Generating Questions and Multiple-Choice Answers using Semantic Analysis of Texts

Jun Araki, Dheeraj Rajagopal, Sreecharan Sankaranarayanan,
Susan Holm, Yukari Yamakawa, Teruko Mitamura
Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
junaraki@cs.cmu.edu, dheeraj@cs.cmu.edu, sree@cmu.edu,
sh4s@andrew.cmu.edu, yukariy@andrew.cmu.edu, teruko@cs.cmu.edu

Abstract

We present a novel approach to automated question generation that improves upon prior work both from a technology perspective and from an assessment perspective. Our system is aimed at engaging language learners by generating multiple-choice questions which utilize specific inference steps over multiple sentences, namely coreference resolution and paraphrase detection. The system also generates correct answers and semantically-motivated phrase-level distractors as answer choices. Evaluation by human annotators indicates that our approach requires a larger number of inference steps, which necessitate deeper semantic understanding of texts than a traditional single-sentence approach.

1 Introduction

Exam questions are an indispensable tool for teachers to assess their students' understanding of material. Thus, automatic question generation from text is a key natural language processing technology to aid teachers in examining learners' reading comprehension. Past studies in education showed that higher-level questions, in contrast to simple factoid questions, have more educational benefits for reading comprehension (Anderson and Biddle, 1975; Andre, 1979; Hamaker, 1986). However, most of existing approaches to question generation have focused on generating questions from a single sentence, relying heavily on syntax and shallow semantics with an emphasis on grammaticality (Mitkov and Ha, 2003; Chen et al., 2009; Heilman and Smith, 2010; Curto et al., 2011; Becker et al., 2012; Lindberg et al., 2013; Mazidi and Nielsen, 2014). A problem with this approach is that the majority of questions generated from single sentences tend to be too specific and low-level to properly measure learners' understandings of the overall contents of text. In other words, what is assessed by such question generation systems ends up essentially being the ability to compare sentences, just requiring learners to find a single sentence that has almost the same surface form as a given interrogative sentence. Results of simple sentence comparisons do little to contribute towards the goal of assessing learners' reading comprehension.

In this work, we propose a question generation approach that engages learners through the use of specific *inference steps* over multiple sentences, namely coreference resolution and paraphrase detection, requiring more semantic understanding of text. We primarily use event and entity coreference as a source of producing questions from multiple sentences. Grounded by the past studies in education, we believe that such high-level questions are more sophisticated and educationally valuable for testing reading comprehension than questions generated by the traditional single-sentence approaches. Our question generation strategy is novel in two ways. First, it employs event and entity coreference between antecedents and referring mentions spanning multiple sentences. This requires learners to resolve the coreference and understand the contents of the text semantically. Second, it makes use of paraphrases when generating questions. The resulting questions are able to check learners' lexical knowledge.

To ensure that the resulting multiple choice question is an adequate test of the learner's reading comprehension, incorrect answers, i.e., distractors are necessary. The distractor candidates must be selected to ensure difficulty (Candidate Selection) and compared against the correct answer to ensure that there is only one correct answer overall and the question remains solvable (Reliability Checking). Most systems

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 1125–1136, Osaka, Japan, December 11-17 2016.

currently generate word-level distractors for testing grammar, and are intimately tied to the question generation method. Our distractor generation method is capable of automatically generating phrase-level distractors leveraging event triggers and event-event relations for the purpose of testing reading comprehension. The results of our experiments show that our method produces comparable results across two different methods of question generation, thus remaining robust to different question generation methods.

Our question generation system is aimed at enhancing the reading comprehension ability of language learners, more specifically, students who learn English as a second language (ESL). Therefore, our ultimate goal is to generate multiple-choice questions from plain texts in an arbitrary domain. However, the state-of-the-art in extracting semantic representations of event and entity relations from text does not perform well enough to support our question generation approach. Thus, the evaluation of question generation relying on automated semantic relation extraction is not practical at this time. Therefore, in this work we use texts and expert human annotations from the ProcessBank corpus[1] (Berant et al., 2014) to facilitate our semantics-oriented question generation.

Our contributions are as follows:

1. This is the first work to automatically generate questions from multiple sentences, involving specific inference steps such as coreference resolution and paraphrase detection. Our experiments show that questions generated by our approach require taking a larger number of inference steps while ensuring comparable grammatical correctness and answer existence, as compared to questions generated by a traditional single-sentence approach.
2. We also present a complementary system which generates questions based on patterns extracted from relationships between events and entities in the passage. This approach produces a higher number of questions without the need for any manual intervention for the generation of patterns, although less reliable compared to the previous question generation system.
3. To complement the complex questions generated from multiple sentences, our system also generates phrase level distractors aimed at challenging comprehension by using event-event relation annotations. The results of our experiments show that the quality of the generated distractors remains robust across two different methods of question generation.

2 Related Work

Among a considerable amount of prior work on question generation spanning nearly four decades, we focus mainly on two families: (1) *syntax-based transformation* and (2) *shallow semantics-based transformation*. Syntax-based transformation uses the output of syntactic parsers, and changes syntactic structures of sentences to convert them into interrogative sentences as questions (Wolfe, 1976; Mitkov and Ha, 2003; Judge et al., 2006; Heilman and Smith, 2010; Curto et al., 2011). In contrast, shallow semantics-based transformation uses the output of shallow semantic parsers and additional knowledge bases, and generates questions from sentences based on their semantic structures (Chen et al., 2009; Mannem et al., 2010; Becker et al., 2012; Yao et al., 2012; Mazidi and Nielsen, 2014). Both groups generate questions based on a single sentence of text, relying heavily on its argument structures in question construction and mostly emphasizing grammaticality in evaluation. Labutov et al. (2015) have recently presented an approach to generate high-level questions using ontology-derived templates. Our approach differs from theirs in that we leverage semantic representations of text to inject specific inference steps, such as event and entity coreferences, into the process of answering system-generated questions.

In the context of computer-assisted language learning, intelligent tutoring systems have been traditionally employed for generating questions in various domains (Koedinger et al., 1997; Vanlehn et al., 2005). Some of the research in this area uses questions as a way of engaging students in ongoing dialogue (Graesser et al., 2001; Piwek and Stoyanchev, 2010). Our underlying motivation to enhance the reading comprehension ability of non-native English readers aligns best with the Reader Specific Lexical Practice (REAP) system (Heilman et al., 2006) and the SmartReader system (Azab et al., 2013). Though very promising, state-of-the-art approaches in this area have not dealt with multiple sentences to generate questions, either. Our work is aimed at addressing this challenge.

[1] See Section 3.1 for details of this corpus.

Prior work in distractor generation has been mostly studied on cloze (gap-fill) questions. Distractor generation for cloze questions often comprises two steps. *Candidate Selection* controls the type and difficulty of the items, and is intimately tied to the intent and target audience for the questions. This step may take advantage of available datasets such as the Lang-8 corpus[2] to identify confusion pairs and use the most frequent learner confusions as distractors (Sakaguchi et al., 2013) or the set of English language prepositions for use as distractors in preposition testing tasks (Lee and Seneff, 2007). *Reliability Checking* ensures that the question remains solvable, i.e., there is a total of one correct answer only (Zesch and Melamud, 2014).

In most cases, however, only the question and the correct answer are available, and distractors have to be automatically generated using these as input. Random generation of distractors (Mostow and Jang, 2012), distractors with a corpus frequency comparable to the correct answer (Hoshino and Nakagawa, 2007), morphologically, orthographically or phonetically similar words as distractors (Pino and Eskenazi, 2009), and semantically similar words selected using taxonomies (Hoshino and Nakagawa, 2007; Mitkov et al., 2009), thesauri (Sumita et al., 2005; Smith et al., 2010) or an additional layering of semantic relatedness to the context (sentence or paragraph) (Pino and Eskenazi, 2009; Agarwal et al., 2011; Mostow and Jang, 2012) are all valid strategies for the generation of distractor items. However, most of these methods generate word-level distractors for testing grammar rather than comprehension. Moreover, distractor generation is often tied to the method of generating questions. Our method automatically generates phrase-level distractors for testing reading comprehension, and is decoupled from the method used for question generation.

3 Generating Questions and Multiple-Choice Answers

Our system comprises a question generation component and a distractor generation component. We show a high-level overview of the system in Figure 1. The question generation component first detects question targets, and then generates questions triggered by the targets as correct answers. This component implements two different template-based algorithms, as described in Section 3.2 and Section 3.3. Given a tuple (text, question, answer) as input, the distractor generation component generates n distractors per question. The resulting tuple (question, answer, distractors) forms a multiple-choice question as the final output of our system.

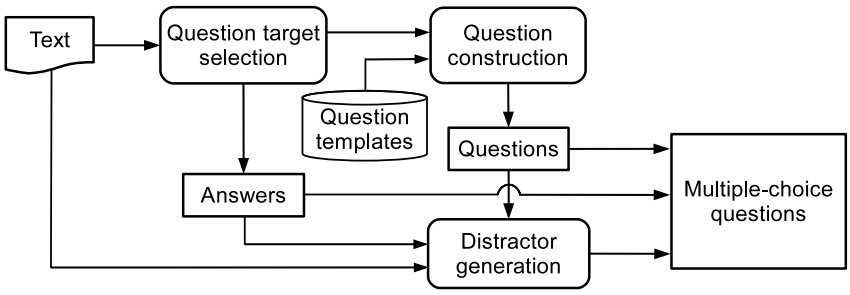


Figure 1: A high-level overview of our multiple-choice question generation system.

3.1 The ProcessBank Corpus

In order to generate questions from multiple sentences using reliable annotation of event and entity coreference, we use the ProcessBank corpus[3], particularly utilizing the expert annotation of events and entities, as described in Section 1. The corpus consists of 200 paragraphs about biological processes, extracted from the high school level textbook *Biology* (Campbell and Reece, 2005). Such textbooks are ideal sources to test learners' reading comprehension from an educational perspective. The corpus includes rich process structures annotated by biologists, shown in Figure 2. More specifically, the expert annotation comprises entity mentions, entity coreference, event triggers (without any event types), arguments (with semantic roles), and event-event relations such as event coreference and causal relations.

[2]A corpus of manually annotated errors by ESL learners, available at `http://cl.naist.jp/nldata/lang-8/`

[3]`http://www-nlp.stanford.edu/software/bioprocess/`

Each event in the corpus represents a biological process. A trigger is defined as a word or a phrase that expresses the process most clearly, and an argument is defined as a phrase denoting entities that participate in the process. The corpus also includes multiple-choice questions per paragraph, created by the biologists. We refer to these expert questions to devise our question templates.

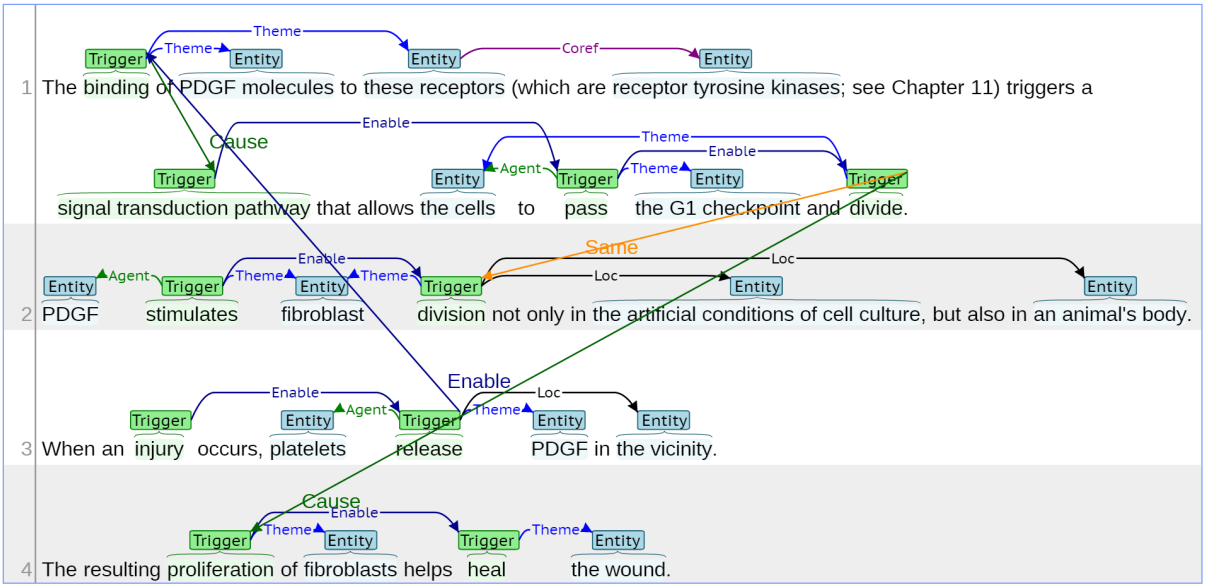


Figure 2: A paragraph with annotation of events, entities and their relations in ProcessBank. A 'Same' link means event coreference, whereas a 'Coref' link means entity coreference.

3.2 Question Generation using Coreferences and Paraphrases

Our first question generation system (QG1) is aimed at generating questions from multiple sentences using three semantic relations: event coreference, entity coreference, and paraphrases. We recognize that one of the key learning points for biology students is the order of biological processes (events) because many of the expert questions in ProcessBank ask about it. Based on this observation, we devise question patterns and templates shown in Table 1. As seen in this table, pattern P2 and P3 focus on the order of biological processes. However, the whole question generation strategy, including P2 and P3, does not rely on any domain-specific knowledge or rules on biology, but instead it makes use of domain-independent semantic relations, such as event coreferences and causal relations. Hence, the question generation strategy is not restricted to the biology domain, and is portable to other domains from an algorithm perspective.

QG1 consists of two parts: **answer generation** and **question construction**. It first finds question patterns applicable to the given text. For example, pattern P3 is applicable to the paragraph of Figure 2 since "divide" (E1) in the first sentence and "division" (E2) in the second sentence are coreferent, and only the former trigger has a 'Cause' relation to "proliferation" (E3). This pattern match means that "proliferation" can be an answer. We then make use of arguments of the answer trigger to generate a more complete answer, obtaining "proliferation of fibroblasts". Next, the algorithm constructs a question given the matched pattern. In the case of the example above, "division" is a nominal trigger. Thus, the algorithm uses question template T5, and creates a question "What is a result of the fibroblast division not only in the artificial conditions of cell culture, but also in an animal's body?" As seen in this example, the algorithm takes advantage of the fact that E2 lacks a piece of information that E1 has in P1, P2 and P3. We only use event coreference E1-E2 where E1 and E2 exist in different sentences, ensuring that questions are generated from multiple sentences.

We also give examples to illustrate how to generate questions based on entity coreferences and paraphrases in Figure 3 and Figure 4, respectively. Pattern P4 applies to Figure 3 because "they" (En1) in the second sentence is coreferent with "a few tumor cells" (En2) in the first sentence, and the former

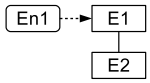
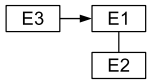
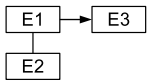
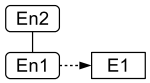
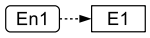
Semantic relation	Question patterns	Answer	Question templates
Event coreference	P1. 	En1	T1. What [*verbal trigger + subsequent arguments*]?
	P2. 	E3	T2. What causes [*nominal trigger + subsequent arguments*]? T3. What makes it happen to [*verbal trigger + subsequent arguments*]? T4. What makes it happen that [*event clause*]?
	P3. 	E3	T5. What is a result of [*nominal trigger + subsequent arguments*]? T6. What happens when [*event clause*]?
Entity coreference	P4. 	En2	T1. What [*verbal trigger + subsequent arguments*]?
Paraphrase	P5. 	En1	

Table 1: Question patterns and templates using event coreference, entity coreference, and paraphrases. In question patterns, En denotes an event trigger, and Enn an entity mention. A straight line denotes a coreference link, a dashed arrow an 'Agent' relation, and a straight arrow a relation which is 'Cause', 'Enable' or 'Result'. An event clause in question templates is defined as a text span including an event trigger and its arguments.

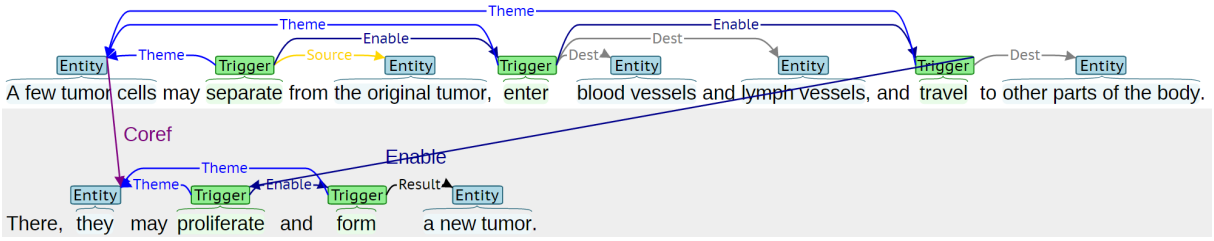


Figure 3: An example text to generate a question using an entity coreference.

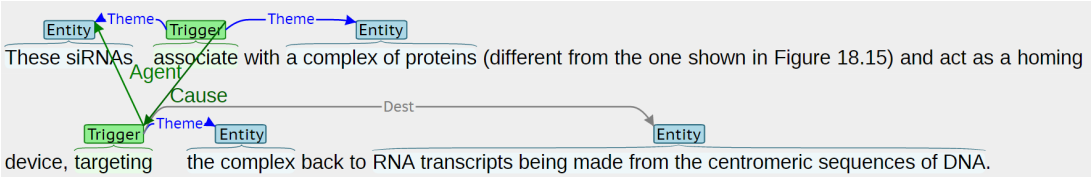


Figure 4: An example text to generate a question using a paraphrase.

entity is an agent of triggers "proliferate" and "form". Given that, "a few tumor cells" can be an answer, and the system can generate a question "What may proliferate and form a new tumor?" Similarly, P5 applies to Figure 4 because "these siRNAs" are an agent of trigger "targeting", and "complex" has a paraphrase "composite". Since "these siRNAs" are coreferent with "siRNAs" in the previous sentence (not appearing in Figure 4), a system can generate an answer "siRNAs" and a question "What targets the composite back to RNA transcripts being made from the centromeric sequences of DNA?" Note that the paraphrase "composite" is inserted into the question.

3.3 Question Generation using Concept Relationships

Our alternate question generation system (QG2) uses question patterns from events and entities in the passage and their relations. In this system, we rely on the fact that we can extract the question patterns based on the relations between entities and events in a passage. The relations between these events and entities span across multiple sentences, and thus most questions are generated from multiple sentences. We first extract generic patterns from the existing questions and relations, and then apply them to unseen passages to generate new questions. Since we rely on generic patterns, a significant number of questions might not have answers in the passage in comparison to QG1. For example, let us consider the relation (*Entity Theme Trigger*), for which one of question patterns is "What happens because of *Entity Trigger*?". A sample question from the passage in Figure 2 is "What happens because of PDFG release?" which does not have a valid answer in the passage. The ambiguity of the *Theme* relation gives rise to question patterns that might or might not have a correct answer in the passage. Alternatively, let us consider the relation (*Trigger_1 Enables Trigger_2*). One possible question can be "*What is enabled by Trigger_1*?" which would have *Trigger_2* as the correct answer. We then apply this pattern to other passages when triggers have the same kind of relations. Patterns from the *Enables* relation tend to have an answer when they are applied to new passages. Table 2 shows a sample of possible relations from the passage shown in Figure 2.

Relation	Entity	Location	Theme	Event Trigger	Event Trigger	Coreference
Enable	–	–	–	injury	release	–
Agent	the cells	–	–	pass	–	–
–	divide	–	–	–	–	division
Theme	–	–	–	binding	PDFG molecules	–

Table 2: A sample of possible relations for the passage in Figure 2.

Some sample questions that can be generated from this passage are shown below.

1. What event enables *Event_Trigger* (division)?
2. What would happen without *Event_Trigger* (the binding) of *Event_Trigger* (PDFG molecules)?
3. What happens after *Event_Trigger* (passage) of a *Entity* (the cells)?

Table 3 shows some of the relations and question templates extracted from the ProcessBank corpus.

Question pattern	Question template
Entity *Result* Trigger	What event should occur before the *Trigger* of *Entity*?
Trigger_1 *Super* Trigger_2	What would happen without *Trigger_1* in *Trigger_2*?
Entity *Theme* Trigger	What would happen without the *Trigger* of *Entity*?
Entity *Location* Trigger	Where was *Trigger* in *Entity*?
Trigger_1 *Cause* Trigger_2	What is caused by *Trigger_1*?

Table 3: Entity-trigger relations as question patterns and their associated question templates.

3.4 Distractor Generation

Our distractor generation method can be described as a sequence of the following steps.

1. **Generation of an event graph using event triggers and event-event relations**. The method first forms an event graph as a basis to find possible distractors. Event triggers act as nodes of the graph whereas event-event relations act as edges of the graph. Coreferent nodes in this initial graph are collapsed together to construct the final event graph.
2. **Mapping a question and its correct answer onto the event graph**. In this step, we identify the nodes (event triggers) corresponding to those present in the question and the correct answer. The identified nodes are not used for the generation of distractors. This ensures the reliability of generated distractors, and serves as our *Reliability Checking* step.
3. **Selection of event triggers for distractor generation**. Nodes (event triggers) in the event graph, which are not the ones identified in the previous step, serve as potential candidates for generating distractor items. While informed selection strategies such as the use of causal relations and information regarding prior and subsequent events could have been used to select distractors, they are effective only if the correct answer is known with complete certainty. To combat the stochasticity around the correctness of the answer generated by our question generation methods, we employ random selection from among the potential candidates. We randomly select three nodes as distractors from the list of nodes identified as potential candidates (after removing the nodes in the question and the correct answer). This serves as our *Candidate Selection* step.
4. **Construction of distractor items**. A distractor item is constructed by selecting the phrase comprised of a selected node (event trigger) and its surrounding entities, provided no comma or period is encountered. We resolve entity and event coreferences associated with terms in this step ito produce the final distractor items.

Let us consider as an example the passage shown in Figure 2. Step 1 generates the event graph obtained from this passage as shown in Figure 5. The nodes "divide" and "division" are coreferent, and therefore have been collapsed into a single node "divide/division" along with their edges. Now consider

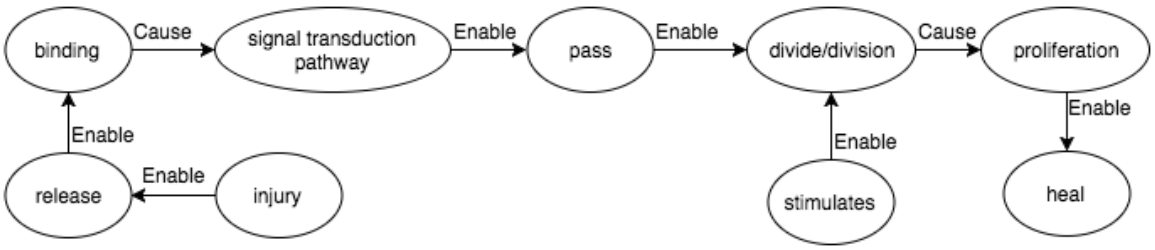


Figure 5: An event graph generated from the passage in Figure 2.

the question and answer we saw associated with this passage in Section 3.2:

- Question: "What is a result of the fibroblast division not only in the artificial conditions of cell culture, but also in an animal's body?"
- Answer: "Proliferation of fibroblasts"

Step 2 recognizes that the nodes "divide/division" and "proliferation" in the event graph are the triggers that the question and answer refer to. We can now use any of the other event triggers from the event graph to generate our distractor items. Since we collapse coreferent nodes, we can guarantee that the nodes selected will not refer to the same triggers as the ones in the question or the answer (*Reliability Checking*). From among the remaining nodes, step 3 randomly selects "binding", "stimulates" and "release" for distractor generation. Lastly, step 4 generates phrases with these triggers and their surrounding entities after entity and event coreference resolution, and constructs the distractor items shown below.

- Distractor 1: binding of PDGF molecules to receptor tyrosine kinases
- Distractor 2: PDGF stimulates fibroblast
- Distractor 3: platelets release PDGF

4 Experiments and Results

To assess the performance of our system, two human annotators evaluate our question generation component and distractor generation component. For a meaningful comparison on question generation, we

use the question generation system by Heilman and Smith (2010) as a baseline. Let MH refer to the baseline. In our experiments, we generate 200 questions from each system, and generate 3 distractors per question.

4.1 Evaluation Criteria for Generated Questions and Distractors

It is important to evaluate generated questions, but this is not straightforward mainly due to the wide variety of acceptable natural language expressions. We use three metrics for the evaluation.

Grammatical correctness judges whether a question is syntactically well-formed. It does not evaluate whether a question is semantically coherent, ignoring the meaning of the question. Our three point scale for this metric is based on the number of grammatical errors.

- 1 (best): The question has no grammatical errors.
- 2: The question has 1 or 2 grammatical errors.
- 3 (worst): The question has 3 or more grammatical errors.

For consistency in counting grammatical errors, we define common grammatical errors in English: spelling errors, run-on sentences, lack of subject-verb agreement, lack of pronoun-antecedent agreement, misplaced modifiers, missing or erroneous quantifiers, prepositions or determiners, erroneous verb forms or nominalization, incorrect word choice, and other errors.

Answer existence identifies whether the answer to a question can be inferred from the passage associated with the question. Note that the answer must be inferred using the passage information only, without relying on external knowledge beyond the passage. Even if a system generates a question while making a specific target its answer, it could be impossible that the target is the answer due to the lack of a valid inference path from the question to the target as its answer. This metric is intended to penalize such questions. Our two-point scale for this metric is:

- 1 (yes): The answer to the question can be inferred from the passage.
- 2 (no): The answer to the question cannot be inferred from the passage.

In addition to answer existence, we also evaluate the correctness of system-generated answers. For this, we use the following three-point scale ratings: correct (1), partially correct (2), and incorrect (3).

Inference steps concern how many semantic relations humans need to understand in order to answer a question. This metric directly evaluates our central idea: inference steps for answering a question. We define the following set of semantic relation types to be considered as inference:

- Event coreference within input text and event coreference between input text and a question.
- Entity coreference within input text and entity coreference between input text and a question.
- Paraphrases in input text and a question.
- Negation, which is a binary relation about logical truthness.

Distractor quality is a rating to measure how appropriate a distractor is for a given question and its correct answer. We set up a three-point scale for rating generated distractors as follows:

- 1 (worst): A distractor is confusing because it overlaps the correct answer partially or completely.
- 2: A distractor can be easily identified as an incorrect answer.
- 3 (best): A distractor can be viable.

As for rating 2, we look for a particular reason for being easily eliminated, such as the distractor is not present in given text.

4.2 Results of Question Generation

We show our results of question generation in Table 4. QG1 achieved more inference steps compared to QG2 by 0.49 and compared to MH by 0.60, while it gained comparable ratings of grammatical correctness and answer existence. We computed the inter-annotator agreement with Cohen's Kappa for each of the criteria mentioned in Section 4.1. Overall, we have a kappa value of 0.55, 0.58 and 0.49 for grammatical correctness, answer existence and inference steps respectively. This result implies moderate agreement. Table 5(a) shows our results for answer correctness. We observe that QG1 tends to generate questions with more incorrect answers than MH.

System	Grammatical correctness			Answer existence			Inference steps		
	Ann 1	Ann 2	Total	Ann 1	Ann 2	Total	Ann 1	Ann 2	Total
QG1	1.52	1.48	1.50	1.17	1.26	1.21	0.80	0.71	**0.76**
QG2	2.13	2.07	2.10	1.58	1.75	1.67	0.31	0.20	0.27
MH	1.42	1.25	**1.34**	1.20	1.14	**1.17**	0.13	0.19	0.16

Table 4: The performance comparison in question generation. Numbers in grammatical correctness and answer existence are average ratings, and lower is better. Numbers in inference steps are average inference steps, and higher is better.

System	Ann 1	Ann 2	Total
QG1	1.35	1.57	1.46
MH	1.08	1.13	1.11

(a) Average ratings of answer correctness in 200 questions. Lower numbers are better. Scores range from 1-3, with 1 a correct answer.

System	Ann 1	Ann 2	Total
QG1	1.98	1.90	1.94
MH	1.93	1.88	1.91

(b) Average ratings of distractors on the same set of 100 questions that the both human evaluators rate as answer existence 1 ("an answer exists"). Higher numbers are better. Scores range from 1-3, with 3 a viable distractor.

Table 5: Results of answer correctness (Table 5(a)) and distractor generation (Table 5(b)).

4.3 Results of Distractor Generation

To make the evaluation of distractor generation reasonable, we need the same set of questions as input for QG1 and MH. Both of the two evaluators rate answer existence 1 on 129 questions generated from QG1 and on 127 questions generated from MH. We randomly select 100 questions from each of the question sets. We show our results of question generation in Table 5(b). Our method achieves an average distractor score of around 2 with a majority of the distractors being rated 2.

5 Discussion

As described in Section 3.2, QG1 attempts to generate questions involving at least one inference step. The average inference step of 0.76 in Table 4 means that the algorithm fails to generate intended questions approximately once out of every four times. A common source of these errors is that some other events can be associated with the event in the question by a different relation (e.g., 'Super'), and they can be an answer to the question. The most common kind of grammatical errors in questions generated by the QG2 system were nominalization ("grow" vs "growth") and verb form ("is uses" instead of "is used"). In terms of inference steps, 'entity coreference' occurred most often. In contrast, QG1 commonly made "determiner" errors in the case of question patterns involving a nominal trigger. For instance, "natural selection" is a noun phrase which does not need an article, but QG1 mistakenly adds "the" in front of it.

The distractor generation component needs to generate distractors as close to a rating of 3 as possible. However, distractors labeled as 2 ("easily eliminated") often occur because they come from events preceding the event described in the question or from events following the results of the events described in the question. A better understanding of cause-and-effect or temporal relations in event graphs might reduce the selection of these easily identified distractors.

We also faced some issues in evaluating system output. The number of grammatical errors assigned can depend on annotators' opinions on how the errors should be corrected. Different corrections could be reached by a different number of steps, changing the evaluation. Thus, differences in opinion on the correct form of the question impacted inter-annotator agreement. Another challenge concerned evaluating the inference steps needed to answer the generated questions. This evaluation requires annotators to identify semantic relations (i.e., event and entity coreferences, negations, and paraphrases) in the texts and count the number of steps needed to answer the question. In some cases, it was not clear how many steps to count, when there were several event mentions in a coreference chain.

6 Conclusion and Future Work

We have presented two different methods that automatically generate questions from multiple sentences. The first method requires learners to take specific inference steps such as event and entity coreferences over multiple sentences. The second method generates questions using patterns extracted from the relations between entities and events in a corpus. Our experiments showed that questions generated by both methods require more inference steps than questions generated by a traditional single-sentence approach. In particular, the first method outperforms the baseline system in terms of the number of inference steps by 0.60 on average, while ensuring comparable grammatical correctness and answer existence. Grounded by past studies in education, we believe that our system-generated questions are more sophisticated and educationally valuable for testing reading comprehension because they require more semantic understanding of text. Our distractor generation approach uses event triggers and event-event relations to generate distractors by building an event graph that lends itself to superior Reliability Checking. We observed that the quality of the generated distractors remains robust across two different methods of question generation.

There are a number of avenues for future work. Although our question generation strategy is domain-independent as described in Section 3.2, some question patterns such as the ones focusing on the order of events (biological processes) might not be useful in other domains. One could explore more domain-adaptable question generation strategies. In another direction, one can extend our system to achieve an end-to-end system which generates multiple-choice questions directly from input text by leveraging automated high-performance semantic parsers, instead of relying on human annotations of a particular corpus. As for distractor generation, one can look at more intelligent ways of generating distractor items from the event graph using causal relations and the knowledge of prior and subsequent events. From a perspective of evaluation, a real user test with non-native English readers is also important. The real-test evaluation will allow us to know how many of the system-generated questions are usable in the real test, obtain the insights of what questions are truly important for language learners' reading comprehension, and capture the rationale behind distractors (e.g., the most confusing distractor patterns) based on an analysis of readers' performance. One could also explore methods to automatically carry out one or more of our evaluation processes. As we argued in this paper, our question generation strategy lends itself to language learners' reading comprehension. In addition, our research can also be useful to the task of creating exam questions and answers, since manual creation is normally quite time-consuming. Automatically generated questions and multiple-choice answers make the creation of exam QAs more efficient.

Acknowledgements

This publication was partly made possible by grant NPRP-08-1337-1-243 from the Qatar National Research Fund (a member of the Qatar Foundation). The statements made herein are solely the responsibility of the authors. Jun Araki is partly supported by an IBM Ph.D. Fellowship and a Funai Overseas Scholarship.

References

Manish Agarwal, Rakshit Shah, and Prashanth 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.

Richard C. Anderson and W. Barry Biddle. 1975. On asking people questions about what they are reading. *Psychology of Learning and Motivation*, 9:90–132.

Thomas Andre. 1979. Does answering higher level questions while reading facilitate productive learning? *Review of Educational Research*, 49(2):280–318.

Mahmoud Azab, Ahmed Salama, Kemal Oflazer, Hideki Shima, Jun Araki, and Teruko Mitamura. 2013. An English reading tool as a NLP showcase. In *Proceedings of IJCNLP 2013: System Demonstrations*, pages 5–8.

Lee Becker, Sumit Basu, and Lucy Vanderwende. 2012. Mind the gap: Learning to choose gaps for question generation. In *Proceedings of NAACL-HLT 2012*, pages 742–751.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *Proceedings of EMNLP 2014*, pages 1499–1510.

Neil Campbell and Jane Reece. 2005. *Biology*. Benjamin Cummings.

Wei Chen, Gregory Aist, and Jack Mostow. 2009. Generating questions automatically from informational text. In *Proceedings of the 2nd Question Generation Workshop*.

Sérgio Curto, Ana Cristina Mendes, and Luísa Coheur. 2011. Exploring linguistically-rich patterns for question generation. In *Proceedings of the UCNLG+Eval: Language Generation and Evaluation Workshop*, pages 33–38.

Arthur C. Graesser, Kurt VanLehn, Carolyn P. Ros, Pamela W. Jordan, and Derek Harter. 2001. Intelligent tutoring systems with conversational dialogue. *AI Magazine*, 22(4):39–51.

Christiaan Hamaker. 1986. The effect of adjunct questions on prose learning. *Review of Educational Research*, 56(2):212–242.

Michael Heilman and Noah A. Smith. 2010. Good question! Statistical ranking for question generation. In *Proceedings of NAACL-HLT 2010*, pages 609–617.

Michael Heilman, Kevyn Collins-thompson, Jamie Callan, and Maxine Eskenazi. 2006. Classroom success of an intelligent tutoring system for lexical practice and reading comprehension. In *Proceedings of the 9th International Conference on Spoken Language Processing*.

Ayako Hoshino and Hiroshi Nakagawa. 2007. Assisting cloze test making with a web application. In *Proceedings of Society for Information Technology and Teacher Education International Conference*.

John Judge, Aoife Cahill, and Josef van Genabith. 2006. QuestionBank: Creating a corpus of parse-annotated questions. In *Proceedings of ACL/COLING 2006*, pages 497–504.

Kenneth R Koedinger, John R Anderson, William H Hadley, and Mary A Mark. 1997. Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, 8:30–43.

Igor Labutov, Sumit Basu, and Lucy Vanderwende. 2015. Deep questions without deep understanding. In *Proceedings of ACL/IJCNLP 2015*, pages 889–898.

John Lee and Stephanie Seneff. 2007. Automatic generation of cloze items for prepositions. In *Proceedings of INTERSPEECH 2007*, pages 2173–2176.

David Lindberg, Fred Popowich, John Nesbit, and Phil 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.

Prashanth Mannem, Rashmi Prasad, and Aravind Joshi. 2010. Question generation from paragraphs at UPenn: QGSTEC system description. In *Proceedings of the 3rd QG Workshop*.

Karen Mazidi and Rodney D. Nielsen. 2014. Linguistic considerations in automatic question generation. In *Proceedings of ACL 2014*, pages 321–326.

Ruslan Mitkov and Le An Ha. 2003. Computer-aided generation of multiple-choice tests. In *Proceedings of the NAACL-HLT 2003 Workshop on Building Educational Applications Using Natural Language Processing*, pages 17–22.

Ruslan Mitkov, Le An Ha, Andrea Varga, and Luz Rello. 2009. Semantic similarity of distractors in multiple-choice tests: extrinsic evaluation. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 49–56.

Jack Mostow and Hyeju Jang. 2012. Generating diagnostic multiple choice comprehension cloze questions. In *Proceedings of the 7th Workshop on Building Educational Applications Using NLP*, pages 136–146.

Juan Pino and Maxine Eskenazi. 2009. Semi-automatic generation of cloze question distractors effect of students' L1. In *Proceedings of SLaTE 2009*, pages 65–68.

Paul Piwek and Svetlana Stoyanchev. 2010. Question generation in the CODA project. In *Proceedings of the 3rd Workshop on Question Generation*, pages 29–34.

Keisuke Sakaguchi, Yuki Arase, and Mamoru Komachi. 2013. Discriminative approach to fill-in-the-blank quiz generation for language learners. In *Proceedings of ACL 2013*, pages 238–242.

Simon Smith, PVS Avinesh, and Adam Kilgarriff. 2010. Gap-fill tests for language learners: Corpus-driven item generation. In *Proceedings of the International Conference on Natural Language Processing 2010*, pages 1–6.

Eiichiro Sumita, Fumiaki Sugaya, and Seiichi Yamamoto. 2005. Measuring non-native speakers' proficiency of English by using a test with automatically-generated fill-in-the-blank questions. In *Proceedings of the 2nd workshop on Building Educational Applications Using NLP*, pages 61–68.

Kurt Vanlehn, Collin Lynch, Kay Schulze, Joel A Shapiro, Robert Shelby, Linwood Taylor, Don Treacy, Anders Weinstein, and Mary Wintersgill. 2005. The Andes physics tutoring system: Lessons learned. *International Journal of Artificial Intelligence in Education*, 15(3):147–204.

John H. Wolfe. 1976. Automatic question generation from text - an aid to independent study. In *Proceedings of the ACM SIGCSE-SIGCUE Technical Symposium on Computer Science and Education*, pages 104–112.

Xuchen Yao, Gosse Bouma, and Yi Zhang. 2012. Semantics-based question generation and implementation. *Dialogue and Discourse, Special Issue on Question Generation*, 3(2):11–42.

Torsten Zesch and Oren Melamud. 2014. Automatic generation of challenging distractors using context-sensitive inference rules. In *Proceedings of the 9th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 143–148.