Answer Presentation in Question Answering over Linked Data using Typed Dependency Subtree Patterns

Rivindu Perera and **Parma Nand**
School of Engineering, Computer and Mathematical Sciences
Auckland University of Technology
Auckland, New Zealand
{rperera, pnand}@aut.ac.nz

Abstract

In an era where highly accurate Question Answering (QA) systems are being built using complex Natural Language Processing (NLP) and Information Retrieval (IR) algorithms, presenting the acquired answer to the user akin to a human answer is also crucial. In this paper we present an answer presentation strategy by embedding the answer in a sentence which is developed by incorporating the linguistic structure of the source question extracted through typed dependency parsing. The evaluation using human participants proved that the methodology is human-competitive and can result in linguistically correct sentences for more that 70% of the test dataset acquired from QALD question dataset.

1 Introduction

In this research we focus on generating a sentence which formulates the answer as a natural language sentence and presents it in a more natural form. In particular, if we ask a question to a person, he/she has the ability to answer with a sentence or sentences which has the answer embedded in a context. This form of answering a question is more natural compared to the bare factoid answer delivered by most QA systems.

The rest of the paper is structured as follows. Section 2 discusses the framework that generates the answer sentences. Section 3 explains the experimental framework that evaluates the framework and the results. We also provide a detailed discussion on results in this section. Related work and comparison of our approach to existing work is discussed in Section 4. Section 5 concludes the paper with an overview of the future work.

2 RealText$_{asg}$ Framework

2.1 Architecture of the framework

We employed the typed dependency parsing (de Marneffe et al., 2014) to determine the linguistic structure of the source question. The core idea in this approach is to identify linguistic patterns based on the typed dependency patterns of source questions and implement answer merging and realization mechanisms for identified patterns. Therefore, new question and answer pairs can be realized to answer sentences using known patterns and by applying associated merging and realization mechanisms. Fig. 1 depicts the schematic representation of the answer sentence generation process. In following sections we first describe the question type identification process and then proceed to a detailed discussion on individual modules of the process.

2.2 Question type identification

Since the answer sentence generation process depends on the question type, it is vital to classify the questions based on the interrogative type before extracting the typed dependency patterns. As the current research concentrates on answer presentation which is the last step of the QA process, we exploited

Proceedings of the Open Knowledge Base and Question Answering (OKBQA) Workshop, pages 44–48, Osaka, Japan, December 11 2016.

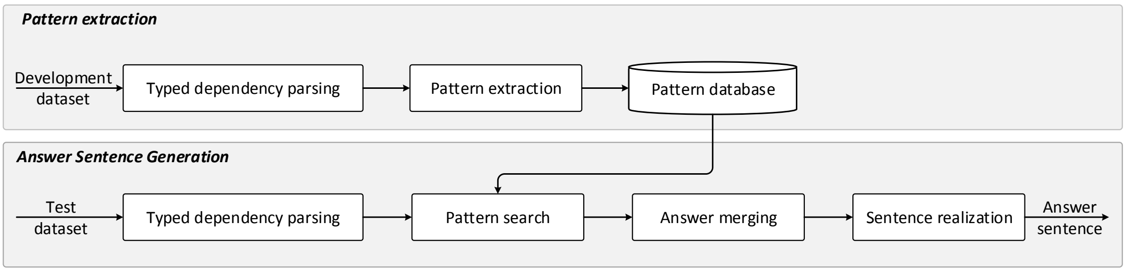


Figure 1: Schematic representation of the answer sentence generation process

both question and the query to classify the questions to the interrogative type. We first classified all questions which require boolean value answers as polar interrogatives. This classification considered both the query and the answer to ascertain that the question is seeking a boolean answer. The rest can be classified as wh-interrogative. However, to further validate this approach, the question text is POS tagged and analysed whether they contain the required POS tags.

It is also important to notice that in this research we do not consider imperative constructs. Imperative constructs are statement which request information such as “Give me information about Steve Jobs”. Although such statements still request information from the user, they do not utilize a linguistic structure of a question.

2.3 Dependency tree and pattern extraction

If a sentence (S) is thought as a sequence of words ($w_0 ... w_n$) in which the w_0 is considered as the root of the sentence, then a dependency tree is a directed tree originated from the w_0 and has the spanning node set V_S. This tree can also be thought as a well-formed graph ($G(V_S, A)$) in which A corresponds to the arcs ($A \subseteq V \times R \times V$) created based on a dependency relation set R. Since, w_0 is the root of the tree and dependency tree satisfies the root property (i.e., there does not exist $w_i \in V$ such that $w_i \to w_0$), w_0 connects the constituents of the tree. Furthermore, if we take a subtree originated from the root, then it can be taken as a phrase given that ordered based on the same subsequence the S is formed of. In essence, the patterns extracted in our approach are first level relations originating from the dependency tree root (w_0). Table 1 depicts some of the syntactic patterns extracted from the dependency tree. We substitute the sub-trees with generic token since their actual words or order of words is not important for patterns except that the relation type originated from the root.

The extracted patterns are preserved as a collection of relations from the root node. In the next section we describe the process of searching for a matching pattern and applying pattern using the specific pattern oriented function.

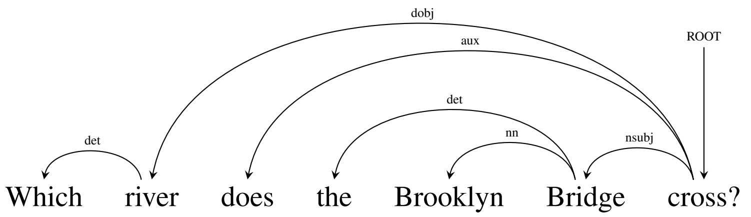
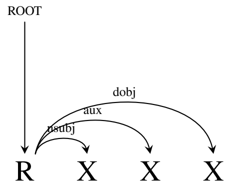
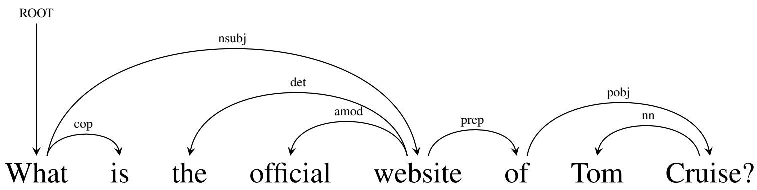
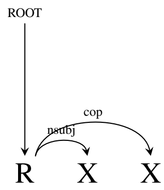
2.4 Pattern search and application

For each of the extracted pattern in Section 2.3, a specific function is defined with the rule set which defines the order of appearance of the dependency relations in a realized sentence. Once a new question is provided, it is first dependency-parsed and the relations from the root node are extracted. Then the matching pattern is identified and the sub-trees in the question are transformed into phrases associating them with the relation type.

2.5 Answer merging and sentence realization

In wh-interrogatives, answer merging process requires embedding another language segment, however for polar interrogatives this component should target on modifying the polar token based on the answer. The model also embeds measurement units and converts numbers to words. We used the Jena (McBride, 2002) 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, 2015a; Perera and Nand, 2015b)) to identify whether it is associated

Table 1: 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
	
	

```
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

with a measurement unit. Listing 1 depicts an example scenario of identifying the measurement unit associated with *height* ontology property of DBpedia.

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 module focuses on realization of periphrastic tense in occasions where the verb can be inflected without compromising the semantics (e.g., does cross ⇒ crosses).

3 Evaluation and results

We were able to identify 18 distinct wh-interrogative patterns 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 the errors in the typed dependency parse (for 1 question).

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. We also carried out a human evaluation using three postgraduate students chosen on the basis of having acceptable level of competency in English. The results show that the participants rated the answer sentences with a Cronbach's Alpha values of 0.842 and 0.771 for accuracy and readability respectively. Fig. 2 depicts the weighted average of rating values provided for both accuracy and readability. According to the figure it is clear that the ratings reside between 4 and 5 in the 5-point Likert scale. Furthermore, weighted average rating average for readability is recoded as 5 for 37 cases (90.24% from the test collection) while weighted average rating average for accuracy is recorded as 5 for 31 cases (75.6% from the test collection). This shows that the framework has achieved reasonable readability and accuracy levels from the user perspective.

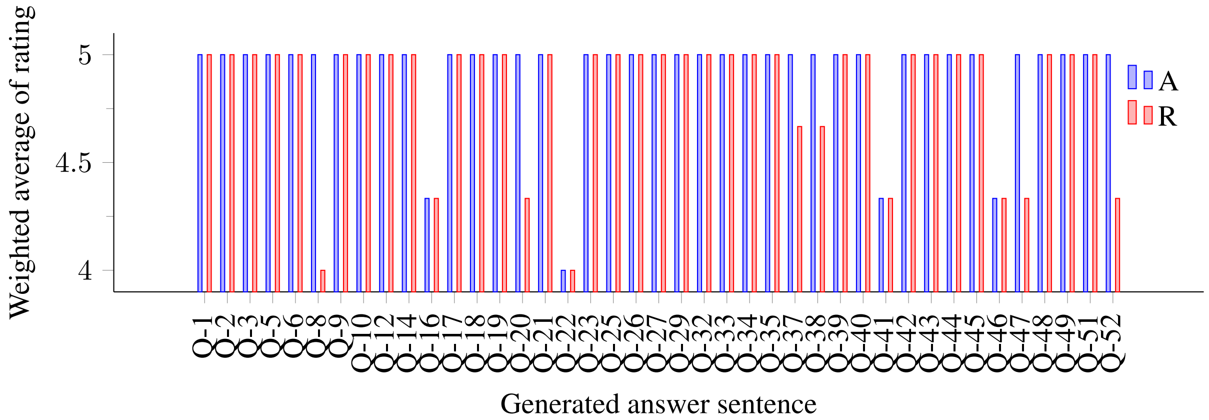


Figure 2: Weighted average ratings provided for generated answer sentences considering both accuracy and readability. (A=Accuracy & R=Redability)

4 Related work

Benamara and Dizier (2003) present the cooperative question answering approach which generates natural language responses for given questions. 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.

A successful attempt to move beyond the exact answer presentation with additional information in sentence form is presented by Bosma (2005) utilizing summarization techniques. In this research Bosma (2005) assumes that a QA system has already extracted a sentence that contains the exact answer. 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.

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 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

This research presented a novel answer presentation mechanism by generating answer sentences utilizing the typed dependency parse of the source question. The generated answer sentence is further realized using rule a based mechanism to generate more natural sentences. The evaluation of the framework covered how extracted patterns provide coverage in the test dataset as well as the human evaluation for both accuracy and readability. The both evaluations showed that framework is performing well in answer sentence generation by producing sentences which emanate human generated language.

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.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D Manning. 2014. Universal Stanford Dependencies: A cross-linguistic typology. In *9th International Conference on Language Resources and Evaluation (LREC'14)*, pages 4585–4592.

Brian McBride. 2002. Jena: A semantic web toolkit. *IEEE Internet Computing*, 6(6):55–58.

Rivindu Perera and Parma Nand. 2015a. 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. 2015b. 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*.

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