Providing A Simple Question Answering System By Mapping Questions to Questions.

Tait Larson
Johnson(Heng) Gong
Josh Daniel

Abstract

Most current QA systems attempt to find answers to input questions by cleverly selecting portions of related documents. In this paper, we discuss an alternative approach for taking advantage of the large amount of pre-answered questions available on the web and finding a similar question that has already been answered. Our approach involves query transformation as breadth-first search with expansion by Wordnet and pruning by language model, in order to transform the question into the language of the question corpus. We use a combination of NLP and IR techniques to return the most relevant pre-answered question. We discuss experimental results comparing our system to other QA systems.

Introduction

In the last several years, question answering systems have become very popular, as evidenced by the TREC QA competition. Standard search requires users to enter keywords to guide a search engine, but an interface that allows users to ask questions directly is much more intuitive for many users. Question answering systems are especially useful in that they allow a user to enter a question, and the system attempts to use the added context in the question to provide a better answer than simply extracting keywords for standard document search.

Most current QA systems attempt to retrieve an answer from a set of documents, or generate an answer from a data source. We propose a slightly different approach. The internet contains vast amounts of knowledge including questions that have already been answered, along with those answers.

This data exists in FAQs, Yahoo answers, Google answers, lawguru.com, and many other resources. Rather than answer the question ourselves, our QA system uses these already answered questions as the answers to an input question. Thus, our system attempts to find the most similar question to that of the user, and then hopefully the stored answer is also a good answer to the input question.

We return the best several questions if they appear to be relevant, and let the user choose which answer he wants to see. If there are no relevant questions, we indicate that we do not have a stored answer, and a complete application could revert to standard information retrieval techniques. Our goal is a system that produces results with better precision than Yahoo answers, but still has reasonable performance.

Algorithms/Method

Overall Description

Given a set of questions and answers as a knowledge base, we take as input a user's question phrased as one to several English sentences. We then attempt to map the user's question to the questions and answers in our repository. Our system returns a set of ranked relevant question and answer pairs to the user.

Note that this is distinctly different from traditional Information Retrieval in that we are attempting to map input questions to stored questions (that we already have answers to), as opposed to mapping input questions directly to documents.

The core of our approach is an attempt to transform input questions into the language of the question corpus, and then find the closest matching repository question. The transformation is done by a breadth-first search via rewriting of words in the input question according to Wordnet synonyms, and pruning the search tree by removing low-probability rewritten sentences according to a language model trained on the question corpus. Once the input question has been rewritten into language close to that of the question corpus, we run tf-idf information retrieval algorithms with only bigrams to find the closest questions. If the final scores are too low, we determine that there is no matching question/answer pair in the repository. A diagram of the process is below.

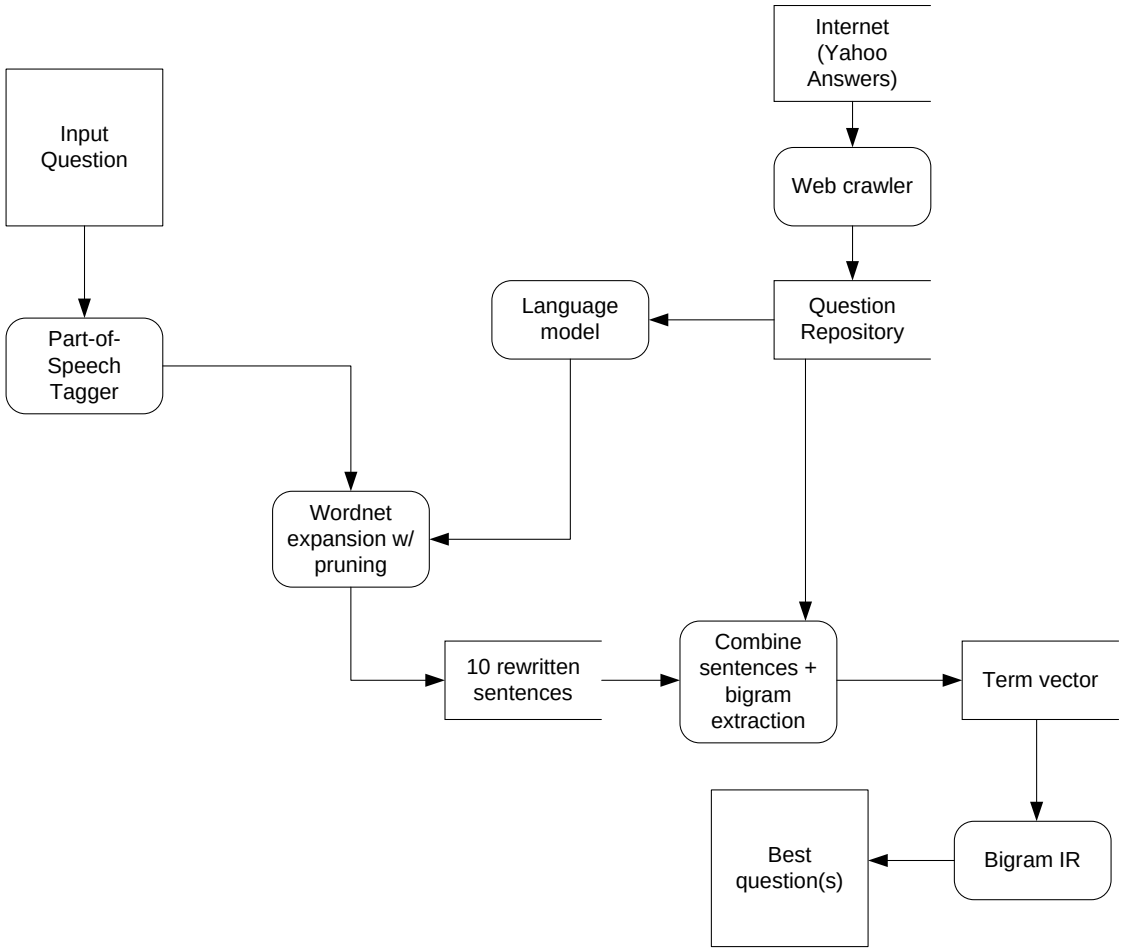

Figure 1: Overall data flow for our Question->Question system.

Data

We chose to use a subset of Yahoo questions as our question repository. This data was easily available to download with a web crawler, and contains a very large number of pre-answered questions. We obtained around 100,000 question/answer pairs from the health section of Yahoo questions, but there are many other categories as well. We used the health questions exclusively in the IR component of our process, but added about 50,000 question/answer pairs from related sections (like pregnancy/parenting, etc.) for training the language model. We chose health because we initially wanted to limit our domain in order to apply domain-specific knowledge from some data source (expert system or semantic web, etc.) Our final project did not actually use a semantic web, so our system should generalize to other questions as well.

Question Rephrasing

Question rephrasing consists of several parts, which are described in the following subsections.

POS tagging

As a preprocessing step, we tag each word in the input question with its part of speech. This is useful for limiting Wordnet to only expand word senses for the correct part of speech. We use the Stanford Log-Linear Part-Of-Speech Tagger for this step.

Language Model

In order to transform the input question into language similar to that of the question corpus, we use a language model to measure how likely a given sentence is in the question corpus. We chose to use our Good-Turing bigram language model with backoff from assignment 1 because it performs well and was easy to implement as we already had working code. We trained this model on the question corpus so that it emphasizes sentences which are likely to occur in the question corpus. We believe there to be enough data in the 140,000 repository questions that a unigram model would be insufficient, so we started with a bigram model, with the option to switch to a trigram model if memory and time permit.

Rewriting words with Wordnet

Each word has been tagged with its part of speech, and Wordnet is used to generate a list of related words to try during query rephrasing. Given a word and its part of speech, Wordnet typically contains several senses of the word (around 3 or 4), and each sense typically contains around 5-10 synonyms. We chose to generate words for nouns, verbs, and adjectives because they are usually the most important content words in each sentence, and expanding other words would significantly increase the computational time required for question rephrasing. Also, we chose to generate all synonyms of all senses for each word (but only for the proper part of speech). In our implementation, we used JWord, a Java version of Wordnet, for word rewriting.

Rephrasing the question

In order to rephrase the question into language similar to that of the question corpus, we perform a breadth-first search, guided by the language model. The search tree begins with the input question, and Wordnet is used to rewrite each important lexical word. We start by generating all sentences with rewrites of the first lexical word, and adding those sentences to the search tree. If necessary, we prune the search tree down to the 100 sentences syntactically closest to the question corpus according to the language model. We then iteratively move to the next lexical word, rewrite it according to Wordnet synonyms, generate all rewrites of the sentences at the leaves of our search tree, and prune the search tree via the language model. Pseudocode for this process follows:

```
Function RephraseQuestion (inputQuestion)
  Tree searchTree(inputQuestion)
  For (position = 0 to inputQuestion.numWords) {
    String[] rewrites = currentQuestion.getNthWord(position).GetRelatedWords()
```

```
   For (each leafNode of the searchTree) {
     currentQuestion = leafNode.getQuestion
     for (each rewrite in rewrites) {
       newQuestion = currentQuestion.replaceNthWord(position, rewrite)
       leafNode.addChild(newQuestion)
     }
   }
   Prune all leaf nodes except top 100
  }
  Return searchTree.getTop10Leaves()
}
```

Pruning with the language model

Pruning is necessary because of the high branching factor. A typical question consists of about 10 words of which around 6 may be content words, and each content word may generate about 10 synonyms. Given these numbers, generating all sentences with all rewritten content words would result in 10^6 sentences, which is too costly in terms of computational time and memory. In addition, there are many sentences longer than this average, for which the problem is even greater. Furthermore, many of these sentences look nothing like proper English or any questions in the question corpus, so they will not contribute much value to the question matching procedure.

To deal with this challenge, at each step of the search process, we prune the search tree down to the 100 most likely sentences according to the language model. This means that on each iteration of sentence rewriting (each iteration corresponds to rewriting one word for each leaf sentence with about 10 synonyms for that word), we typically start with 100 sentences, grow to about 1000 sentences, and then prune down to 100 sentences again. At the end of this process, we want to end up with the 10 best rewritten sentences, so we hope that keeping 100 sentences after each iteration does not prune away too many potentially good sentences. Our intuition is that keeping around 10 times as many sentences as we actually want to end up with is reasonable, as the branching factor for each iteration is also 10, and we have a margin for error of one order of magnitude. An example portion of the search tree is shown below in figure 2.

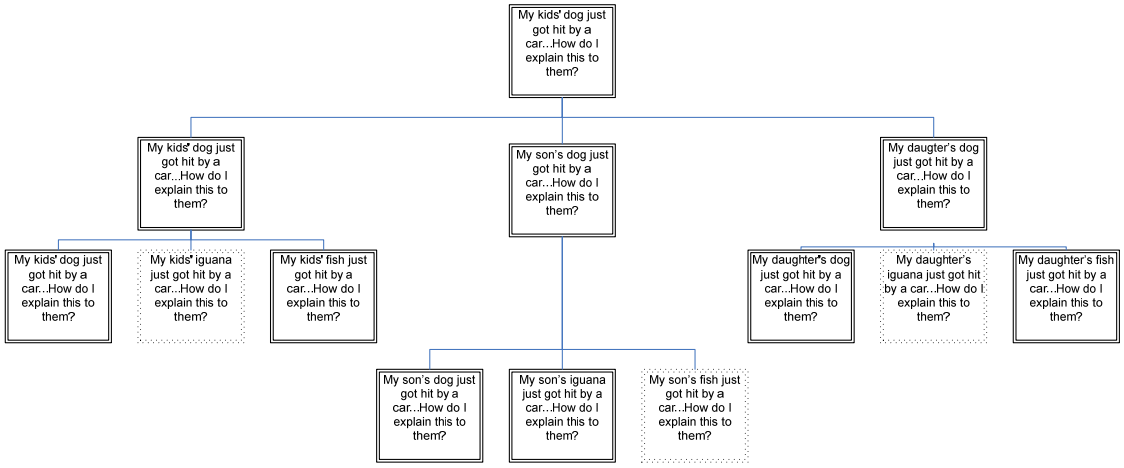

Figure 2: Sample branching of question rephrasing search process. Note that nodes with dotted borders will be pruned off due to low language model probabilities.

Ensuring relevant terms are expanded

The language model works as a very weak yet computationally efficient word sense disambiguator. Given a set of synonyms of a word, the language model does an extremely effective job of determining whether or not each of these synonyms appropriately fits into a candidate sentence. Unfortunately, one drawback of using the language model to prune our search space is that very common words get expanded. Consider the following example:

"Were Einstein and Godel very good friends?"

The most common bigrams in this sentence are "very good" and "good friends". Such terms and their synonyms will dominate the score given by the language model. Synonyms for "very", "good", and "friends" will populate the majority of expanded sentences. Yet "Einstein" and "Godel" are the terms most relevant to the question. In order to ensure our model expands the most relevant terms, we delineate our sentences into a number of positions. Each position stores one word in the initial sentence and contains an associated weight. Probabilities, as determined by the language model, are multiplied by their corresponding position weights when determining the full sentence probability. In the rare case where a term is expanded to two words, only one of the two words is used to determine the probability of that position; therefore, all positions of all expanded sentences contain the same part of speech. By default, each position was given a weight of one.

Effectively any method could be used to determine the most relevant terms. We simply assume all nouns (except pronouns) are most relevant and give higher weights to positions containing nouns.

Information Retrieval

Once we have chosen the 10 sentences most similar to the question corpus, we extract all bigrams from those sentences and create a term vector with those bigrams. Much like tf-idf information retrieval, we then find the cosine between the generated term vector and the term vector for each repository question. The repository questions with the highest scores are returned as potential matches. If the scores are too low for all repository questions (according to a hand-tuned threshold), we then determine that our repository does not contain a question that is a good match. Information retrieval was implemented using Lucene.

Bigrams vs. unigrams

Note that typical tf-idf IR uses only unigrams for its term vectors, whereas we use only bigrams. We use bigrams because the order of words in a question is more important than in typical search/IR. For example, bigrams typically capture the context of the question (e.g. the subject precedes a verb, and the object follows a verb), and the short length of repository questions allows us to be more context-sensitive than if we were looking at the full contents of an answer document.

Many IR solutions choose not to incorporate bigrams in their index. Indexing all bigrams leads to a blow up in the index size because of the number of unique bigrams and their associated entries. We can get away with this because we're indexing questions much smaller than documents.

By combining sentences, we have more bigrams to use for information retrieval. As each of these bigrams was generated by synonyms of input words, and these bigrams are likely to occur in the question corpus (because of the language model), these bigrams should be a good intermediate representation of the question that is suitable for tf-idf scoring, and should help precision. Note that we exclude unigrams because they cause noise in the data due to multiple word senses, and there should (hopefully) be enough bigrams to cover most phrases containing the constituent unigrams.

Stop words and stemming

Note that we do not use any stop words (unlike typical IR), because those pieces of our bigrams are useful for distinguishing parts of speech or word senses, while they do not provide much value in unigram IR. For example, "the fires" refers to burning objects, while "she fires" refers to an act of terminating someone's employment or using a gun. We also chose not to stem, because is unclear how to stem bigrams, and stemming the component unigrams would lead to a loss of contextual information.

Term frequency, and inverse document frequency

In addition, we determined that idf was definitely important for our modified IR as it emphasizes the importance of bigrams that occur infrequently in the question corpus (and therefore contain more discriminatory power). For example, common phrases like "why

does" occur extremely often and we don't want to place too much weight on those phrases.

On the other hand, there was some internal debate as to the importance of using term frequency values. Tf values may be useful as they emphasize bigrams that occur frequently in a particular question. For example, "My cat is not potty trained, how can I potty train my cat?" contains "my cat" twice and "potty train" once or twice depending on stemming, and the question is clearly about those bigrams that occur twice.

Results

Testing methodology

Our testing regimen included several instances of our program, and Yahoo's question search as a benchmark. Our question repository contained approximately 100,000 questions in health and related topic areas, and 50 reference questions. Similarly to older TREC competitions, we measured the Mean Reciprocal Rank of the first relevant repository question for each system. The rank is the position of the first relevant result (limited to the top 5 results), so the reciprocal rank is 1/4 if the fourth result is the first relevant result.

The general population is constantly asking new questions, and is the best source of representative questions. Our 50 reference questions were chosen randomly from among the most recently asked and unresolved questions on Yahoo health to obtain a diverse, representative sample. Our primary goal was to outperform Yahoo question search, but we also mention results from other systems. Relevancy was manually determined by an unbiased judge who was unaware of which questions were from our systems and from Yahoo. Note that our judge actually looked at the answers linked to the result questions, and determined whether the answer was relevant to the input question. This is important because it demonstrates the effectiveness of our system at obtaining good answers even though the process only looks at question text. Our 50 reference questions can be found in appendix A.

Results

On our 50 reference questions, Yahoo question search scored an MRR of 0.1466. Our baseline system scored an MRR of 0.2382. This is a significant improvement, and demonstrates the success of our system. Further improvements on our system yielded even better results, as can be seen in the following graph.

Version 2 contained a modification to idf calculations in the IR phase. After our initial run we noticed that sentences containing many common bigrams would match other sentences containing the same structure ignoring important uncommon bigrams. For instance sentences like "Is there any natural way of getting rid of hayfever?" matched questions in our corpus containing the substring "Is there any natural way of getting rid of". This occurs because the idf weight of "of hayfever" though large cannot compensate

for the combined weights of the common bigrams. To compensate for this we modified the comon idf score of idf=log(N/d) to idf = log(N/d) + square_root(N/d). This allowed uncommon bigrams to more frequently dominate question scores.

Version 3 boosts the IR score of candidate bigrams containing a noun. We found that some relatively uncommon yet useless bigrams such as "a reasonable" dominated question similarity scores. Adding weight to bigrams with nouns compensates for this effect.

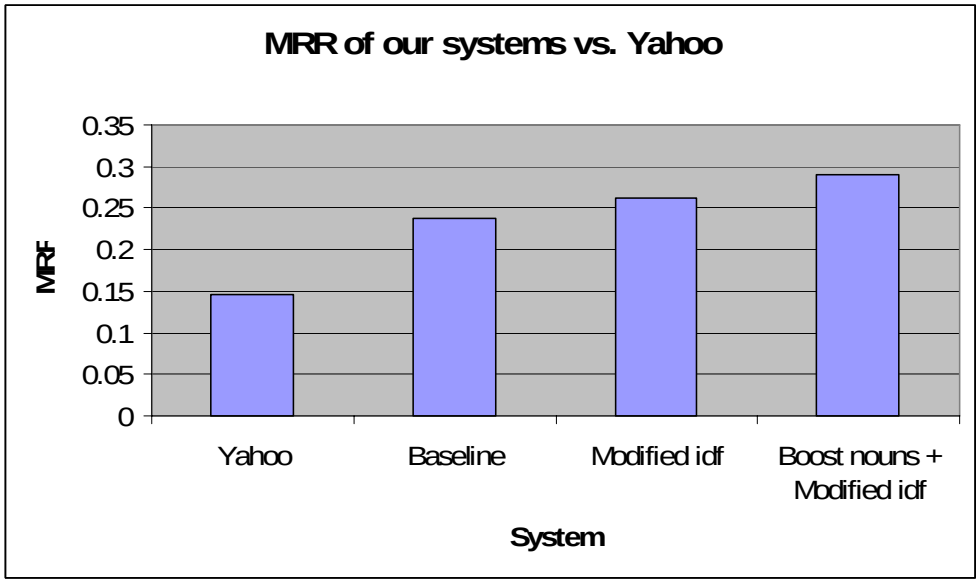

Figure 3: Cumulative Mean Reciprocal Rank results of our systems.

Note that top scores at recent TREC proceedings have been around 0.70, but their problem is significantly easier. Their set of documents is well defined, the questions are much less ambiguous, and questions are written in proper English. Also, all TREC questions contain answers in their corpus. Our questions are taken from the *unresolved* Yahoo questions with the hope that they have been answered in the past, but many of them have not. Also, most TREC questions are looking for short, factual answers (e.g. "What movies do James Dean appear in?") while Yahoo questions tend to be more open-ended (e.g. "Yeah I'm a teenager who needs help on weight?"), and are therefore more difficult to answer.

We've included several interesting pairs of input questions and the repository questions they return. All these results were in the top 3 results for their input question, and are from our baseline system.

Input: a sharp pain in the center of the chest breastbone area?
Result: keep getting a throbbing pain in the middle of my rib cage . any idea what it could be?

Note that many words are different here, but the question rephrasing translated “sharp” to “throbbing” with Wordnet, and the language model scored it highly, so “throbbing pain” was found in the IR stage. It appears that Wordnet did not actually translate “chest breastbone” to “rib cage”, as several similar questions were also returned with other body parts in that part of the sentence, but there were enough bigram hits to make IR successful. Unigram IR would probably not correctly retrieve this question.

Input: my contact lens ripped is there any way that i can buy like one 1 lens without having to buy a sixpack?
Result: does anyone know where i can order contact lens without having a prescription verified?!?

Note that Wordnet does some of the rewriting very well (e.g. “buy” -> “order”), so there are a few key bigrams used in the IR phase (“contact lens”, “can order”, “without having”), but the two queries still do not match. In this case, bigram matching is not sufficient to establish the proper meaning of the sentence. Also, note the word “like” in the input question and the repeated “one 1” – many questions, like this one, are more similar to casual spoken English than proper written English, and are more difficult to process.

Input: what are some cures 4 the flu or if i have gastondridous?
Result: What are some good home remedies for the flu?

Here is another successful example, where Wordnet rewrote “cures” as “home remedies”, and 4 as “for”. As a result, there are a lot of bigrams in common between the two questions.

Input: is weed bad for asthma?
Result: marijuana for asthma?

Here, “weed” was rewritten as “marijuana”. Wordnet was useful for this question, but unigram IR would probably perform just as well as bigram IR because “marijuana” and “asthma” capture the meaning of the question well.

In addition, a significant subset of our reference questions are virtually impossible to answer. For example, questions like “i would like advice on what i should do” do not contain any useful information in the question itself, and one would need to look at the full text of the message body or in the answers to properly process these questions.

Discussion/Analysis

There are several linguistic assumptions inherent to our models. One is that bigrams represent the context of each question much better than unigrams. This is true, but bigrams are clearly not sufficient to capture all information in the question, and deeper semantic structures are important to fully understand questions. As a step in this direction, we used Wordnet (guided by the language model) to capture semantic information. This

was partially successful, but Wordnet has its limitations, and deeper semantic structures would be useful.

The most obvious feature and limitation of our system was that we only looked at question text in the repository. Obviously, using the answer as well would provide more information, but our goal was to focus on NLP techniques and do a more thorough job of exploiting the structure of the question. One could certainly run IR techniques on the answers, and use the result as another feature for determining the best question/answer pairs to return.

The most interesting feature of our system was the language model, and it (along with Wordnet) is the foundation on which the rest of our system is based. One interesting decision to note is that we train the language model on the same corpus that we use for evaluation. Normally, it is considered “cheating” to train and test on the same corpus. However, we are not testing the accuracy of the language model, but rather the results of the entire process, and the language model is merely a tool used in the middle of the process to rewrite questions. Language models are typically used for predicting the future. In our case, the language model is used to guide query rewriting towards the language of the question corpus. It also serves roughly as a word sense disambiguator, and tends to emphasize sentences with bigrams that are relevant to both the input question and the question corpus. Wordnet generates new sentences, and the language model keeps those similar to questions in the corpus.

Much other work on question answering systems used FAQs or other resources for their question repositories. We chose to use Yahoo answers because it seemed more interesting than FAQs as it contains more realistic, casual questions asked by typical users as opposed to very specific questions chosen by site administrators. The downside of using this data source is that a large number of users are lazy or careless, and many questions contain improper English. For example, “y” is used instead of “why”, capitalization is often lacking, and punctuation is often used incorrectly. The primary result of this is that the Part-of-Speech Tagger, which was trained on Wall Street Journal text, misclassifies many words that are improper English (e.g. “i" is a foreign word, but is intended to be used as “I”, which is a pronoun). As we only expand certain parts of speech, many of these misclassifications are irrelevant, but some do adversely affect our results. We expect that we would obtain better results if we used questions that correspond more closely to proper English, or if we could train the POS tagger on a more appropriate, labeled corpus.

We noticed that Yahoo question search contained highly stratified results, and was overall not particularly impressive. Yahoo was very successful with some questions, and their first returned result was highly relevant. In this case, the next several results were typically good as well. On many other questions, Yahoo was very unsuccessful, and no results were even close to the input question. There were very few input questions for which the first result was irrelevant but any other highly ranked results were relevant. Our interpretation of this is that Yahoo’s information retrieval algorithm works well on certain easy questions, but not a lot of effort has gone into searching harder questions,

and commercial interests tempt the user to click on sponsored links for these failed questions. In addition, this is a hard problem. Many users are asking questions because they can't find the answers elsewhere on the web. Often, this is because the answer doesn't already exist in the Yahoo database, or because the user is inexperienced with web search and is using ambiguous terms.

Another difficulty we faced was the sparse coverage of Wordnet. Wordnet does not contain many drug names, it lacks common proper nouns (like "Honda"), and often has a limited set of synonyms for words that it does contain. We attempted to improve the performance of Wordnet by including hypernym/hyponym relationships in addition to direct synonyms. Another difficulty we had was dealing with verb tenses. Wordnet could find synonyms for a past tense verb, but the synonyms were generated as present tense, and this adversely affected our language model and IR bigram scores.

Initially, we used only the Yahoo health section for both the language model and IR. Based on the number of unknown bigrams generated by Wordnet rewriting, we decided to add more questions from similar domains of Yahoo questions to training of the language model. In general, increasing the corpus size of a language model is always a good thing (if the new data is relevant to the original corpus). In particular, we hoped to store more rare words in the model, and recognize that those words are less likely than unknown words.

Also, our first attempt did not contain POS tagging, and it was so computationally intensive that we didn't even attempt a costly full evaluation on 50 sentences. The problem is that there are a huge number of words in English that can be either nouns or verbs. Because we didn't know the part of speech of a particular instance of such words, we had to expand all noun and verb senses of these words, and this was simply too noisy. Adding the POS tagger was a necessary step to allow the more interesting portions of our system to work properly.

Related work

The TREC Question Answering competition has been going for about 10 years, and much research has been conducted into question answering systems. The majority of these systems attempt to find an answer directly, rather than finding a similar question. One fairly typical example of this is AskMSR, which categorized questions into types, performed some syntactic rewrites of input questions, searched a large repository of documents for the transformed text, and combined the most frequent search results into an answer.

More successful QA systems have also been developed using deeper semantic understanding of questions such as (Harabagiu, 2001). Most of these systems involve reformulating an input question and expanding words, and many also involve question type recognition, semantic processing, and applying world knowledge.

Early question answering systems based on question to question mapping were attempted by Ask Jeeves. Their system was based on question paraphrasing, and used a set of manual rules to find similar questions based on predefined sentence patterns. When successful, it returned the answers to those questions. As the rules needed to be created manually, the system was very inflexible, and many answerable questions did not fall into the preset patterns. Their backup system was pure information retrieval.

(Lytinen, 2002) use several features to match input questions to FAQ questions. They use a sophisticated approach to question types involving 12 categories, machine learning on about 90 content-independent prepositions, and similarity metrics between question types. Their other features include cosine similarity between the input question and repository questions, coverage (percent of input question words in repository questions), and semantic similarity based on Wordnet distances between pairs of input and repository words.

Future Work

There are many additional features we'd like to include in our system. The most important of these is question type similarity. A simple possibility is to categorize questions by their first words (e.g. who, what, why, how, etc.) but this has been shown to perform somewhat poorly. (Lytinen, 2002) demonstrate a much more successful approach to question types, as discussed in the previous section.

Also, we'd like to add a simple IR feature that runs basic tf-idf scoring on the unigrams of the input question against repository questions. This would help out with cases where our question rephrasing technique transforms the sentences too much from their actual meaning towards irrelevant questions in the repository.

Another obvious feature is the content of answer documents. By only looking at the repository question, we can better analyze its meaning. But the document contains much useful information, and running IR on the answer document is another rich source of data.

Given these additional features (and potentially others as well), we could combine these features with our rephrased question bigram similarity metric, and learn weight for each feature via some machine learning technique. This could be something simple like maximum likelihood estimation, or more complex like a maximum entropy model.

Another possibility would be to replace or augment Wordnet with a more complete data source. Wordnet is useful, but is missing many words, particularly proper nouns like "Honda" or drug names. If we limited the domain of the questions, then we could use a richer data source like a semantic web or knowledge base containing domain-specific word associations.

In addition, word sense disambiguation techniques could be applied to limit the expansion of words such that only words related to the correct sense are generated. This would improve precision, and speed up computation.

Collaboration

We spent the majority of our project time working together and discussing ideas and approaches, so it is difficult to specify an exact breakdown of responsibilities. That said, the division of labor was roughly as follows. The overall design was entirely collaborative. Tait and Johnson were primarily responsible for the coding, and Josh was primarily responsible for the writeup.

Tait wrote the crawler for retrieving Yahoo questions. Josh wrote the initial Wordnet code, and Johnson and Tait both enhanced it. Johnson and Tait were both partially responsible for the query rephrasing search process. Tait was primarily responsible for implementing the language model and the POS tagger, and Johnson was primarily responsible for the IR component. Results were processed and analyzed by all group members.

References

S. Lytinen, N Tomuro, *The Use of Question Types to Match Questions in FAQFinder*, American Association for Artificial Intelligence, 2002

Marius Pasca and Sanda Harabagiu. High-performance question/answering. In SIGIR Conference 2001.

Erika F. de Lima and Jan O. Pedersen. Phrase Recognition and Expansion for Short, Precision-biased Queries based on a Query Log. http://citeseer.ist.psu.edu/558356.html

Zhai, C. (1997). Fast Statistical Parsing of Noun Phrases for Document Indexing. In Proceedings of the Fifth Conference on Applied Natural Language Processing, Washington, DC. To appear.

Ana-Marie Popescu, Oren Etzioni, and Henry Kautz. Towards a theory of natural language interfaces to databases. In Proceedings of the conference on Intelligent User Interfaces, 2003.

Wordnet: JWord, http://www.seas.gwu.edu/~simhaweb/software/jword/

Stanford POS Tagger, http://nlp.stanford.edu/software/tagger.shtml

Lucene, http://lucene.apache.org/

Appendix A: Reference questions for testing

do bush's baked beans give you gas?
How can I get rid of my pimples by tomorrow without products?
Where can I purchase alpha hydroxy acid for mild acne? is there a specific brand that works good?
What tests might the doctor order for a man with low sex drive?

How does someone get eggsma?
Does anyone know what these may be symptoms of ?: Extreme rapid weight loss and trembling hands? Thank you!?
What are the best things to put in a home gym?
A sharp pain in the center of the chest (breastbone area)?
How can I pass a saliva drug screening?
Have any men had problems with laser hair removal in the face/neck? Also, what is a reasonable cost for it?
Help with infected and swollen acne cyst please!?
What is the best way to get rid anxiousness?
What's a good way to do a stomach crunch without tearing up your back?
Yeah I'm a teenager who needs help on weight?
Question about depression and anxiety.?
Lexapro and weight gain?
What can happen to someone who abuses laxitives and is it ever safe to take everyday?
When I bend my big toe upwards, the bottom of my foot hurts?
what type of schooling do you need for tattoo removal?
what causes heart palpitations.?
What are the symptoms of low sodium blood levels?
How can someone help stabilize their cholesterol? What foods can someone eat to lower the levels?
I would like advice on what i should do?
Where can I find or how can I get in touch with a specialized sex counselor or specialist?
health hazards of the mineral lime?
Teeth filing...?
Why does my resting heart rate of 60-65 speed up at night to 70-76 and I hear swishing noise in both ears?
My contact lens ripped, is there any way that I can buy like ONE(1) lens without having to buy a sixpack.?
I think I have a small medial cartilage tear- Some pain, full ROM, no swelling. How long before it heals?
Can Prozac make you extremely tired?
Chemical burns in mouth?
What should I do if I'm eating right and exercising but still fell sleepy during the day?
what are some cures 4 the flu,or if i have gastondridous.???.....?
Why I sweat? and How can I stop this problem?
I have acid in my back how can get ready of as soon as possible. my birthday party is soon and I need my back?
How can I stop my hives from itching?
Hello, just wondering If anyone can help with sleep suggestions. Ive tried ambien, to expensive. Cant sleep.?
I have a pain on the left side of my waist, anyone know what that may be?
why is Provigil 200mg so expensive? mine cost $448.00 30 day supply for sleep disorder.?
hello....ive being diagnosed with hypothyroidism since i was 16...and now am 21...iam usually into depression?
how do i cure my paranoia or however u spell it?
Can swimming after a mantoux test cause a false positive?
ECG Abnormal?
Where can I get Nitrofuradantoin 100 mg tablets?
what to take to pass a drug test for crystal meth?
My wife was diagnosed with kidneystones, what are good/bad food/drinks for her?
Is it true that postmenopausal women under 50 should take birth control for two years after their last period?
i got a spider bite and its now big and its purple in the middle should i worry?
is weed bad for asthma?
is there any natural way of getting rid of hayfever?