Simple Answering Questions System

Gon-soo Moon

Master Student of Computer Science
University of Texas at Dallas,
Richardson, TX, USA
gonsoomoon@gmail.com

Abstract

In the major natural language processing (NLP) applications, a question answering application is one of the challenging and harder one. This paper as a starting point focuses on simple what-type question based on sources such as Wikipedia, WordNet and Google as well as scalable system implementation.

Introduction

This paper handles how to answer this kind of a simple question, "What is something?" like "What is a logical entailment?" In order to answer one, with algorithm implementation to answer, the Simple Answering Questions System (SAQS) is developed and deployed on the cloud service, Amazon Web Services (AWS). As a logical process, to handle the question, the following steps in brief are involved: 1) the user types a question, 2) the syntactic and semantic parser finds word dependencies, identifies what-type question and then extracts a relevant word which is called a headword, 3) with the headword, SAQS queries it to Wikipedia, WordNet and Google. For WordNet, the original question as a context is passed in addition. Each returned result per source is used to extract a specific answer using its own algorithm, 4) aggregating all answers, SAQS generates one HTML document and then delivers it as an answer to the user. The Fig. 1 shows as below an overall flow based on what I mentioned.

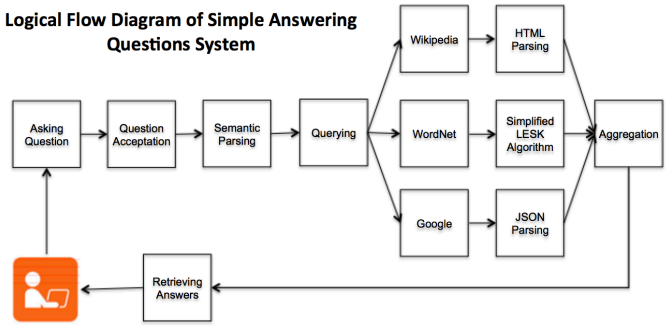

Fig. 1. Logical Flow Diagram of Simple Answering Question System

From next sections, I will explain the brief logical process in detail, how to implement SAQS from a scalable system perspective, experiments and then discuss possible future work.

Approach

To explain details, I will use Fig. 2, Component Diagram of Simple Answering Questions System as below getting along with Fig. 1.

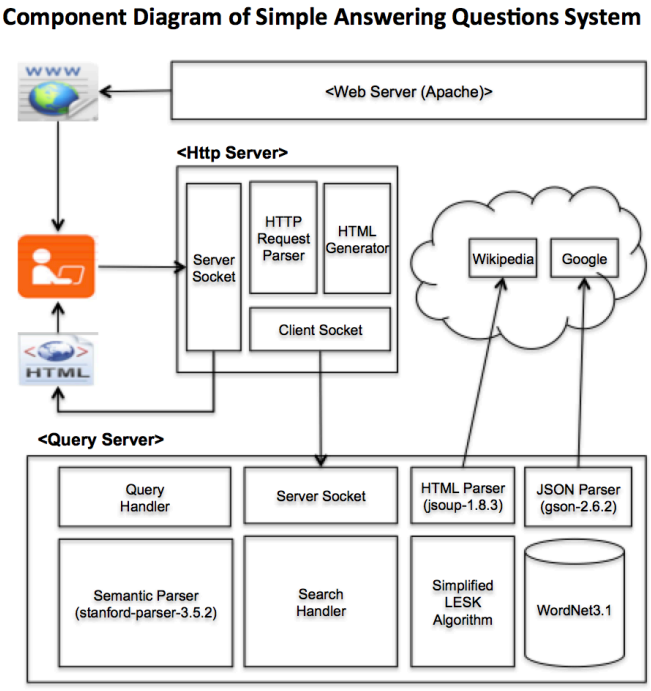

Fig. 2. Component Diagram of Simple Answering Questions System

First, a user requests query.html on the web server (Apache). Once the user retrieves it in its web browser, he or she types any what-type question like "What is a logical entailment?" and submits it. The Http Server, denoted as <Http Server> in Fig. 2, accepts the request through TCP/IP Server Socket, extracts the question from the HTTP POST request and sends

it to the Query Server, denoted as <Query Server> in Fig. 2, via TCP/IP Server/Client Socket.

Second, once Query Server retrieves the question, it is sent to Semantic Parser, standford-parser-3.5.2 [6]. It parses the question and makes a result in the form of "Relation (Governor word, Dependent word)". For an example of "What is a logical entailment?" a result is a set of relations such as cop (What-1, is-2), det (entailment-5, a-3), amod (entailment-5, logical-4), nsubj (What-1, entailment-5). The current goal is to identify this relation, nsubj (nominal subject), and to extract as what-type question a governor word, "What", and as a headword a dependent word, "entailment" in the nsubj relation. The headword extracted is used to query to Wikipedia, Google and WordNet.

Wikipedia returns an article related to the headword, and then the HTML Parser, jsoup-1.8.3 [3], retrieves the first paragraph in the article. As the next source, Google returns a list of search results and JSON parser, gson-2.6.2 [2], selects the first result. Last, as to WordNet, I apply Simplified LESK algorithm in shown on Fig 3. In addition to the headword, the question, "What is a logical entailment?" is used to feed into the Simplified LESK algorithm [4]. In order to access the electrical WordNet database, it uses a WordNet interface, Java Libraries for Accessing the Princeton WordNet [5], and provides "best-sense". Finally the three results from the three sources are sent to the Search Handler that will deliver them to Http Server via TCP/IP Server/Client Socket.

```
function SIMPLIFIED LESK(word, sentence) returns best sense of word

  best-sense ← most frequent sense for word
  max-overlap ← 0
  context ← set of words in sentence
  for each sense in senses of word do
    signature ← set of words in the gloss and examples of sense
    overlap ← COMPUTEOVERLAP(signature, context)
    if overlap > max-overlap then
        max-overlap ← overlap
        best-sense ← sense
  end
  return(best-sense)
```

Figure 20.3 The Simplified Lesk algorithm. The COMPUTEOVERLAP function returns the number of words in common between two sets, ignoring function words or other words on a stop list. The original Lesk algorithm defines the *context* in a more complex way. The *Corpus Lesk* algorithm weights each overlapping word w by its $-\log P(w)$ and includes labeled training corpus data in the *signature*.

Fig. 3. Simplified LESK Algorithm

Once Http Server retrieves a set of answers from the three sources, it generates as an answer one HTML document and then eventually delivers it to the user.

Experiments

To experiment, SAQS is deployed in AWS. For the web query interface shown in Fig. 4, query.html file is deployed on http://www.imaginehappier.org/user/gsmoon/query.html and two server programs, Http Server and Query Server, are deployed, listening to 7000 and 7001 port respectively on the AWS virtual server, www.imaginehappier.org. I selected three questions as experiments even though a variety of what-type question types are plausible because the three questions can present a major of features on SAQS and points to be able to be improved for future work.

The first question is "What is a logical entailment?" shown in Fig. 4 and an answer is shown in Fig. 5. With the "entailment" as a headword SAQS extracted the first paragraph of the article on http://en.wikipedia.org/wiki/entailment, did best sense of entailment on WordNet and did the first result of a set of the Google search results.

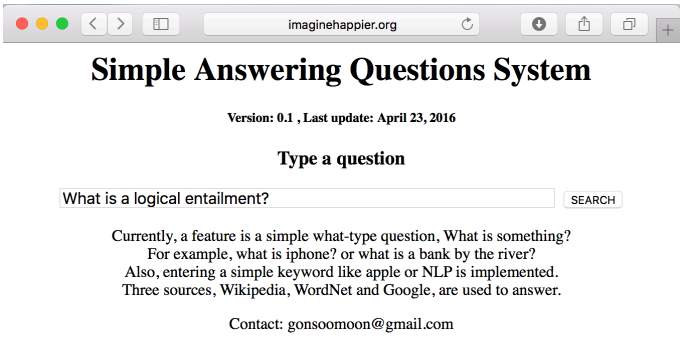

Fig. 4. Web Query Interface with 'What is a logical entailment?'

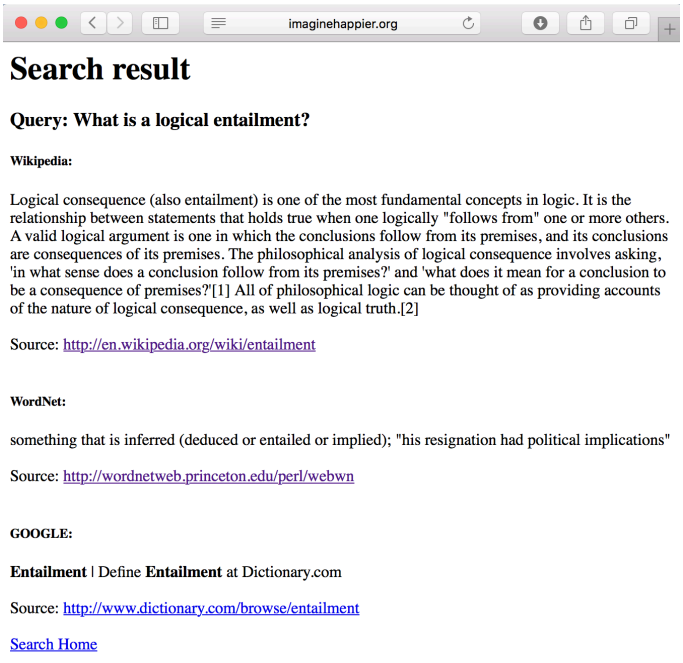

Fig. 5. Web Query Answer with 'What is a logical entailment?'

The second question is "What is a bank by the river?" in Fig. 6 and an answer is on Fig.7. Only with the "bank" headword, just as Wikipedia returns a definition of a bank as a financial

institution, Google does a little different result that is Bank of America. However, with two pieces of information such as the same headword and the question as a context piece, WordNet through Simplified LESK algorithm provides best sense among senses. For reference, if I type the question on Google, www.google.com, it displays information about a sloping land.

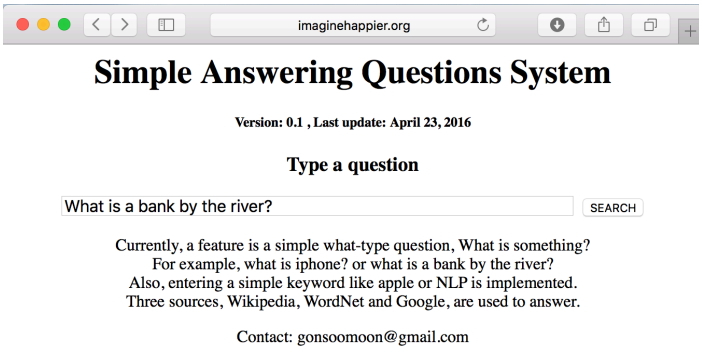

Fig. 6. Web Query Interface with 'What is a bank by the river?'

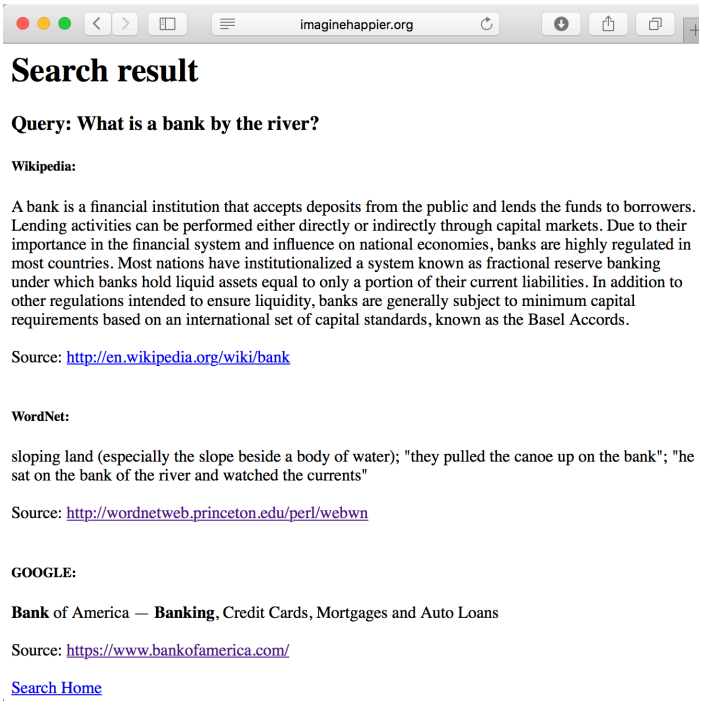

Fig. 7. Web Query Answer with 'What is a bank by the river?'

The third question, "What is snapchat?" and answer are on Fig. 8 and Fig. 9 respectively. Wikipedia and Google return an exact result that would be intended by the user. However, WordNet doesn't have information because the headword "snapchat" is recently made, which refers to an image messing application software product Wikipedia displays on Fig. 9.

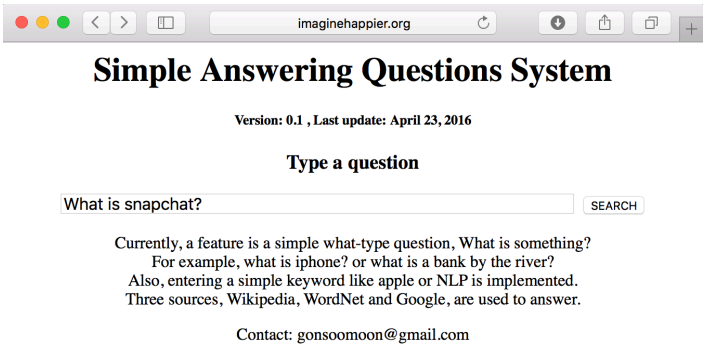

Fig. 8. Web Query Interface with 'What is snapchat?'

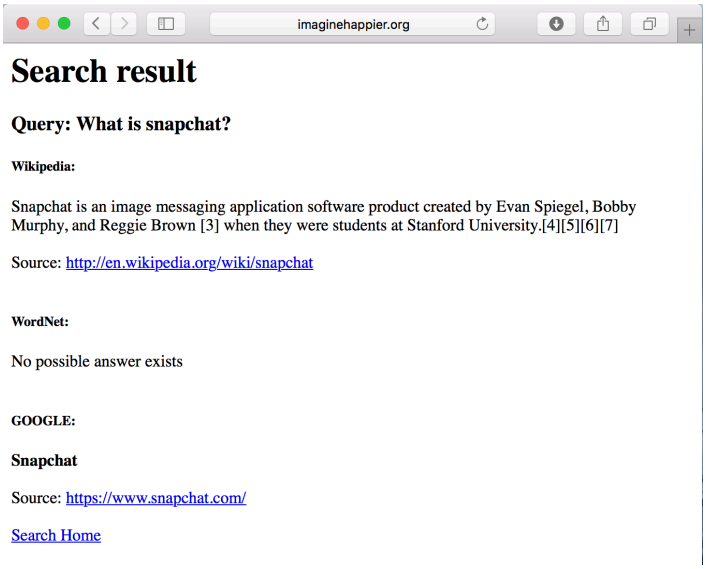

Fig. 9. Web Query Answer with 'What is snapchat?'

Only with three examples I mentioned, the experiment is able to show features SAQS has as a straightforward way. On the URL, http://www.imaginehappier.org/user/gsmoon/query.html, you can experiment more types of what-type question. Also, it is possible to experiment a few words without "what" on the URL. Consequently, three examples show particular characteristics per source and algorithm I applied. Through experiments and results, SAQS shows many points to be able to be improved. In next section, I will discuss them.

Discussion

To better handle simple what-type question, "What is a bank by the river?" like WordNet through Simplified LESK algorithm, two pieces of information such as the headword and

context phrase could be necessary to be passed into Google and other sources. However, in the case of Wikipedia, it is hard to feed both because it cannot process both. Accordingly, depending on a source, SAQS needs to determine whether headword and context phrases are needed. To step ahead, for a specific what-type question like "What is the telephone number for the University of Texas, Dallas?" related articles and information are needed for more elaborate processing such as question formulation, question classification, source documents indexing, passage retrieval processing and answering processing which can be called information retrieval (IR) [4]. Furthermore, in addition to what-type question, the other types of who, when, where and others can be expanded. Also, the more elaborated subjects involve knowledge base (KB) representation and reasoning.

Conclusion

A term project in the natural language processing class goes by, I have learned more knowledge compared to when I prepared for this term project proposal in the middle of semester. Accordingly, I thought that my term project proposal had limitation to implement better answering questions application. However, obtaining and implementing new knowledge, I have a starting point and will be able to improve answering question application. That is what I achieve in the project.

References

[1] Christiane Fellbaum (1998, ed.) WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.
[2] Google (2008), gson (Version 2.6.2) [Software]. Available from https://github.com/google/gson
[3] Jonathan Hedley (2009-2016), jsoup HTML parser (Version 1.8.3) [Software]. Available from https://jsoup.org
[4] Jurafsky Daniel and Martin, James H, Speech and Language Processing, 2nd ed. Prentice Hall, 2008, pp. 646.
[5] Finlayson, Mark Alan (2014) Java Libraries for Accessing the Princeton Wordnet: Comparison and Evaluation. In H. Orav, C. Fellbaum, & P. Vossen (Eds.), Proceedings of the 7th International Global WordNet Conference (GWC 2014) (pp. 78-85). Tartu, Estonia. Available from http://projects.csail.mit.edu/jwi/
[6] Stanford NLP Group (2015), Stanford-parser (Version 3.5.2) [Software]. Available from http://nlp.stanford.edu/software/lex-parser.html