Building a Pipeline of Question Answering for Biomedical Question Answering

As both biomedical research and question answering systems have been discussed very frequently these years, we try to combine these hot topics together for better helping research and commercial application. In our project we use UIMA framework to build our QA system with a focus on list questions. Using the test data we got, we have results shown in result section.

1. Introduction

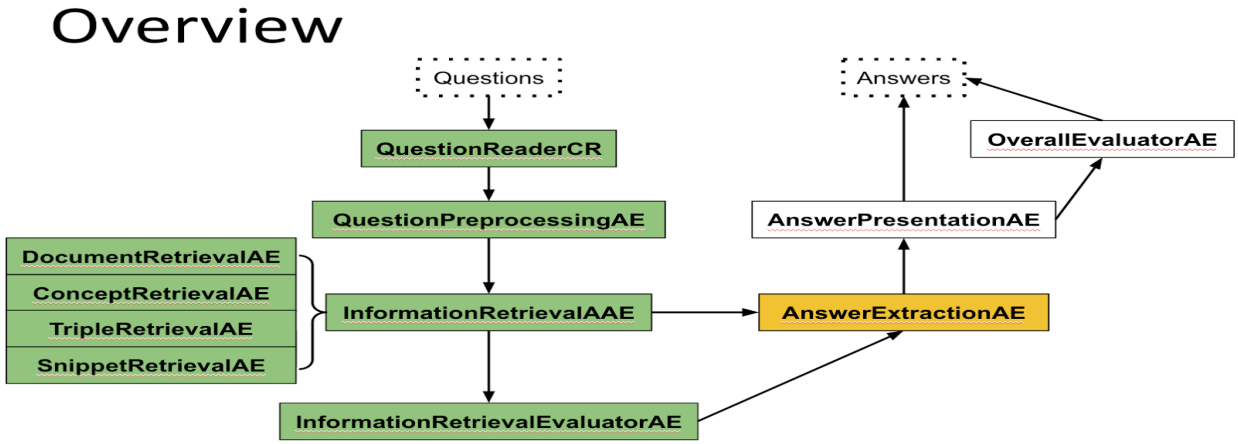
NLP components such as parsing, part of speech tagging, named entities recognition, which are important tools for question answering systems, are improving year by year. Moreover, NLP researchers in these fields are paying more attention to the Bio domain these years, which helps a lot in the domain specific (Biomedical) Question Answering system.

In our system we extend the initial framework (baseline system with basic NLP tools) to get an improvement on List question answering. We have found that the performance of a QA system highly relies on information retrieval in different stages includes query expansion, document retrieval, snippets retrieval, etc. And in information retrieval system, how to define similarity is very important.

To achieve better results, we used Word2Vec to project the word into vector space and applied it with other NLP tools in our system. The outline of this report is as follows. In section 2, we introduce the framework of our system. In section 3 we show the experiments and results. In section 4 we conclude our work and provide a general idea about future works.

2. System Architecture

The system arhchitecture can be generally separated into two parts. The first part is about making useful annotations, for example using NER, Pos tagging to annotate the elements in document, concept and generate snippet. The second part is how to rank the these elements given the annotation and then generate k-best answers.



<u>For all the word similarity process, we project the word to vector space using Word2Vec.</u>

2.0 Data expansion

We only have **8** list questions in BioASQ-SampleData1B.json, which will easily overfit our system. That's why we we extracted **91** list questions out of BioASQ-trainingDataset2b.json to use in experimentation.

2.1 NLP Resources we use

Stanford Stemmer
Stanford POS tagger
Lingpipe
ABNER
Word2Vec (We will introduce this tool with detail in 2.6)

2.2 Concept Retrieval

The gold standard urls contains 2012 and the document contains 2014, so we change the 2014 to 2012.

We experiment on different threshold to filter bad concepts. The final threshold we are using is 0.15.

2.3 Document Retrieval

We tried to expand our query to get more and better documents. Basically we use NER tool(Abner) and POS tagger to get important word. And then we try to find the synonyms of these words using both UMLS synonyms dictionary and Word2Vec.

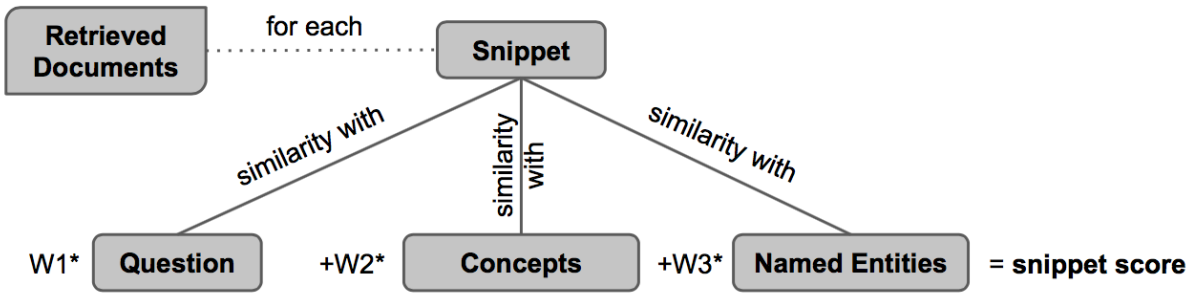
We also tried to use concept and bigrams in the query expansion part. After we can all the words we use OR and AND operator in PubMed.

2.4 Snippet Retrieval

For snippet retrieval, we compute similarity between all snippets and three other things related to a question. First we compare snippet with original/normalized question, and compare to the concepts (concatenate all the concepts with space and remove duplicates and replace the double quotes with empty string) we retrieve in the last section, and compare the named entities from snippet with the name entities / proper nouns we retrieve from the query.

We weighted the three similarities and experimented with different configurations. The best configuration was (0, 1, 0) which means only use the similarity with the retrieved concepts. We tried (1, 1, 1), (1, 0, 0), (0, 1, 0), and (0, 0, 1).

All the similarities are computed using our *pubvec* web service.

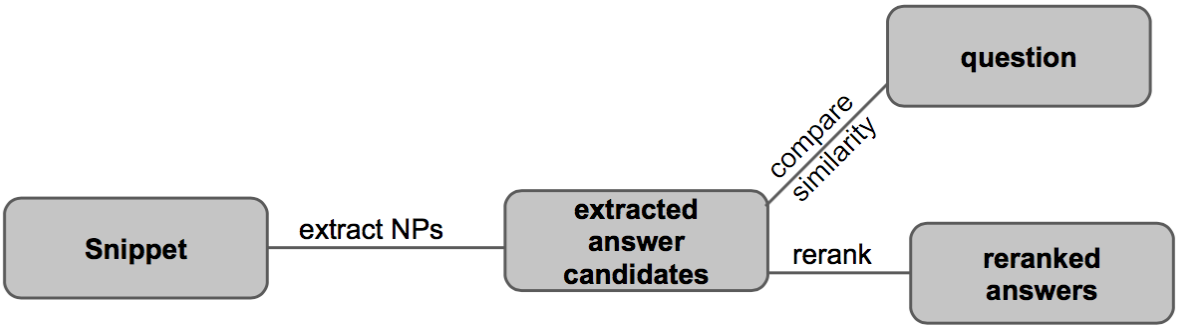


2.5 Answer Extraction

We extract all answer candidates from the Snippets. Since two-word noun phrases are quite rare in the training data and three-word NPs are even fewer, we assume it is safe to assume 3 is the longest possible NP length. We thus extract all consecutive NN* in the snippets, with max. length=3.

NEs in Snippets with higher ranking should be considered more important, so we use DCG to tune both word to word similarity and the ranking of the Snippets where the word comes from.

After extracting candidates, we rank them using cosine similarity to the question and which snippets they are come from. We use Discount Similarity Score to tune between similarity and ranking of snippets.



2.6 Word2Vec Server

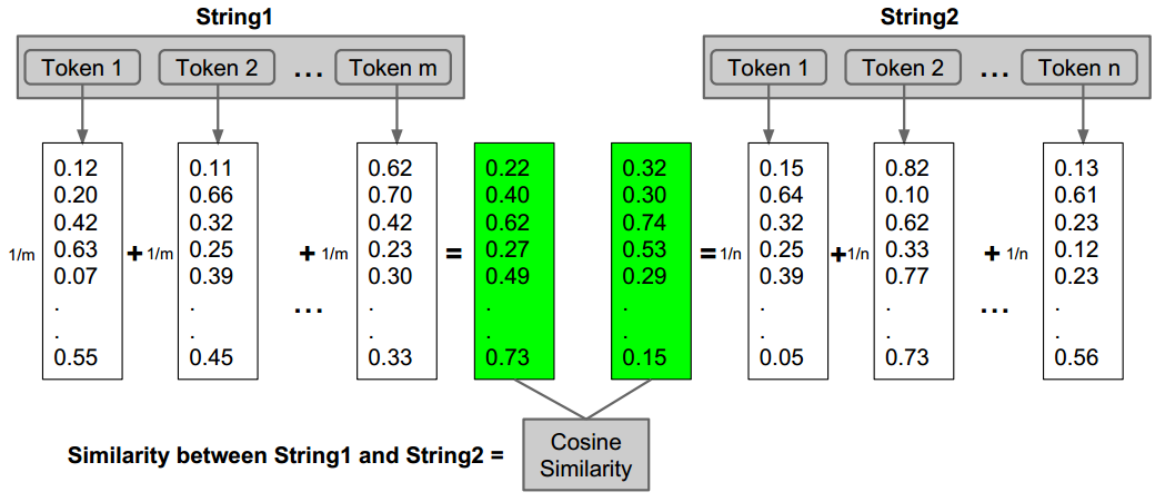
We implemented a web service to speed up working with word vectors using Ruby on Rails. The original vectors are provided for 1,701,632 words (types). We included only words that start with either a letter or a digit which gives around 1,400,000 words. Using mysql, we create an index on the words.

- Given a word: return its 200-length vector.
- Given a string and n, return the closest n words (in cosine similarity). We used it for query expansion.
- Given two strings, returns the cosine similarity between their two vectors, a vector of a string is the addition of its words vectors. We used it in snippet retrieval and answer extraction

All parameters are sent as GET parameters as following:

- <home_url>?w=<word> to retrieve the vector of the word.
- <home_url>/expand?query=<query>&n=<numberOfWordsToReturn> to expand the query with n number of additional words.
- <home_url>/sim?m=<string1>&n=<string2> to compute similarity between string1 and string2.

The following figures shows our algorithm to compute the similarity between two strings:



3. Experiment & Results

	Precision	Recall	F1	MAP	GMAP
Concept	0.1548	0.3824	0.2203	0.3589	0.1761
Document	0.0130	0.0923	0.0228	0.0715	0.0260
Snippet	0.0015	0.0083	0.0026	0.0005	0.0104
Match	0.0091	0.2800	0.0176	0.0158	0.0170

4. What we learnt & Future Work

In this project we provide a framework for building a BioMedical QA system pipeline focusing on List question. In the process, we leant more about UIMA, about QA systems, about how to use different NLP tools, about using SCRUM, about Word2Vec, and about Software Engineer.

Future work includes code refactoring, better use of WordVec2 to improve the performance in document retrieval, Optimize the query expansion and support other types of questions.

Acknowledgements

Thanks to Prof. Eric Nyberg who gave us the lectures and knowledge on how to build intelligent information systems. Also thanks to the TAs for their guidance.

Reference

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of NIPS, 2013.