Ranking Automatically Generated Questions Using Common Human Queries

Yllias Chali and **Sina Golestanirad**
University of Lethbridge
Alberta Canada

Abstract

In this paper, we challenge a form of paragraph-to-question generation task. We propose a question generation system which can generate a set of comprehensive questions from a body of text. Besides the tree kernel functions to assess the grammatically of the generated questions, our goal is to rank them by using community-based question answering systems to calculate the importance of the generated questions. The main assumption behind our work is that each body of text is related to a topic of interest and it has a comprehensive information about the topic.

1 Introduction

Human beings are not very good at asking questions about topics. They are often forgetful, which causes difficulties in expressing what is in their minds (Hasan, 2013). Also sometimes, Humans, in front of a search engine, have difficulties to express their needs and intents as query terms. Imagine that you want to find out what was the first logo for Apple Inc. You may use a search engine such as *Google* and the search query *Apple Logos*, the result might have the exact information that you need. However, they may also include other information, such as who designed the logo or where it was designed or any other information that you are not interested in. We believe if before showing the list of websites the search engine had shown some suggested queries you would benefit from this question generation (QG) system. This way search engines' users will be able to use right queries to gain what they are looking for. Suggestions could be questions like: *What is the logo of Apple Inc.? What was the first logo designed for Apple Inc.? Do we have any information about where Apple's logo was designed?* etc. In this paper, we address the challenge of generating questions from topics, which is motivated by the fact that people do not always obtain the desired results from search engines. In this task, we assume that for each search-engine user's query there is a body of text having useful information about it. Our goal is to generate and show a few questions to the user in order to help her/him to find exactly what she/he is looking for. We need to rank the questions because the number of generated questions could be many to be shown and we may have to show only top-ranked ones.

We generate the questions for a given topic in two steps. First, we tag the name entities in the topic and its associated body. Then, we apply some general rules and generate the basic questions. At this level, the answers for the basic questions may not be in the body of the text, but the reason of generating them is to have more variety. Second, we use predicates and their arguments from the sentences in the given body of text to generate specific questions, which answers can be generated from the text. As the number of generated questions may be too large we rank them and show the top ones to the users. The ranking of question consists of two steps. First, we investigate other questions being asked by people in community-based question answering (CQA) systems such as Yahoo! Answers to see how common our questions are. Second, we apply the tree kernel functions in order to compute the syntactic similarity

Proceedings of The 9th International Natural Language Generation conference, pages 217–221,
Edinburgh, UK, September 5-8 2016.

between each question and the text from which the question is generated. This way we can determine the correctness of the grammatically of the generated questions. Then, the questions are ranked by their importance and grammatical correctness.

2 Related Work

An automated question generation system can also be used for educational purposes, and some works address the task of automatically generating questions from reading materials in order to advance educational assessment and practice (Mitkov and Ha, 2003; Wang et al., 2008; Xu et al., 2009; Rus and Graesser, 2009; Heilman and Smith, 2010a; Agarwal and Mannem, 2011; Agarwal et al., 2011). Heilman and Smith (2010b) proposed a system that overgenerates some questions and then uses a model, which has been trained on a dataset in order to rank the generated questions. Liu et al. (2010) proposed an automatic question generation system which helps students to write literature reviews. Gate (2008) developed a question generation system that generates questions in order to help students while reading an article rather than afterwards. Lindberg et al. (2013) introduced a sophisticated template based system which merges semantic role labels into a system that automatically generates natural language questions to support online learning. Mazidi and Nielsen (2014) proposed an automatic question generator which benefits from semantic pattern recognition to generate questions which have different depth and type for tutoring or self-study purposes. Rokhlenko and Szpektor (2013) challenged the task of automatically generating questions which are relevant to a given text but do not exist in the text. Labutov et al. (2015) developped an approach for generating deep (i.e, high-level) comprehension questions from novel text. Chali and Hasan (2015) addressed the problem of automatically generating questions from topics.

3 Question Generation

Our question generation approach is built in five steps. In the first step, we tag named entities from a text which is related to the query. In the next step, we use question templates to generate basic questions, based on the tags from the previous step. In the third step, we apply a text simplifier to all of the sentences in the text and then we use a semantic role tagger to tag all of the arguments and predicates in these sentences. Fourth step is about applying another set of question rules to the extracted arguments and predicates in order to generate specific questions. In the final step, we use our proposed algorithm to rank all of the generated questions.

3.1 Generating Basic Questions

The named entities, which are in the topic and its relevant text are tagged using the Illinois Named Entity Tagger. Then, we apply our general rules to generate the basic questions. At this level, the answers to these questions may not be in the text and the reason for generating these questions is to have more diversity in our question pool. We designed 265 question templates and an algorithm that generates the basic questions with regard to the tagged named entities.

3.2 Generating Specific Questions

The grammar of the sentences in the body of the text may be complicated, that is why the sentences have to be simplified before we can generate more accurate questions. To do the task of simplification, we use the simplified factual statement extraction toolkit (Heilman and Smith, 2010a). This model simplifies sentences by changing semantic and syntactic structures, eliminating phrase types, etc.

For the next step, we need to parse sentences in the text semantically. To this end, we use Automatic Statistical SEmantic Role Tagger (ASSERT)[1]. When a sentence is given to ASSERT, it applies a full syntactic analysis of that sentence, identifies all of the verb predicates, then extracts features for constituents within the parse tree relative to the predicate, and eventually identifies and tags the constituents with the appropriate semantic arguments. The outputs contain verbs (predicates) with their arguments (semantic roles). Those arguments can be used to generate specific questions.

In order to generate the specific questions, we used 350 rules to transform the tagged sentences into questions. These rules are designed in a way that the answer words in a sentence could be discovered and replaced by question words.

[1] Available at http://www.cemantix.org/.

4 Ranking of the Questions

As the number of generated questions is usually too large, we have to rank and show the top N ones. We score the questions regarding to their importance and syntactic correctness. We give equal weight to both importance and correctness.

4.1 Importance of the Generated Questions

In Chali and Hasan (2015), the importance of the generated questions was estimated by their similarities to the text. We believe that there is another source to use in order to do the task of ranking the importance of the generated questions. Nowadays, it is becoming a common habit for people to ask their questions in online forums, which are called community-based question answering (CQA) systems, such as Yahoo! Answers' web site. We believe that there are many common questions being asked by people that can be used to study what people need and what they are mostly curious about. The key point is that people using CQA systems use more informative sentences to ask and so if these sentences are similar to the search engines' queries then we can extract additional information about the users' probable interests. Our algorithm predicts what the user might be looking for by investigating other questions asked by other people, then this knowledge can be used to rank the importance of the generated questions.

4.1.1 Database

Our algorithm needs a CQA system, we use then Yahoo! Answers dataset. Yahoo! Answers is growing quickly, it is suggested that researchers increasingly use Yahoo! Answers dataset and it is becoming a popular source of information, such as advice or opinion (Liu and Agichtein, 2008). The data that we use in our experiments is Yahoo! Answers corpus as of 10/25/2007. In Yahoo! Answers people usually ask their questions in two steps. First, they ask a short and informative question which is called *subject*. Then, they try to explain the question in a few sentences, which is called *content*. The content part does not often provide more information.

4.1.2 Ranking Algorithm

1. To begin, we extract all subjects from Yahoo! Answers database.

2. When a user performs a Search Engine Query (SEQ), we calculate the semantic similarity between the SEQ and each extracted subject.

3. Then we store the top scored subjects in an array, named Top-Subjects and also the scores of these Top-Subjects in another array called Top-Subjects-Scores.

4. At this step, we find the similarity scores of the first generated question with all Top-Subjects.

5. We store these scores in an array called Generated-Question-Similarities-to-Top-Subjects.

6. To obtain an overall score we take the average of all scores in both vectors Top-Subjects-Scores & Generated-Question-Similarities-to-Top-Subjects.

At this point, we have one score showing us how similar the generated question is to the questions that people have asked in Yahoo! Answers. The same steps will be taken for each generated question resulting in one similarity score per generated question. Then we sort the generated questions by these scores and show the user as many top ones as required.

In our experiments, we use the semantic similarity toolkit SEMILAR[2]. It has different methods for calculating the semantic similarity scores (Rus et al., 2013a; Rus et al., 2013b). We use LDA-Optimal method as its accuracy is quite acceptable.

4.2 Judging Syntactic Correctness

It is strongly believed that a question has a similar syntactic structure to the sentences from where it is generated (Chali and Hasan, 2015). Therefore, to judge the syntactic correctness of each generated question, we apply tree kernel functions (Collins and Duffy, 2001) in order to compute the syntactic similarity between each question and its associated body of text. To measure the syntactic similarity between two sentences, we first parse them syntactically which results in a parse tree for each sentence, then we apply tree kernel functions to these trees.

[2]Available at http://www.semanticsimilarity.org/.

The tree kernel function produces the syntactic similarity score between each sentence in the given body of text and the generated question. Each sentence contributes a score to the questions and then the questions are ranked by considering the average of their similarity scores.

5 Experiments

5.1 Corpus

In our experiments, we use the dataset from the Question Generation Shared Task and Evaluation Challenge (Rus and Graesser, 2009; Rus et al., 2010) to tackle the task of automatically generating questions. The dataset consists of 60 paragraphs, each related to 60 topics. They are originally selected from several articles such as OpenLearn, Wikipedia and Yahoo!Answers. The paragraphs are constructed from approximately 57 sentences, a total number of 100,200 tokens including punctuations. As mentioned before, to do our task, we assume that there exists a text related to each query containing useful information about it, so we consider the topics as queries and treat paragraphs as the associated body of the texts.

5.2 Evaluation Setup

Our methodology to evaluate the performance of our automated question generation system is inspired by Hasan (2013). Three unknown native English speakers were chosen to judge the result of our system. They were asked to score the generated questions according to two criteria: syntactic correctness and topic relevance. Judges give scores between 1 (very poor) and 5 (very good). There were four scores for each generated question. To evaluate the topic relevance criterion, judges were given three aspects, and they score each question according to each aspect. Aspects were: 1) questions' semantic correctness 2) question type correctness and 3) clarity of referential. For syntactic correctness, they score the generated questions considering if they are grammatically correct or not. Then the average of the judges' scores is calculated for each question.

To evaluate our system, we compare it with the state-of-the-art question generation system proposed by Chali and Hasan (2015). To do so, we use a publicly available question generation system by Heilman and Smith (2010a) as a benchmark. In our evaluation, we generated the questions from 20 randomly chosen texts and then select the top 10 ranked questions. We also generate the questions for the same texts by Heilman and Smith (2010a) question generation toolkit and again select top 10 ranked ones. The human judges were presented with 20 questions per text, top 10 from our system and top 10 from the system proposed by Heilman and Smith (2010a). We have 20 texts for the total number of 400 questions for each judge. After comparing our system with Heilman and Smith (2010a) system, we calculate our system advancement in comparison with the one created by Chali and Hasan (2015).

5.3 Results and Discussion

Table 2 lists the average of syntactic correctness and topic relevance scores for each system. These results confirm that our proposed automated question generation system outperforms the Heilman and Smith system (2010a) by 29.39%, and 18.71%, and over the Chali and Hasan system (2015) by 25.38%, and 14.04%, respectively. In this paper, we have shown that by using semantic similarity between a topic of interest and a group of pre-asked questions we can extract related ones to the concept of the topic and then we can use them to find the importance of a new generated question.

Systems	Syntactic Correctness	Topic Relevance
Heilman and Smith	3.13	3.42
State-of-the-art	3.23	3.56
Proposed QG System	4.05	4.06

Table 1: Syntactic correctness and topic relevance scores

6 Conclusion

We presented a novel system for automatically generating questions for topics of interests. The main assumption is that each topic is associated with an informative text. We have designed 265 templates to generate basic questions, and 350 rules to generate specific questions. The main aspect of this proposed method is the use of CQA systems to improve ranking of the generated questions. We used CQA to investigate the importance of questions and tree kernel functions to gauge how grammatically they are

correct. We believe that there might be some ways in which this research could be continued, for example, our proposed system is rule-based, however, one of the ways to scale up these rules is learning them using learning techniques, in other words, the templates may be learned / acquired from a corpus of CQA data.

References

M. Agarwal and P. Mannem. 2011. Automatic gap-fill question generation from text books. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 56–64.

M. Agarwal, R. Shah, and P. Mannem. 2011. Automatic question generation using discourse cues. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9.

Y. Chali and S. Hasan. 2015. Towards topic-to-question generation. *Computational Linguistics*.

M. Collins and N. Duffy. 2001. Convolution kernels for natural language. In *Advances in neural information processing systems*, pages 625–632.

D. M. Gates. 2008. Automatically generating reading comprehension look-back strategy: Questions from expository texts. Technical report, DTIC Document.

S. Hasan. 2013. *Complex question answering: minimizing the gaps and beyond*. Ph.D. thesis, University of Lethbridge, Alberta, Canada.

M. Heilman and N. A. Smith. 2010a. Extracting simplified statements for factual question generation. In *Proceedings of QG2010: The Third Workshop on Question Generation*.

M. Heilman and N. A. Smith. 2010b. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617.

I. Labutov, S. Basu, and L. Vanderwende. 2015. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 889–898.

D. Lindberg, F. Popowich, J. Nesbit, and P. Winne. 2013. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 105–114, Sofia, Bulgaria.

Y. Liu and E. Agichtein. 2008. On the evolution of the yahoo! answers qa community. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 737–738.

M. Liu, R. A. Calvo, and V. Rus. 2010. Automatic question generation for literature review writing support. In *Intelligent Tutoring Systems*, pages 45–54.

K. Mazidi and R. D. Nielsen. 2014. Linguistic considerations in automatic question generation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 321–326.

R. Mitkov and L. A. Ha. 2003. Computer-aided generation of multiple-choice tests. In *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing-Volume 2*, pages 17–22.

O. Rokhlenko and I. Szpektor. 2013. Generating synthetic comparable questions for news articles. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 742–751.

V. Rus and A. C. Graesser. 2009. *The question generation shared task and evaluation challenge*. The University of Memphis. National Science Foundation.

V. Rus, B. Wyse, P. Piwek, M. Lintean, S. Stoyanchev, and C. Moldovan. 2010. The first question generation shared task evaluation challenge. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 251–257.

V. Rus, M. C. Lintean, R. Banjade, N. B. Niraula, and D. Stefanescu. 2013a. Semilar: The semantic similarity toolkit. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 163–168.

V. Rus, N. Niraula, and R. Banjade. 2013b. Similarity measures based on latent dirichlet allocation. In *Proceedings of the Computational Linguistics and Intelligent Text Processing*, pages 459–470.

W. Wang, T. Hao, and W. Liu. 2008. Automatic question generation for learning evaluation in medicine. In *Advances in Web Based Learning – ICWL*, pages 242–251.

Y. Xu, A. Goldie, and S. Seneff. 2009. Automatic question generation and answer judging: a q&a game for language learning. In *Proceedings of the SIGSLaTE*.