IEICE TRANS. INF. & SYST., VOL.E102–D, NO.7 JULY 2019

PAPER

Rule-Based Automatic Question Generation Using Semantic Role Labeling

Onur KEKLIK[†], *Nonmember*, **Tugkan TUGLULAR**[†a)], *Member*, *and* **Selma TEKIR**[†], *Nonmember*

SUMMARY This paper proposes a new rule-based approach to automatic question generation. The proposed approach focuses on analysis of both syntactic and semantic structure of a sentence. Although the primary objective of the designed system is question generation from sentences, automatic evaluation results shows that, it also achieves great performance on reading comprehension datasets, which focus on question generation from paragraphs. Especially, with respect to METEOR metric, the designed system significantly outperforms all other systems in automatic evaluation. As for human evaluation, the designed system exhibits similar performance by generating the most natural (human-like) questions.

key words: *question generation, rule-based, semantic role labeling, METEOR*

1. Introduction

Humans have curios nature. They ask questions to gain more knowledge and try to link them with other things they know. Our daily lives include asking questions in conversations. For instance, student questions play an important role in their learning process and help them to learn more from their teachers and teacher questions help students to assess their performance. In a nutshell, questions are one of the primary sources of learning from daily conversations to verbal tutorings and assessments.

Most of learners are not good at asking questions. Hacker et al. (1998) state that learners have a problem with identifying their own knowledge deficits. Therefore, they ask very few questions. Automation of question generation systems can help learners to find out their own knowledge gaps by helping them to reach their valuable inquiries.

Automation of question generation systems helps automated question answering systems such as IBM Watson to perform self training (IBM Watson Ecosystem, 2014). Automatic question generation systems automatize the process of defining ground truth answers from the questions. Intelligent tutoring systems can also get benefit from that. Rather than relying on human experts to manually extract questions from study materials, each end user can define its own tutoring system automatically from the study material.

Question generation is a fundamental activity in educational learning. Questions serve to different levels of complexity in the educational learning process. In terms of complexity, Rus and Graesser (2009) divide questions into deep and shallow categories. If a learner wants to acquire difficult scientific and technical material, deep questions (such as why, why not, how, what-if, what-if-not) can be asked. These questions involve more logical thinking than shallow questions. Conversely, shallow questions focus more on facts (such as who, what, when, where, which, how many/much and yes/no questions).

Question generation for reading comprehension uses a paragraph as an information source and creates questions that test its understanding. Question generation from a sentence, on the other hand, poses factual questions related to the given input sentence.

Recently, Question Generation (QG) and Question Answering (QA) in the field of computational linguistics has got enormous attention from the researchers (Rus and Graesser, 2009). Twenty years ago, receiving answers to the same questions would take hours or weeks through documents and books. After the computers and internet, wealth of information becomes available and this field holds great promise for making sophisticated question asking and answering facilities mainstream in the future.

Most of the question generation systems tackles the problem with a rule-based approach. In the proposed approach, sentence-to-question transformation is performed by applying external rules or internal templates to the syntactic representation of the given input sentence. However, syntactic representations are not sufficient to reach high-level abstractions in question generation. There is a strong need to consider the semantic roles of words to increase the comprehension level of generated questions. Thus, question generation should be enriched by the process of understanding meaning behind a sentence. Mazidi and Tarau (2016) highlights this fact by stating that natural language understanding (NLU) is a missing piece of the puzzle in question generation.

This work proposes a rule-based approach to question generation from sentence. To be specific; dependency-based, named entity recognition-based, and semantic role labeling-based templates/rules are used. In terms of rules, our contribution can be outlined as follows:

1. For dependency-based rules, the following patterns are newly added:
   - S-V-oprd is an object predicate that defines the subject.
   - S-V-xcomp is an open clausal complement with-

Manuscript received June 6, 2018.
Manuscript revised January 3, 2019.
Manuscript publicized April 1, 2019.

[†]The authors are with the Department of Computer Engineering, Izmir Institute of Technology, Izmir, 35430 Turkey.
a) E-mail: tugkantuglular@iyte.edu.tr

DOI: 10.1587/transinf.2018EDP7199

out an internal subject.
- S-V-ccomp (clausal complement) is a clause with an internal subject.

2. NER-based rules, which are S-V-number, S-V-location, S-V-date, S-V-person, are added.
3. More importantly, new semantic role labeling-based templates/rules are constructed:
   - S-V-ARGM-CAU: cause clause.
   - S-V-ARGM-MNR: manner marker.
   - S-V-ARGM-PNC: purpose clause.
   - S-V-ARGM-LOC: locative.
   - S-V-ARGM-TMP: temporal marker.
4. Any rule can be enabled or disabled. That means deep and shallow questions can be generated individually, selectively, or all together.

Automatic question generation can be enriched by exploiting semantic roles of words in a diverse set of rules to generate both shallow and deep questions. Semantic role labeling promises deep question generation as semantic roles are not simple functions of a sentence's syntactic structure, but are proved (Lapata and Brew, 1999) useful as cues to reveal the meaning of a word in a particular context.

The proposed approach uses dependency-based, NER-based, and SRL-based semantic rules. Semantic roles of words contribute to comprehensive question generation in two ways:

- Using different parsers in combination (namely Dependency parser-SRL and NER-SRL).
- Using SRL-based templates (exclusively examining SRL parser to look for more sentence patterns).

To test the effectiveness of our approach, we compare against two state-of-the-art systems: Du et al.'s (2017) learning-based system for question generation for reading comprehension and the best rule-based system by Heilman and Smith (2010). Our automatic evaluation through objective neural translation metrics show that our system has superior performance and outperforms the others in BLEU-2, METEOR, and ROUGE-L metrics. Our superior performance especially in METEOR metric can be attributed to its recall-based nature. By diversifying and extending the rule sets, our approach generates an expanded set of questions out of those that can possibly be asked.

We performed human evaluation as well. In our experimental setup, 25 sentences are randomly sampled from SQuAD. From this set, 126 questions are generated and rated by four different professional English speakers on a 1-5 scale (5 for the best). We compare our results with those of Du et al. (2017) and Heilman & Smith (2010) system. In both difficulty and correctness metrics, our system produces closest scores to the real human-generated questions, indicating that our proposed system outperforms the others and is the most natural (human-like) system.

In the remaining part of the paper, first in Sect. 2, background is provided along with the related work. In Sect. 3, the proposed approach is explained in detail. After that, automatic evaluation and human evaluation are presented in separate sections. Finally, in Sect. 6, the paper is concluded with some remarks and possible future directions.

2. Background

2.1 Related Work

There are various types of questions and researchers proposed different taxonomies for organizing them. In terms of complexity, as Rus and Graesser (2009) state, questions can be divided into two categories, namely deep questions and shallow questions. If a learner wants to acquire difficult scientific and technical material, deep questions (such as why, why not, how, what-if, what-if-not) can be asked. These questions involve more logical thinking than shallow questions. On the other hand, shallow questions focus more on facts (such as who, what, when, where, which, how many/much and yes/no questions).

Most of the question generation systems tackle the problem with a rule-based approach. Majority of the steps are: first get the syntactic representation of the given input sentence, then use external templates or internal rules to apply sentence-to-question transformation. However, relying only on syntactic representations, which don't tell anything about the semantic role of words, force us to use only low-level abstractions in question generation. Du et al. (2017) state that the rule-based approaches make use of the syntactic roles of words, but not their semantic roles.

Heilman and Smith (2011) use a multi-step process to generate factual questions from text. The process begins with NLP transformations for the input sentence. Then, manually encoded transformation rules are applied for sentence-to-question conversion, and finally a linear regression-based ranker assigns acceptability scores to questions to eliminate the unacceptable ones.

Most of the prior works mentioned here arrange sentence constituents with respect to grammar rules to generate as many possible questions as they can. In contrast, Mazidi and Tarau (2016) introduced NLU-approach that focuses on constituent patterns in a sentence. These patterns are key to detect the type of question that should be asked. They also used multiple parsers (both syntactic and semantic parsers) instead of depending on only one because each parser tells its own particular viewpoint about the sentence. In their evaluation of the top 20 questions, their system generated 71% more acceptable questions than other state of the art question generation systems by augmenting the generation process with NLU techniques.

Labutov et al. (2015) used a completely different approach. They used crowd sourcing method to generate deep comprehension question templates. First, they obtain the high-level question templates from the crowd. Then, they retrieve the subset of collected templates. For example, category-section pairs for an article about Albert Einstein contains (Person, Early life), (Person, Awards), and (Per-

son, Political views). Articles about persons have similar subsections, so that templates formed for one person should transfer reasonably well to others. Their relevance classifier decides on whether category-section pairs match or not. However, in some cases, it transfers irrelevant content and raises the false positive errors.

Du et al. (2017) used another innovative approach to generate questions for reading comprehension. They used a neural language model with a global attention mechanism to generate questions. They study several variations of this model, from sentence focused models to paragrahs, reading passages and other variations to determine the importance of pre-trained vs. learned word embeddings. Moreover, they don't rely on hand-crafted rules in their approach.

The first Question Generation Shared Task Evaluation Challenge (QGSTEC, 2010) is one of the campaigns that follows the same tradition of STECs (such as Text REtrieval Conference, TREC) in Natural Language Processing. The campaign consists of two tasks. The first task focuses on question generation from paragraphs, whereas the second task focuses on question generation from sentences. Data sets, evaluation criteria and guidelines are prepared with respect to these tasks. For the second task, input sentences were selected from Wikipedia, OpenLearn and Yahoo! answers (30 inputs from each source). Participants were also provided with the list of target question types (who, where, when, which, what, why, how many/long, yes/no) that they need to generate. Finally, human evaluators evaluate the submitted questions according to evaluation criteria, which are relevance, question type (who/what/why), syntactic correctness, fluency, ambiguity, and variety.

Using human evaluators to evaluate machine generated questions is a time and resource consuming process. However, automatic evaluation metrics are key to manage this process much efficiently. Since the release of IBM's BLEU metric (Papineni et al, 2002) and the closely related NIST metric (Doddington, 2002), automatic evaluation metrics have been widely recognized and extensively used by machine translation (MT) community. Compared to the human evaluations, evaluating an MT system using such automatic metrics is cheaper, faster and time-saving. This is why Du et al. (2017) used BLEU, METEOR and ROUGE-L metrics to evaluate their neural question generation system. They also perform human evaluations to complement their results.

BLEU metric is first proposed by IBM (Papineni et al, 2002). It is the exact matches of words and matches against a set of reference translations for greater variety of expressions. It calculates geometric average of the n-gram scores (size 1 to 4) for precisions. It has no recall, but uses exponential brevity penalty to reduce the score of overly short sentences in order to compensate for recall.

METEOR metric is first proposed by Denkowski and Lavie (2009). It is a recall oriented metric, which combines recall and precision as weighted score components. METEOR calculates the similarity score between generations, references and semantic equivalents such as inflections, synonyms and paraphrases. Instead of relying on higher order n-grams, METEOR uses a direct word-ordering penalty.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metric, which is first proposed by Lin (2004), is designed to compare n-grams recall of machine produced translations against human-produced translations. ROUGE-L is measured according to the longest common subsequence.

Our implementation uses the evaluation package released by Chen et al. (2015) that includes the implementation of BLEU-1, BLEU-2, BLEU-3, BLEU-4, METEOR and ROUGE-L metrics.

Constructing a question answering or reading comprehension dataset is also a challenging problem. Richardson et al. (2013) curated MCTest, where each question is paired with 4 answer choices. Despite the dataset contains challenging human generated questions, its size is too small to support data-demanding question answering models. After that, many datasets are released and most of them generate the questions in a synthetic way. bAbI (Weston et al, 2015) is a fully synthetic reading comprehension dataset which features 20 distinct tasks. In order to solve these tasks different types of reasoning is required. Hermann et al. (2015) constructed a corpus of cloze style questions by predicting entities in abstractive summaries of Daily News / CNN articles. With this approach they collected a million new stories. However, Chen et al. (2016) concluded that the dataset is quite easy and current neural networks almost reached ceiling performance.

Finally in 2016, Rajpurkar et al. (2016) released the Stanford Question Answering Dataset (SQuAD). SQuAD is a reading comprehension dataset that consists of 100,000+ questions on 500+ articles. Questions are posed by crowdworkers on a set of Wikipedia articles. It overcomes the semi-synthetic and small size data issues that mentioned above.

2.2 Parsers

In this section parsers, which are the main building blocks of question generation, are explained one by one, with their own particular viewpoint examples.

2.2.1 Part-of-Speech Tagging

Part-of-speech tagging (POS tagging) is a task that intends to identify the syntactic role of each word in the given sentence such as singular noun, adjective. Part-of-speech tagging uses different tagsets based on language and corpus that was collected from different sources. The designed system relies on SENNA's (Collobert et al., 2011) part-of-speech tagging algorithm which uses Penn Treebank tagset (Marcus, Marcinkiewicz and Santorini, 1993). An example part-of-speech of the sample sentence "The Bill of Rights gave the law federal government greater legitimacy" can be seen in Table 1.

Table 1 An example of POS and chunk tags of sample sentence.

Token	POS Tag	Chunk Tag
The	DT	B-NP
Bill	NNP	E-NP
of	IN	S-PP
Rights	NNPS	S-NP
gave	VBD	S-VP
the	DT	B-NP
new	JJ	I-NP
federal	JJ	I-NP
government	NN	E-NP
greater	JJR	B-NP
legitimacy	NN	E-NP

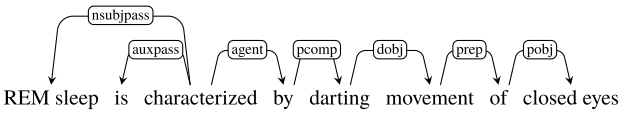

Fig. 1 An example dependency tree of sample sentence.

2.2.2 Chunking

Chunking (also called shallow parsing) is a process that first detects constituents in a sentence, then segments them to chunks of syntactically related word groups. Each word is labeled with its own unique tags in the groups. The designed system uses SENNA's chunking implementation. For this implementation, chunk tags have two parts. The first part indicates the beginning, continuation or end of a chunk. The second part indicates the type of the current word group. For example, beginning of a chunk noun phrase is labeled with B-NP, continuation of a chunk verb phrase is labeled with I-VP. Example chunking tags of the sample sentence "The Bill of Rights gave the law federal government greater legitimacy" can be seen in Table 1.

2.2.3 Dependency Parsing

The main idea of dependency parsing is that each word is connected to each other by directed links. These links are called dependencies in linguistics. The main goal is to reveal the syntactic structure of the sentence by looking at these dependencies. Structure is determined by the relation between a head word and its dependents (childs). The most widely used syntactic structure is a parse tree. It allows to navigate through generated parse tree using head and child tokens. An example dependency tree of the sentence "REM sleep is characterized by darting movement of closed eyes" can be seen in Fig. 1.

2.2.4 Named Entity Recognition

Named Entity Recognition (NER) is an information extraction task that labels words into various semantic categories such as person, date, location, facility. There are various NER approaches based on rule-based techniques and statistical models, i.e. machine learning. Rule-based approaches

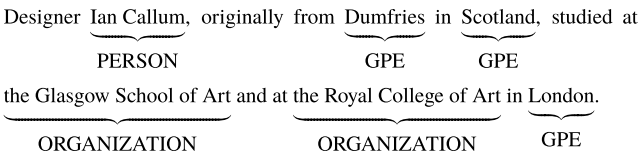

Fig. 2 Example NER tags in sample tense.

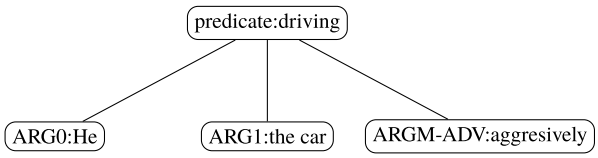

Fig. 3 An example SRL representation of a sample sentence.

rely on hand-crafted rules which is done by experienced linguists, whereas statistical approaches need large set of training data. Statistical models also allow to train custom models and define new categories according to problem domain. Found named entity tags of the sentence "Designer Ian Callum, originally from Dumfries in Scotland, studied at the Glasgow School of Art and at the Royal College of Art in London" can be seen in Fig. 2.

2.2.5 Semantic Role Labeling

Semantic Role Labeling (SRL) is a process that aims to reveal the semantic structure of a sentence by labeling word groups and phrases. It is an essential part of NLU and extracts the semantic word groups in a sentence. In SRL representation, predicate is the root and word groups accompanying the predicate is considered as arguments. Depending on their role in the sentence, predicates are assigned to different semantic categories. These categories are decided by the Proposition Bank (Palmer, Kingsbury, Gildea, 2005) which adds predicate-argument layer to syntactic structures of the Penn Treebank. The Proposition Bank provides numbered argument tags (such as ARG0, ARG1) and assigns functional tags to all verb modifiers, such as cause (CAU), temporal (TMP), purpose (PNC) and others (Malaya, 2005). The designed system relies on AllenNLP's (Gardner et al., 2017) semantic role labeler which includes high quality trained models. An example SRL representation of the sentence "He is driving the car aggressively." can be seen in Fig. 3.

3. Proposed Approach

3.1 Predefined Rules (Templates)

In this section, templates and their generation are explained in detail. In order to consider a sentence pattern as a template, it should satisfy the criteria which are previously stated by Mazidi and Tarau (2016). These are: (1) The sentence pattern should be working on different domains. (2) It should extract important points in the source sentence and

Table 2 Dependency-based template examples.

Template and Example
1. S-V-acomp is an adjective phrase that describes the subject. **S:** It has been argued that the term "civil disobedience" has always suffered from ambiguity and in modern times, become utterly debased. **Q:** Indicate characteristics of the term "civil disobedience".
2. S-V-oprd is an object predicate that defines the subject. **S:** Brain waves during REM sleep appear similar to brain waves during wakefulness. **Q:** How would you describe brain waves during REM sleep?
3. S-V-attr is a noun phrase, usually following copula and defines the subject. **S:** The fourth Yuan emperor, Buyantu Khan (Ayurbarwada), was a competent emperor. **Q:** How would you describe the fourth Yuan emperor, Buyantu Khan (Ayurbarwada)?
3. S-V-xcomp is an open clausal complement without an internal subject. **S:** Some of Britain's most dramatic scenery is to be found in the Scottish Highlands. **Q:** What is some of Britain 's most dramatic scenery?
4. S-V-dobj (direct object) is a noun phrase that is the accusative object of a verb. **S:** In 1996, the trust employed over 7,000 staff and managed another six sites in Leeds and the surrounding area. **Q:** What did the trust manage in Leeds and the surrounding area in 1996?
5. S-V-ccomp (clausal complement) is a clause with an internal subject. **S:** He says that you like to swim. **Q:** What does he say?
6. S-V-dative indicates an indirect object. **S:** The Bill of Rights gave the new federal government greater legitimacy. **Q:** What did give the new federal government greater legitimacy?
7. S-V-pcomp is a complement of a preposition that modifies the meaning of a prepositional phrase. **S:** REM sleep is characterized by darting movement of closed eyes **Q:** What is REM sleep characterized by?

create an unambiguous question. (3) Semantic information that is transferred by sentence pattern should be consistent across different instances.

Templates are categorized below with respect to the parsing method(s) they used. Each sentence pattern is detected by using different parsers.

3.1.1 Dependency-Based Templates

Using semantic role labeling, first the semantic structure of a sentence is revealed, then using dependency parsing, semantic arguments are matched with the found dependency tags to check if the sentence template corresponds to any template. In Table 2, templates can be seen with example sentences. S-V-acomp, S-V-attr, S-V-dobj, S-V-pcomp and S-V-dative are previously mentioned by Mazidi and Tarau (2016). S-V-xcomp and S-V-ccomp are changed and S-V-oprd is added according to the criteria that we have mentioned in Sect. 3.1.

Table 3 NER-based template examples.

Template and Example
1. S-V-number. **S:** In 1996, the trust employed over 7,000 staff and managed another six sites in Leeds and the surrounding area. **Q:** How many staff did the trust employ in 1996?
2. S-V-location. **S:** In 1996, the trust employed over 7,000 staff and managed another six sites in Leeds and the surrounding area. **Q:** Where did the trust manage another six sites in 1996?
3. S-V-date. **S:** In 1996, the trust employed over 7,000 staff and managed another six sites in Leeds and the surrounding area. **Q:** When did the trust employ over 7,000 staff?
4. S-V-person. **S:** The fourth Yuan emperor, Buyantu Khan (Ayurbarwada), was a competent emperor. **Q:** Who was a competent emperor?

3.1.2 NER-Based Templates

Using semantic role labeling, first the semantic structure of a sentence is revealed, then using named entity recognition, the designed system detect words that are labeled as person, location, date or number. So, the designed system can ask who, where, when or how many questions accordingly. Then, the tagged word is removed from the corresponding semantic argument. Finally, using the rest of the semantic arguments, the designed system checks if the constructed sentence has reasonable components (subject, direct object, verb,etc.).

However, "who" question is problematic. Let's consider an example sentence: "Atop the Main Building's gold dome is a golden statue of the Virgin Mary." In this scenario, NER detects "Virgin Mary" as a person. However, if we look at the full noun phrase: "statue of the Virgin Mary" is an object, not a person. So, "who" question is not an appropriate choice in this situation. The designed system tackles this problem through the use of chunking. So, the full noun phrase can be detected and this problematic case can be eliminated. With the use of chunking, the designed system also detects relative clauses to ensure a sentence really mentions a person, not an object. NER-based templates and examples can be seen in Table 3.

3.1.3 SRL-Based Templates

Using semantic role labeling, the designed system reveals the semantic structure of the sentence under consideration. Using only numbered argument labels and modifiers, the designed system generates questions. Templates and examples can be seen in Table 4.

3.2 The Designed System

The designed system focuses both on deep and shallow

Table 4 SRL-based template examples.

Template and Example
1. S-V-ARGM-CAU: cause clause.
S: On 24 March 1879, Tesla was returned to Gospic under police guard for not having a residence permit.
Q: Why was Tesla returned to Gospic on 24 March 1879?
2. S-V-ARGM-MNR: manner marker.
S: Tesla's work for Edison began with simple electrical engineering and quickly progressed to solving more difficult problems.
Q: How did Tesla's work for Edison progress quickly to solving more difficult problems?
3. S-V-ARGM-PNC: purpose clause.
S: To secure further loans, Westinghouse was forced to revisit Tesla's AC patent, which bankers considered a financial strain on the company (at that point Westinghouse had paid out an estimated $200,000 in licenses and royalties to Tesla, Brown, and Peck)..
Q: For what purpose was Westinghouse forced to revisit Tesla's AC patent, which bankers considered a financial strain on the company?
4. S-V: don't have any modifier, using only numbered argument labels to generate yes/no questions.
S: The Bill of Rights gave the new federal government greater legitimacy.
Q: Did the Bill of Rights give greater legitimacy?
5. S-V-ARGM-LOC: locative.
S: Mr. Bush met him privately, in the White House, on Thursday.
Q: Where did mr. Bush meet him on Thursday?
6. S-V-ARGM-TMP: temporal marker.
S: Mr. Bush met him privately, in the White House, on Thursday.
Q: When did mr. Bush meet him in the White House?

questions which have mentioned in Sect. 2.1. As shallow questions, the designed system can generate the following question sentences:

- What…?
- Who…?
- Where…?
- How many…?
- When…?

As deep questions, the designed system can generate:

- Why…?
- How…?
- How would you describe…?
- Indicate characteristics of…?
- For what purpose…?

As seen in Fig. 4, the designed system takes an input sentence, which is preprocessed. In the preprocessing stage, contractions are expanded first. For instance, "would've" is expanded into "would have". Contractions can be problematic for parsers, as most of them get parsed wrong and generate unexpected results. So, in order to detect verb groups perfectly and reduce errors, the designed system uses our predefined dictionary to handle contractions.

In the second step of preprocessing, idiomatic language is eliminated. Idiomatic language is another problematic case for question generation systems. For instance, the sentence "Mary has to learn to bite the bullet and face her fears

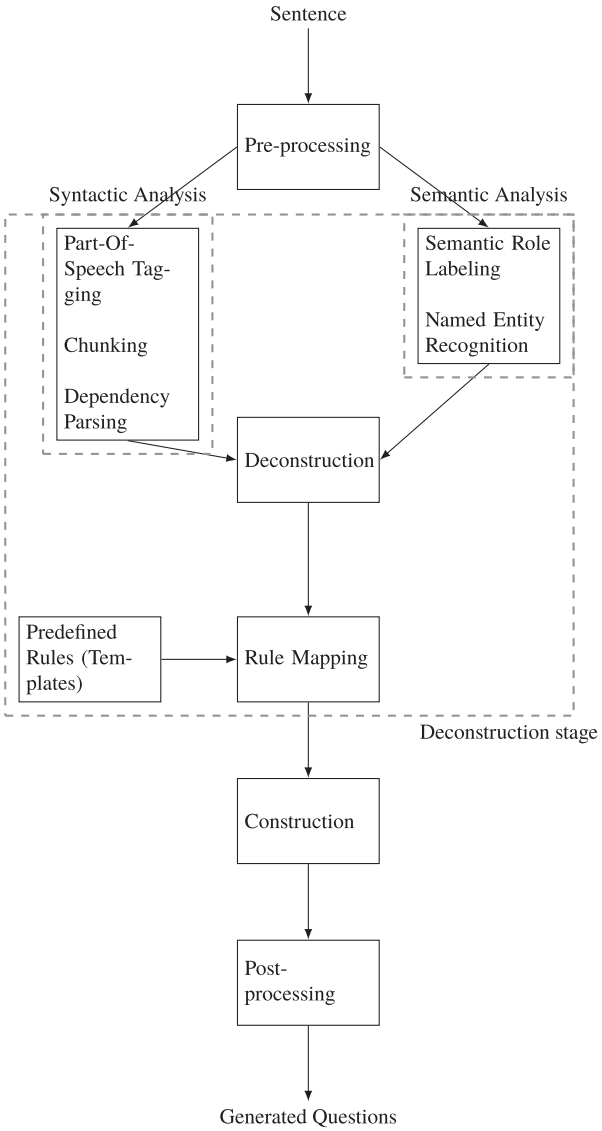

Fig. 4 Flowchart of the designed system.

of flying" results in the generated question: "What does Mary have to learn to bite?". However, the generated question is vague and out of context. In this case, "the bullet" grammatically is the direct object, which is why this question was generated, but "bite the bullet" is an idiom. In the preprocessing stage, the designed system detects idioms by using our predefined dictionary and eliminates these problematic cases.

The designed system uses multiple semantic and syntactic parsers because each parser tells its own particular viewpoint about the sentence and adds to its understanding. In Algorithm 1, dependency and SRL parsers (lines 9-10), NER and SRL parsers (lines 13-14), exclusively SRL parser (line 19) are used. By exploiting synergies between these parsers, the designed system searches for various templates. As a result, the designed system generates different types of questions for the templates found. Question generation examples can be seen in Sect. 3.3. Following the preprocess-

ing, in the deconstruction stage, dependencies between the words, the named entity information and also semantic word groups are extracted. For extracting dependencies between the words and gathering the named entity information, the designed system relies on Spacy's[†] algorithms. Finally, the designed system uses AllenNLP's (Gardner et al., 2017) semantic role labeler for extracting semantic word groups in a sentence.

The main objective of deconstruction stage is to get an intermediate representation in order to determine the sentence pattern. Sentence pattern is crucial to detect the type of question to be generated. Systematic arrangement of words (such as Subject + Verb + Object + Complement) in a sentence is called the sentence pattern.

Intermediate representation consists of sentence elements. In English, a part of the sentence is classified as a certain sentence element such as subject, direct object, verb, subject complement, etc. Full verb detection is also needed in this stage. The designed system relies on chunking algorithm to extract verb groups and phrases. For chunking algorithm, the designed system uses SENNA's (Collobert et al., 2011) implementation.

Algorithm 1 shows the pseudo code of the deconstruction stage. The deconstruction stage exploits synergies between SRL and dependency parser, SRL and NER parser. Also, it exclusively examines SRL parser to look for more sentence patterns. First, dependency list, NER list and SRL list are defined with constant string names. These are the templates mentioned in Sect. 3.1.

First loop starting in line 9 to 12 focuses on searching dependency-based templates. Dependency tags found are checked one by one. If any tag matches with our predefined list of dependencies, the designed system checks the validity of the sentence by examining the corresponding semantic representation. This process is performed by examining numbered argument tags (such as ARG0, ARG1) in the semantic representation (please see Sect. 2.2.5 for details). The least numbered argument becomes the subject of the sentence and the other one becomes the object. If the semantic representation has no problem (has a subject, object, verb) and head node of the dependency found matches with verb of a semantic representation, then deconstruction stage begins. Using semantic representation and chunk tags, sentence is separated into its parts, then added to the deconstructed list. If extra fields exist in the semantic representation (such as ARGM-TMP and ARGM-LOC), they are also added.

Next loop starting in line 13 to 18 focuses on searching NER-based templates mentioned in Sect. 3.1.2. It has a similar process with the previous loop. Instead, this time, found NER phrase is removed from the corresponding semantic representation and remaining parts of it becomes the object of the question.

In SRL-based templates starting in line 19 to 24, each key in SRL set is searched in the predefined semantic role labeling list. If the semantic representation conforms to the subject, verb, object pattern, it is valid. Then deconstruction stage takes place for SRL-based templates. Once the semantic representation is valid, there is an extra option for generating a yes/no question. If this option is active, the system is ready to generate questions using the corresponding SRL object.

Algorithm 1 Deconstruction

```
1: function DECONSTRUCT(sentence)
2:     depList ← ["dobj", "acomp", "attr", "pcomp", "ccomp", "oprd", "dative"]
3:     nerList ← ["location", "date", "person", "number"]
4:     srlList ← ["argm-cau", "argm-pnc", "argm-mnr", "argm-loc", "argm-tmp"]
5:     srlTags ← set of found srl lists in the sentence.
6:     depTags ← dependency representation of the sentence.
7:     nerTags ← detected ner tags in the sentence.
8:     chunkTags ← shallow chunking representation of the sentence.
9:     for each d ∈ depTags do
10:        for each s ∈ srlTags do
11:            if depList.match(d.dep_) and s['V'] == d.head.text and
               isValid(s) then
12:                HANDLEDECONSTRUCTION(s, chunkTags, null, d.dep_)
13:    for each d ∈ nerTags do
14:        for each s ∈ srlTags do
15:            for each value ∈ s do
16:                if nerList.match(n.label_) and value.match(n.text) and
                   isValid(s) then
17:                    value ← remove(n.text,value)   ▷ remove found ner
18:                    HANDLEDECONSTRUCTION(s, chunkTags, value,
                       n.label_)
19:    for each s ∈ srlTags do
20:        for each key ∈ s do
21:            if srlList.match(key) and and isValid(s) then
22:                HANDLEDECONSTRUCTION(s,chunkTags)
23:            else if isValid(s) then ▷ option for creating yes/no question
24:                HANDLEDECONSTRUCTION(s, chunkTags, null, key)
25:
26:
27: function HANDLEDECONSTRUCTION(srlObject,chunkTags,modifiedValue,
    type)
28:    fullVerb ← getFullVerb(srlObject['V'],chunkTags)
29:                                                ▷ get sentence parts
30:    if modifiedValue != null then
31:        object ← modifiedValue
32:    else
33:        object ← getObject(srlObject)
34:    subject ← getSubject(srlObject)
35:    extaField ← getExtraField(srlObject)
36:    ADDDECONSTRUCTEDPARTS(fullVerb,subject,object,extaField,type)
```

Algorithm 2 Construction

```
1: function CONSTRUCT
2:     for i in 0...types.length do
3:         verbParts ← convertVerbTense(fullVerb[i])
4:         question ← buildQuestion(types[i], verbParts, objects[i], subjects[i],
5:         extraFields[i])
6:         formattedQuestion ← postProcess(question)
7:         foundQuestions.push(formattedQuestion)
       return formattedQuestion
8:
```

In construction stage, as shown in Algorithm 2 the designed system matches the sentence pattern with predefined rules, i.e.templates. If a rule matches with the sentence pattern, a question can be generated. By examining extracted verb groups with part-of-speech tagging, grammatical tense is detected. The designed system relies on SENNA's (Collobert et al., 2011) part-of-speech tagging algorithm, which uses English Penn Treebank tagset (Marcus, Marcinkiewicz and Santorini, 1993). Then, using Python package named

[†]https://github.com/explosion/spaCy/

as Pattern, which is developed by Smedt and Daelemans (2012), the designed system gets the base form of the verb. Finally, sentence elements are put together to form a question with respect to the detected template type. After checking all templates one by one, the system returns the generated questions as an output.

Our implementation is written in Python 3.6. The designed system relies on

- SENNA's part-of-speech tagging and chunking algorithm,
- AllenNLP's semantic role labeling algorithm,
- SpaCy's dependency parsing and NER algorithm, and
- Python package named as Pattern to get the base form of verbs.

Moreover, two dictionaries are constructed to handle contractions and detect idioms. The source code is available at github[†].

3.3 Trace of Question Generations

In the first place, dependency parser-SRL synergy will be exemplified. Assume that we want to trace an example of question generation for the sentence: "Inflammation is one of the first responses of the immune system to infection". The sentence does not have any contractions to expand, neither idioms to detect. So, we skip the pre-processing stage. Before we begin with the deconstruction stage, sentence is parsed with using parsers explained above. The obtained semantic representation and dependency tags are given below:

Semantic representation: [Inflammation]$_{\text{ARG1}}$ [is]$_{\text{V}}$ [one of the first responses of the immune system to infection]$_{\text{ARG2}}$

Dependency tags:

Then, we deconstruct the sentence. First, we look for dependency based templates in Algorithm 1 in lines 9-12. There is an attr (attribute) dependency between "is" and "one". So, we will search for S-V-attr pattern mentioned in Sect. 3.1.1. S-V-attr pattern is one of the deep question generation templates. In order for a generated sentence to be valid, corresponding semantic representation should consist of a verb and two numbered arguments (such as ARG0 and ARG1). In addition, in dependency based templates (line 11), head of the detected dependency tag should be the same as the verb of the corresponding semantic representation and the tail of the detected dependency tag should belong to one of the numbered arguments in the corresponding semantic representation. Then we can conclude that the template is valid and can be used for question generation.

In dependency tree of the example sentence (Fig. 5), the verb "is" belongs to the head of the detected dependency tag and it is also the verb of the corresponding semantic representation. A similar case also applies to the tail of the detected dependency tag "one". It is a part of one of the numbered arguments in the corresponding semantic representation. We conclude that S-V-attr template is valid.

[†]https://github.com/OnurKeklik/Qg-Iztech

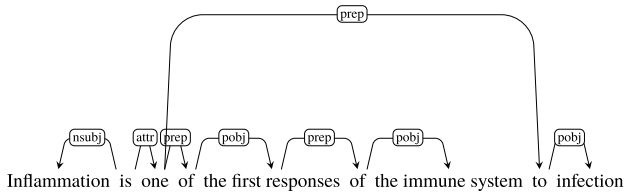

Fig. 5 Dependency tree of example sentence.

Next, we deconstruct the sentence (line 12). In S-V-attr template, the least numbered argument in semantic representation, which is "inflammation", should be the object of the generated question, because we are going to ask the description of "inflammation". For this template, we only need "inflammation" as a deconstructed part. Other parts of the to be generated question are static and selected with respect to detected template type in the construction stage (Algorithm 2, line 4). Object of the to be generated question "inflammation" and template type "attr" are added to the queue for the construction stage (Algorithm 1, line 36).

Since there are no detected NER tags, algorithm passes NER based template generation and jumps to line 19. In SRL based templates, yes/no question is generated for the sample sentence (generation of other predefined SRL based templates in Sect. 3.1.3, can be seen in the next example). In yes/no questions, the least numbered argument in the semantic representation, which is "inflammation" in our example, becomes subject of the question to be generated and the other numbered argument "one of the first responses of the immune system to infection" becomes the object. The verb "is" of the semantic representation becomes the verb of the question to be generated. Then, these question sentence parts are added to the queue for the construction stage. As mentioned, in order to detect the tense of the sentence, the algorithm looks for verb phrases in chunking representation. So, we conclude that there is only one verb and the sentence is in present simple tense form.

In construction stage (Algoritm 2), each deconstructed sentence is iterated. First, if the base form of the verb is needed, it is converted into its base form (line 3). Then, question word is determined and the question is built with respect to detected template type, which is S-V-attr template in our example (line 4). After arranging punctuation and capital letters (with respect to proper nouns), question sentence is ready. So, for the sample sentence given above, two questions, one deep and one shallow, are generated.

- How would you describe inflammation? (S-V-attr) (deep question)
- Is inflammation one of the first responses of the immune system to infection? (S-V) (shallow question)

For the next example, we will only examine SRL based templates. SRL representation of the sample sentence "Phrasal verbs tend to be more common in speech than in writing as they are less formal" can be seen below.

Semantic representation: [Phrasal verbs]$_{\text{ARG1}}$ [be]$_{\text{V}}$ [more common]$_{\text{ARG2}}$ [in speech]$_{\text{ARGM-LOC}}$ [than in

writing]$_{\text{C-ARG2}}$ [as they are less formal]$_{\text{ARGM-CAU}}$

We look for SRL based templates in Algorithm 1 in lines 19-24. There are ARGM-CAU (cause clause) and ARGM-LOC (locative) tags in the SRL representation. Our semantic representation is also valid (it has two numbered arguments and verb). In SRL based templates, the least numbered argument in the semantic representation, which is "Phrasal verbs" in our example, becomes subject of the question to be generated and the other numbered argument "more common" becomes the object. The verb "tend to be" of the semantic representation becomes the verb of the question to be generated. The designed system successfully detects verb phrases using chunking representation of the sample sentence. Then, these question sentence parts are added to the queue for the construction stage (both for ARGM-CAU and ARGM-LOC).

In construction stage (Algoritm 2), question word is determined and the question is built with respect to detected template types, which are S-V-ARGM-CAU and S-V-ARGM-LOC templates in our example (line 4). After arranging punctuation and capital letters (with respect to proper nouns), question sentences are ready. So, for the sample sentence given above, two SRL based questions, one deep and one shallow, are generated.

- Why do phrasal verbs tend to be more common in speech? (S-V-ARGM-CAU) (deep question)
- Where do phrasal verbs tend to be more common? (S-V-ARGM-LOC) (shallow question)

In this study, first, the designed system reveals the semantic structure of the sentence under consideration by using AllenNLP's semantic role labeling. Then, the designed system exploits the synergies between different parsers. If a semantic representation has valid sentence parts, the sentence is deconstructed. Another advantage of using predefined rules with semantic representations is that shallow and deep question generations can be generated distinctively and selectively. Since it is a rule based system, we exactly know which templates generate shallow and deep questions. So, we can select the type of questions we want to generate. For instance, if the question generation is used on tutoring system, only deep question generation can be enabled. In this way, the learner can acquire difficult scientific and technical material in a precise way.

4. Automatic Evaluation

In the automatic evaluation stage, the proposed system's performance is measured using automatic evaluation metrics. Despite the proposed system focuses on question generation from sentences, its performance is compared with Du's reading comprehension system (Du et al. 2017) that focuses on question generation from paragraphs. In order to compare two systems, the proposed system sets up the same evaluation environment as the other system. Both systems use Stanford Question Answering Dataset (SQuAD), which is released by Rajpurkar et al. (2016). Also, both systems use the same evaluation package released by Chen et al. (2015) that includes the implementation of BLEU-1, BLEU-2, BLEU-3, BLEU-4, METEOR and ROUGE-L metrics. Table 5 shows BLEU 1-4, METEOR and ROUGE-L scores of different systems. Our automatic evaluation through these objective neural translation metrics show that our system has superior performance and outperforms the others in BLEU-2, METEOR, and ROUGE-L metrics. Especially for METEOR metric, the proposed system gets highly significant difference. Banerjee et al. (2005) demonstrated that METEOR has significantly enhanced correlation with human evaluators. They also demonstrated that when obtaining high level correlation with human evaluators, recall plays more significant role than precision. Because of the diversity of the templates that the proposed system using, recall thus METEOR score is significantly higher than the other competitive systems.

Table 5 BLEU 1-4, METEOR and ROUGE-L scores of different systems.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
IRBM25	5.18	0.91	0.28	0.12
IREdit Distance	18.28	5.48	2.26	1.06
MOSES+	15.61	3.64	1.00	0.30
DirectIn	31.71	21.18	15.11	11.20
H&S	38.50	22.80	15.52	11.18
Vanilla seq2seq	31.34	13.79	7.36	4.26
Du's Model (no pre-trained)	41.00	23.78	15.71	10.80
Du's Model (w/ pre-trained)	**43.09**	25.96	**17.50**	**12.28**
Proposed Approach	41.90	**26.90**	16.90	10.61

Model	METEOR	ROUGE-L
IRBM25	4.57	9.16
IREdit Distance	7.73	20.77
MOSES+	10.47	17.82
DirectIn	14.95	22.47
H&S	15.95	30.98
Vanilla seq2seq	9.88	29.75
Du's Model (no pre-trained)	15.17	37.95
Du's Model (w/ pre-trained)	16.62	39.75
Proposed Approach	**25.01**	**40.38**

5. Human Evaluation

Human evaluation studies were also performed to measure the quality of generated questions. Human evaluators evaluate the submitted questions according to evaluation criteria, which are difficulty, relevance, syntactic correctness, and ambiguity.

Difficulty is rated to ensure that there is a syntactic divergence between the input sentence and the generated question. That is, some reasoning is necessary to answer the question. The difficult question should assess the reader's knowledge about the input sentence. Relevance is rated to ensure that the question can be answered based on the input sentence. Syntactic correctness is rated to ensure that the question generated is grammatically correct. Finally, ambiguity is rated to ensure that the question makes sense when asked with no context. Typically, an unambiguous question will have one very clear answer.

Table 6 Human evaluation results for generated questions.

Total Questions	Difficulty	Ambiguity	Correctness	Relevance
126	2.85	3.10	3.60	3.65

Table 7 The total number of questions, their types, and the total number of answers.

Question Type	Total Questions	Total Answers
dobj	16	64
acomp	2	8
attr	4	16
pcomp	1	4
date	14	56
number	4	16
person	2	8
location	26	104
direct (yes/no)	45	180
manner (how)	7	28
what	5	20
TOTAL	126	504

Table 8 ANOVA p-values: Question types against the evaluation criteria.

Difficulty	Ambiguity	Correctness	Relevance
9.1e-10	0.0485	1.14e-09	4.22e-10

Relevance, syntactic correctness and ambiguity were previously stated on QGSTEC (2010). 25 sentences are randomly sampled from SQuAD (2016) and QGSTEC (2010). From this set, 126 questions are generated and rated by four different professional english speakers on a 1-5 scale (5 for the best). Table 6 shows the average scores for each evaluation criterion in human evaluation.

Within the generated question set; the total number of questions, their types, and the total number of answers are given in Table 7.

In order to evaluate the suitability of the evaluation criteria in measuring the performance of questions of different types, we applied analysis of variance (ANOVA). ANOVA is used to test the null hypothesis that question types are independent from the evaluation criteria, which are difficulty, ambiguity, correctness, and relevance. As a result, all but one of these criteria got so small p-values that we can safely reject the null hypothesis meaning that there is dependency between these criteria and the generated question types. In other words, these three criteria are proved useful in distinguishing the performance among different question types. Only the criterion of ambiguity does not give a statistically significant p-value (Table 8).

In a study on evaluating evaluation methods in text generation (Stent et al. 2005), although the mean scores for adequacy (a variant of ambiguity) distinguishes one system from the other, the analysis of adequacy scores of a judge on the same set of questions (paired-sample t test) states that his rating for each generated text does not make a difference between the two given systems. However, scores for other human evaluation measures make a statistically significant difference at the individual question level.

Answerability is another ambiguity-related metric that depends on the presence of relevant information such as question type, entities, relations, etc. In a recent study, Nema and Khapra (2018) test based on the human evaluations whether answerability depends on question types or not. Their experimental results show that dependence on question types behaves differently across different datasets, namely WikiMovies, SQuAD, and VQA.

Table 9 Human evaluation results.

	Difficulty	Ambiguity	Correctness	Relevance
H&S	1.94	-	2.95	-
Du's Model	3.03	-	3.36	-
The Proposed Approach	2.85	3.10	3.60	3.65
Human	2.63	-	3.91	-

Our results and similar evidences from the current literature show that ambiguity scores do not manage to distinguish different types of questions. Statistical independence of ambiguity scores from question types can be explained by the fact that this measure is hard for humans to rate consistently across different (types of) questions.

To rate ambiguity, human evaluator is presented with an input sentence, and then a question generated from it, and expected to evaluate the generated question with respect to its independence from the given context as if she was not provided with the input sentence. For humans, it is hard to consistently rate this measure correctly for all along the generated questions considering the context variations due to different input sentences. Seeing both the input sentence and the generated question may cause human evaluators to make slight errors in rating ambiguity thus explaining the neutralization effect among different question types.

Du et al. (2017) also performed human evaluation studies. They both evaluated the performance of the H&S system and their own system. They used two criteria: difficulty and naturalness. Naturalness indicates the grammatical correctness and fluency. It corresponds to correctness criterion that we have mentioned above. They randomly sampled 100 sentence-question pairs in SQuAD and asked four professional English speakers to rate the sentence-question pairs in terms of difficulty and naturalness on a 1-5 scale (5 for the best). They also rated real human-generated sentence-question pairs (ground truth questions) in SQuAD.

As noted above, in our experimental setup, 25 sentences are randomly sampled from SQuAD. From this set, 126 questions are generated and rated by four different professional English speakers on a 1-5 scale (5 for the best). Since our experimentation setup is very similar to Du et al.'s setup, we can compare their results with our own results. Table 9 shows human evaluation results for Du et al.'s system, H&S system and the proposed system. For difficulty metric, the designed system gets score of 2.85 and real human-generated questions gets score of 2,63. Also, for correctness metric, the proposed system gets score of 3,60 and real human-generated questions gets score of 3,91. Although Du et al.'s system generated more difficult questions than the designed system, our difficulty score is much closer to real human-generated questions. Since human-generated

Table 10 Sample questions generated by H&S, Du et al. and the designed system.

Sentence 1: Inflammation is one of the first responses of the immune system to infection.
Human: What is one of the first responses the immune system has to infection? (shallow question)
H&S: What is inflammation one of? (shallow question)
Du: What is one of the first objections of the immune system to infection? (shallow question)
The Designed System: • How would you describe inflammation? (deep question)
• Is inflammation one of the first responses of the immune system to infection? (shallow question)
Sentence 2: However, the rainforest still managed to thrive during these glacial periods, allowing for the survival and evolution of a broad diversity of species.
Human: Did the rainforest managed to thrive during the glacial periods? (shallow question)
H&S: What allowed for the survival and evolution of a broad diversity of species? (shallow question)
Du: Why do the birds still grow during glacial periods? (deep question)
The Designed System: • Did the rainforest manage to thrive during these glacial periods still? (shallow question)
• When did the rainforest manage to thrive? (shallow question)
• What did the rainforest manage still? (shallow question)

questions are taken as ground truth questions, we can say that the question generation system, which has more similar scores to human-generated questions, is better. As can be seen from the results, the designed system significantly outperforms all other systems and turns out to be the most natural (human-like) system.

For further evidence, sample questions generated by H&S system, Du's system, and the designed system can be seen in Table 10. For H&S system, top ranked question is taken from their list. When we look at the sample sentence 1, H&S system failed to detect the object of the sentence. H&S system's pure syntactic approach on question generation sometimes causes failures to detect parts of a sentence. However, the proposed system combines the knowledge of syntactic parsers with semantic roles. This increases the correctness of the generated questions. Moreover, through the use of SRL-based templates like S-V-ARGM-CAU, S-V-ARGM-MNR, and S-V-ARGM-PNC; correct, relevant, and unambiguous deep questions can be generated. From the sentence, H&S and Du's systems generate shallow questions. On the other hand, the designed system generates one shallow and one deep question. The deep question is generated through S-V-attr dependency-based template. The template uses SRL's knowledge to complement the knowledge coming from the dependency parser.

When we look at the sample sentence 2, Du's system generates an over difficult and ambiguous question. This can be attributed to its learning-based algorithm from paragraphs. The designed system does not require any ranking algorithm like H&S system, because it does not overgenerate questions in order to eliminate unacceptable ones. All generated questions with respect to detected templates are accurate, unambiguous and relevant. In addition, using semantic representations the designed system can conclude that if the question has valid parts or not. Again, human evaluation results given in Table 9 confirm this.

6. Conclusion and Future Work

This paper presents a rule-based automatic question generation system. Especially, with respect to METEOR metric, the proposed approach significantly outperforms all others in automatic evaluation stage. Banerjee et al. (2005) demonstrated that METEOR has significantly enhanced correlation with human evaluators. So, our results confirm that statement by performing human evaluation study as well. In conclusion, the proposed approach significantly outperforms all other systems in human evaluation study by generating the most natural (human-like) questions.

Currently, our templates are designed to generate questions from sentences. To improve the performance of paragraph-based questions, we need to investigate how to better use the paragraph-level information. Also, some templates might fit better with some topics than others. This will be explored in future work.

References

[1] D.J. Hacker, J. Dunlosky, and A.C. Graesser, "Metacognition in educational theory and practice," Routledge, 1998.
[2] IBM, IBM Watson Ecosystem - Getting Started Guide, 2014.
[3] V. Rus and C.G. Arthur, The question generation shared task and evaluation challenge workshop report, The University of Memphis, National Science Foundation, Citeseer, 2009.
[4] K. Mazidi and P. Tarau, "Infusing nlu into automatic question generation," Proceedings of the 9th International Natural Language Generation conference, pp.51–60, 2016.
[5] M. Lapata and C. Brew, "Using subcategorization to resolve verb class ambiguity," Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, 1999.
[6] X. Du, J. Shao, and C. Cardie, "Learning to ask: Neural question generation for reading comprehension," Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp.1342–1352, 2017.
[7] M. Heilman and N.A. Smith, "Rating computer-generated questions with mechanical turk," Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's mechanical turk, pp.35–40, Association for Computational Linguistics, 2010.
[8] M. Heilman, Automatic Factual Question Generation from Text, PhD Thesis, Carnegie Mellon University, 2011.
[9] I. Labutov, S. Basu, and L. Vanderwende, "Deep questions without deep understanding," Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), vol.1, pp.889–898, 2015.
[10] V. Rus, B. Wyse, P. Piwek, M. Lintean, S. Stoyanchev, and C. Moldovan, "The first question generation shared task evaluation challenge," Proceedings of the 6th International Natural Language Generation Conference, pp.251–257, Association for Computational Linguistics, 2010.
[11] V. Rus, B. Wyse, P. Piwek, M. Lintean, S. Stoyanchev, and C. Moldovan, "A detailed account of the first question generation shared task evaluation challenge," Dialogue & Discourse, vol.3, no.2, pp.177–204, 2012.
[12] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method

for automatic evaluation of machine translation," Proceedings of the 40th annual meeting on association for computational linguistics, pp.311–318, Association for Computational Linguistics, 2002.
[13] G. Doddington, "Automatic evaluation of machine translation quality using n-gram co-occurrence statistics," Proceedings of the second international conference on Human Language Technology Research, pp.138–145, Morgan Kaufmann Publishers Inc., 2002.
[14] A. Lavie and M.J. Denkowski, "The meteor metric for automatic evaluation of machine translation," Machine translation, vol.23, no.2-3, pp.105–115, 2009.
[15] C.-Y. Lin, Rouge: A package for automatic evaluation of summaries, Text Summarization Branches Out, 2004.
[16] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C.L. Zitnick, "Microsoft coco captions: Data collection and evaluation server," arXiv preprint arXiv:1504.00325, 2015.
[17] M. Richardson, C.J.C. Burges, and E. Renshaw, "Mctest: A challenge dataset for the open-domain machine comprehension of text," Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp.193–203, 2013.
[18] A.M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp.379–389, 2015.
[19] K.M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," Advances in Neural Information Processing Systems, pp.1693–1701, 2015.
[20] D. Chen, J. Bolton, and C.D. Manning, "A thorough examination of the cnn/daily mail reading comprehension task," Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp.2358–2367, 2016.
[21] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp.2383–2392, 2016.
[22] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," Journal of Machine Learning Research, vol.12, pp.2493–2537, Aug. 2011.
[23] M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of english: The penn treebank," Computational linguistics, vol.19, no.2, pp.313–330, 1993.
[24] M. Palmer, D. Gildea, and P. Kingsbury, "The proposition bank: An annotated corpus of semantic roles," Computational linguistics, vol.31, no.1, pp.71–106, 2005.
[25] O. Babko-Malaya, Propbank annotation guidelines, URL: http://verbs.colorado.edu, 2005.
[26] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. Liu, M. Peters, M. Schmitz, and L. Zettlemoyer, Allennlp: A deep semantic natural language processing platform.
[27] T.D. Smedt and W. Daelemans, "Pattern for python," Journal of Machine Learning Research, vol.13, pp.2063–2067, June 2012.
[28] S. Banerjee and A. Lavie, "Meteor: An automatic metric for MT evaluation with improved correlation with human judgments," Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization, pp.65–72, 2005.
[29] A. Stent, M. Marge, and M. Singhai, "Evaluating Evaluation Methods for Generation in the Presence of Variation," Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing, vol.3406, pp.341–351, 2005.
[30] P. Nema and M.M. Khapra, "Towards a Better Metric for Evaluating Question Generation Systems," Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp.3950–3959, 2018.

Onur Keklik received the B.S. degree in software engineering from Izmir University of Economics in 2014. He is a M.Sc. student in computer engineering at Izmir Institute of Technology. His current research interests include natural language processing and computer vision. He is actively working in game development industry.

Tugkan Tuglular received the B.S., M.S., and Ph.D. degrees in Computer Engineering from Ege University, Turkey in 1993, 1995 and 1999, respectively. He worked as a research associate at Purdue University from 1996 to 1998. He joined Izmir Institute of Technology in 2000. Currently, he is working as an assistant professor at Izmir Institute of Technology, Department of Computer Engineering. His current research interests include model-based development, model-based testing, and natural language processing.

Selma Tekir studied B.S. in Computer Engineering at Ege University. She received her M.S. in Computer Engineering from Izmir Institute of Technology. In 2009, she worked as a visiting researcher at Faculty of Computer Science Chair for Databases, Data Analysis and Visualization at University of Konstanz-Germany. Tekir received a Ph.D. in computer engineering from Ege University in 2010. Her research interests include text mining, news analysis, semantics, natural language processing, and information warfare. Currently, she is working as an assistant professor at Izmir Institute of Technology Department of Computer Engineering.