Exploring NLP and Information Extraction to Jointly Address Question Generation and Answering

Pedro Azevedo, Bernardo Leite(✉), Henrique Lopes Cardoso, Daniel Castro Silva, and Luís Paulo Reis

Artificial Intelligence and Computer Science Lab (LIACC), Department of Informatics Engineering (DEI), Faculty of Engineering, University of Porto (FEUP),
Rua Dr. Roberto Frias, s/n, 4200-465 Porto, Portugal
{pedro.jazevedo,bernardo.leite,hlc,dcs,lpreis}@fe.up.pt

Abstract. Question Answering (QA) and Question Generation (QG) have been subjects of an intensive study in recent years and much progress has been made in both areas. However, works on combining these two topics mainly focus on how QG can be used to improve QA results. Through existing Natural Language Processing (NLP) techniques, we have implemented a tool that addresses these two topics separately. We further use them jointly in a pipeline. Thus, our goal is to understand how these modules can help each other. For QG, our methodology employs a detailed analysis of the relevant content of a sentence through Part-of-speech (POS) tagging and Named Entity Recognition (NER). Ensuring loose coupling with the QA task, in the latter we use Information Retrieval to rank sentences that might contain relevant information regarding a certain question, together with Open Information Retrieval to analyse the sentences. In its current version, the QG tool takes a sentence to formulate a simple question. By connecting QG with the QA component, we provide a means to effortlessly generate a test set for QA. While our current QA approach shows promising results, when enhancing the QG component we will, in the future, provide questions for which a more elaborated QA will be needed. The generated QA datasets contribute to QA evaluation, while QA proves to be an important technique for assessing the ambiguity of the questions.

Keywords: Question Generation · Question Answering · Named Entity Recognition · Part-of-speech tagging · Open Information Extraction

1 Introduction

Artificial intelligence has its impact on all areas of society, and Education is not left out. Posing questions is a central piece in the area of Education. From Professor-generated questions, it is possible not only to test the acquired knowledge but also to contribute to the student's continuous learning process. As the

Published by Springer Nature Switzerland AG 2020
I. Maglogiannis et al. (Eds.): AIAI 2020, IFIP AICT 584, pp. 396–407, 2020.
https://doi.org/10.1007/978-3-030-49186-4_33

amount of information available is increasing exponentially, accurately selecting relevant information has become an important goal. The task is a daily concern for teachers. The ability to automatically generate questions opens up several possibilities in this context. The prospect of generating questions by giving a text document as input, can provide substantial assistance for reading comprehension. Additionally, it is very important in the Education field since it allows the use of automatic mechanisms to find relevant information and create questions about those contents. It can also be useful since it teaches people how to search for information online correctly. Answering these questions does not always imply knowing where the answers are located. In fact, when developing Question Answering (QA) systems, we often need to rely on Information Retrieval to locate relevant information on the web [28].

With this in mind, we have developed a tool capable of generating factual questions and also answering them, regardless of the genre of textual content. Generated questions include factual questions such as *Who?*, *Where?, Which country?*, *When?*, *What?*, *How much?*, and *What organization?*. Factual questions allow us to question about specific facts. Usually, these facts may refer to information about people, places, dates, events, monetary values or even about certain organizations/institutions. Our tool uses Named Entity Recognition (NER) to extract entities from the text. In addition, we use POS tagging to find patterns in sentences and extract information that can be questioned. Questions are generated from well-defined rules and with the use of regular expressions to match certain patterns.

In tandem with the Question Generation (QG) component, we have developed an independent QA mechanism. Searching within a corpus, the QA mechanism is capable of ranking candidate sentences. These sentences are also evaluated by their content using Open Information Extraction mechanisms [12].

This combined approach is sensible because both tasks can help each other. QA can be used to assess if the generated question is correctly formulated, by retrieving relevant candidate sentences; or if the question is ambiguous, by finding more than one viable answer. On the other hand, QG can help evaluate the QA mechanism by creating a large number of questions. By manipulating the subject or the manner that questions are generated, it is possible to evaluate the robustness of the QA mechanism.

In Sect. 2, we present related work in the two topics of Question Answering and Question Generation. In Sect. 3, we explain the methodology used in our research, and the development of the question generation and answering components. Experimental results are presented in Sect. 4. Finally, in Sect. 5 we conclude and point out some observations about both current work and new ideas for future work.

2 State of the Art

Most existing works address either QG or QA. Hence, this section discusses the state of the art for QG and QA in a separate way. The works that do address QG together with QA focus only on creating QG models to improve QA [6,11,27,31].

This is done by augmenting QA datasets, or by training the QA model with the generated questions and fine-tuning on QA datasets.

2.1 Question Generation

Typically, QG can be divided into three distinct categories: syntax-based, semantic-based and template-based.

In a syntax-based approach [5], the goal is to convert declarative sentences into interrogative using several transformations. With a semantic approach [7], it is possible to obtain the semantic parse of a sentence using semantic role labeling (SRL). This approach may provide a deeper level of analysis, when compared to the syntax-based one. It also applies the necessary transformations. Finally, in a template-based approach [15] there are no transformation rules. This method extracts relevant content from the text and uses predefined question templates. Recent approaches make use of neural networks [8,13] in order to automatically generate questions from large datasets.

Some specific algorithms have been developed regarding automatic question generation. Topic modeling and noun phrase extraction have been used to create questions from different text passages with a holistic approach [17]. The use of an agent for generation of factual questions has been proposed [24] in order to assess the knowledge of learners and verify their understandings. Factoid gap-filling questions are usually employed [1]: the system extracts informational sentences from the paragraphs in order to generate a gap or a set of gaps that will be hidden from the sentences. These elements will have to be filled out. Through semantic analysis it has been possible to extract important features such as semantic roles and then work with sentence patterns [7]. The generation of factoid questions with Recurrent Neural Networks (RNN) [22] uses a novel neural network approach in order to convert texts or facts into Natural Language questions. Then, the generated questions and their answers may be evaluated by both evaluation metrics and humans.

QG has been used to support several areas of study such as language learning [25], history [20], vocabulary and grammar [10], science [4] and technologies [30].

Our approach aims to bridge the studies made for generating factual questions in English through a syntactic analysis combined with Named Entity Recognition. We also generate factual questions from Dependency Analysis.

2.2 Question Answering

A base structure, for question answering, needs to be defined in order to find answers, so a generic pipeline is created with three major modules [21]: Question Analysis, Passage Retrieval and Answer Extraction. In Question Analysis, the main goal is to analyze the question. The module receives an input with unstructured text and identifies semantic and syntactic elements that define the question. This information will be encoded in a structured way to be used in the remainder modules. Passage Retrieval can be based on a search engine. Using the given query, the most similar passages or sentences are retrieved. Queries are formulated based on the information extracted in the Question Analysis stage, and are used to find information suitable for answering the posed question. Different candidates are then evaluated, for which dynamic sources such as the Web and online databases can be explored. Using the appropriate representation of the question and each candidate passage, candidate answers are extracted from the passages and ranked in terms of probable correctness in the Answer Extraction module. An answer can be formulated based on this information.

There are different ways to approach the analysis of a question. Most known approaches perform stop-word removal, conversion of the inflected term to its canonical form, query term expansion, and syntactic query generation. For example, Pakray et al. [19] uses the Stanford Dependency parser to recognize query and target result types. IBM Watson uses rule- and classification-based methods to analyze different parts of the input query [9]. Phrase-level dependency graph was adopted by Xu et al. [29] to determine question structure and the domain dataset was used to instantiate created patterns.

The Information Retrieval phase of any QA system is always a challenge, considering that sources, in some cases, are not the same. Problems arise on the credibility of the sources or how the information is structured. Thus, an IR system needs to be generic in order to extract information from different types of texts: structured and unstructured. Therefore, to deduce the intention of the query different approaches are used, such as a syntactic analysis to extract information from the query [26] as well as the use of structured information such as knowledge graphs able to disambiguate information [23].

Widely well-known techniques on Answer Extraction (AE) are n-grams, patterns, named entities and syntactic structures. A very important AE technique introduced a merging score strategy based on relevant terms [14].

3 System Overview

In our approach, we produced a tool capable of automatically generating questions from any text source, which are then sent to an independent module that analyzes the questions and tries to answer them based on a provided corpus. Figure 1 shows a system overview in which the red blocks represent QG operations, and blue ones compose the QA module. The dashed line aims to highlight the input and output elements that together intersect the QG and the QA modules, enabling the evaluation of the latter. As visible in the diagram, the same documents serve as input for both mechanisms. The key generated in QG is matched against the answer generated in QA in the evaluation step. "SPO (*subject*, *predicate*, *object*) Tuples" is the information acquired from the Question Analysis, to be further explained in Sect. 3.2.

3.1 Question Generation

NLTK[1] was used to tokenize the obtained sentences. We used spaCy[2] for POS tagging and NER, supporting the identification of the following entity labels:

[1] http://www.nltk.org/.

[2] https://spacy.io/models/pt.

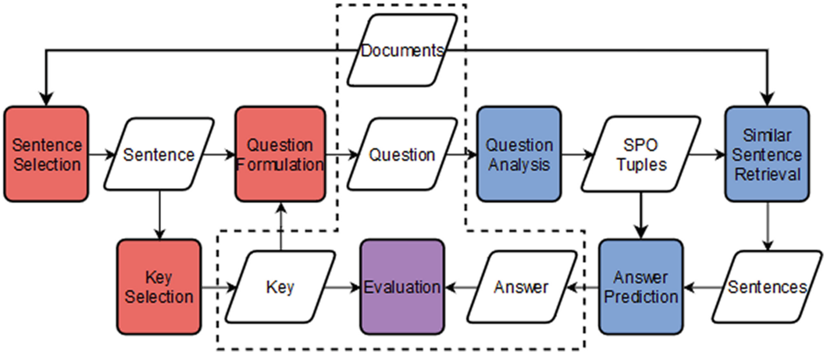

Fig. 1. System Overview, including QG steps (in red) and QA steps (in blue). (Color figure online)

PER (People, including fictional), DATE (Absolute or relative dates or periods), GPE (Countries, cities, states), LOC (Non-GPE locations, mountain ranges, bodies of water), ORG (Companies, agencies, institutions), MONEY (Monetary values, including unit) and EVENT (Named hurricanes, battles, wars, sports events). Additionally, spaCy also identifies context-specific token vectors, POS tags, dependency parse and named entities.

The first step for generating questions is directly related to the selection of sentences with one or more questionable facts. To address this phase, we use POS tagging combined with NER. Through the information obtained by these two tasks, it is possible to understand the structure of a sentence with some depth. Through POS we can know the grammatical class of each of the words and by identifying the entities we can find out if a phrase is expressing information about people, places, dates, countries, cities, states, money, or events. For instance, when a sentence begins with an entity type person followed by a verb, it is likely that the sentence is related to a certain person performing a particular action; we can thus ask who is the person performing that specific action.

Once the candidate sentence is selected, it is necessary to find out if its structure corresponds to at least one of the established rules. These rules allow us to check for the existence of questionable facts and to find patterns. For that, we use regular expressions. If there are matches on the established rules, we assume that a question can be generated from that sentence.

The next phase is to select the word or set of words that will be the answer (key selection) to the generated question. These words are the questionable facts and may be referring to people (*Who? What people?*), locations (*Where?, Which country?*), dates (*When?*), amounts (*How much?*), organizations (*Which organization?*) and events (*Which event?*).

The last phase is responsible for making the necessary transformations from the original sentence in order to create the interrogative sentence. Transformation rules take into account the position of pronouns, auxiliary and main verbs as well as phrases that are in active and passive form. The detail of the entire process is shown in Table 1.

Table 1. Steps from sentence handling to question generation.

Sentence pattern before POS and NER	**The French Revolution** was a period of intense political and social upheaval in France
Sentence pattern after POS	<DET> <PROPN> <PROPN> <AUX> <DET> <NOUN> <ADP> <ADJ> <ADJ> <CCONJ> <ADJ> <NOUN> <ADP> <PROPN> <PUNCT>
Sentence pattern after POS and NER	<**EVENT**> <AUX> <DET> <NOUN> <ADP> <ADJ> <ADJ> <CCONJ> <ADJ> <NOUN> <ADP> <PROPN> <PUNCT>
Expression used as rule in QG	<**EVENT**> <(?:AUXorVERB)>.*? <PUNCT>
Generated question	**Which event** was a period of intense political and social upheaval in France?

We have considered several particular cases where the sentences needed certain adjustments. These cases are listed as follows:

- The phrase is in the passive form. The system will transform it to active form;
- The phrase has the main verb and the auxiliary verb. The system will create the question in a way that both verbs stay in the correct positions;
- The phrase has more than one entity. The system will detect the given entities and, if it finds a relationship between them, it will question accordingly;
- The main verb is in the past and, for that case, it changes the verb tense.

Table 2 showcases several examples of questions generated according to the type of entities, number of entities and word position.

3.2 Question Answering

When designing the Question Answering module, it is necessary to not create assumptions, allowing this module to be completely independent of the Question Generation one. Three major steps identified in Subsect. 2.2 were followed to develop this module.

As the generated questions are of the *Wh* type, an abstraction of the analysis of the question was made. Thus, NER is performed to analyze the question. Each question may or may not have an entity. Thus, the extraction of all subjects and all objects are done. In order to do this, the Stanford CoreNLP [16] was used to obtain a Dependency Parsing Tree and also POS tagging. The returned entities, subjects and objects will be referenced as keywords.

Table 2. Original sentences and generated questions

Entity/Entities	Sentence and question
PER = Paul	S: **Paul** was the son of Henry of Burgundy and Teresa, the illegitimate daughter of King Alfonso VI of León and Castile Q: **Who** was the son of Henry of Burgundy and Teresa?

PER = Anne
PER = Henry
	S: **Henry and Anne** reigned jointly as count and countess of Portugal
	Q: **What people** reigned jointly as count and countess of Portugal?

GPE = Portugal	S: **Portugal** was conquered by Afonso I Q: **Which country** was conquered by Afonso I?
ORG = The Congress of Manastir	S: **The Congress of Manastir** had chosen the Latin script as the one to be used to write the language Q: **Which organization** had chosen the Latin script as the one to be used to write the language?
MONEY = 50 thousand dollars	S: One bedroom apartment costs **50 thousand dollars** Q: **How much** costs the one bedroom apartment?
PER = Henry	S: A car is cleaned by **Henry** Q: **Who** did clean a car?
DATE = 1109	S: Paul was born in **1109** Q: **When** was Paul born?

After analyzing the question, we retrieve sentences that may contain the information that can answer the question. The sentences are extracted from a predefined set of documents regarding a knowledge topic relevant to the question. To achieve this purpose, a metric of similarity is created between the keywords extracted in the previous step and the sentences that are more likely to contain the necessary content to answer the question. This metric consists in the statistical measure of the Term Frequency–Inverse Document Frequency (TF-IDF) followed by a cosine-similarity. The score has a range between 0 and 1. The sentences retrieved are the x most relevant sentences that scored a minimum y of the similarity metric. The variables x and y are hyper-parameters and need to be fine-tuned for different datasets. In this case, they were set as $x = 3$ and $y = 0.75$, extracting a sufficient number of sentences that would not overload the AE system and maintaining the most relevant sentences.

To extract the answer, we process the obtained sentences by extracting triples, consisting of subject (S), predicate (P) and object (O) [3]. To perform this extraction we use the Stanford OpenIE [2].

As an example, from the text *José Saramago was born in Portugal. Bernardo Azevedo wrote this sentence.* the following triples are extracted:

- '**subject**': 'José Saramago', '**predicate**': 'was', '**object**': 'born'
- '**subject**': 'José Saramago', '**predicate**': 'was born in', '**object**': 'Portugal'
- '**subject**': 'Bernardo Azevedo', '**predicate**': 'wrote', '**object**': 'sentence'

With these extracted triples, we perform lemmatization to every predicate. Since at this stage the objective is to obtain the answer, it is necessary to verify which predicate in the triple matches the one in the question. All triples in which the sentence relationship is not present are discarded. After this selection, we check if the question contains the subject or the object. If the question contains both, the triple is discarded. If the question has only the subject/object, the object/subject will be retrieved as a possible answer. In the end, there will be a list of all predicted answers.

An example is presented as follows, which shows the extracted triples that were used to predict the answer.

- **Question:** What people reigned jointly as count and countess of Portugal?
- **Most relevant sentence:** Henry and Anne reigned jointly as count and countess of Portugal.
  - **Triple 1:** ‘subject’: ‘Anne’, ‘predicate’: ‘reigned jointly as’, ‘object’: ‘count of Portugal’,
  - **Triple 2:** ‘subject’: ‘Henry’, ‘predicate’: ‘reigned jointly as’, ‘object’: ‘countess of Portugal’
  - **Predicted Answer:** Anne Henry
  - **Correct Answer:** Henry and Anne

4 Experimental Evaluation

We used the *Wikipedia Sentences 2*[3] dataset, available on *Kaggle*. It consists of a collection of 7.8 million sentences from August 2018 English Wikipedia dump. From this dataset, we created 50 different documents, each containing 20 randomly selected sentences. A small control group (handwritten sentences) of 3 documents was also created. This enabled us to analyze specific cases in order to improve the process of question generation.

Bearing in mind that there is no standard method for evaluating the quality of the generated questions, have developed a pilot test with five English teachers. Our survey contained 10 generated questions from the same text source (sentences from our dataset) and each question is evaluated according to the following criteria:

- **Objectivity of the Question** - Do you consider the question objective? (1 - Nothing objective, 5 - Very objective)
- **Question Extension** - How do you characterize question extension? (1 - Not long, 5 - Too long)
- **Grammatically** - Do you consider the question to be grammatically correct? (1 - Very Incorrect, 5 - Totally Correct)
- **Answerability** - How many answers do you think this question might have? (No answer, One, Two or more)

The results of the first three metrics can be seen in Table 3.

[3] https://www.kaggle.com/mikeortman/wikipedia-sentences.

Table 3. Averages scores for Objectivity, Grammatically and Question Extension

Metric	Avg. score
Objectivity (1–5)	3,64
Extension (1–5)	3,14
Grammaticality (1–5)	3,42

Regarding answerability, little consensus was achieved regarding the number of answers given a question. We assume this happens because there are several interpretations that can be caused by the presence of multiple entities in the sentence or external knowledge (in addition to what is written in the sentence).

Overall, the teachers considered the questions to be objective. Some ambiguities aroused when there were multiple entities identified for the same sentence. In other cases, generic questions have also introduced ambiguity. Regarding grammatically, we conclude that the questionable term (used at the beginning of the question) may not be the most appropriate in some cases. Also, the main inconsistencies are due to verb conjugation. Question extension is adequate most of the times but it needs treatment, mainly to remove unnecessary parts. The appropriateness of the question length may not always be the most suitable. To improve that, we would have to better understand the context in which the question is asked, that is, how long the question needs to be.

Assessing whether the answer found is the right one is not enough to evaluate the QA mechanism. Understanding the question is important and, for example, predicting "Paul" when the correct answer is "John" should not be considered as bad as "London". With this in mind, a metric of similarity was created. This consists of a model of word2vec embeddings [18], trained with the dataset available in the *Gensim* API called *text8*. This allows us to neutralize the score if the predicted and correct answers have a similar type or meaning.

To evaluate the QA mechanism, a pipeline was created to join the different tasks. Thus, all the documents generated from the *wikisentences* were read, followed by the generation of the questions and performing a search for their answers. Results are shown in Table 4. These results are divided on five groups: Wiki Documents, Controlled Documents, Entity Questions, Entity Questions without the generating question with the entity ORG and Dependency Questions. Evaluation metrics include: Short Answer (using the similarity metric), Correct Triple (whether the triple contains the answer) and Correct Sentence (as compared to the one used by the QG module to generate the question). The type of questions generated from Wiki Documents and Controlled Documents contained Entity and Dependency questions.

Overall, the results were good. Thus, we find that by adding different types of questions that include new entities, the QA system can generalize well, which demonstrates its robustness. This is supported by the fact that, when adding questions about the *ORG* entity, results are similar. However, results were not as good for questions generated through dependency analysis. This is probably

Table 4. Results of the different question sets and evaluated in 3 metrics

Dataset	Question generated	Short answer	Correct triple	Correct sentence
Wiki documents	311	78,5%	87,4%	96,4%
Controlled documents	43	89,3%	95,2%	100%
Entity questions w/o ORG	301	80,9%	88,5%	98,8%
Entity questions	334	81,2%	89,8%	98,1%
Dependency questions	20	32,7%	80,0%	100%

due to the fact that an extracted triple does not contain the necessary information. It is possible to state that creating different types of questions can help to evaluate the robustness of the QA module. These results also revealed a decrease in the performance of the QA mechanism for questions generated from the Wiki Documents when compared to the Controlled Documents. This is due to the fact that, in some cases, there are very broad questions and some questions are not being asked in the best possible way, reveling ambiguity problems like the appearance of different answer possibilities. After a closer look into the system, ambiguities have been found. For example, in the question, from the wiki documents, *What did Maria have?* the answer *New Pet* was predicted, although the correct answer could be *A Motorcycle*. This happened because there were two sentences that contained two reasonable answers: *Maria has a motorcycle.* and *Maria and Bob have adopted a new pet.*

5 Conclusions and Future Work

Question Generation and Question Answering are two independent yet highly related tasks. If it is true that the amount of available data has increased exponentially, it is also true that there is a need for discernment and responsibility to select important and valid information. This is important to filter information that can be questionable.

From a learning perspective, access to trustworthy information provides a variety of contents for the teachers. For the students, the available content can help them to enhance their knowledge. Bearing this in mind, the developed tool has a clear advantage: provide content such as generated questions and their answers. This helps the teacher to automatically generate questions and make slight changes if needed. For the student, it allows to test their skills with different questions from different contexts.

The results of the presented tool are promising. Even so, we are aware that there is a lot to improve in each of the modules. Our main goal is to present a direction of research on the possible bidirectionality of the QG and QA tasks. The analyzed results show a possible way to evaluate the robustness of the QA mechanism based on the ambiguity of the generated questions.

We intend to expand our tool with other types of approaches that allow a more extensive analysis of the sentences. In addition, we intend to decrease the number of grammatical errors that can be verified in the process of generating questions. Generating factual questions for the English language has a lot of possibilities and some of the suggestions can be: generating questions from a combination of more than one sentence, the ability to handle more complex text and to have standard evaluation techniques. The possession of gold-standard test data is also very important.

The most important conclusion that we can draw from our study is the possibility of generating datasets with question and answer pairs in a completely automatic manner. This way, less human intervention will be necessary to create this type of content. In this automatic generation process (both for questions and answers) it is necessary to guarantee the quality of the generated content. We see a promising future for these tasks using modern Machine Learning techniques.

References

1. Agarwal, M., Mannem, P.: Automatic gap-fill question generation from text books. In: Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications, IUNLPBEA 2011, pp. 56–64. Association for Computational Linguistics, USA (2011)
2. Angeli, G., Premkumar, M.J.J., Manning, C.D.: Leveraging linguistic structure for open domain information extraction. In: 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), pp. 344–354 (2015)
3. Banko, M., Cafarella, M.J., Soderland, S., Broadhead, M., Etzioni, O.: Open information extraction from the web. IJCAI **7**, 2670–2676 (2007)
4. Conejo, R., Guzmán, E., Trella, M.: The SIETTE automatic assessment environment. Int. J. Artif. Intell. Educ. **26**, 270–292 (2015)
5. Danon, G., Last, M.: A syntactic approach to domain-specific automatic question generation. CoRR (2017)
6. Duan, N., Tang, D., Chen, P., Zhou, M.: Question generation for question answering. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 866–874 (2017)
7. Flor, M., Riordan, B.: A semantic role-based approach to open-domain automatic question generation. In: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, pp. 254–263. Association for Computational Linguistics, New Orleans, June 2018
8. Harrison, V., Walker, M.A.: Neural generation of diverse questions using answer focus, contextual and linguistic features. CoRR abs/1809.02637 (2018)
9. High, R.: The Era of Cognitive Systems: An Inside Look at IBM Watson and How it Works. IBM Corporation, Redbooks (2012)
10. Hoshino, A., Nakagawa, H.: Predicting the difficulty of multiple-choice close questions for computer-adaptive testing. Nat. Lang. Process. Appl. 279 (2010)
11. Hu, S., Zou, L., Zhu, Z.: How question generation can help question answering over knowledge base. In: Tang, J., Kan, M.-Y., Zhao, D., Li, S., Zan, H. (eds.) NLPCC 2019. LNCS (LNAI), vol. 11838, pp. 80–92. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32233-5_7
12. Khot, T., Sabharwal, A., Clark, P.: Answering complex questions using open information extraction. arXiv preprint arXiv:1704.05572 (2017)
13. Kumar, V., Ramakrishnan, G., Li, Y.F.: A framework for automatic question generation from text using deep reinforcement learning. arXiv abs/1808.04961 (2018)

14. Le, J., Zhang, C., Niu, Z.: Answer extraction based on merging score strategy of hot terms. Chin. J. Electron. **25**(4), 614–620 (2016)
15. Le, N.-T., Pinkwart, N.: Evaluation of a question generation approach using semantic web for supporting argumentation. Res. Pract. Technol. Enhanc. Learn. **10**(1), 1–19 (2015). https://doi.org/10.1007/s41039-015-0003-3
16. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Association for Computational Linguistics (ACL) System Demonstrations, pp. 55–60 (2014)
17. Mazidi, K.: Automatic question generation from passages. In: Gelbukh, A. (ed.) CICLing 2017. LNCS, vol. 10762, pp. 655–665. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77116-8_49
18. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
19. Pakray, P., Bhaskar, P., Banerjee, S., Pal, B.C., Bandyopadhyay, S., Gelbukh, A.F.: A hybrid question answering system based on information retrieval and answer validation. In: CLEF (Notebook Papers/Labs/Workshop) (2011)
20. Papasalouros, A., Kanaris, K., Kotis, K.: Automatic generation of multiple choice questions from domain ontologies. In: Proceedings of the IADIS International Conference e-Learning 2008, vol. 1, pp. 427–434 (2008)
21. Prager, J., Chu-Carroll, J., Brown, E.W., Czuba, K.: Question answering by predictive annotation. In: Strzalkowski, T., Harabagiu, S.M. (eds.) Advances in Open Domain Question Answering. TLTB, vol. 32, pp. 307–347. Springer, Dordrecht (2008). https://doi.org/10.1007/978-1-4020-4746-6_10
22. Serban, I.V., et al.: Generating factoid questions with recurrent neural networks: the 30m factoid question-answer corpus. CoRR abs/1603.06807 (2016)
23. Shekarpour, S., Marx, E., Ngomo, A.C.N., Sina, S.: Semantic interpretation of user queries for question answering on interlinked data. Elsevier-Web Semantics (2015)
24. Stancheva, N.S., Popchev, I., Stoyanova-Doycheva, A., Stoyanov, S.: Automatic generation of test questions by software agents using ontologies. In: 2016 IEEE 8th International Conference on Intelligent Systems (IS), pp. 741–746, September 2016
25. Susanti, Y., Tokunaga, T., Nishikawa, H., Obari, H.: Evaluation of automatically generated English vocabulary questions. Res. Pract. Technol. Enhanc. Learn. **12**, Article no. 11 (2017)
26. Unger, C., Bühmann, L., Lehmann, J., Ngonga Ngomo, A.C., Gerber, D., Cimiano, P.: Template-based question answering over RDF data. In: Proceedings of the 21st International Conference on World Wide Web, pp. 639–648. ACM (2012)
27. Wang, T., Yuan, X., Trischler, A.: A joint model for question answering and question generation. arXiv preprint arXiv:1706.01450 (2017)
28. Wu, P., Zhang, X., Feng, Z.: A survey of question answering over knowledge base. In: Zhu, X., Qin, B., Zhu, X., Liu, M., Qian, L. (eds.) CCKS 2019. CCIS, vol. 1134, pp. 86–97. Springer, Singapore (2019). https://doi.org/10.1007/978-981-15-1956-7_8
29. Xu, K., Zhang, S., Feng, Y., Zhao, D.: Answering natural language questions via phrasal semantic parsing. In: Zong, C., Nie, J.-Y., Zhao, D., Feng, Y. (eds.) NLPCC 2014. CCIS, vol. 496, pp. 333–344. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45924-9_30
30. Zampirolli, F., Batista, V., Quilici-Gonzalez, J.A.: An automatic generator and corrector of multiple choice tests with random answer keys. In: 2016 IEEE Frontiers in Education Conference (FIE), pp. 1–8, October 2016
31. Zhang, S., Bansal, M.: Addressing semantic drift in question generation for semi-supervised question answering. arXiv preprint arXiv:1909.06356 (2019)