Automatic Question Generation from Sentences

Husam Ali   Yllias Chali   Sadid A. Hasan
University of Lethbridge
Lethbridge, AB, Canada
{ali,chali,hasan}@cs.uleth.ca

Abstract. Question Generation (QG) and Question Answering (QA) are some of the many challenges for natural language understanding and interfaces. As humans need to ask good questions, the potential benefits from automated QG systems may assist them in meeting useful inquiry needs. In this paper, we consider an automatic Sentence-to-Question generation task, where given a sentence, the Question Generation (QG) system generates a set of questions for which the sentence contains, implies, or needs answers. To facilitate the question generation task, we build elementary sentences from the input complex sentences using a syntactic parser. A named entity recognizer and a part of speech tagger are applied on each of these sentences to encode necessary information. We classify the sentences based on their subject, verb, object and preposition for determining the possible type of questions to be generated. We use the TREC-2007 (Question Answering Track) dataset for our experiments and evaluation.

Mots-clés : Génération de questions, Analyseur syntaxique, Phrases élémentaires, POS Tagging.

Keywords: Question Generation, Syntactic Parsing, Elementary Sentence, POS Tagging.

1 Introduction

Ideal learners are often curious question generators who actively self-regulate their learning. That is, they identify their own knowledge deficits, ask questions that focus on these deficits, and answer the questions by exploring reliable information sources. Unfortunately, this idealistic vision of intelligent inquiry is rarely met, as most learners have trouble identifying their own knowledge deficits (Rus & Graesser, 2009). Question asking and Question Generation (QG) are important components in advanced learning technologies such as intelligent tutoring systems, and inquiry-based environments. QG is an essential element of learning environments, help systems, information seeking systems, and a myriad of other applications (Lauer *et al.*, 1992). A QG system would be useful for building an automated trainer for learners to ask better questions, and for building better hint and question asking facilities in intelligent tutoring systems (Graesser *et al.*, 2001). Another benefit of QG is that it can be a good tool to help improve the quality of the Question Answering (QA) systems. Available studies revealed that humans are not very skilled in asking good questions. Therefore, they would benefit from automated QG systems to assist them in meeting their inquiry needs (Rus & Graesser, 2009).

In this paper, we consider a form of Text-to-Question generation task, where the input text are sentences. The QG system would then generate a set of questions for which the sentence contains, implies, or needs answers. We experiment with the TREC-2007 (Question Answering Track) [1] dataset. The scenario for the main task in the TREC 2007 QA track was that an adult, native speaker of English is looking for information about a target of interest (Dang *et al.*, 2007). The target could be a person, organization, thing, or event. The user was assumed to be an "average" reader of U.S. newspapers. The main task required systems to provide answers to a series of related questions. A question series, which focused on a target, consisted of several factoid questions, one or two list questions, and exactly one Other question. We use these data to act oppositely in this research. That is, using the given target, we filter out the important sentences from the large sentence pool and generate possible questions from them. So, we call our Sentence-to-Question generation system as target-driven. For this research we consider the factoid type questions only. A factoid question can be any of these types : "What...", "Where...", "When...", "Who...", and "How many / How much...". For example, considering "WWE" as target, we can generate these questions : *"Who is the chairman of WWE ?"*, *"Where is WWE headquartered ?"*, *"What is "WWE" short for ?"*, *"WWE evolved from what earlier organization ?"*, *"What cable network airs WWE ?"*.

The rest of the paper is organized as follows : Section 2 describes the related work on Question Generation followed by Section 3 that discusses the details of our QG system. Section 4 shows the evaluation results. Finally, in Section 5 we give conclusion and future directions.

2 Related Work

Recently, tackling Question Generation (QG) in the field of computational linguistics has got immense attention from the researchers. Twenty years ago it would take hours or weeks to receive answers to the same questions as a person hunted through documents in a library. In the future, electronic textbooks and information sources will be mainstream and they will be accompanied by sophisticated question asking and answering facilities. As a result, it is believed that the Google generation is meant to have a much more inquisitive mind than the generations that relied on passive reading and libraries (Rus & Graesser, 2009). In the last few years, new preoccupations appear for automatic question generation. In (Andrenucci & Sneiders, 2005), they introduced a template-based approach to generate questions on four types of entities. The authors in (McGough *et al.*, 2001) used WTML (Web Testing Markup Language), which is an extension of HTML, to solve the problem of presenting students with dynamically generated browser-based exams with significant engineering mathematics content. In (Wang *et al.*, 2008), they generated the questions automatically based on question templates that are created by training on many medical articles. In (Brown *et al.*, 2005), an interesting approach was described to automatically generating questions for vocabulary assessment.

3 Sentence-to-Question Generation

In this section, we describe the overall framework of our Question Generation (QG) system. We discuss all the modules in detail. As we said before, we consider the question generation task given a target. We use the TREC-2007 (Question Answering Track) dataset for this purpose. There are 70 topics in the dataset with 50 files for each topic. A target and a set of related questions for each topic are given. When the results were released, they also provided the actual answers and the file names that contain the answers.

1. http ://trec.nist.gov/data/qamain.html

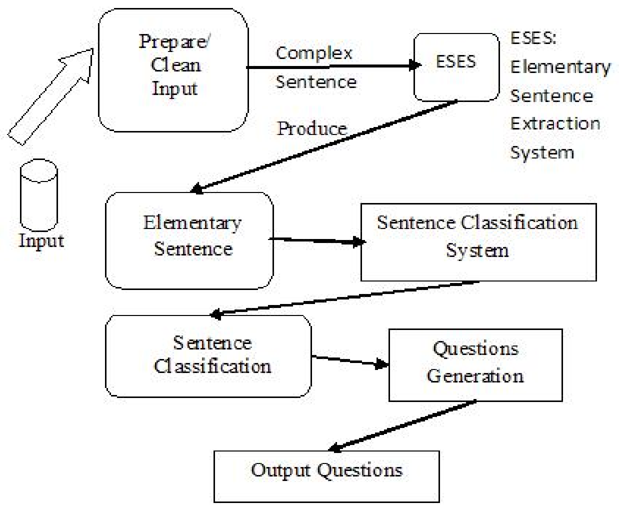

FIGURE 1 – Basic architecture of our QG system

Initially, we clean up the data in order to get rid of unnecessary information. We use the actual answers and the targets to find the sentences that are relevant to the target or the answer or both. The sentences might have complex structure with multiple clauses. So, it would be difficult to generate accurate questions from the complex sentences. Therefore, we simplify the process by extracting elementary sentences from the complex sentences using syntactic information. We classify the sentences based on their subject, verb, object and preposition for determining the possible type of questions to be generated. The basic architecture of our QG system is depicted in Figure 1. We divide the whole task into three modules :

3.1 Data Preprocessing

Cleaning and processing of raw data is the important initial part of any NLP task. We remove the redundant tags and text from all the documents and the questions provided in the TREC dataset. We use the Oak system[2] to tokenize the sentences and the questions. The text file given in the dataset has the information about the actual answers for the questions. It is parsed to extract the topic number, the answer, and the file name that contains the answers to the questions. The system opens the file that contains the answer, and searches for the relevant sentences that contain the answer or the target or both. Thus, we reduce the number of sentences to be processed. Then, we pass the relevant sentences to the Named Entity (NE) tagger and the Parts of Speech (POS) tagger. We use the Oak system to generate the POS tagged sentence. The POS tagged sentences will provide us with the information about the verbs and their tenses. We extract all the verbs from a sentence based on this information. Again, we employ the Oak system to generate the NE tagged sentences. A sentence may include a certain Named Entity types (among the 150 NEs possible) such as : PERSON, LOCATION, ORGANIZATION, GPE (Geo-Political Entity), FACILITY, DATE, MONEY, PERCENT, TIME, etc.

2. http ://nlp.cs.nyu.edu/oak/

3.2 Elementary Sentence Construction

Sentences in the dataset may have complex grammatical structure with multiple embedded clauses. Therefore, we extract elementary sentences from the complex sentences with the intention to generate more accurate questions. We syntactically parse each complex sentence to accomplish this. We perform this by using the Charniak parser[3]. This module constructs a syntactic tree representation, from the bracketed representation of the parsed sentence. While building the tree process, we construct 3 arrays, one for the Noun Phrases (NPs), one for the Verb Phrases (VPs) and one for the Prepositions (PPs) with their location in the tree, a fourth array is generated, from the tree to represent the depth first sequence of the tree nodes and leaves structure. We combine the NPs with the VPs and PPs by reading the NPs till the scope of the VPs and the PPs that are in the VPs scope and thus, we get the elementary sentences.

3.3 Sentence Classification and Question Generation

Elementary sentences are the inputs of this module. Based on the associated POS and NE tagged information, we get the subject, object, preposition and verb for each elementary sentence. We use this information to classify the sentences. We follow a two-layered taxonomy to represent a natural semantic classification for the sentences. Our sentence classifier module makes use of a sequence of two simple classifiers. The first classifies the sentences into fine classes (Fine Classifier) and the second into coarse classes (Coarse Classifier). This is a similar but opposite approach to the one described in (Li & Roth, 2002). The second classifier influences the first in that its candidate labels are generated by reducing the set of retained fine classes from the first into a set of coarse classes. This set is treated as the confusion set for the first classifier, the confusion set keep shrinking till we find the Coare classes that the word belongs to. OAK System has 150 types that can be tagged. They are included in a hierarchy. This information is used to make candidate fine and coarse classes. We define the five coarse classes as :

1. ***Human :*** *Any subject or object that is a name of a person.*
2. ***Entity :*** *Includes animals, plant, mountains and any object.*
3. ***Location :*** *Words that represent locations, such as country, city, school, etc.*
4. ***Time :*** *Words that represent time, date or period such as year, Monday, 9 am, last week, etc.*
5. ***Count :*** *Hold all the counted elements, such as 9 men, measurements like weight etc.*

We process each sentence in a top-down manner to get it classified. Let, the confusion set of any sentence be $C_0 = \{c_1, c_2, \cdots, c_n\}$, , the set of all the coarse classes. Initially, the fine classifier determines the fine classes. Then, the set of fine classes is reduced to a coarse class determined by the class hierarchy. That is, the set $\{f_{i1}, f_{i2}, \cdots, f_{im}\}$ of fine classes is mapped into the coarse class c_i. Based on the coarse classification, we consider the relationship between the words in the sentence. For example, if a sentence has the structure : "Human Verb Human", it will be classified as "whom and who" question types. If it is followed by a preposition that represents time, then we add the "When" question type to its classification. We check the coarse classes according to the word-to-word interaction rules. The rule check will produce the type of questions that can be generated while considering the verb tense and stem. Indeed, the output of the system will be the questions of the type that is suggested here. In this research, we define a set of 90 interaction rules like "Human Verb Human", "Human Verb Entity", "Human Verb Human Time" ... etc.

3. available at ftp ://ftp.cs.brown.edu/pub/nlparser/

Type	Q_g	Q_a	Recall	Type	Q_g	Q_a	Recall
What	7	52	0.135	Which	3	10	0.300
Who/Whom	11	20	0.550	How many/much	3	13	0.231
Where	4	8	0.500	When	1	2	0.500
Over all	29	105	0.276				

TABLE 1 – Individual factoid types and Over all Recall

Type	Q_g	Q_r	Precision	Type	Q_g	Q_r	Precision
What	105	43	0.410	Which	57	23	0.404
Who/Whom	144	106	0.736	How many/much	43	17	0.395
Where	117	89	0.761	When	71	37	0.521
Over all	537	315	0.587				

TABLE 2 – Individual factoid types and Over all Precision

4 Evaluation Results and Discussion

TREC-2007 provides the dataset along with the questions and their corresponding answers. We use this information to evaluate our system. In this research we employ the widely used evaluation measures : *Recall* and *Precision* of our system. We define *Recall* and *Precision* as follows :

$$Recall = \frac{Q_g \bigcap Q_a}{Q_a} \tag{1}$$

$$Precision = \frac{Q_g \bigcap Q_r}{Q_r} \tag{2}$$

where, Q_g is the number of questions generated by our QG system, Q_a is the number of actual questions given for each topic in the TREC dataset and Q_r is the number of related questions generated by the system excluding questions with gramatical error.

For Recall evaluation we experiment with 70 topics from the TREC 2007 dataset. For 20 topics, our system could generate the questions given the target. However, for the other topics our system was unable to generate any questions. We get a Recall of 0.000 for the other topics. From Table 1 we see that for the "When", "Where" and "Who" type questions, the Recall is similar. For the type "What", we get the lowest Recall of 0.135. We also show the overall Recall considering all the question types.

For Precision evaluation we experiment with 5 topics from the 20 topics that we could generate question from, from TREC 2007 dataset. In this analysis we rejected the grammatically incorrect generated question from the relevant set. Table 2 shows that the precision was high for the types "Who and Where", the type "When" was still above 0.5, the other types was hovering around 0.4, the reason for that is not considering the grammatically wrong constructed questions as a valid relevant questions, we believe that if the system included a semantic analysis for the sentence and its components the results will be higher, and the grammatically wrong sentences to decrease in number.

5 Conclusion and Future Work

In this paper, we proposed an approach to automatically generate questions given sentences. We used the dataset provided by the TREC 2007 Question Answering Track and evaluated the performance of our system using *Recall* and *Precision*. We filtered out important sentences from the dataset by following a target-driven method. We simplified the process by extracting elementary sentences from the complex sentences using syntactic information. After classifying the sentences based on their subject, verb, object and preposition, we generated the questions automatically from them using a predefined set of interaction rules. We plan to extend the number of interaction rules in the future. We will also focus on the sentence classification module to make it more robust. Since human generated questions always tend to have words with different meanings and senses, the system can be improved with the inclusion of semantic information and word sense disambiguation. We hope to carry on these ideas and develop additional mechanisms to question generation based on the dependency features of the answers and answer finding (Li & Roth, 2006; Pinchak & Lin, 2006).

Références

ANDRENUCCI A. & SNEIDERS E. (2005). Automated Question Answering : Review of the Main Approaches. In *Proceedings of the 3rd International Conference on Information Technology and Applications (ICITA'05)*, Sydney, Australia.

BROWN J. C., FRISHKOFF G. A. & ESKENAZI M. (2005). Automatic Question Generation for Vocabulary Assessment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Vancouver, British Columbia, Canada.

DANG H. T., KELLY D. & LIN J. (2007). Overview of the TREC 2007 Question Answering Track. In *Proceedings of the 16th Text REtreival Conference*, Gaithersburg, Maryland.

GRAESSER A. C., VANLEHN K., ROSE C. P., JORDAN P. W. & HARTER D. (2001). Intelligent Tutoring Systems with Conversational Dialogue. *AI Magazine*, **22(4)**, 39–52.

LAUER T. W., PEACOCK E. & GRAESSER A. C. (1992). Questions and Information Systems.

LI X. & ROTH D. (2002). Learning Question Classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics*, p. 1–7, Morristown, NJ, USA : Association for Computational Linguistics.

LI X. & ROTH D. (2006). Learning Question Classifiers : The Role of Semantic Information. *Journal of Natural Language Engineering*, **12(3)**, 229–249.

MCGOUGH J., MORTENSEN J., JOHNSON J. & FADALI S. (2001). A Web-based Testing System with Dynamic Question Generation. In *ASEE/IEEE Frontiers in Education Conference*.

PINCHAK C. & LIN D. (2006). A Probabilistic Answer Type Model. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, p. 393–400.

RUS V. & GRAESSER A. C. (2009). The Question Generation Shared Task and Evaluation Challenge. In *Workshop on the Question Generation Shared Task and Evaluation Challenge, Final Report*, The University of Memphis : National Science Foundation.

WANG W., TIANYONG H. & WENYIN L. (2008). Automatic Question Generation for Learning Evaluation in Medicine. In *LNCS Volume 4823*.