2019 International Conference on Sustainable Technologies for Industry 4.0 (STI), 24-25 December, Dhaka

Information Extraction from WWW using Structural Approach

Md. Zahidur Rahman*, Md. Hasan Hafizur Rahman† and Md. Faisal Bin Abdul Aziz‡
Department of Computer Science & Engineering*†‡
Comilla University
Cumilla - 3506, Bangladesh
mzahidur.bd@gmail.com*, hhr@gmail.com† and faisal_245cse@yahoo.com‡

***Abstract*—The expeditiously growing World Wide Web (WWW) consists of a large number of data sources from diverse organizations all over the world. The heterogeneous nature of these data poses challenges to researchers to extract specific information. In this regard, the area of finding answers of a specific question from available web contents is an emerging area of research. Questions are normally expressed in natural language and for finding answers to natural language questions from web contents; Question Answering (QA) is the most promising framework, which can be implemented on either closed domain or open domain. In this paper, we propose an automated *QA* system which can answer binary and wh-interrogated questions about closed domain using wikipedia articles as its knowledge source. The system allows us to generate questions from wikipedia pages and then to extract answers to questions from wikipedia pages in real time.**

***Index Terms*—Question-Answering System, Wikipedia, Linguistic approach, Question Generation, Answer Extraction.**

I. INTRODUCTION

In recent years, there has been a marked increase in the amount of data available on the Internet. Now-a-days global users have specific questions in mind, for which they want to get answers from the WWW. They expect these answers would be short and precise, even they prefer to express their questions in native language without being restricted to a specific query language, query formation rules, or even a specific knowledge domain. In this regard, the classical *Natural Language Processing (NLP)* tools alongside statistical and machine learning approach provide a Question Answering (QA) framework, to carry out a complete genuine investigation of the question from a linguistic perspective and to attempt to understand what the user really means.

In addition, the *QA* problems have addressed in the literature since the beginning of computing machines. Earlier experiments in this direction implemented systems that operate in very restricted domains to understand the nature of the questions and classify them in a way similar to how human beings understand and answer questions.

QA can work for both closed and open domains. Open-domain *QA* fulfills a user's information need by giving direct answers to natural language queries, is a challenging but important problem. Closed domain *QA* helps researchers to gather important information about a certain context like judiciary documents, medical reports, collection of articles, e-commerce site, etc. Therefore, the motivation for this research work has come from the recent growing interest in the closed domain field that inspires us to work on a certain context. This proposed system will help in building an effective system for researcher to analyze and learn more on a certain context in the field of computer science & engineering of ongoing fourth industrial revolution (Industry 4.0).

II. RESEARCH GAP

Previously, some beautiful works done on a closed domain like QARAB[1] on a collection of Arabic newspapers, Dialog Navigator[2] serviced by Microsoft Japan, WEBCOOP[3] on tourist domain, ExtrAns[4] on genomics area, enquireMe[5] on health-related issues. The aim of work in closed domain is, to help researcher to explore an area of his/her interest to learn more about it.

For this work, we treat wikipedia articles as the knowledge source. The reason we choose wikipedia articles is that their contents are general and random, which contains information on all branches of knowledge and present a neutrally written summary of existing mainstream knowledge in a fair and accurate manner; acts as an online encyclopedia. In this paper, we propose a method to integrate wikipedia articles with *QA* framework that will help researcher to summarize the important information from a collection of wikipedia articles on a certain context and also help in self-learning.

III. METHODOLOGY

This paper presents a QA (Question Answering) System based on linguistic approach, which is one of the famous approaches for extracting information from WWW and building *QA* framework. Linguistic approach understands natural language text, linguistic and common knowledge and emphasizes on the syntactic structure of a sentence according to some context-free grammar. The core idea is to transform a syntactic tree of a declarative sentence into a precise query by implementing some common linguistic techniques such as parsing[7], tokenization[8] and POS tagging[9]. And then extract the respective answer of that query from the source. All operations are straightforward from a syntactic point of view.

Green's *QA* system[10] reads the question from punched cards and examine the phrase structure and other syntactic

978-1-7281-6099-3/19/$31.00

facts and then determines the answer from the source. In [11],[12],[13] system provides users with multi-media information in response to questions formulated in English using web as their knowledge resource. Unlike the above mentioned systems, our system can generate arbitrary number of questions from the knowledge source and put emphasis on certain context to know deeply about this. To provide computationally expensive options in the information extraction process, we employ Stanford NER tagger, Stanford parser instead of using corresponding tools from Natural Language Toolkit (NLTK)[14]. We consider wikipedia articles as the knowledge resource of this *QA* system.

A. Challenges

There lie certain challenges in the way of developing a conceptual model in linguistic approach, including lexical gap[15], ambiguity[16], multilingualism, question's entity identification etc. The most prominent challenge is the lexical gap. It is transparent in the difference between questions expressed in natural language and the semantically structured information of the knowledge base. Ambiguity is the phenomenon of the same expression having different meanings; this can be structural, syntactic and semantic. There is not a single language that is always used in Web documents in WWW, documents can be expressed in various languages at WWW, so it is challenging work to process such documents for formulating QA pair. Another most prominent challenge is the question's entity identification, especially in questions involving multiple entities. The lexical gap can negatively affect this issue and increase the difficulty of entity identification.

To explore an appropriate model to meet all of these research challenges describe above in the area of information extraction from WWW will be investigated and this will enable us to develop a standard idea in a more robust way. Firstly, a conceptual model for parsing information from WWW(i.e. wikipedia articles) and generating questions from retrieved information will be developed. Then, a conceptual model for understanding questions, question type and extracting answers to those questions will be developed.

IV. SYSTEM ARCHITECTURE

The question module takes a wikipedia article and a number as input and output a number of questions. The answer module takes wikipedia article and a question file corresponding the wikipedia article and output the answers according to the provided questions. The architecture of our system is depicted in figure 1. The system is decomposed into the following modules: -

1) Question module
2) Answer module

The detailed description of these modules will be provided in section IV-A and IV-B.

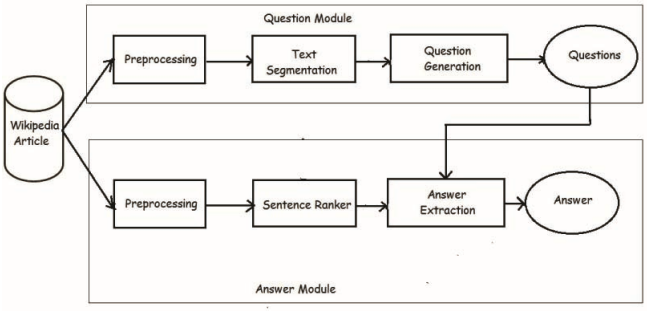

Fig. 1: System Architecture

A. Question module

The aim of question module is to generate questions from a wikipedia article. These questions then fed into answer module for extracting answers.The detailed description of components used in this module will be provided in IV-A1,IV-A2 and IV-A3.

1) Preprocessing: Preprocessing mainly aims to simplify the input in a manner that is suitable for the system to process. As we will be working with wikipedia article i.e. html file, we need to preprocess the file. A html file contains a lot of tags, citation, etc. Here, we focus mainly on paragraph tags because important information resides mainly in these tags. Using Beautiful Soup (a python package for parsing HTML and XML documents), we perform our preprocessing[8].

2) Text Segmentation: Text segmentation is the process of partitioning written text into meaningful units, for example, words, sentences, or topics. In this step, we perform following tasks, which make the base for creating questions.

a) Part-of-speech tagging: Part-of-speech tagging is one of the most prominent text analysis tasks is used to classify words into their part-of-speech and label them according to the tag set, where tag set is a collection of tags used for the Part-of-speech tagging (pos-tagging). These tags, in turn, can be used as features for higher level tasks, for example, building parse trees, which can in turn be used for Named Entity Resolution and Question Answering[17].

b) Named-entity recognition (NER): Named-entity recognition (NER) is a subtask of information extraction that tries to find and arrange named entities in text into predefined categories, for example, the names of persons, organizations, locations, etc [18]. For NER tagging, we will use the Stanford NER tagger which provides computationally inexpensive options than provided by Natural Language Toolkit (NLTK). In information extraction, a named entity is a real-world object, for example, persons, locations, organizations, products, etc., that can be denoted by a proper name. It tends to be abstract or have a physical existence.

c) Parsing: Parsing is done by parser. A natural language parser is a program which deals with the grammatical structure of sentences, for example, which groups of words go together

(as 'phrases') and which words are the subject or object of a verb. Parser generates parse tree[19] which represents the syntactic structure of a string according to some context-free grammar(CFG)[20]. In our system, we will use Stanford Parser[7], which provide greater benefits than *NLTK*.

3) Question Generation: This is the main part of Question module. Here, we intend to generate binary (i.e. Yes/No) questions and Wh-questions using some grammars. The concepts behind generations of these types of questions are discussed below.

1)Yes/no: Yes/No question is also named as binary question. To generate binary question, we first need to determine the first word of question such as 'do','did','does' etc.. Using *NLTK* POS tagging, we can find these helping verbs(MD) and can adjust the string accordingly. So as to create questions beginning with 'do','did' or 'does', we first need to check if the sentence begins with either a preposition, a singular proper noun, or a plural proper noun, then we expel '-ed'/'-s' for a VBD[past tense verb]/VBZ [3rd person singular present tense verb] and substitute the period with a question mark.

input:
[Jack's folks constantly headed out to Sweden for excursion., David Beckham has got a nice car.]
output:
[Did Jack's folks constantly headed out to Sweden for excursion?, Has David Beckham got a nice car?]

2)Who/what: Having subject in a sentence is a prerequisite to generate who/what question. Each word of each sentence need to be tagged using tagger to find the noun/pronoun. To generate question from a sentence having noun labeled as 'NNP'[Proper noun], we simply replace the 'NNP' with 'who' and substitute the period with a question mark.

input:
[Faisal is a student., He should go to office.]
output:
[Who is a student?, Who should go to office?]

To generate 'What' question, same method is used.

3)When/where: When/where question can be generated based on yes/no question. In this case, we need to convert the sentence into a yes/no question first, then identify named entity such as PREP [preposition], TIME, LOCATION, ORGANIZATION etc using Stanford NERtagger and POStagger. Then, to generate question,we simply substitute PREP + TIME with 'when', PREP + LOCATION/ORGANIZATION with 'when' and move the substitution to the front.

input:
[Jamil worked at HSBC in London after he graduated from Stamford University.]
output:
[Where did Jamil work after he graduated from Stamford University?]
[When did Jamil work at HSBC in London?]

4)Why: Why question can be generated when a sentence contains subordinating conjunctions such as 'because', 'since', 'so' and 'due to'. The 'reason' or 'consequence' of a sentence is determined by seeing which subordinating conjunction contained in that sentence. Then, question is generated by using the format - "Why (consequence)?" and altering the word orders in the (consequence) to make it grammatically correct.

input:
[Beckham attended church every week with his parents, because that was the only way he could play football for their team.]
output:
Here, in the sentence, reason is: "because that was the only way he could play football for their team."
and consequence is : "Beckham attended church every week with his parents." So the question is:
[Why did he attend church every week with his parents?]

5)How many: How many question can be generated when a sentence contains a quantity such as cardinal number: one, two etc. and structure like "Cardinal Number (CD) + Noun (NN/NNS/NNP/NNPS)". Then, question is generated by using the format - "How many + Noun + (the rest of the sentence reordered)?"

input:
[Orion contains seven brightest stars.]
output:
[How many brightest stars does orion contain?]

B. Answer module

Purpose of answer module is to extract answers from the article for the generated questions from question module. The best answer should be correct, concise and fluent. The detailed description of components used in this module will be provided in IV-B1,IV-B2 and IV-B3.

1) Preprocessing: Alongside removing tags from article file as described in IV-A1, we have to perform following tasks in answer module -

a) Removing stop words: A stop word is a frequently used word such as 'the', 'a', 'an', 'in' etc., which are normally ignored by search engines in both for searching and retrieving result of a query [21]. We also would not want these words taking up space in our database, or taking up valuable processing time.

b) Stemming: Documents use different forms of word of grammatical reason, for example, organize, organizes, and organizing. Also, there are groups of derivationally related words having similar meanings, such as democracy, democratic, and democratization. In many cases, it seems as if it is well enough to search one of these words to return documents that contain another word in the set[22]. The aim of stemming is to decrease inflectional forms and derivationally related forms of a word to a common base form.

c) Term Frequency Vector: Term Frequency Vector, tf-idf, is a numerical statistic that is intended to reflect how valuable a word is to a document in a collection or corpus. It is usually used as a weighting factor in hunts of information retrieval, text mining, and user modeling. The value of tf-idf increases proportionally to the number of times a word appears in the document and is offset by the frequency of the word in the corpus[23].

2) Sentence Ranker: Based on the term frequency vector, Sentence Ranker finds a list of ranked sentences from which we can find the best ranked sentence that will yield our expected answer.

3) Answer Extraction: Using this component, we will extract answers for questions those are generated in question module. For answer extraction purpose, we may need to perform parsing, Named-entity recognition (NER) or call Sentence Ranker component.

1)Yes/no: The answer of yes/no or binary question is simply Yes or No. To extract the answer, we first need to parse the question as described in IV-A2; then, we need to find out the best ranked sentence as described in IV-B2. Depending on the presence of negation words such as 'not', 'no', 'never' in best ranked sentence but not in the question; we determine the answer as 'No'; otherwise 'Yes'.

input:
[Has David Beckham got a nice car?, Did George Cukor direct the movie?]
output:
[Yes, No]

2)Who/what: Who/what question generated based on 'NNP'[proper noun] tag. So, we first need to parse the question and find out the corresponding types of phrases containing 'NNP'[proper noun] tag, which implies the correct answer. When we found more than one candidate answer, we performed Named-entity recognition (NER) as described in IV-A2 and considere the named entity 'PERSON' to find the answer.

input:
[Who was a Manchester United mascot for a match against West Ham United in 1986?]
output:*[Beckham.]*

For extracting answer from 'What' question, same method is used.

3)When/where: When question generated based on TIME/DATE named entity and Where question generated based on LOCATION/ORGANIZATION named entity. So, we first need to parse the question and find out the best ranked sentence. Then, we utilize Stanford NERtagger and POStagger to find out the named entity, which implies the correct answer.

input:
[Where did Jamil work after he graduated from Stamford University?]
output:
The best ranked sentence is "Jamil worked at HSBC in London after he graduated from Stamford University." Thus, answer is, *[At HSBC in London.]*

4)Why: Why question is generated with subordinating conjunctions. Depending on which subordinating conjunction we see, we need to determine the 'reason' and the 'consequence'. The reason token is the answer of "why" question.

input: *[Why did beckham attend church every week with his parents?]*
output: Here, reason is: "because that was the only way he could play football for their team."
consequence is : "Beckham attended church every week with his parents"
[Because that was the only way he could play football for their team.]

5)How many: How many question is generated when a sentence contained a quantity such as cardinal number. Answer of the how many question is a sentence which contains "CD (cardinal number:one,two ...)" in its structure. So, we need to parse the question and find out the best ranked sentence which has the structure "CD (cardinal number:one,two ...) + NN(singular noun)/NNS(plural noun)/NNP(proper noun)/NNPS(plural proper noun) + (the rest of the sentence)".

input:
[How many brightest stars does orion contain?]
output:
[Orion contains seven brightest stars.]

V. EXPERIMENTAL RESULT

In order to evaluate our system, we treated a collection of wikipedia articles as our dataset and implemented the process described in section IV. We can either save the article file in our directory or directly work with the article.

The question module of the system takes a wikipedia article(figure 2) and a number as input and generates a list of random questions like as follows -

Where was Beckham born at Whipps Cross University Hospital?
Where was Beckham a Manchester United mascot for a match?
When was he captain for six years , earning 58 caps?
When was beckham a Manchester United mascot for a match against West Ham United?
When did beckham sign a five-year contract with Major League Soccer club LA Galaxy?

Fig. 2: Question-Answer extraction from Wikipedia

And answer module takes wikipedia article and a question file corresponding this wikipedia article and output the answers according to the given questions as bellow -

In Leytonstone , London , England.
Against West Ham United in 1986.
He was captain for six years, earning 58 caps during his tenure.
In 1986.
In July 2007.

Note that generated questions may only focus on a special point of a certain fact, e.g. "When did beckham sign a five-year contract with Major League Soccer club LA Galaxy?". In this case, if the answers reported by our system can obviously show the aspect, like telling when beckham sign a five-year contract, we say that we correctly answer this question.

When target sentence does not contain answer, it will output wrong answer or just give the target sentence. Using True Success Rate (True) as a performance metric, we get 85% accurate result for our generated questions. The ability of this system can enhance by adding more rules/patterns.

VI. CONCLUSIONS

In this research work, we have developed a system that can generate questions from a wikipedia article and can provide answers to those questions. We applied, a list of grammars on the article to find our targeted question type, make the question and extract answers to these questions. There can be many, many combinations of questions and answers, so we are willing to apply more grammars in our system that will give our system a huge performance boost in future work.

REFERENCES

[1] Bassam Hammo, Hani, Abu-Salem and Steven Lytinen, "QARAB: A Question Answering System to Support the Arabic Language", 2002

[2] Yoji Kiyota, Sadao Kurohashi, Fuyuko Kido, "Dialog Navigator: A Question Answering System based on Large Text Knowledge Base", 2002

[3] Fabio Rinaldi, James Dowdall, Gerold Schneider, Andreas Persidis, "Answering Questions in the Genomics Domain", 2004

[4] Paulo Quaresma, Irene Rodrigues, "A question-answering system for Portuguese juridical documents", 2005

[5] Wilson Wong, John Thangarajah, and Lin Padgham, "Contextual Question Answering for the Health Domain", 2012

[6] Sindhu L Unmesh Sasikumar. "A survey of natural language question answering system", 2014.

[7] "Stanford-parser".[Online].Available:http://nlp.stanford.edu/software/lex-parser.shtml.[Accessed: 2018-07-26]

[8] "Beautiful-soup".[Online]. Available: http://www.crummy.com/software/BeautifulSoup/bs4/doc/.[Accessed: 2018-07-15]

[9] "Named-entity".[Online].Available: http://en.wikipedia.org/wiki/Named_entity.[Accessed:2018-07-15]

[10] Green BF, Wolf AK, Chomsky C, and Laughery K,"Baseball: An automatic question answerer", in Proceedings of Western Computing Conference, Vol. 19, 1961, pp. 219–224.

[11] Katz B.,"Annotating the World Wide Web using natural language", in Proceedings of the 5th RIAO conference on Computer Assisted Information Searching on the Internet, 1997, pp. 136-159.

[12] Chung H, Song YI, Han KS, Yoon DS, Lee JY, and Rim HC, "A practical QA System in Restricted Domains", in Workshop on Question Answering in Restricted Domains, 42nd Annual Meeting of the Association for Computational Linguistics (ACL), 2004, pp. 39-45.

[13] Cai D, Dong Y, Lv D, Zhang G, Miao X, "A Web-based Chinese question answering with answer validation", in Proceedings of IEEE International Conference on Natural Language Processing and Knowledge Engineering, pp. 499-502, 2005.

[14] "Natural Language Toolkit".[Online].Available: https://www.nltk.org/.[Accessed:2018-07-15]

[15] "Question-answering-challenges".[Online].Available:http://realkm.com/2017/11/03/developments- challenges- and- trends- in- question-answering- systems/.[Accessed: 2018-07-15]

[16] Edgard Marx Ricardo Usbeck Jens Lehmann Axel-Cyrille Ngonga Ngomo Konarad Hoffner, Sebastian Walter, "Survey on challenges of question answering in the semantic web", 2016.

[17] "Part-Of-Speech Tagging and POS Tagger".[Online]. Available:http://textminingonline.com/dive-into-nltk-part-iii-part-of-speech-tagging-and-pos-tagger. [Accessed: 2018-07-15]

[18] "Named-entity-recognition".[Online].Available:http://en.wikipedia.org/wiki/Named-entity_recognition.[Accessed: 2018-07-15]

[19] "Parsetree".[Online].Available:https://www.quora.com/What-is-a-parse-tree-in-nlp-and-for-what-is-it-used.[Accessed: 2018-07-15]

[20] Mats Rooth. "A Tree Syntax of Natural Language"

[21] "Stop words". [Online]. Available:https://www.geeksforgeeks.org/removing-stop-words-nltk-python/.[Accessed: 2018-07-26]

[22] "Stemming".[Online].Available:https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html.[Accessed: 2018-07-26]

[23] "Term-Frequency-Vector".[Online].Available:https://en.wikipedia.org/wiki/Tf%E2%80%93idf.[Accessed: 2018-07-26]