worked well in IR experiments. The relative entropy between the relevance model  $R$  and a document model  $D$  is defined as:

$$KL(R \| D) = \sum_w P(w|R) \log \frac{P(w|R)}{P(w|D)} \quad (4)$$

This measure produces consistently better retrieval results than either the query-likelihood or document-likelihood approaches. It also simplifies to the query-likelihood model when maximum-likelihood estimates are used for the query terms.

Both the query-likelihood and relevance model approaches have been used for passage-based retrieval of documents [10]. In this case, document language models are replaced with passage language models. Relevance models can be constructed from either documents or passages. Liu and Croft [10] show that passage retrieval is approximately as effective for retrieving relevant documents, and more robust when searching databases containing very heterogeneous documents.

In this paper, we are developing a passage retrieval model specifically for retrieval of answer passages, not relevant documents. The specific passage type used for this work is described in the next section. The baseline for the query-likelihood model is equation (1) assuming uniform prior probabilities. In other words, we rank passages by

$$P(A|Q) \propto \prod_{i=1}^n P(q_i|A) \quad (5)$$

where  $A$  is an answer passage. As with all language model approaches, the smoothing of probabilities is a major issue. In our experiments, we used interpolation with a collection model and Dirichlet smoothing [11]. The collection probabilities were estimated using the whole TREC-9 collection.

In the case of the relevance model, equation (2) is used with a uniform prior ( $p(d)$ ). The relevance model is built from the top ranked passages rather than documents (in our case, the top 30 passages).

In order to incorporate additional information about an answer model, we used the prior probability in both the query-likelihood and relevance model equations. In other words,  $p(d)$  (or  $p(a)$  in this case) is modified based on the probability of the text given a particular question class. To calculate these probabilities, we constructed answer models from training data for the main TREC question classes. This is described in more detail in section 5.3. A number of approaches were used to incorporate Wordnet synonyms. They are described in section 5.5.

### 3. SYSTEM ARCHITECTURE

The goal of our research is to effectively rank passages directly from the query. Currently, however, the toolkit we are using (Lemur<sup>1</sup>) does not support direct passage retrieval. For these experiments, therefore, we first retrieve documents, then split these documents into passages. The passages are then ranked using language model techniques.

Passages were created using the following procedure. The top 20 retrieved documents were selected (early tests showed that increasing this number had no effect in system performance). These selected documents were split into sentences using a heuristic sentence segmenter.

The sentences were sequentially formed into passages that were at most 250 bytes in length and possibly overlapping with neighboring passages. If a sentence was longer than 250 bytes, as did occur, it was dropped. We call the passages produced by this procedure “sentenced” passages. Our procedure yielded an average of 434 passages per question that needed to be ranked.

Alternatively, we could have “windowed” the passages without respecting sentence boundaries. We have studied this alternative and have found system performance increased. However, we were interested in developing a system that could be seen as a pre-processor for an NLP system that needed grammatical sentences while at the same time respecting the 250-byte limit so our performance could be compared to that of other systems that have performed the same task.

These sentenced passages were subsequently indexed with no stemming and a stop list consisting of single characters only. We then performed retrieval on these passages applying the techniques we described above. Our retrieval for each question was limited to only those passages that came from its retrieved documents. Doing otherwise would be contrary to our whole approach: start with retrieved documents for a question and then identify passages within those documents that are likely to contain its answer.

## 4. EXPERIMENTAL SETUP

### 4.1 Dataset

The TREC-9 QA Dataset was used for our experiments. It consists of 979,000 documents with about 3GB of text from various news sources (AP newswire, Wall Street Journal, San Jose Mercury News, Financial Times, LA Times, and FBIS) [4].

The questions set for TREC-9 consists of 693 questions that are generally characterized as being of the “factual” type: “What is the longest river in the world?”, “Who is Colin Powell?”, etc. It

<sup>1</sup> [www.cs.cmu.edu/~lemur/](http://www.cs.cmu.edu/~lemur/)

was found during the TREC-9 evaluation that some questions were too ambiguous or did not have an answer within the corpus [4]. Thus, 682 questions were finally used to report evaluation metrics for the TREC-9 QA tasks, and it is this restricted set that we use in the experiments reported here.

The TREC-9 questions have been classified into “question types” by Thomas S. Morton of the University of Pennsylvania (private email communication). We used six of his question type classifications:

- Amount (A): “How many zip codes are there in the U.S.?”
- Famous (F): “What is Francis Scott Key best known for?”
- Location (L): “Where is Glasgow?”
- People (P): “Who invented basketball?”
- Time Point (T): “What year did Montana become a state?”
- Other(X): This is a generic, catch-all for questions that did not fit the other types

There are various reasons for using these question types. Many QA systems have a question classification component. Thus it was reasonable to investigate if our statistical approach benefits from knowing a question’s type. In addition, we were interested in seeing if training data improves the performance of a statistical approach. The A questions are the smallest set in our group with 52 questions. We took this as the practical limit for testing approaches that divided the questions into training and testing portions. And finally, this AFLPTX set contains 589 out of the 682 questions evaluated in TREC-9.

So the results we present here are for each individual question in the AFLPTX set, the AFLPTX set combined and all of the TREC-9 evaluated questions. The individual question type results allows us to see if performance varies across questions types. The combined result would be the performance of a system that had perfect question classification performance. And finally the results for all questions allow an approximate comparison between our work and previously reported results for the TREC-9 evaluation.

## 4.2 Task and performance metric

The TREC-9 QA track had two experimental conditions: returning answer strings limited to 50 bytes, and answer strings limited to 250 bytes. The results of the TREC-9 evaluation make it clear that performance correlated with answer string length – longer strings improved a system’s performance. All experiments reported were done for the 250-byte task [4].

The performance metric for TREC-9 was the Mean Reciprocal Rank (MRR) measured down to the top 5 answer strings returned for each question. So systems were rewarded for returning a correct answer within the top 5 ranked answer strings, but received no credit for answers returned below the 5<sup>th</sup> position.

In lieu of using human judges to decide if retrieved answer passages did answer the questions, we utilized the regular expressions developed by NIST in an attempt to develop a “reusable test collection”. These regular expressions were constructed so that “almost all” strings judged to be correct by the original judges of the TREC-9 evaluation would be matched [4]. The Kendall  $\tau$  association between system performance measured

by human judges and that measured using regular pattern matching was found to be 0.94 for the systems that participated in the TREC-9 250-byte task [4].

## 5. EXPERIMENTAL RESULTS

### 5.1 Query Likelihood Baseline

The query likelihood baseline was described in section 2. It returns a ranked list of passages using equation (3).

After examining the initial results, we found that just using the query likelihood ranking may not be the most effective approach. Passages at the top of a ranked list may be overlapping and come from the same document. To test the effect this has on the results, we tested two strategies for removing passages from the same document. The results from the original ranking are shown in the first column of Table 1 under the heading “Unvetted”. The second column, “one per document”, only allows the top ranked passage from a given document to remain in the ranked list. The second strategy, “non-overlapping”, allows multiple passages from the same document by removing passages that overlap with a higher ranked passage. It is clear from these results that vetting the list returned by the query likelihood ranking produces a small improvement for every question class, although there is not much difference between the two strategies for removing passages.

**Table 1. Query Likelihood MRR performance (%)**

Question Set	Unvetted	One per Document	Non-overlapping
A (52)	21.2	22.6	21.9
F (84)	56.4	57.6	58.0
L (109)	43.6	44.1	44.8
P (109)	53.4	55.4	55.4
T (71)	27.3	27.7	27.6
X (164)	34.9	36.4	36.3
AFLPTX (589)	40.9	42.1	42.2
All(682)	38.2	39.4	39.7

### 5.2 Relevance Model Results

The relevance model baseline was done using equation (2). As mentioned before, the relevance models are built using the top-ranked passages from an initial retrieval. As one can see from the baseline results for query likelihood, however, the best performance is obtained when we vet the ranked list by requiring that only non-overlapping sentenced passages be allowed in the list. This implies that the relevance model results could also be improved by vetting. Table 2 shows that this is the case for all question types.

In general, the relevance model was more effective than the query-likelihood model, sometimes substantially. This is a little surprising given that relevance models are equivalent to a massive query expansion [5], and answer passage retrieval is often done in other systems using strict matching criteria. Two of the question classes, People and Time, did not perform better with relevance models (although the results were not worse either). This suggests

that the effectiveness of query expansion may depend on the type of question.

**Table 2. QA Relevance Models MRR performance (%)**

Question Set	Unvetted	One per Document	Non-overlapping
A (52)	21.2	23.4	23.4
F (84)	65.2	66.8	66.6
L (109)	49.0	50.1	50.3
P (109)	53.4	55.4	55.4
T (71)	25.8	26.8	27.4
X (164)	37.2	38.2	39.2
AFLPTX (589)	43.6	45.0	45.3
All (682)	39.2	41.1	41.4

It is interesting to note that the relevance models always had their best performance at a smaller value for the Dirichlet constant than the query likelihood results. This is encouraging since it means that the probability estimates derived from the passages were weighted more than in the query likelihood baseline. That is, relevance models depend less on the use of smoothing. In addition, whereas the number of words used to estimate the relevance models for document retrieval is in the order of hundreds, for this task the best value was obtained when the words used to estimate the models were in the order of ten words.

### 5.3 Bigram Answer Model Results

Both of the results reported so far, query likelihood and relevance models have used the assumption that the prior probability of an answer passage is constant. We wanted to investigate how both methods would be affected by the use of probability priors for the answer passages. We investigated this approach by looking at obtaining passage priors from bigram language models for correct answer passages.

For example, one would expect that location questions (L type) would invariably have a location within the correct answers. This suggests that answer models trained on data where various entities are abstracted from the text could lead to improvements in the query likelihood baseline.

We tested this hypothesis by limiting ourselves to the 6 question types that had more than 50 questions (A, F, L, P, T, X).

We began by developing a “global” vocabulary file as follows. The text for all candidate answer passages for the AFLPTX questions were processed using BBN’s IdentiFinder [12] to tag for the following entities:

- o PERSON
- o PERCENT
- o DATE
- o MONEY
- o LOCATION
- o ORGANIZATION

These tags were used to replace the tagged text itself. Thus, for example, eliminating most mentions (the tagger is not perfect) of specific locations to a single token: `_location_`. This procedure yielded a vocabulary of about 53K words for our AFLPTX candidate answer passages versus a vocabulary of about 94K words if no tagging with replacement had occurred.

We then proceeded to train answer models specific to each question type with the following procedure. The question set for each question type was divided into ten random partitions assigning 90% of the questions to a training set and 10% to a testing set.

Each random partition of a question type now had 90% of the questions to be used to train an answer model. We then trained bigram models using the known correct answer text to the training questions. This training correct answer text also was processed by IdentiFinder with tags similarly replaced as in our procedure to create a global vocabulary file. We used absolute discounting to deal with unseen words and count one words since our training data was so small and it is generally assumed that absolute discounting works better than other schemes such as Turing smoothing for small amounts of training text.

We then tested the use of these models by ranking answer passages using equation (1) and estimating  $p(a)$  using the answer model.

Since different answer passages contain different amounts of words and we want to use a probability that is not biased in favor of shorter passages, we used the inverse of the candidate answer passage perplexity under the bigram answer model to estimate our prior probabilities for the testing answer passages.

As can be seen by the results in Table 3, answer model priors always improved the baseline, sometimes markedly.

**Table 3. Bigram answer models MRR performance (%)**

Question Set	Unvetted	One per Document	Non-overlapping
A (52)	21.6	22.2	22.6
F (84)	62.9	64.0	64.7
L (109)	46.9	48.9	49.3
P (109)	58.3	59.3	59.6
T (71)	31.9	32.4	32.6
X (164)	53.2	54.3	54.6
AFLPTX (589)	47.3	48.4	48.7

### 5.4 Answer Models with Relevance Models

Since bigram answer models improved the baseline query-likelihood performance, we also tested how the use of the model priors would improve the relevance model performance. This was done using the prior in equation (2).

The results are shown in Table 4. The performance is the best of all the models we have tried. Both query expansion and the incorporation of answer models improve the effectiveness of the answer passage ranking.

**Table 4. RMs with bigram AM priors MRR performance (%)**

Question Set	Unvetted	One per Document	Non-overlapping
A (52)	22.1	24.7	24.7
F (84)	64.7	67.4	67.2
L (109)	52.4	54.7	55.4
P (109)	58.3	59.5	60.0
T (71)	31.9	32.4	32.6
X (164)	53.8	55.7	55.6
AFLPTX (589)	50.5	52.3	52.5

## 5.5 Incorporating Wordnet Synonyms

Our motivation in exploring the use of WordNet as a resource in Question Answering was in part to tease out where question expansion might be most beneficial. It seems reasonable that using synonyms in a probabilistic framework would help overcome the polysemy barrier. It is also possible that the use of synonyms would be more effective in question answering than in document retrieval, because questions have a more restricted use of multiple terms for the same concept (as compared to keyword queries, for instance). We investigated using WordNet [18] synonyms to expand the question, both as a pre-processing device, and as part of a translation model. All experiments were done on AFLPTX questions, without stemming. The WordNet synonym groups were constructed from the synonyms and hypernyms of the terms in the question, for all terms except those that were in a list of generic stop words. The stop words list was chosen randomly from a set of stop words designed for other unrelated tasks, and is representative of a general-purpose stop word list one might use for any number of other tasks, and was not specifically related to the TREC-9 data. Stop words were not removed from the questions, the just were not expanded with synonyms.

The following two subsections describe two sets of experiments: the first uses question pre-processing to insert expansion terms, the second uses a translation model commonly used for cross-lingual information retrieval.

### 5.5.1 Question Expansion by Pre-Processing

Voorhees demonstrated in [15] that poor quality synonyms degrade performance, and good quality synonyms offer small benefit. With this in mind, we created two sets of synonyms. The first expands terms with noun hypernyms, and if there are none, chooses verb hypernyms. Words that have no noun or verb hypernyms are not expanded.

The second synonym group expands each word with the non-overlapping union of synonyms (the `syns[ar|v|n]` operator in WordNet) and hypernyms, for each word as a noun, verb, adjective or adverb. This second set was almost twice as large as the first, with an average of 10.5 synonyms per term, compared to 5.8 synonyms per term in the smaller set.

**Table 5. Question expansion with WordNet MRR (%)**

	Small Synonym	Large Synonym
A	14.6	14.6
F	55.0	55.8
L	42.8	43.9
P	50.1	50.0
T	24.8	25.8
X	33.2	35.4
AFLPTX	38.6	39.6

	Set	Set
A	14.6	14.6
F	55.0	55.8
L	42.8	43.9
P	50.1	50.0
T	24.8	25.8
X	33.2	35.4
AFLPTX	38.6	39.6

Table 5 shows the comparison of the average MRR scores for question types A,F,L,P,T, and X with question expansion with the smaller synonym group and the larger. Our results indicate that poor quality synonyms hurt performance, and better quality synonyms did not offer a significant improvement.

### 5.5.2 Using Translation Models

We used a framework for cross-lingual information retrieval used in Xu et al, [16] and Federico and Bertoldi [17], to construct a translation model to “translate” a term to its set of synonyms. In cross-lingual information retrieval, the problem is to find documents in a target language that are relevant to a question in a source language:

$$P(Q_S | D_T) \propto \prod_{w_s \in Q_s} P(w_s | D_t) \quad (6)$$

where  $w_s$  is a source-language term in the question  $Q_s$ , and  $D_t$  is a document in the target language. We estimated this by introducing a latent variable  $t$  that represents the “translation” of a term  $w$ . Thus,

$$P(w | D) \propto \sum_{t \in T} P(w | t) P(t | D) \quad (7)$$

where the synonym group  $T$  for a word is the set of translations of the original word, including the original word itself.

To account for the fact that some translations are better than others, we introduced a tuning parameter  $\alpha$  (which takes values from 0 to 1):

$$\begin{aligned} P(w | D) &\propto \sum_{t \in T} P(w | t) P(t | D) \\ &= P(w | w) P(w | D) + \sum_{t \neq w} P(w | t) P(t | D) \\ &= (P(w | w) + (1 - \alpha)(1 - P(w | w))) P(w | D) \\ &\quad + (1 - \alpha) \sum_{t \neq w} P(w | t) P(t | D) \end{aligned} \quad (8)$$

We estimated the translation probabilities  $P(w|t)$  in four ways. We assume that  $P(w | t) \approx P(t | w)$ , that is, that without actual statistics on word frequency, we can approximate the true translation probabilities with methods that assume that the probability of a word given a translation is equivalent to the

probability of a translation given a word. In cross-lingual information retrieval, very simple methods of estimation are successful. We used the following methods to estimate the translation probabilities.

### 5.5.2.1 Translation probabilities are constant

We set  $P(w/t) = 1$ . This means that the probability of a word given a document is the sum of the probabilities of its synonyms given the document. In this estimation, we did not normalize the probabilities to sum to one. This simple method performed well, over all, with an overall gain of 1.3 average MRR score. (Method 1 in Table 6)

### 5.5.2.2 Translation probabilities are equal

We used two different formulations of this idea. In the first, we let the translation probabilities equal one over the number of synonyms that appear in the document. Thus the translation probabilities are different for every document, for every synonym, and are not constant across documents. This method of estimating the translation probabilities yields an improvement of 0.3 over all. (Method 2 in Table 6)

In the second, the translation probabilities are estimated to be one over the total number of synonyms, whether they exist in the document or not. The translation probabilities are constant across documents because the probability estimation does not depend on the occurrence of the term in the document. This method of estimation performed well, with an over all gain in average MRR score of 1.4, with the biggest gain for “T” questions, of 4.4 in the average MRR score. (Method 3 in Table 6)

### 5.5.2.3 Translation probabilities are estimated from the corpus

Intuitively, an indication of how likely a synonym is, is its frequency in the corpus. Most closed-class words (such as “the” and “where”) don’t have synonyms in WordNet, so the frequency of a term in the corpus only contributes to content words. This method of estimating the translation probabilities led to a significant drop in performance, of 3.4 over all. (Method 4 in Table 6)

**Table 6. Translation Models MRR performance (%)**

	<b>Method 1</b>	<b>Method 2</b>	<b>Method 3</b>	<b>Method 4</b>
A	17.1	20.2	17.0	17.1
F	61.0	55.9	63.0	55.9
L	44.8	44.8	44.8	46.2
P	54.0	53.9	54.2	53.3
T	31.2	28.1	31.2	27.6
X	34.4	33.8	33.7	21.1
AFLPTX	41.9	40.8	42.0	37.1

### 5.5.3 Summary of results

Our experiments indicated that expanding the query as a pre-processing step did not improve our results. Treating the

synonyms as a set of translations, and employing a translation model showed some benefit, even with simple methods of estimating the translation probabilities. The simplest methods of estimating the translation probabilities work well. The frequency of the synonym in the corpus did not provide an adequate estimation of likelihood of synonymy.

## 6. RELATED WORK

The use of passage retrieval for question answering has been studied before. The IR-n system [13] from the University of Alicante has focused on this task. It uses a heuristic measure to retrieve passages with a fixed number of sentences, with best performance obtained at around 20 sentences. Besides our use of language models, this work differs from ours because we only considered passages that conformed to the 250-byte TREC-9 QA task.

The University of Sheffield [14] has also used passage retrieval in their QA system. But as in the IR-n system, performs the passage retrieval with a heuristic measure and retrieves passages longer than the 250-byte limit.

WordNet [18] has been used in a number of QA systems. In Prager et al [7], WordNet hypernyms, which represent IS-A relationships, are used to answer definition questions. Hovy, et al [19] use WordNet synonyms to rescore retrieved sentences according to their similarity to the question, giving a certain portion of the score for exact matches of words appearing in the sentence that are synonyms of words appearing in the question. Use of WordNet noun hypernyms for query expansion in document retrieval in Voorhees [15] is found to degrade performance over all, even when the synonyms are chosen by hand.

## 7. CONCLUSIONS AND FUTURE WORK

The results shown in this paper demonstrate that it is possible to significantly improve answer passage ranking by incorporating additional features related to passage quality into the retrieval model. Table 7 summarizes the percentage improvements of different approaches compared to the baseline query likelihood model. This shows that query expansion via the relevance model generally improves performance, but the incorporation of answer models is better for some question classes. The incorporation of answer models into the priors for the relevance model produces the best results with consistent and substantial improvements in all question classes. The incorporation of Wordnet synonyms does not have a significant effect, but neither does it hurt performance.

The best results in this table are competitive with the best results achieved in TREC-9, but they are somewhat difficult to compare directly because answer models were not constructed for all question classes. It should be emphasized that this high level of performance was achieved in what is effectively a single retrieval pass with no additional *ad-hoc* filtering done afterwards, as is common with most other systems.

In future work, we plan to investigate other features that could be incorporated into the retrieval model. To do this, we may use maximum entropy language models, which are more flexible with

regard to the types of features in the model [8]. We also will test different forms of answer models. We have recently used mining algorithms based on Web data to build new models, similar to the work of Ravichandran and Hovy [14], and have encouraging preliminary results.

**Table 7. Relative percentage improvement over baseline**

Question Set	RMs	AMs	RMs + AMs
A (52)	6.8	3.2	12.8
F (84)	14.8	11.6	15.9
L (109)	12.3	10.0	23.7
P (109)	0.0	7.6	8.3
T (71)	-1.0	18.1	18.1
X (164)	8.0	50.4	53.2
AFLPTX (589)	7.3	15.4	24.4

## 8. ACKNOWLEDGEMENTS

This work was supported in part by the Center for Intelligent Information Retrieval, in part by SPAWARSYSCEN-SD grant numbers N66001-99-1-8912 and N66001-02-1-8903, and in part by Advanced Research and Development Activity under contract number MDA904-01-C-0984. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

## 9. REFERENCES

- [1] Ittycheriah et al. IBM's Statistical Question Answering System. In Proceedings of the Ninth Text Retrieval Conference (TREC-9), 2000.
- [2] Elworthy, D. Question Answering using a large NLP System. In Proceedings of the Ninth Text Retrieval Conference (TREC-9), 2000.
- [3] J. Ponte, W.B. Croft, "A language modeling approach to information retrieval", *Proceedings of ACM SIGIR 1998*, 275-281, 1998.
- [4] A. Berger, J. Lafferty, "Information Retrieval as statistical translation", *Proceedings of ACM SIGIR 1999*, 222-229, 1999
- [5] V. Lavrenko, W.B. Croft, "Relevance-based language models", *Proceedings of ACM SIGIR 2001*, 120-127, 2001
- [6] Voorhees, E. Overview of the TREC-9 Question Answering Track. In Proceedings of the Ninth Text Retrieval Conference (TREC-9), 2000.
- [7] Prager et al. Use of WordNet Hypernyms for Answering What-Is Questions. In Proceedings of the Tenth Text Retrieval Conference (TREC-10), 2001.
- [8] R. Rosenfeld, "Two decades of statistical language modeling: where do we go from here?", *Proceedings of the IEEE*, 88(8), 1270-1288, 2000.
- [9] K. Sparck Jones, S. Walker, S E. Robertson, "A probabilistic model of information retrieval: development and comparative experiments", Parts 1 and 2, *Information Processing and Management*, 36(6): 779-840, 2000.
- [10] Liu, X. and Croft, W.B., "Passage Retrieval Based On Language Models," *Proceedings of CIKM '02 conference*, 375-382, 2002.
- [11] C. Zhai, J. Lafferty, "A study of smoothing methods for language models applied to ad hoc information retrieval", *Proceedings of ACM SIGIR 2001*, 334-342, 2001.
- [12] Bikel, Daniel M., Schwartz, Richard L., and Weischedel, Ralph M. An Algorithm that Learns What's In a Name. *Machine Learning* vol. 34, p. 211-231, 1999.
- [13] Llopis, Fernando, Vicedo, Jose Luis, and Fernandez, Antonio. Passage Selection to Improve Question Answering. In Workshop on Multilingual Summarization and Question Answering, 2002.
- [14] Ravichandran, D. and E.H. Hovy. 2002. Learning Surface Text Patterns for a Question Answering System. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002) Conference*. Philadelphia, PA, July 7-12.
- [15] E. M. Voorhees. "On Expanding Query Vectors with Lexically Related Words." In *TextREtrieval Conference*, pages 223-232, 1993.
- [16] J. Xu, R. Weischedel, and C. Nguyen. "Evaluating a Probabilistic Model for Cross-lingual Information Retrieval." In *Proceedings of the 24<sup>th</sup> Annual International ACM SIGIR Conference*, 2001.
- [17] M. Federico and N. Bertoldi. "Statistical Cross-language Information Retrieval Using N-Best Query Translations." In *Proceedings of the 25<sup>th</sup> Annual International ACM SIGIR Conference*, 2002.
- [18] G. A. Miller. "WordNet: A Lexical Database." *Communications of the ACM*, 38(11):39-41, 1995
- [19] E. Hovy, U. Hermjakob, C.-Y. Lin and D. Ravichandran. Using Knowledge to Facilitate Factoid Answer Pinpointing." In *Proceedings of the Conference on Computational Linguistics (COLING)*, Taipei, Taiwan, 2002.