The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Automated Question Answering System for Community-Based Questions

Chanin Pithyaachariyakul, Anagha Kulkarni
cpithyaa@mail.sfsu.edu, ak@sfsu.edu
San Francisco State University
1600 Holloway Ave, San Francisco, California, 94132

Abstract

The emergence of community question answering sites, such as, Yahoo! Answer (Y!A), and Quora, indicate that for certain information needs, users prefer receiving focused answers to their questions, rather than a list of URLs from search results. This trend has sparked a rich area of investigation at the intersection of Information Retrieval (IR), Natural Language Processing (NLP), and Machine Learning (ML) of Automated Question Answering (QA). In this paper, we present our attempt at developing an efficient QA system for both factoid and non-factoid questions from any domain. Empirical evaluation of our system using multiple datasets demonstrates that our system outperforms the best system from the TREC LiveQA tracks, while keeping the response time to under less than half a minute.

Introduction

Question Answering (QA) problem has been researched extensively by IR, NLP, and ML communities (Agichtein et al. 2016). To accomplish the QA task, the following sub-problems have to be addressed: (i) transforming free-text questions into well-formed boolean queries (ii) compiling sources of documents that may contain the answer, (iii) extracting short units of texts as candidate answers from the retrieved documents, and (iv) selecting the best answer using effective ranking algorithms (Wang et al. 2015). To solve each of these sub-problems, we designed our QA system, *SF-State-QA*, into four components: Question Formulation Module (QFM) applies effective tools to transform verbose questions into key-phrase queries; Document Retrieval Module (DRM) searches knowledge-rich resources to obtain answer-bearing web-pages; Candidate Answer Extraction Module (CAEM) generates short passages from the web-pages for candidate answers, and Answer Selection Module (ASM) utilizes ML techniques to select the best answer. We use TREC LiveQA datasets to evaluate our system performance with the official ranking metrics. The results demonstrate that our system outperforms the best systems from the TREC LiveQA 2015 and 2016 competitions.

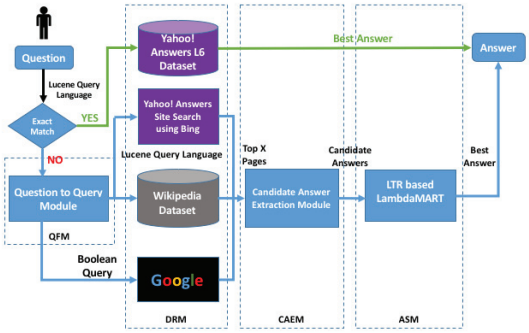

Figure 1: System's Architecture

System Overview

Figure 1 illustrates the architecture of our system. Next we describe the four modules of the system in details.

Query Formulation Module (QFM)

The QFM transforms the free-text questions to well-formed boolean conjunctive queries that can be evaluated by search engines. This is one of the most difficult processes because human-generated questions are often lengthy and ambiguous, containing many unnecessary connecting words for human-friendly readers. However, the additional words create the long queries that mislead the main idea of the questions, or even do not return any search result. Thus, we need to analysis the complex questions to generate the queries that are precise for search engines to highly obtain relevant documents. In this research, we applied Stanford dependency parser library (Chen and Manning 2014) to detect phrases in order to generate the key-phrase queries based on grammatical rules. For instance, the question: *"Why's juice from orange peel supposed to be good for eyes?"* is transformed to a boolean query: *"(orange peel) AND (good for eyes) AND (juice)."* This keyword-phrase query is more explicit than single-word query because *orange juice* has more dominant presence on the sites than *orange peel juice*. Consequently, with the single-word query, documents about *orange juice* have higher opportunities to be retrieved that is not relevant to the question. Therefore, key-phrase queries are more unambiguous to obtain higher amount of accurate documents.

In the Y!A datasets, each question provides the category of the question such as health, pet sport etc. We use this category classification to apply a new query expansion approach for health questions. Since health questions contain many diversity of language, the query expansion technique is widely used to optimize the queries to solve the vocabulary mismatch (Zhu et al. 2014). In this work, we applied Metamap tool to discover clinical terms and expand the queries with the term that might be relevant using the UMLS Metathesaurus. For instance, as a question: *"How to treat type 2 diabetes without medication?"*, The MetaMap tool is used to expand and formulate the following query: *"(medication OR pharmaceutical preparations) AND (non-insulin-dependent) AND (type 2 diabetes OR diabetes mellitus)"*. This query is effective to solve the vocabulary mismatch in medical publications to improve retrieval performance.

Document Retrieval Module (DRM)

For this paper, we utilize two search engines: Bing Y!A site search API, and Google Search API, and two datasets: L6 Y!A dataset, English Wikipedia dataset. The L6 Y!A dataset, which contain 4.4 million questions, is used to check if these exists an exact match from the old questions. If there is an exact match, then the best answer from the previous question is returned, and that is end of the answer generation for that question. However, if an exact matching question is not found, then Wikipedia, which contains 5+ million articles, is used to search for relevant articles. In parallel, Bing Y!A site search is used to search for the similar questions from current Y!A site. Also in parallel, Google search API, is applied to retrieve documents from the WWW. The top three documents are retrieved by each of the above searched to obtain a total of nine documents at most.

Candidate Answer extraction Module (CAEM)

We applied jsoup library to download and convert HTML pages into the lists of plain-text sentences. We used the slicing window technique to concatenate the sentences into concise passages, which contain at most 1,000 characters. We then calculate and rank the passages based on its semantic similarity to the queries using Okapi BM25. The top passages are returned as candidate answers to the next pipeline.

Answer Selection Module (ASM)

ASM is built to select the best answer from the lists of candidate answers. We applied LambdaMART (Burges 2010), a Learning to Rank algorithm. We used the TREC LiveQA datasets to train the model. The labeled data is based on the following features: TF.iDF, Okapi BM25, cosine similarity, number of overlapping terms, number of characters, number of words, and number of non-alphanumeric characters. The candidate answers are labeled by its score of the features and run through the Ranklib library based LambdaMART model. The candidate answer, which obtains the highest score, is selected as the best answer.

Experiments, Results, and Analysis

For empirical evaluation, we used TREC LiveQA 2015 and 2016 datasets, which contain 1,000+ questions each. We randomly selected 200 questions from the datasets and performed manual assessment using TREC LiveQA 4-level Likert scale from 1 (poor) to 4 (excellent). We evaluated the system's performance based on the official TREC LiveQA metrics: avgScore(0-3) and Success@i+. The avgScore is average score over all questions (tranferring 1-4 to 0-3). The Success@i+ (i=1..4) is the number of question with score i or above, divided by the total number of questions. We compared our system's performance with the highest ranking systems from each competition. Table1 reports the end-to-end results of TREC LiveQA 2015 and 2016 datasets. Our system outperformed the best systems in all evaluation metrics providing at least 1.42 of avgScore with approximately 26 seconds of processing time per question.

Table 1: End-to-end Results

	avgScore(0-3)	Success@		
		2+	3+	4+
TREC 2015				
SF-State-QA	1.420	0.650	0.430	0.340
OAQA	1.081	0.532	0.359	0.190
TREC 2016				
SF-State-QA	1.570	0.760	0.500	0.330
EmoryCrowed	1.260	0.620	0.421	0.220

Conclusions

We presented an automated QA system that employs simple but effective query generation approach, uses multiple document sources to compile a strong pool of candidate answers, and then identifies the best answer using trained answer ranking models. Empirical evaluations demonstrate that our system performs 31% and 25% better than the best performing system at TREC LiveQA 2015 and 2016, respectively, with average response time of less than half a minute.

References

Agichtein E., Carmel D., Pelleg D., Pinter Y., Harman D., 2016. Overview of the TREC 2016 LiveQA Track. *Proceedings of TREC, 2016* 1–10.

Burges C., 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11: 23-581.

Chen D., and Manning D. C. 2014. A Fast and Accurate Dependency Parser using Neural Networks. In *Proceedings of EMNLP, 2014* 740–750. Doha, Qatar.: EMNLP Press.

Wang D., and Nyberg E. 2015. CMU OAQA at TREC 2015 LiveQA: Discovering the Right Answer with Clues. In *Proceedings of TREC LiveQA, 2015.*

Shtok A., and Szpektor I. 2012. Learning from the past: Answering new questions with past answers. In *Proceedings of the 21st International Conference on World Wide Web, 2012* 759-768. Lyon, France.: ACM Press.

Zhu, D., Wu, S., Carterette, B., and Liu, H. (2014). Using Large Clinical Corpora for Query Expansion in Text-based Cohort Identification. Journal of Biomedical Informatics, 0, 275-281.