Paragraph Similarity Matches for Generating Multiple-choice Test Items

Halyna Maslak and **Ruslan Mitkov**
University of Wolverhampton
Wolverhampton, UK
halyna.maslak@gmail.com
r.mitkov@wlv.ac.uk

Abstract

Multiple-choice questions (MCQs) are widely used in knowledge assessment in educational institutions, during work interviews, in entertainment quizzes and games. Although the research on the automatic or semi-automatic generation of multiple-choice test items has been conducted since the beginning of this millennium, most approaches focus on generating questions from a single sentence. In this research, a state-of-the-art method of creating questions based on multiple sentences is introduced. It was inspired by semantic similarity matches used in the translation memory component of translation management systems. The performance of two deep learning algorithms, doc2vec and SBERT, is compared for the paragraph similarity task. The experiments are performed on the ad-hoc corpus within the EU domain. For the automatic evaluation, a smaller corpus of manually selected matching paragraphs has been compiled. The results prove the good performance of Sentence Embeddings for the given task.

1 Introduction

The Multiple-choice test items (MCQs) are frequently used for knowledge assessment, e.g., ongoing assessment or during an examination. They are also used for career-significant assessment in medical training and certification, e.g., the Medical College Admission Test (MCAT) or the United States Medical Licensing Examination (USMLE) as well as in law schools and as a part of the Multistate Bar Examination (MBE). Some employers also use MCQs to evaluate the knowledge of candidates applying for a job since it is an easy and fast way to do so. Tests of this kind are also extremely popular in game shows like Who Wants to Be a Millionaire? and alike, in myriads of mobile apps or Facebook quizzes. Moreover, the Covid-19 pandemic has already affected significantly the way assessment is conducted in education institutions switching to extensive use of technology. MCQs are recommended to be used for online education, and many virtual learning environments offer in-built tools for the composition of such tests (Burnett and Fuentes, 2020).

For our research, we decided to experiment with the EU law domain. MCQs are used to assess knowledge of job seekers applying for positions in in the European Union institutions. Furthermore, MCQs are frequently used during exams in law schools and universities. Therefore, the primary reason for selecting this domain is our motivation to assist education professionals and offer a helpful tool for knowledge assessment. We conduct the experiments on EU textbooks used in classrooms.

Despite the popularity and extensive use of multiple-choice tests, their manual creation is a laborious and time-consuming task (Mitkov and Ha, 2003; Mitkov et al., 2006). Ambiguously worded questions, too easy or too difficult distractors can lead to the poor performance of test-takers and misleading results.

There have been several attempts to automatically generate multiple-choice questions starting from various NLP techniques, including the use of WordNet, shallow parsing, corpora, ontologies (Mitkov and Ha, 2003; Papasalouros et al., 2008). For the last couple of years, researchers have been experimenting with machine learning (Guo et al., 2016). In recent papers on this topic, deep neural networks are successfully applied for this task (Kumar et al., 2018; Martinez-Gil et al., 2019).

Proceedings of the Student Research Workshop associated with RANLP-2021, pages 99–108,
held online, Sep 1–3, 2021.
https://doi.org/10.26615/issn.2603-2821.2021_015

Furthermore, various strategies have been used for choosing distractors or testing on various types of instructive material (such as linguistics (Mitkov and Ha, 2003) or medical texts (Karamanis et al., 2006), sports domain (Majumder and Saha, 2015), as well as Wikipedia articles (Singh et al., 2013). Most research papers are based on the sentence-based approach, where questions are generated from one sentence that contains a key term (Wang et al., 2018). In recent research, the questions are based on the whole paragraph with the use of deep neural models (Zhao et al., 2018).

We aim to evaluate the performance of two existing deep neural models, doc2vec and SBERT, for the generation of multiple-choice test items based on multiple sentences.

Our contributions are as follows:

- Compiling a corpus of MCQs based on multiple sentences within the EU law domain. As to our knowledge, no such corpus exists or is publicly available.
- Providing a state-of-the-art method for finding semantically similar paragraphs with the use of paragraph embeddings.

The MCQ corpus is needed for the experiments. It should contain the question-answer (QA) pair and the paragraph on which the question is based. In addition, we aim to create the corpus that can be used for knowledge assessment of EU law. Therefore, the QAs should follow the criteria of MCQs used in classroom, i.e. they should not focus on too narrow topics or specific cases. In addition, we want to expand this research in the future by generating distractors automatically. That is why the answer should be concise. The requirements of MCQ corpus design are listed in Section 3.1. More detailed description of the corpus and experiments can be found in the master's dissertation (Maslak, 2021).

2 Related Work

Early research was focused on the use of traditional natural language processing methods for generating questions and distractors automatically. For example, Mitkov and Ha (2003), who were the pioneers in generating MCQs automatically, used such NLP techniques as term extraction, parsing, a corpus, and WordNet. Their system-generated test questions and distractors based on a linguistics textbook. A few years later, Karamanis et al. (2006) generated Multiple-Choice Test Items from medical text using Rapid Item Generation (RIG) and the UMLS thesaurus. A medical textbook served as the source texts, while a much more extensive collection of MEDLINE texts was used as the reference corpus. In another research, Mitkov et al. (2006) employed various NLP techniques such as automatic term extraction, shallow parsing, sentence transformation and computing of semantic distance as well as corpora and ontologies. In contrast to the methodologies that highly depend on the used domain, Papasalouros et al.'s (2008) approach is domain-independent as they employed specific ontology-based strategies and OWL.

Singh Bhatia et al. (2013) proposed selecting sentences using existing test items in the Web as well as presented a technique for creating named entity distractors from Wikipedia. Alsubait et al. (2014) used OWL ontologies to generate multiple-choice test items and proposed a psychologically-based theory to control the question difficulty.

Afzal and Mitkov (2014) suggested an unsupervised dependency-based approach to identify the most important named entities and terms and define semantic relations between them. This approach did not use any prior knowledge about the semantic types of the relations but was based on a dependency tree model. The results were evaluated in respect of their readability, usefulness of semantic relations, relevance, acceptability of questions and distractors and general usability of multiple-choice test items.

A couple of years ago, the focus of the research community shifted towards the use of neural networks for natural language processing tasks. For example, Liang et al. (2018) investigated how machine-learning models, in particular feature-based and neural net (NN) based ranking models, can be used for distractor selection. Gao et al. (2019) proposed a hierarchical encoder-decoder framework to generate question items for reading comprehension questions from real examinations. Susanti et al. (2018) investigated methods for automatically generating distractors for multiple-choice questions on English vocabulary and used semantic similarity and collocation information. Shin et al. (2019) used a topic modeling procedure, machine learning, and natural language processing to generate distractors based on students' misconceptions.

In the aforementioned words, various approaches were used to generate questions or distractors automatically. Mostly, the questions were generated on the basis of a single sentence. In our research, we do not create questions automatically. Instead, we employ the most recent methods of semantic textual similarity matching to find similar paragraphs in the reference corpus. The question-answer pair attached to the incoming paragraph that is fed to the model is transferred to the newly found paragraph and then post-edited to create a new question-answer pair. This approach aims at reducing the teachers' effort to create new test items from scratch.

3 Methodology

This research was inspired by translation memory (TM) matching used in computer-assisted translation (CAT) tools. TM consists of aligned on the sentence level source-and-target pairs (Sikes, 2007). These are the previous translations done by a human translator on the same or similar topic. Basically, after the source text is segmented according to segmentation rules, usually according to sentence-closing punctuation marks, the source sentence is searched in the translation memory. When a similar sentence in the source language is found, its translation is suggested for the target segment. The translation memory matches are retrieved according to a certain threshold chosen by a translator based on how similar the segments are. This score is normally presented in a percentage value. Totally exact segments constitute a 100% match, while exact segments within the same context, i.e., the same sentences or phrases before and after this segment, are called 101% matches or context-matches. The other matches below the 100% score are referred to as fuzzy matches.

In this work, an analogous method is used for finding similar paragraphs. However, there are several differences. Firstly, only one language, English, is used in the experiments. Moreover, the segmentation rules are different. Instead of sentences, paragraphs are used. A paragraph is defined as a sequence of characters ending with closing punctuation marks, i.e., a full stop, exclamation mark, question mark, followed by a new line symbol. Moreover, in the case of lists, such as the text presented in the form of bullet points, additional rules apply. If colons and semicolons are followed by a new line symbol, the new line is removed so that such phrases comprise a part of a larger paragraph. Otherwise, if left unchanged, each new bullet point phrase would make a separate paragraph. The paragraph consisting of a few words is not useful for this research.

To imitate the translation memory matches, two corpora are needed. The Paragraph cell in the above-mentioned MCQ corpus serves as a source text segmented into individual characters. A paragraph is searched for in the huge reference corpus that represents the translation memory. When a similar paragraph is found, it is suggested to a user. The question, anchor, and distractors attached to the source paragraph are used now for the target paragraph. They can be post-edited to reflect the contents of the new paragraph better. Although the reference corpus used in this research does not include any translations, for the simplicity of explaining the mechanism behind the research idea, it is called the translation memory, or TM corpus.

3.1 MCQ Corpus Compilation

For the purposes of this research, a corpus of MCQs is required. It is needed for the experiments to find similar paragraphs in another textbook. Such a corpus is manually compiled and consists of the following elements:

- The paragraph from a chosen book on which the question is based.
- The question referring to the corresponding paragraph.
- The answer.

Such question answering datasets as The Natural Questions corpus (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), HotpotQA (Yang et al., 2018), QuAC (Choi et al., 2018), QASPER (Dasigi et al., 2021), CaseHOLD (Zheng et al., 2021), OpenBookQA (Mihaylov et al., 2018), SQuAD (Rajpurkar et al., 2016), and MultiRC (Khashabi et al., 2018) were revised to determine whether they can be used in this research. Due to the fact that most of them contain question-answer pairs derived from Wikipedia articles, we did not find them suitable. Instead, we focus on textbooks and other teaching materials for the experiments to imitate real-life in-classroom applications. Moreover, in some cases, it is not

possible to create the reference corpus for the above-mentioned QA pairs since it is not clear what materials were used for those datasets. In addition, in some cases, the questions were not based on multiple sentences. Therefore, it would not be reasonable to use such datasets while we aim to work with the European Union law domain and with questions based on a whole paragraph. Consequently, it was decided to develop a new corpus of multiple-choice test items that is based on the selected principles of design.

The criteria for choosing a book for the MCQ corpus were the following:

- The contents of the book cover general information about the European Union, particularly the history of its creation, the founding treaties, the four freedoms, the sources of law, and other basic concepts and legislation. The book should not include any specific details regarding specialised legislation cases. For instance, home affairs law, EU immigration and asylum law, commerce or terrorism are too narrow topics and were not touched upon in the corpus.

- The information presented in the book should mostly be written in full paragraphs rather than in bullet points or stored in tables.

- It should be possible to copy the text easily from the book. Otherwise, retyping the paragraphs would take too much time. Moreover, it is easier to have typos or other errors in this case. Thus, not being able to copy directly from the book constitutes a challenge in terms of both time and human effort.

The chosen book consists of sixteen chapters and covers the following topics from the history and legislation of the European Union: the origins and character of EU law; the development from Community to Union; the political and legal institutions of the European Union; the sources of EU law; the legislative process; enforcement of EU law; Article 267 of the Treaty on the Functioning of the European Union (TFEU); the relationship between EU law and national law – supremacy; the relationship between EU law and national law – direct effect; the Internal Market; citizenship of the Union; the free movement of workers; freedom of establishment and the freedom to provide and receive services under Articles 49 and 56 TFEU; the free movement of goods and Articles 34 and 35 TFEU; Article 28 TFEU and customs tariffs and Article 110 TFEU and discriminatory internal taxation; EU competition law; social policy; discrimination law and Article 157; the wider social influence of the EU. All of these topics are mentioned in the MCQ corpus.

The following criteria for the corpus were selected:

- Paragraphs have to be included for all the questions. Since the research is based on paragraph similarity matching, without the relevant paragraph the question cannot be used for the purposes of this task. In the context of this research, a paragraph refers to either an actual paragraph from the book or a couple of sentences on which the question is based. The Paragraph cell should contain only those sentences that are necessary to answer the question. Therefore, sometimes the actual paragraph from the book has to be cut or shortened to include only the sentences needed to answer the question. It is possible to add sentences from different paragraphs as long as they make sense and the text in the Paragraph cell is 2-5 sentences long. The length of the paragraphs may vary. It is worth underlying that only one sentence cannot constitute a paragraph for this research no matter its size. The number of sentences in the paragraph highly depends on the asked question and the answer. Although it is possible to include longer paragraphs in the MCQ corpus, it is rather challenging to make a question that is based on the information in all the sentences in a large paragraph.

- All the questions in the MCQ corpus should be based on the information in all the sentences in the Paragraph cell. Yes/No or True/False questions are not possible. The questions should be worded as actual questions rather than incomplete statements.

- The answer should consist of one or a few words rather than a long phrase. The

answer cannot consist of a whole sentence. It is of paramount importance for the automatic generation of distractors. Although the distractors are not generated for this study, they will be used in future research. In order to retrieve good quality distractors, the anchor should be concise, ideally consist of a couple of words or a number. However, the MCQ corpus is aimed to be easy to use in classrooms. Therefore, the above-mentioned rule about the length of the anchor can be neglected in some cases for the sake of a good multiple-choice test item that can be used unchanged.

In total, the MCQ corpus consists of 200 paragraphs and question-answer pairs. The corpus was checked with a writing tool Grammarly to spot grammar and spelling mistakes, typos, extra spaces and other issues. Furthermore, all the questions were revised by a native speaker of English who is a professional proofreader and qualified linguist. Unfortunately, due to the lack of time and resources, the MCQ corpus was not revised by an expert in EU law. It is an area of further research and developments.

3.2 The reference corpus (TM corpus)

The preparation of the TM corpus started with choosing the relevant book. A procedure similar to choosing the book for the MCQ corpus was used. Overall, the book had to cover the same topics in the EU law and be available in a machine-readable or easy-to-process format. The information should be presented in text in paragraphs rather than in diagrams, charts, tables or images. In contrast to the MCQ corpus, the TM corpus should be of a considerably larger size. While looking for such a book, we encountered several problems. Many books on the EU law are available in hard copies but do not have the copies in the electronic format. In many cases, if it is available in the PDF format, the printed book is scanned. Therefore, even with the use of the optical character recognition (OCR) software, the processed output would be of very low quality and will require a significant amount of time and human resources to be revised manually.

The selected book consists of 1,198 pages and includes similar chapters as the one used for the MCQ corpus. The PDF is processed with a Python program in order to write the text into a text file. The information unrelated to the contents of the relevant paragraphs is omitted. For example, the table of contents, page numbers, headings, headers, footnotes, the list of references as well as any tables or images are not included in the TM corpus file.

Moreover, in the original file, the words at the end of the line are hyphenated. It is a typographical hyphen that has to be removed from the corpus. If it is not done, a substantial amount of information can be lost since for the machine, the same word written with or without a hyphen constitutes different words. Besides, a new line character follows this typographical hyphen, so the word is split into two parts. The hyphens followed by a new line character were deleted, and the parts of the word were glued together. All the hyphenated words with a new line character inside were checked in the Enchant module. It is a spell-checking library for Python. After the new line symbol is deleted, if a hyphenated word is found in the library, it is left untouched. Otherwise, it is glued together.

The other issues with processing the file included the following ones: many chapters include quotes that constitute a large paragraph. They should not be included in the TM corpus since the MCQ corpus can contain the same quotations. For the purposes of this research, we want to retrieve fuzzy matches instead of the exact matches. Therefore, we do not want to have exactly the same paragraphs in both corpora. Since such quotes are written in a different font and size, it was possible for the Python program to omit them.

3.3 Comparison of the MCQ and TM corpora

As mentioned above, the method for finding similar paragraphs resembles the one used for retrieving translation memory matches in a translation management system. Since the MCQ corpus represents a source text, and the reference (TM) corpus embodies the translation memory, it is essential that those corpora have many features in common. Hence, a comprehensive contrastive analysis of both corpora was conducted.

The corpora are compared in terms of the word count, the number of unique words, the average number of words per paragraph, most common words. As previously stated, the MCQ corpus includes 200 multiple-choice test items together with the referencing paragraphs. It estimates to 12,754 words. A word in this context is a sequence of alphanumeric characters split by a space.

Conversely, the reference corpus equals to 399,196 words. It encompasses more than a thousand pages of the processed text. Thus, it is more than 31 times larger than the multiple-choice questions corpus.

The average number of words in a paragraph in the MCQ corpus is 63 words. Likewise, for the TM corpus the number is 57 words.

4 Experiments

Experimenting with semantic textual similarity has been within the scope of recent research in natural language processing. For example, Ranashinghe et al. (2019a) evaluated the implications contextual word embeddings have on unsupervised semantic textual similarity methods. They carried out their experiments with several dataset including the SICK dataset and bio-medical dataset for English as well as the dataset in Spanish. Such methods as cosine similarity using average vectors, Word Mover's Distance and cosine similarity using Smooth Inverse Frequency with contextualised word embeddings were evaluated to calculate semantic similarity between pairs of texts. They came to a conclusion that contextual word embeddings can be employed for various languages and domains for unsupervised machine learning tasks.

Ranasinghe et al. (2019b) experiment with Siamese neural networks (Bromley et al., 1993) for semantic textual similarity. They mention that "Siamese networks contain two or more identical sub-networks. The networks are identical in the sense that they have the same configuration with the same parameters and weights. In addition, parameter updating is mirrored across these sub-networks". The authors state that this type of neural networks performs well for finding similarity or a relationship between two comparable things, e.g., signature verification, face verification, image similarity as well as sentence similarity. In this research, we will use sentence transformers for the paragraph similarity tasks that also use Siamese and triplet network structures.

In Ranasinghe et al. (2020), sentence encoders are used to improve the matching and retrieving process in Translation Memories systems. In our research, we use a similar approach while experimenting with paragraph encoders.

4.1 Doc2vec model

In our experiments, we use a similar technique as in Řehůřek (2014) and Shperber (2017). We use Gensim, a free Python library for topic modelling (Rehurek and Sojka, 2010). It is helpful for training large-scale semantic natural language processing models; representing text as semantic vectors; and finding semantically related documents. It allows to train vector embeddings fast, can be run on any operating system as well as any other platform that supports Python and NumPy. The library is also open-source.

For each paragraph from the MCQ corpus, we retrieve two similar paragraphs from the TM corpus. We also calculate the similarity score for each match.

4.2 Sentence-BERT

Transformer models like BERT are the state-of-the-art nowadays in semantic textual similarity and are salient in natural language processing in general. They are widely used in machine translation and time series prediction; document summarization; document generation; named entity recognition; image processing; video understanding and other tasks. Sentence-BERT (SBERT) is "a modification of the pretrained BERT network that use Siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity" (Reimers and Gurevych, 2019). It was presented as an advancement of BERT, which requires a lot of computational power and is very slow. For example, BERT needed 65 hours to find the most similar pair in a collection of 10,000 sentences, while SBERT did the same task in five seconds. Although the authors claim that SBERT is very fast and computationally efficient, such speed and performance requires the use of a GPU. For this research, the experiments were conducted on a regular laptop using a CPU only. Therefore, our results are far from such small numbers in terms of speed. The motivation behind the approach of using a CPU only for this research is explained by the fact that the aims of this research is to help education professionals and reduce the teachers' effort in preparing the assessment materials. Therefore, if the used methods require very powerful computers and a lot of computational power with the use of a GPU, it will not be feasible

to replicate this method in a real-life classroom setting.

The procedure for finding the similar paragraphs in this research is analogous to the one used with doc2vec. Particularly, for each paragraph from the small ad-hoc corpus of multiple-choice test items, two matching paragraphs are found in the large reference corpus. The similarity scores are also calculated for all the retrieved paragraphs.

In this research, we took the SBERT approach to obtain paragraph vectors. We used the paraphrase-MiniLM-L12-v2 model.

5 Evaluation

In this study, a simple method of automatic evaluation is used. Specifically, the paragraphs from the MCQ corpus are fed to the algorithms. With the help of either doc2vec or SBERT, semantically similar paragraphs are found in the reference corpus. The results are written to a file together with the paragraphs similarity score and the paragraph manually found as a gold standard. Paragraph similarity scores resemble fuzzy matches in translation memory systems. After that, the number of identical matches, i.e., paragraphs produced by the algorithms and the paragraphs from the evaluation corpus, is calculated. The performance of the algorithms is presented in the percentage of correct matches out of 50 paragraphs.

For the purposes of evaluation, a small corpus of fifty paragraphs was made. It consists of a Paragraph column with the paragraphs from the MCQ corpus; a Question column with the questions attached to those paragraphs; and a Match column with the paragraphs similar to the ones in column 1. All the matches are selected manually. In order to do so, the reference corpus is searched for key words manually identified in the paragraph from the MCQ corpus. It is done with the Find option. After that, the found results are checked in terms of their content and size. The paragraph from the MCQ corpus and its match from the reference corpus should talk about the same thing and include the same key words. Moreover, it is important that the question from the MCQ corpus attached to one paragraph can be answered if shown only the paragraph from the reference corpus. In addition, the correct answer (anchor) for this question should also be included in the new match. It was required that the length of both paragraphs was roughly similar where possible.

5.1 Evaluation Results

It was calculated that the performance of doc2vec during the automatic evaluation was 0%, i.e. zero matches out of fifty paragraphs in the evaluation corpus were found. In contrast, in 44% of the cases, SBERT found the correct matches from the evaluation corpus. More rigorous human evaluation is required to check to what extent the automatically obtained paragraphs are similar to the ones in the MCQ corpus. It is assumed that some of the question answer pairs could be post-edited and used in the in-classroom environment. It was also noticed that in some cases, SBERT outperformed human judgement and found the paragraphs with higher similarity scores. In terms of efficiency, SBERT was rather slow when used on a CPU, which is its main drawback.

Table 1 below includes the paragraph similarity scores for both models. It is evident that the difference in performance of the two algorithms is rather significant. The highest score of doc2vec is roughly the same as the lowest score of SBERT. Moreover, the average score of SBERT is higher than any of the results of doc2vec.

doc2vec	SBERT
10 highest scores	
0.58	0.86
0.57	0.86
0.57	0.85
0.57	0.85
0.57	0.85
0.57	0.83
0.57	0.83
0.56	0.83
0.56	0.82
0.56	0.82
Highest score	
0.58	0.86
Lowest score	
0.37	0.57
Average score	
0.51	0.70

Table 1: Similarity scores for doc2vec and SBERT

Furthermore, the efficiency of both algorithms was compared in terms of time needed to perform the task. The time in seconds was recorded for each found match with the help of the Python module

Timeit(). After that, the average time needed to find a single match was calculated. The results for the two algorithms are as follows: doc2vec – 46 seconds; SBERT – 268 seconds (4.4 minutes).

It is worth mentioning that usually the experiments with SBERT are conducted on a GPU. It was done deliberately to check the efficiency of the algorithms under the typical conditions teachers face at schools and non-technical departments in universities. Therefore, although SBERT demonstrated impressive results and even outperformed human judgement, it is very slow without using a GPU.

6 Conclusion and Future Work

In this research, the ways to automatically generate multiple-choice test items from a whole paragraph are investigated. In particular, it was inspired by translation memory matches in modern translation management systems.

The experiments are conducted within the European Union law domain. A small corpus of 200 multiple-choice test items together with the corresponding paragraphs is manually created from scratch. The larger reference corpus (TM corpus) that consists of 1,000+ pages of processed text from a textbook on the same topic was also created.

The performance and efficiency of two deep learning models, doc2vec and SBERT, was tested. Both algorithms provide paragraph vectors used to find semantically similar paragraphs. Although doc2vec was almost six times faster than sentence transformers when run on a regular computer with a CPU, it failed to produce the desired results. In fact, the performance of doc2vec was 0% when evaluated automatically. In contrast, SBERT achieved 44% performance in the same automatic evaluation task. Moreover, in some cases, it outperformed human judgement and found the paragraphs with higher similarity scores than those selected manually. Nevertheless, the fact that SBERT is not computationally efficient when used on a regular laptop remains its main drawback.

In the future, this research can be broadened to include automatic generation of distractors and their evaluation. Moreover, the corpus of multiple-choice test items can be extended. It would also be beneficial to have this corpus revised by an expert on the European Union law. Moreover, in-classroom experiments with the students of law and teachers are needed to fully evaluate the quality of produced multiple-choice test items and to test whether the suggested method truly reduces the teachers' effort in performing knowledge assessment.

References

Naveed Afzal and Ruslan Mitkov. 2014. Automatic generation of multiple choice questions using dependency-based semantic relations. In *Soft Computing* 18.7, pages 1269-1281.

Tahani Alsubait, Bijan Parsia, and Uli Sattler. 2014. Generating Multiple Choice Questions From Ontologies: Lessons Learnt. In *OWLED*, pages 73-84.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Sackinger, and Roopak Shah. 1993. Signature verification using a siamese time delay neural network. In *IJPRAI*, 7, pages 669–688.

Tim Burnett and Stefania Paredes Fuentes. 2020. Assessment in the Time of Pandemic: A Panic-free Guide. In *The Economics Network*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. *Quac: Question answering in context*, arXiv preprint arXiv:1808.07036. https://doi.org/10.18653/v1/D18-1241

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. *A Dataset of Information-Seeking Questions and Answers Anchored in Research Papers*, arXiv preprint arXiv:2105.03011. https://doi.org/10.18653/v1/2021.naacl-main.365

Yifan Gao, Lidong Bing, Piji Li, Irwin King, and Michael R. Lyu. 2019. Generating distractors for reading comprehension questions from real examinations. In *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. No. 01. 2019, pages 6423-6430.

Qi Guo, Chinmay Kulkarni, Aniket Kittur, Jeffrey P. Bigham, and Emma Brunskill. 2016. Questimator: Generating knowledge assessments for arbitrary topics. In *IJCAI-16: Proceedings of the AAAI Twenty-Fifth International Joint Conference on Artificial Intelligence*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers, pages 1601–1611. https://doi.org/10.18653/v1/P17-1147

Nikiforos Karamanis and Ruslan Mitkov. 2006. Generating multiple-choice test items from medical text: A pilot study. In *Proceedings of the fourth international natural language generation conference,* pages 111-113. https://aclanthology.org/W06-1416

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Volume 1 (Long Papers), pages 252-262. https://doi.org/10.18653/v1/N18-1023

Vishwajeet Kumar, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018. *A framework for automatic question generation from text using deep reinforcement learning*, arXiv preprint arXiv:1808.04961.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Rhinehart, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. In *Transactions of the Association of Computational Linguistics.* https://doi.org/10.1162/tacl_a_00276

Chen Liang, Xiao Yang, Neisarg Dave, Drew Wham, Bart Pursel, and C. Lee Giles. 2018. Distractor generation for multiple choice questions using learning to rank. In *Proceedings of the thirteenth workshop on innovative use of NLP for building educational applications*, pages 284-290. https://doi.org/10.18653/v1/W18-0533

Mukta Majumder and Sujan Kumar Saha. 2015. A system for generating multiple choice questions: With a novel approach for sentence selection. In *Proceedings of the 2nd workshop on natural language processing techniques for educational applications*, pages 64-72. https://doi.org/10.18653/v1/W15-4410

Jorge Martinez-Gil, Bernhard Freudenthaler, and A. Min Tjoa. 2019. Multiple choice question answering in the legal domain using reinforced co-occurrence. In *International Conference on Database and Expert Systems Applications*, pages 138-148.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. *Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering*, arXiv preprint arXiv:1809.02789. https://doi.org/10.18653/v1/D18-1260

Halyna Maslak. 2021. Automatic generation of multiple-choice test items with deep learning (Unpublished master's dissertation). University of Wolverhampton, Wolverhampton, United Kingdom.

Ruslan Mitkov. 2003. Computer-aided generation of multiple-choice tests. In *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing*, pages 17-22. https://aclanthology.org/W03-0203

Ruslan Mitkov, Le An Ha, and Nikiforos Karamanis. 2006. A computer-aided environment for generating multiple-choice test items. *In Natural language engineering 12(2)*, pages 177-194.

Andreas Papasalouros, Konstantinos Kanaris, and Konstantinos Kotis. 2008. Automatic Generation Of Multiple Choice Questions From Domain Ontologies. In *e-Learning*, pages 427-434.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. *SQuAD: 100,000+ Questions for Machine Comprehension of Text*, arXiv preprint arXiv:1606.05250. https://doi.org/10.18653/v1/D16-1264

Tharindu Ranashinghe, Constantin Orasan, and Ruslan Mitkov. 2019a. Enhancing unsupervised sentence similarity methods with deep contextualised word representations. In *RANLP*. https://doi.org/10.26615/978-954-452-056-4_115

Tharindu Ranasinghe, Constantin Orăsan, and Ruslan Mitkov. 2019b. Semantic textual similarity with siamese neural networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. https://doi.org/10.26615/978-954-452-056-4_116

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2020. *Intelligent Translation Memory Matching and Retrieval with Sentence Encoders*, arXiv preprint arXiv:2004.12894. https://aclanthology.org/2020.eamt-1.19

Radim Rehurek. 2014. Doc2vec tutorial.

Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*, pages 45-50.

Nils Reimers and Iryna Gurevych. 2019. *Sentence-bert: Sentence embeddings using siamese bert-networks*, arXiv preprint arXiv:1908.10084. https://doi.org/10.18653/v1/D19-1410

Jinnie Shin, Qi Guo, and Mark J. Gierl. 2019. Multiple-choice item distractor development using topic modeling approaches. In *Frontiers in psychology* 10: 825.

Gidi Shperber. 2017. A gentle introduction to Doc2Vec.

Richard Sikes. 2007. Fuzzy Matching in Theory and Practice. In *Multilingual*, 18(6), pages 39–43.

Arjun Singh Bhatia, Manas Kirti, and Sujan Kumar Saha. 2013. Automatic generation of multiple choice questions using wikipedia. In *International conference on pattern recognition and machine intelligence*, pages 733-738.

Yuni Susanti, Takenobu Tokunaga, Hitoshi Nishikawa, and Hiroyuki Obari. 2018. Automatic distractor generation for multiple-choice English vocabulary questions. In *Research and practice in technology enhanced learning 13.1*, pages 1-16.

Yuanlong Wang, Ru Li, Hu Zhang, Hongyan Tan, and Qinghua Chai. 2018. Using Sentence-Level Neural Network Models for Multiple-Choice Reading Comprehension Tasks. In *Wireless Communications and Mobile Computing 2018*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. *Hotpotqa: A dataset for diverse, explainable multi-hop question answering*, arXiv preprint arXiv:1809.09600. https://doi.org/10.18653/v1/D18-1259

Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3901-3910. https://doi.org/10.18653/v1/D18-1424

Lucia Zheng, Neel Guha, Brandon R. Anderson, Peter Henderson, and Daniel E. Ho. 2021. When Does Pretraining Help? Assessing Self-Supervised Learning for Law and the CaseHOLD Dataset. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*, pages 159-168.