Non-Factoid Question Answering

Information Retrieval Final Project

Rawan Manasra
rwan.mnasra@gmail.com

Jumana Rayan
jumana.rayan@gmail.com

ABSTRACT

In this paper, we will describe the approaches and technologies we used to get the best answers for non-factoid questions, using apache Lucene and yahoo database and starting with Shai project as a base project.

INTRODUCTION

Most of the work about question answering has focused on the task of factoid QA, where questions match short answers, usually in the form of named or numerical entities. But the approach for answering factoid questions won't be affective for non-factoid questions, we have reason, manner and caution questions, that for each one of these categories there are different type of answers.
In this paper we will discuss the approach that we used and other approaches that we tried.

Data Base

We ran our program on yahoo database, which contains questions and each one of them has several answers, and a chosen best answer, the problem that we needed to solve is retrieving 5 best answers for each question.

We translated this database into a hash map, which it's key is the answer id, (i.e. the first answer in the corpus got id = 0 and the id was increasing with each answer) and it's value is an answer.

Open Source

We used Lucene Apache 6.0.0 and the Shai Erera project.

Approach

The approach that we used was simple and classic for searching mechanisms:

1. Filtering stop words.
2. Indexing.
3. Scoring documents by exact match terms frequency.
4. Sorting retrieved answers.

Indexing

As we explained in section 2, each document (i.e. answer) got a unique id.
And we used inverted index, with SnapshotDeletionPolicy which adds a persistence layer so that snapshots can be maintained across the life of our application. The snapshots are persisted in the index Directory and are committed as soon as snapshot() or release(IndexCommit) is called.

Data processing and analysis

For data processing we used stopping for proving it's effectiveness, we tried tokenizing but the accuracy results dropped by half.

Stopping

we used English analyzer and created a set of stop words that contains the default English analyzer stop

words and we add more words to it, so after getting a query, we filtered the stop words from it, then we send it to the searcher.
If after filtering the query, we got an empty string (i.e. all the query was stop words) then we replaced the query with "verb noun" string.

tokenizing

we tried to use the tokenizing method but due to it's bad results we ditch it. The tokenizer we built used wn_s.pl [3] as rules for Synonym Map, extending the analyzer we used Classic Tokenizer and standard and lowercase filters.
But by using this analyzer the accuracy decreased by 50% and my guess is that the wn_s.pl file was not the best ft for our data set, not even close enough, and the alternative was to build new one or skip this method, as we did.

retrieval approach

we used the same searcher from Shai project with minor changes: updating the maximum passage length to be 40000 as the length of the longest document, also we ditched the overlap ratio parameter. we tried to add VSM formulas and add VSM Similarity for the index searcher, but the accuracy results dropped to 19% (the best result was 23.6%).
The algorithm we used was like that:

1. creating an index searcher, and retrieving the top 5 documents, according to the document score (i.e. regular index searcher).
2. For each one of the top 5 documents we extracted the exact terms that appeared in the query and saved their positions.
3. Then each doc was rescored like this: the total number of query terms it contains multiplied by its own score.

- $new - doc - score = tf * doc - initial - score$

4. Ranking the top 5 documents was like this: first we prefer the documents that have the highest term scores, if they were equal then we sort by the document score, if they were equal, then by the start offset of the term, then if they were equal, by the document id.
5. Returning the top 5 answers for the given question.

Finally

the results that we got for these methods were much higher than other methods that we mentioned above.

ACKNOWLEDGMENTS

Our thanks to Shai Erera for his project that was our base project, and for all the forum questions that helped a lot.

REFERENCES

[1] Mihai Surdeanu, Massimiliano Ciaramita, Hugo Zaragoza, 2011. Research Learning to Rank Answers to Non-Factoid Questions from Web Collections.
[2] Junichi Fukumot, 2007. Question Answering System for Non-factoid Type Questions and Automatic Evaluation based on BE Method.
[3] https://raw.githubusercontent.com/kenoir/polysemous/master/src/wordnet/wn_s.pl
[4] https://lucene.apache.org/core/6_0_0/core/org/apache/lucene/search/similarities/package-frame.html