RealText$_{asg}$: A Model to Present Answers Utilizing the Linguistic Structure of Source Question

Rivindu Perera, Parma Nand
School of Computer and Mathematical Sciences
Auckland University of Technology
Auckland, New Zealand
{rivindu.perera, parma.nand}@aut.ac.nz

Abstract

Recent trends in Question Answering (QA) have led to numerous studies focusing on presenting answers in a form which closely resembles a human generated answer. These studies have used a range of techniques which use the structure of knowledge, generic linguistic structures and template based approaches to construct answers as close as possible to a human generate answer, referred to as human competitive answers. This paper reports the results of an empirical study which uses the linguistic structure of the source question as the basis for a human competitive answer. We propose a typed dependency based approach to generate an answer sentence where linguistic structure of the question is transformed and realized into a sentence containing the answer. We employ the factoid questions from QALD-2 training question set to extract typed dependency patterns based on the root of the parse tree. Using identified patterns we generate a rule set which is used to generate a natural language sentence containing the answer extracted from a knowledge source, realized into a linguistically correct sentence. The evaluation of the approach is performed using QALD-2 testing factoid questions sets with a 78.84% accuracy. The top-10 patterns extracted from training dataset were able to cover 69.19% of test questions.

1 Introduction

Question Answering (QA) comprises of four main tasks; question processing, answer search, answer extraction, and answer presentation. The first three tasks focus on extracting the answer while the last aims to present the extracted answer in a human-like format. With the rise of trend towards building human-competitive QA systems, there been a corresponding demand for the extracted to be presented in a human competitive form rather than the bare answer as a single word or a phrase. A wide range of answer presentation schemes have been reported including user tailored answers (Mendes and Coheur, 2013; Maybury, 2008; Kolomiyets and Moens, 2011; Perera and Nand, 2014a), justification based answers (Mendes and Coheur, 2013; Maybury, 2008; McGuinness, 2004; Saint-Dizier and Moens, 2011), presentation of paragraph level text summaries with the extracted answer (Mendes and Coheur, 2013; Lin et al., 2003; Perera, 2012b; Perera, 2012a), presentation of hot links with answers (McGuinness, 2004), and presentation of navigable related answers and contextual information (Saint-Dizier and Moens, 2011; Perera and Nand, 2015a; Perera and Nand, 2014b; Perera and Nand, 2014c). All of the mentioned models aim to build an answer which closely resembles a human generated answer. However, an approach that has not been explored in the mentioned models is to exploit the structure of the question in the formulation of the answer. A human generated answer is based both on the answer structure as well as how the question was formulated (Singer, 2013). For example, given the question "Which river does the Brookyln Bride cross?", the expected answer sentence would be of the form of "The Brookyln Bridge crosses East River".

It is essential to understand the types of questions and their linguistic structure in order to successfully generate a sentence with the answer embedded in it. The questions can be divided in to two main categories based on their interrogative categories; wh-interrogative and polar interrogatives. A wh-interrogative is aimed at getting an answer which represents another entity or a property of a resource mentioned in the question, on the other hand a polar interrogative requests a true/false (yes/no) answer. These two types require two different answer sentence generation schemes; wh-interrogatives require to embed the answer to the modified source question linguistic structure and the polar interrogatives need to transform the same question without further embedding, however it still needs modification based on the answer. Table 1 shows the interrogative types with examples and Part-Of-Speech (POS) tags associated with them and the expected answer sentences.

This paper focuses on answer sentence generation based on typed dependency parsing. To the best of our knowledge, no previous study has investigated this method of generating an answer sentence utilizing the source question's linguistic structure. The methodology we introduce here is based on linguistics. The core idea is that the generation of an answer sentence is initiated by identifying the root of the parse tree and then proceed to build the sentence using the nominal subject, a key feature of a Subject-Verb-Object (SVO) style language such as English. In order to identify the grammatical relation that holds the parts of question with the root, we employ typed dependency parse of the complete question. The typed dependency based patterns extracted using a training dataset are used to construct the framework. Answer merging and further realization of the sentence are implemented in order to provide a human-like natural language answer sentence. The complete framework (implemented in Java) and the datasets are available for download from the project website[1].

The remaining part of the paper proceeds as follows. Section 2 introduces the methodology of answer sentence generation. We discuss the process under four main themes; extracting syntactic patterns, applying patterns to new questions, answer merging, and further realization. Section 3 describes the experimental framework including the results. A discussion on related work which investigates different answer presentation methods in natural language is presented in Section 4. Section 5 concludes the paper with an overview of future work.

2 Answer Sentence Generation

In this section we explain the Answer Sentence Generation (ASG) process which has a pipeline architecture as shown in Fig.1. The process is comprised of three main components; pattern processing (pattern extraction and application), answer merging, and sentence realization. The pattern processing component is responsible of deriving typed dependency based patterns to transform a question back into a natural sentence. It is also responsible for identifying and applying the appropriate pattern based on the typed dependency parse of a question. The answer merging module embeds the answer in wh-interrogatives preserving the naturalness of the sentence. The sentence realization module applies grammar based realization if needed and further realizes the answer sentence.

2.1 Pattern extraction

The pattern identification process first identifies the interrogative type of the question. We employed the Stanford parser[2] to parse the question and to identify the POS tag of a wh-determiner. However, POS tagging itself cannot be used to classify questions because of two reasons. Firstly, a sentence can be formed using an embedded interrogative such as "I wonder what he likes to eat for the dinner" or "Do only what is assigned to you". In aforementioned examples, the former is an embedded interrogative to explain the speaker's perspective and the latter is a command, however both cannot be considered as questions. The second reason is that when forming a sentence using relative clauses (both restrictive and non-restrictive) the joining token is also POS tagged as wh-determiner. For instance, the sentence "Chess is a good game that is interesting too" contains the wh-determiner (*WRB*) POS tag (based on Penn Treebank guidelines) associated with token "*that*". Due to these factors, we consider three features, the POS

[1] `http://rivinduperera.com/information/realtext.html`

[2] `http://nlp.stanford.edu/software/lex-parser.shtml`

Table 1: Interrogative types with examples and associated POS tags. POS tags are compliant with the Penn Treebank guidelines.

	Wh-interrogative	Polar interrogative
Interrogative tokens	Who, What, Where, Which, When, How	Is, Are, Was, Were
POS tags	WP, WRB ,WDT	VBZ, VBP, VBD
Question - 1	Which river does the Brooklyn Bridge cross?	Was Natalie Portman born in the United States?
Answer	East River	Fasle/No
Answer sentence	The Brookyln Bridge crosses East River.	Natalie Portman was not born in the United States.
Question - 2	How many films did Hal Roach produce?	Is Cristian Bale starring in Batman Begins?
Answer	509	True/Yes
Answer sentence	Hal Roach produced 509 films.	Cristian Bale is starring in Batman Begins.

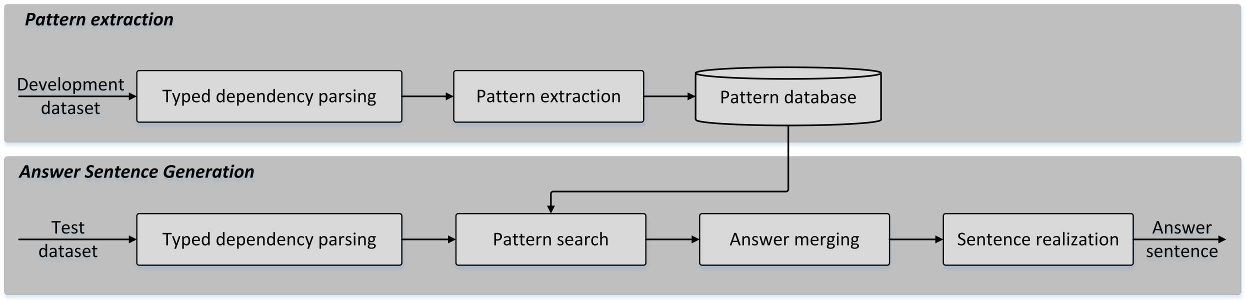


Figure 1: Schematic representation of the overall Answer Sentence Generation (ASG) process

tag, the relative position in the sentence, and whether they belong to the wh-lexicons (who, when, what, etc.).

Once the interrogative type is identified, the patterns can be extracted using a development question set. Each pattern is a collection of first level typed dependency relations as a directed graph based on the root node of the parse tree whose nodes are generic. The order of typed dependency relations are not significant as linguistic structure may vary based on the formation of the question. Table 2 shows some patterns and with example questions.

Each extracted pattern is associated with rule sets to generate a sentence. These rules specify how nominal subject, direct object, clausal complements, and other typed dependency relations should be aggregated to form a natural sentence.

2.2 Identifying and applying patterns

Once the pattern database is built, an appropriate pattern for a given question can be retrieved by analysing the typed dependency parse of that question. In the identification process, the order of the relations in the typed dependency pattern is insignificant as we only consider the relations from the root node to another generic node.

When applying the pattern, the parse tree is transformed to a list of phrases based on the root-based relations. Also more importantly, the phrase is generated based on the order of appearance of tokens in the source question. An example scenario of phrase extraction is shown in Table 3. These phrases are then aggregated based on the rules specified for each pattern.

Table 2: Syntactic patterns extracted from Typed dependency relations. The pattern is derived from the typed dependencies from the root token. The sign *X* represents a slot which can be replaced with a single or multiple tokens even if there exist typed dependency relations among those multiple tokens. The sign *R* represents the root token of the parse tree.

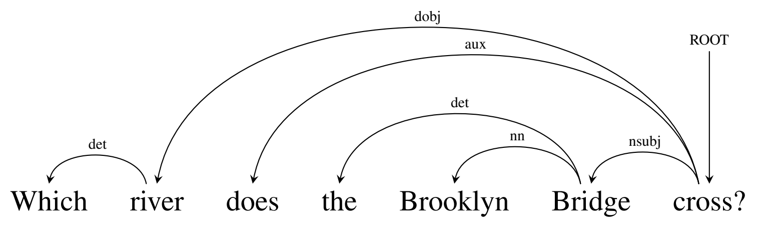
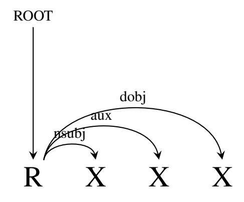
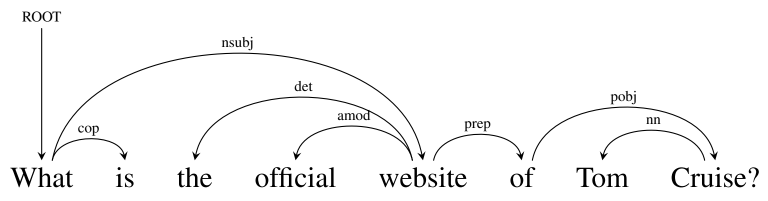
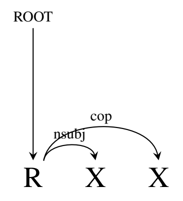
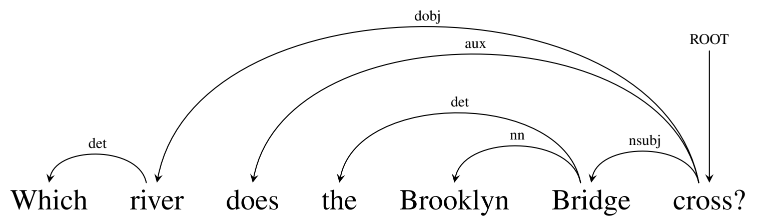
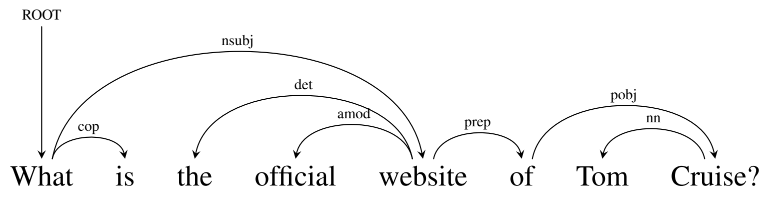
Type dependency	Extracted pattern
 dobj aux ROOT det det nn nsubj Which river does the Brooklyn Bridge cross?	
 ROOT nsubj det amod cop prep pobj nn What is the official website of Tom Cruise?	

Table 3: Phrase extraction from typed dependencies

Type dependency	Extracted phrases
 dobj aux ROOT det det nn nsubj Which river does the Brooklyn Bridge cross?	(i) Which river (ii) does (iii) the Brooklyn Bridge (iv) cross
 ROOT nsubj det amod cop prep pobj nn What is the official website of Tom Cruise?	(i) What (ii) is (iii) the official website of Tom Cruise

2.3 Answer merging

It is also required to embed the answer to the syntactic structure when the pattern has been identified to transform the question back into natural language sentence. In wh-interrogatives this require embedding another language segment, however for polar interrogatives this component should target on modifying the polar token based on the answer.

For wh-interrogatives, we have designed the model to embed the answer based on the type of the wh-token. This model is depicted in Table 4 for six different wh-tokens. It is also important to note that the wh-token "*why*" is not considered, since the current paper focuses only on factoid questions (e.g., how tall is Claudia Schiffer?) and definitional questions (e.g., why the sky is blue?) which often start with wh-token "*why*" are out of the current scope of the paper. The main reason for this elimination is that definitional questions does not require answer sentences as answer is a explanation.

Table 4: Answer merging schema for wh-interrogatives. Existing preposition is a one that is already appeared in the phrase and new prepositions are added based on the answer.

Wh-token	Merging schema	Example phrases	Merged answer
Which	Existing preposition + Answer	in which country	in New Zealand
What	Existing preposition + Answer	for What city	for London
Whom	Existing preposition + Answer	for whom	for Barack Obama
How	Naturalized answer (once/ twice/ thrice)	how often	twice
	Answer + Rest of the Phrase	how many films	509 films
When	New preposition + Answer	When	in 1990
Where	New preposition + Answer	Where	in New York

In addition to the answer merging schema shown in Table 4, the model also embeds measurement units and converts numbers to words. For example, if a number has appeared as the first word of an answer sentence, they are converted to a lexical form (e.g., 31 ⇒ Thirty one). If a question requires the height of an entity (e.g.,person or mountain) as the answer then appropriate measurement unit is added which is extracted from the knowledge source utilized for the answer. However, the query built to extract the answer needs to be analysed in order to determine whether answer requires a measurement unit. The queries generated in QA systems highly depend on the answer extraction source. For example, a QA system which utilizes a database will employ SQL as the query language and transforming the natural language question into a SQL query will be a major task for the query processing module of the QA system. The experiments described in this paper utilizes a Linked Data resource, DBpedia, as the answer extraction data resource (more information on this selection is described in Section 3.1). Use of DBpedia as the data resource required us to implement a SPARQL[3] (*S*PARQL *P*rotocol and *R*DF *Q*uery *L*anguage) processing module. In particular, we used the Jena[4] to parse the SPARQL query and identify queried predicate from the SPARQL. The module then searches the queried predicate in a local lexicon database (this is built as a different task in this research (Perera et al., 2015; Perera and Nand, 2015b; Perera and Nand, 2015c)) to identify whether it is associated with a measurement unit. Listing 1 depicts an example scenario of identifying the measurement unit associated with *height* ontology property of DBpedia.

[3]SPARQL is the query language used for Linked Data

[4]`https://jena.apache.org/`

2.4 Sentence realization

The sentence realization is based on a linguistic realization module which can further realize the answer sentence. However, by this stage, the answer sentence is nearly built except for the verb inflections. Therefore, this modules focuses on realization of periphrastic tense in occasions where the verb can be inflected without compromising the semantics (e.g., does cross ⇒ crosses). Also more importantly the formation of active voice based questions (identified using POS tagging) often requires periphrastic tense embedded in the question (e.g., Which river *does* the Brooklyn Bridge *cross*? ⇒ does cross ⇒ crosses). We used a specially built verb information database to identify different inflections of verbs. This database was built using VerbNet and contains 3773 records where each corresponds to a unique verb. However, VerbNet does not provided verb inflections. Therefore, we reverse engineered the Porter stemming algorithm (Porter, 1980) to generate the verb inflections.

An example set of records of this database is shown in Table 5.

3 Experimental framework

This section explains the experimental framework used to evaluate the answer sentence generation process. Due to absence of a method that can be directly compared to, we report the experiments in various dimensions; syntactic accuracy, execution time, and memory requirements. The last two factors has been added to the evaluation as QA has now moved closer

```
PREFIX res: <http://dbpedia.org/resource/>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT DISTINCT ?height
WHERE {
res:Claudia_Schiffer dbo:height ?height .
}

⇒ ?height ⇒ dbo:height ⇒ meters(m)
```

Listing 1: An example scenario of identifying the measurement unit associated with queried predicate by parsing the SPARQL query

Table 5: An example set of records from verb information database. (Base = Base form of the verb, PT = Past tense, PP = Past participle, TPS = Third person singular, Frames = aggregation of all frames found for the verb in VerbNet)

Base	PT	PP	ING	TPS	Frames
abandon	abandoned	abandoned	abandoning	abandons	NP V NP.initial_location
abase	abased	abased	abasing	abases	NP V NP.patient, NP V ADV-Middle, NP V NP PP.instrument
abash	abashed	abashed	abashing	abases	NP.cause V NP, NP V NP, NP V ADV-Middle, NP V NP-PRO-ARB, NP V NP ADJ

to achieving the long-held illusive goal of accuracy, hence we have now started to also look at real-time performance and computational efficiency.

3.1 Datasets

The experiments were carried out using factoid questions extracted from QALD-2 datasets. The QALD-2 training dataset is used to extract typed dependency patterns (as the development dataset) and the testing dataset is used to evaluate the framework.

The statistics related to the dataset is summarized in Table 6. The training set contained a record "Give me the homepage of Forbes". This record does not form a linguistic representation of any type of interrogative question and therefore could not be considered for pattern extraction.

3.2 Results and discussion

The evaluation of the framework focused on two aspects; the syntactic and semantic accuracy of the approach, and the memory and processing requirement. The latter was employed to identify the viability of the methodology for real-time systems.

We were able to identify 18 distinct wh-interrogative patterns and 7 polar interrogative patterns. However, based on the syntactic structure identified, these patterns can be generalized under certain interrogative patterns. Throughout this study we will be referring to the extended list of patterns (18 wh-interrogative and 7 polar interrogative patterns). Using these patterns, answer sentences were generated for the testing dataset with a 78.84% accuracy. Except for 11 questions where the framework completely failed to generate answer sentences, all others were syntactically and semantically accurate. These 11 questions include 5 wh-interrogatives and 6 polar interrogatives. The framework failed to generate answer sentences for these questions mainly due to the absence of rules (for 10 questions) and

Table 6: Statistics related to the question sets

	Total	Factoid	Wh-interrogatives	Polar interrogatives
Training set	100	50	41	8
Testing set	99	52	43	9

the errors in the typed dependency parse (for 1 question). Table 7 shows a selected set of generated answer sentences together with questions.

It is also important to analyse the coverage provided for the test dataset by the extracted patterns from training dataset. Fig. 2 shows the number of testing dataset cases covered by top-k patterns extracted from training dataset. The top-10 patterns were able successfully cover 69.19% of the questions from the testing dataset. Furthermore, the coverage of 51.91% of the questions through top-4 patterns shows that the top patterns are highly representative.

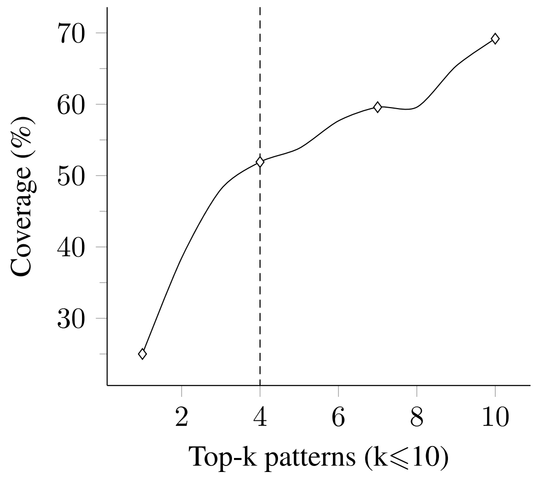
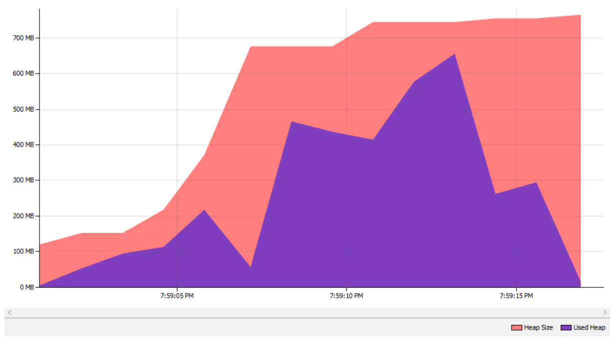


Figure 2: The coverage provided for the test dataset by the extracted patterns from training dataset. The coverage is depicted as a percentage of questions transformed accurately by top-k patterns which are extracted from training dataset. The dashed line shows the point where 50% coverage is reached.

Another aspect of the evaluation of the framework was to analyse the memory and execution analysis. Fig.3(a) and Fig.3(b) depicts the heap memory allocation for the answer sentence generation process and execution time analysis respectively. The remarkably high heap memory allocation (more than 600 Mb) indicates that the current architecture needs further work to work in limited-memory environments. However, the execution time analysis shows that the framework can be adapted to real-time systems.



(a) Heap memory allocation

Typed dependency parsing	2539ms	96.81%
Interrogative type identification	1.82ms	0.06%
Wh-interrogatives	80.1ms	3.04%
Rule identification	2.5ms	0.09%
Rule application	77.6ms	2.95%
Polar interrogatives	1.7ms	0.06%
Rule identification	0.208ms	0.01%
Rule application	1.492ms	0.05%

(b) Execution time for key components

Figure 3: Memory and execution analysis for ASG process. The analysis is performed for test dataset which contains 52 factoid question.

4 Related work

A considerable amount of literature has been published on answer presentation. However, to the best of our knowledge answer sentence generation is not studied in any present model. Therefore, this section provides a broader review of answer presentation.

Benamara and Dizier (Benamara and Dizier, 2003) present the cooperative question answering approach which generates natural language responses for given questions. The idea of cooperative question answering dates back to 1986 with the invention of cooperative interface for information systems. Then several researchers involved the cooperative interfaces for QA systems. In essence, a cooperative QA system moves a few steps further from ordinary question answering systems by providing an explanation of the answer, describing if the system is unable to find an answer or by providing links to the user to get more information for the given question. However, the development of this description is entirely based on the external knowledge source, and the linguistic structure is not considered to present the answer.

A successful attempt to move beyond the ex-

Table 7: Selected set of generated answer sentences together with questions

QALD Id	Question	Answer sentence
2	Who was the successor of of John F. Kennedy?	The successor of John F. Kennedy was Lyndon B. Johnson.
4	How many students does the Free University in Amsterdam have?	The Free University in Amsterdam has 22730 students.
20	How tall is Michael Jordan?	Michael Jordan is 1.9812m tall.
53	What is the ruling party in Lisbon?	The ruling party in Lisbon is Socialist Party (Portugal).
78	Was Margaret Thatcher a chemist?	Margaret Thatcher was a chemist.

act answer by presenting users with additional information in sentence form is presented by Bosma (Bosma, 2005) utilizing summarization techniques. In this research Bosma (Bosma, 2005) assumes that a QA system has already extracted a sentence that contains the exact answer. Then based on this candidate sentence, Bosma (Bosma, 2005) tries to generate an answer response by utilizing information acquired from a collection of sentences. He coins the term, an intensive answer to refer to the answer generated from the system. The process of generating intensive answer is based on summarization using rhetorical structures.

Another answer presentation method based on summarization is presented by Demner-Fushman and Lin (Demner-Fushman and Lin, 2006). They introduce the concept of extractive summaries to present with extracted answer. This model has some similarity with the one presented by Bosma (Bosma, 2005), but like in Bosmas model, Demner-Fushman and Lin (Demner-Fushman and Lin, 2006) do not make use of specifically generated weighted graph to identify relevant sentences to include in the summary. Instead Demner-Fushman and Lin (Demner-Fushman and Lin, 2006) generates the answer by aggregating the top three ranked sentences by a regression model.

Vargas-Vera and Motta (Vargas-Vera and Motta, 2004) present an ontology based QA system, AQUA. Although AQUA is primarily aimed at extracting answers from a given ontology, it also contributes to answer presentation by providing an enriched answer. The AQUA system extracts ontology concepts from the entities mentioned in the question and present those concepts in aggregated natural language. However, the research does not contribute towards identifying the most appropriate context for the given question and answer. In addition, no specific effort is taken to identify how the aggregated concepts need to presented. However, the benefit that researchers achieved by building the enriching module on top of an ontology is that the related information can be easily acquired using the relations in the ontology.

5 Conclusion and future work

The purpose of the current study was to build a framework to generate answer sentences using the linguistic structure of the question with embedded answer. At the core of the framework, we used typed dependency based patterns extraction and the generated sentence was further realized through multiple strategies. We designed the experiment to determine the accuracy and as well as the resource requirements. The results confirmed that the approach can be successfully applied in QA systems with a reasonable level of accuracy. However, the study found that the framework needs further work to extend it to the mobile platforms as the current architecture consumes considerable memory during execution, mainly due to loading of the pre-trained dependency parser model. In addition, the current study focused only on grammatical accuracy. We plan to extend the evaluation to consider other factors of answer sentences including suitability and cohesion. In addition to aforementioned experimental investigations, further studies need to be carried out to enrich the proposed strategy as well as combine it with other possible strategies to further advance the answer presentation.

References

Farah Benamara and Patrick Saint Dizier. 2003. Dynamic Generation of Cooperative Natural Language Responses in WEBCOOP. In *9th European Workshop on Natural Language Generation (ENLG-2003) at EACL 2003*, Budapest, Hungary. Association for Computational Linguistics.

Wauter Bosma. 2005. Extending answers using discourse structure. In *Recent Advances in Natural Language Processing*, Borovets, Bulgaria. Association for Computational Linguistics.

Dina Demner-Fushman and Jimmy Lin. 2006. Answer extraction, semantic clustering, and extractive summarization for clinical question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL - ACL '06*, pages 841–848, Morristown, NJ, USA, July. Association for Computational Linguistics.

Oleksandr Kolomiyets and Marie-Francine Moens. 2011. A survey on question answering technology from an information retrieval perspective. *Information Sciences*, 181(24):5412–5434, December.

Jimmy Lin, Dennis Quan, Vineet Sinha, Karun Bakshi, David Huynh, Boris Katz, and David R. Karger. 2003. What Makes a Good Answer? The Role of Context in Question Answering. In *Interact*.

Mark Maybury. 2008. New Directions In Question Answering. In Tomek Strzalkowski and Sanda M. Harabagiu, editors, *Advances in Open Domain Question Answering*, volume 32 of *Text, Speech and Language Technology*, chapter New Direct. Springer Netherlands, Dordrecht.

D L McGuinness. 2004. Question answering on the semantic Web.

Ana Christina Mendes and Luisa Coheur. 2013. When the answer comes into question in question-answering: survey and open issues. *Natural Language Engineering*, 19(01):1–32, January.

Rivindu Perera and Parma Nand. 2014a. Interaction history based answer formulation for question answering. In *International Conference on Knowledge Engineering and Semantic Web (KESW)*, pages 128–139.

Rivindu Perera and Parma Nand. 2014b. Real text-cs - corpus based domain independent content selection model. In *IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 599–606.

Rivindu Perera and Parma Nand. 2014c. The role of linked data in content selection. In *Pacific Rim International Conference on Artificial Intelligence (PRICAI)*, pages 573–586.

Rivindu Perera and Parma Nand. 2015a. Answer presentation with contextual information: A case study using syntactic and semantic models. In *28th Australasian Joint Conference on Artificial Intelligence*.

Rivindu Perera and Parma Nand. 2015b. Generating lexicalization patterns for linked open data. In *Second Workshop on Natural Language Processing and Linked Open Data collocated with 10th Recent Advances in Natural Language Processing (RANLP)*, pages 2–5.

Rivindu Perera and Parma Nand. 2015c. A multi-strategy approach for lexicalizing linked open data. In *International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 348–363.

Rivindu Perera, Parma Nand, and Gisela Klette. 2015. Realtext-lex: A lexicalization framework for linked open data. In *14th International Semantic Web Conference*.

Rivindu Perera. 2012a. Ipedagogy: Question answering system based on web information clustering. In *IEEE Fourth International Conference on Technology for Education (T4E)*.

Rivindu Perera. 2012b. *Scholar: Cognitive Computing Approach for Question Answering*. Honours thesis, University of Westminster.

M.F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Patrick Saint-Dizier and Marie Francine Moens. 2011. Knowledge and reasoning for question answering: Research perspectives. *Information Processing and Management*, 47:899–906.

Murray Singer. 2013. *Psychology of Language: An Introduction to Sentence and Discourse Processes*. Psychology Press.

M Vargas-Vera and E Motta. 2004. AQUAontology-based question answering system. In *Mexican International Conference on Artificial Intelligence*, Mexico City, Mexico. Springer-Verlag.