

# **TDK-Thesis**

**Asmae Azzi**

Faculty of Mechanical Engineering, Informatics and Electrical Engineering

Informatics Department

Asmae Azzi

First year Master of Business Informatics

**Supervised By:**

Dr. Ferenc Erdős, PhD (Associate Professor)

## **Comparative Analysis of NLP-Driven MCQ Generators from Text Sources**

# TABLE OF CONTENTS

|            |                                         |           |
|------------|-----------------------------------------|-----------|
| <b>1</b>   | <b>Introduction .....</b>               | <b>5</b>  |
| <b>2</b>   | <b>Tools Presentation.....</b>          | <b>8</b>  |
| <b>3</b>   | <b>Methodology.....</b>                 | <b>13</b> |
| <b>3.1</b> | <b>Dataset.....</b>                     | <b>13</b> |
| <b>3.2</b> | <b>Evaluation Metrics .....</b>         | <b>13</b> |
| <b>3.3</b> | <b>Collaborative Evaluation .....</b>   | <b>15</b> |
| <b>4</b>   | <b>Comparative Analysis .....</b>       | <b>17</b> |
| <b>4.1</b> | <b>Quality of the questions .....</b>   | <b>17</b> |
| <b>4.2</b> | <b>Quality of the right answer.....</b> | <b>19</b> |
| <b>4.3</b> | <b>Quality of the distractors .....</b> | <b>21</b> |
| <b>4.4</b> | <b>Secondary metrics .....</b>          | <b>23</b> |
| <b>5</b>   | <b>Conclusion.....</b>                  | <b>25</b> |
| <b>6</b>   | <b>Summary.....</b>                     | <b>26</b> |
| <b>7</b>   | <b>References .....</b>                 | <b>27</b> |

## List of Tables and Figures

**Figure 1:** Steps of training of GPT-3

**Figure 2:** Multipartite Graph

**Figure 3:** Example of the relationship between hypernyms and hyponyms

**Figure 4:** Example of the Levenshtein Distance operations

**Table 1:** Evaluation Syllabus

**Table 2:** results for “Quality of the questions” metric

**Table 3:** results for “Quality of the right answer” metric

**Table 4:** results for “Quality of the distractors” metric

## List of Acronyms

**AI:** Artificial Intelligence

**BERT:** Bidirectional Encoder Representations from Transformers

**CEFR:** Common European Framework of Reference for Languages

**GPT-2:** Generative Pre-trained Transformer 2

**GPT-3:** Generative Pre-trained Transformer 3

**GPT-3.5:** Generative Pre-trained Transformer 3.5

**GPT-4:** Generative Pre-trained Transformer 4

**LLM:** large language model

**MCQ:** Multiple Choice Questions

**MMR:** Maximal Marginal Relevance

**ML:** Machine Learning

**NLP:** Natural Language Processing

**PaLM 2:** Pathways Language Model version 2

**PKE:** Python Keyphrase Extraction

**RLHF:** Reinforcement Learning From Human Feedback

**RNN:** Recurrent Neural Network

**SQuAD:** Stanford Question Answering Dataset

**LaMDA:** Language Models for Dialog Applications

**T5:** Text-to-Text Transfer Transformer

## Abstract

Recent studies on technology's impact on student learning reveal that contemporary tools enhance learner engagement. Nevertheless, technological tools that can be beneficial for both students and professors remain underutilized in the education sector. One of the most powerful technologies is artificial intelligence (AI), which has been rapidly evolving in recent years and has revolutionized the academic field. A prime example is natural language processing (NLP), a subfield of AI that focuses on addressing human languages, can be utilized in the educational domain by developing automated tutoring systems that understand and assist students, providing personalized guidance and aid. One noteworthy application of NLP in the educational sector is generating multiple-choice questions (MCQs) based on the individual text entered. If such a tool provides high-quality tests, considerable manual work can be avoided by the instructor.

The research delves into the comparative analysis of available solutions on the market, from the commonly used large language model-based chatbots like ChatGPT to MCQ-specific AI solutions such as Quizgecko. The analysis involved testing each software using a uniform dataset of input texts, which varied from simple to complex ones. The evaluation metrics comprised the text summarization, accuracy of keyword extraction, pertinence of the generated questions, and distractors quality, the latter being essential to assess the responder's comprehension and reasoning, making it harder to pick the right answer.

Additionally, this research presents my software designed mainly for MCQ generation, where I have used NLP concepts. The software employs multiple algorithms to summarize, extract keywords, and generate questions. Detailed within, this paper analyses different algorithms and NLP techniques such as: the Multipartite algorithm. A comparative analysis of WordNet, Sense2Vec, and BERT-WSD for distractors' generation further enriches the discourse.

# 1 Introduction

MCQs are a sort of examination questions in which the person responding is presented with different possible answers to a question from which they need to select the right answer (Haladyna & Downing, 2004). MCQs are among the most prevalent forms of quizzes used in education, and they are also employed in job interviews and surveys (Haladyna & Downing, 2004). However, it might be time-consuming to generate high-quality MCQs especially for educators that deal with a huge workload.

As the artificial intelligence field is rapidly evolving, multiple tools have been developed to help users in diverse areas, including education. This led to the creation of a multitude of software designed for the automatic generation of MCQs from an input text.

This paper starts by examining different studies conducted in this field highlighting the importance of the following literature review.

When generating questions from plain texts, the tools usually transform the declarative sentences in the text to questions by focusing on the main ideas of each sentence based on some keywords. For instance, “AUTOQUEST” (Wolfe & H., 1976) was among the first tools that use a similar technique, even though it is not any more on the market, it still represents an important example of MCQ generation programs. It was mainly used for testing students’ understanding after reading a text by generating a set of questions for every sentence. The program’s vocabulary was based on a dictionary of different grammatical categories ranging from pronouns to verb stems (Wolfe & H., 1976). The first step “AUTOQUEST” follows is to preprocess the given sentence, then develop a set of patterns based on the sentence’s characteristics, and finally postprocess those questions before outputting them.

In addition to the usage of nearly the same approach to generate MCQs, the tools we can find on today’s market have the same base architecture that was first presented by Vaswani et al. in 2017 which is the transformer architecture. This latter follows the ‘encoder-decoder architecture’ and is based on “self-attention” which can be defined as an attention system that links different points in some sequence to compute its illustration (Vaswani, et al., 2017). However, even though most of today’s tools that treat NLP tasks are based on the transformer architecture, we can notice some differences. We can distinguish three types of models: encoder-only, decoder-only, and encoder-decoder model. For instance, BERT is based on an encoder-only architecture, it is mainly a stack of transformer encoder layers that include many self-attention heads (Rogers, et al., 2020). Rogers et al. (2020) explains the workflow in BERT

as each head computes key, value, and query vectors (Rogers, et al., 2020) for every input token present in some pattern, which is utilized to produce a weighted depiction. Every output coming from the same layer's head are merged and routed via a fully linked layer (Rogers, et al., 2020). Every layer is preceded by layer normalization and completed with a skip connection. Programs that are based on encoder-only architecture like BERT are usually used for tasks that focus on sequence embedding, text classification, information retrieval, or anomaly detection. Additionally, they can also be used for MCQ generation especially if integrated with other algorithms. For the second type, decoder-only architecture programs are based on the transformer architecture but only uses the decoder part. It is formed by a stack of transformer decoder layers that include: "multi-head self-attention" and "position-wise feed forward networks" (Vaswani, et al., 2017). For example, GPT-2 follows the same architecture and can be used to perform a multitude of NLP tasks ranging from translation to question answering. Furthermore, GPT-3 and later models also follow the same architecture but can output higher quality results mainly because they have larger parameters and were trained and fine-tuned on more diverse datasets. (Floridi & Chiriatti, 2020) For instance, GPT-3 uses 175 billion parameters while GPT-2 used only 1.5 billion of them. (Floridi & Chiriatti, 2020) Finally, it is worth mentioning the tools that use both parts of the transformer architecture: the encoder and decoder such as T5 (Text-To-Text Transfer Transformer) that treats every task concerning text processing as a "text to text" situation. (Raffel, et al., 2020) The T5 model can perform a variety of tasks such as: translation, summarization, checking grammar, and checking similarity between sentences. Additionally, the pre-training of T5 model was performed on large datasets such as: "WMT' 14 English to French" (Raffel, et al., 2020) and "Colossal Crawled Corpus" (Raffel, et al., 2020), while the fine-tuning was done on diverse NLP tasks like translation and summarization. (Raffel, et al., 2020) The way those two aspects were completed differentiate the T5 model from the original transformer architecture. The T5 model has tremendous promise for a range of NLP applications, including MCQ production (Raffel, et al., 2020). The flexibility of this tool allowed developers and researchers to create software solutions that make use of this model.

This correlates with the studies conducted by Kurdi, Haffari, and Rasooli (Kurdi, et al., 2020) who provided a thorough review of the current state of automatic MCQ generation from an input text. They divided the MCQ generation techniques into two groups: rule-based methods and data-driven methods (Kurdi, et al., 2020). For the first group, the systems employ a set of manually produced rule-based guidelines, whereas the second type uses machine

learning. The authors explore a variety of automatic MCQ generating techniques, including open-source programs, research datasets, and commercial solutions such as: Quillionz, Quizizz, Quizlet, and others. Additionally, the authors mention many obstacles in the field of MCQ production like producing high-quality MCQs or ones in various domains and several languages (Kurdi, et al., 2020).

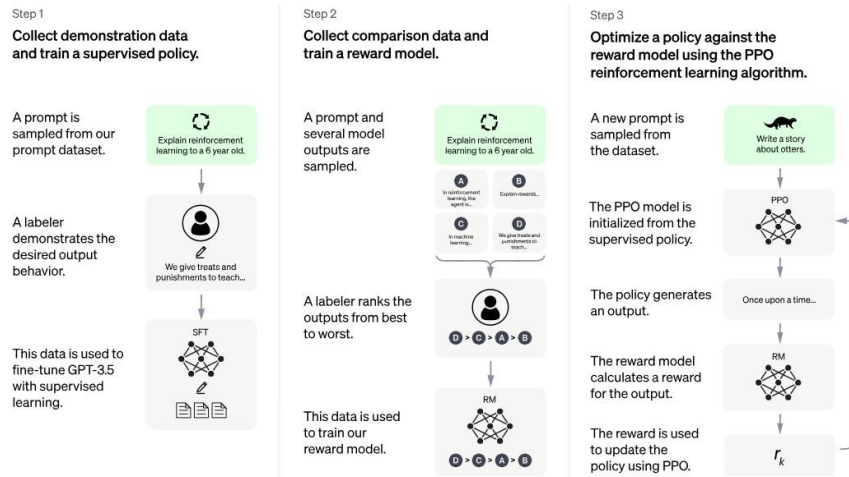
Even though the previously reviewed paper compared some tools that automatically generate MCQs, novel large language model-based generative AI technologies like ChatGPT or Google Bard, and other MCQ-specific solutions which can represent a crucial information deficiency. This research gap highlights the need of conducting a comparative analysis that may give significant insights into the distinct advantages and drawbacks of each tool while adapting to the individual demands of instructors at different levels and other quizzes creators. Additionally, the absence of deployed software solutions that use NLP techniques to produce MCQs in schools motivated me to create my own program that performs this task and serves as a teacher assistant. In this paper, I aim to present a comparative analysis of cutting-edge multiple tools available on the market ranging from general to specific for the task of MCQ generation including my own software solution.

## **2 Tools Presentation**

In this comparative analysis, I have chosen a range of tools that can perform the MCQ generation task. These tools were divided into two groups: large language model-based chatbots and MCQ-specific AI solutions. For the first group, I chose ChatGPT and Google Bard, while for the second group, I chose QuestGecko and Questgen. These tools are notable examples of current AI-driven MCQ generators since they rely on sophisticated language models. The second set was chosen since QuestGecko and Questgen are especially built for MCQ generation. These programs are designed to generate MCQs from textual information, and they represent specific solutions in the subject. Moreover, I added to this comparison my software that I created specifically to automatically generate MCQs from an input text. Before delving into the comparative analysis of the tools, it was important to learn about each one of them and study their different characteristics.

### **2.1 ChatGPT**

This AI software is a chatbot that has gained enormous popularity during the last year, it had a free version: GPT-3.5 and a paid one: GPT-4, both available via a web site developed by the program's creator, OpenAI. (Thorp & H.Holden, 2023) This tool was built on a decoder-only architecture and was trained to respond to a prompt by following an instruction and providing a thorough answer (OpenAI, 2022). ChatGPT has been trained using RLHF (Reinforcement Learning From Human Feedback), that uses similar approaches to InstructGPT which is a variant of GPT-3. However, there has been some minor variations in data gathering configuration (OpenAI, 2022). Concerning the Fine-tuning, OpenAI stated that they used supervised fine-tuning to train the initial model by having human AI trainers that offered discussions where they played both sides: the user and the AI assistant (OpenAI, 2022). The new dialogue dataset was then combined with the InstructGPT dataset which helped to convert it into a dialogue structure (OpenAI, 2022). The following figure available on the official website of OpenAI represents in more detail the steps followed to train the GPT-3 model.



**Figure 1:** Steps of training of GPT-3 (OpenAI, 2022)

ChatGPT is a valuable tool for researchers at all phases of a study, giving pertinent information, direction, and assistance to maximize productivity (Azaria, et al., 2023). This tool has performed well in multiple applications such as: summarization, ranking, categorization, and translation. This can revolutionize the education field by creating personalized learning experience and automating diverse tasks done by educators (Azaria, et al., 2023). However, ChatGPT has its own limitations that lead to generating incorrect or biased answers which can negatively impact the students' learning. Furthermore, ChatGPT may not comprehend the nuances of human language, which might lead to misunderstandings (Azaria, et al., 2023). Finally, as ChatGPT's training data was updated until 2021 (Azaria, et al., 2023), it was not able to generate answers to queries about events or information that occurred after that date.

## 2.2 Google Bard

Bard was initially built on top of Google AI's LaMDA (Language Models for Dialog Applications), which is a set of transformer-based neural language models characterized by their amazing scalability, with up to 137 billion parameters (Ahmed, et al., 2023). In 2022, Google conducted a study that proved LaMDA's tremendous promise in enhancing conversational AI programs (Ahmed, et al., 2023). It produced excellent results while generating plausible replies through the use of techniques like response creation, safety filtering, and others (Ahmed, et al., 2023). However, Bard has evolved and is now a free to use LLM (large language model) chatbot (Rezwan & Rahat, 2023) that is powered by PaLM 2 (Pathways Language Model version 2) and can perform multiple tasks such as: engaging in open-ended discussions, generating texts, translating, writing creative material, and providing

answers to users' queries (Rezwan & Rahat, 2023). For PaLM 2 also, it uses a standard transformer model architecture (Vaswani, et al., 2017) specifically following a decoder only architecture but with some modifications (Chowdhery, et al., 2022). Google Bard is capable of accessing real-time data, papers, and websites which extends its range of data retrieval and can be helpful for users' queries.

### **2.3 QuizGecko**

This tool is an AI question generator which generates different types of questions ranging from MCQs to fill-in-the-blank style questions (QuizGecko, 2023) It offers a free and a paid version of the system and the ability to customize, analyze, and publish the generated quizzes. Additionally, this tool allows users to generate quizzes from YouTube videos too.

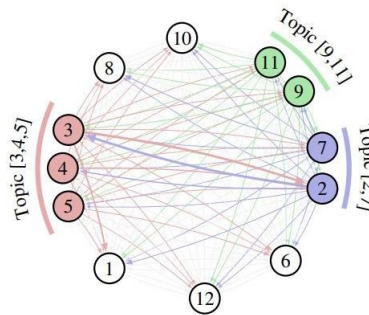
### **2.4 Questgen**

This tool is an AI-powered quiz generator that can create different types of quizzes such as: MCQs, true or false questions, fill-in-the-blank, or higher-order questions (Questgen, 2023). The quizzes can also be customized based on users' needs whether it is concerning the difficulty, number of questions generated, or the type of quizzes. This tool also offers a free and a paid version depending on the needs of the user. Questgen employs three T5 models to keep questions relevant. One for generating Boolean questions, one for MCQs, paraphrasing, and one for answering questions (gouthamg, 2023).

### **2.5 My software**

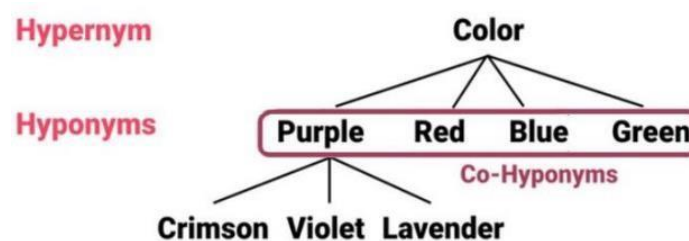
Concerning my tool, it is also an AI-powered MCQ quiz generator, however, this tool is much more focused only on the generation of MCQs which is different from the other tools. It offers the user a summary of the text entered paired with the MCQ generated. In my software, I relied on the pre-trained T5 model: T5-base, to perform the summarization task. This transformer model was trained using Teacher forcing which is utilized to train RNNs (Recurrent Neural Network) especially sequence-to-sequence models (Azzi, 2023). Additionally, cross entropy loss was also used in the training of T5, and this method computes the performance of a classification model that generates a probability value from 0 to 1 (Glossary, 2017). After the summarization phase, the system goes on extracting the keywords or keyphrases that represent the major topics in the sentences. This process was performed using the Multipartite Rank Algorithm that is available in PKE (Python Keyphrase Extraction) library. This algorithm is specialized in key extraction, it is graph-based and expresses the interconnections between the

words, key terms, and documents using a multipartite graph structure (Boudin & Florian, 2018). The following figure is an illustration of the multipartite graph, the nodes are the topics or the keyphrases and the edges are the co-occurrence relationships (Boudin & Florian, 2018).



**Figure 2:** Multipartite Graph (Boudin & Florian, 2018)

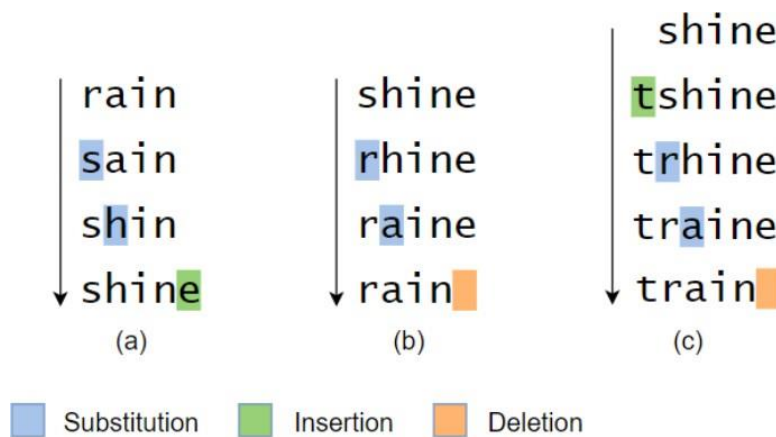
After the keyword and keyphrase extraction, the program generates questions using another pre-trained T5 model that is available for public usage on Hugging face Model Hub. This model was fine-tuned on the SQuAD v1 (Stanford Question Answering Dataset) dataset which is mostly used for training ML (Machine Learning) models for the task of question answering (Rajpurkar, et al., 2016). The following step is distractors generation, where my software uses WordNet and Sense2Vec to create relevant and efficient distractors. For this comparative analysis, I used only the Sense2Vec generated distractors for all MCQs, however, the user can choose between both tools depending on their preferences. The difference between the two methods is that the first one is an English lexical database containing relationships like hyponymy and hypernymy. These relationships are used to extract the relevant distractors by choosing the co-hyponyms of the hypernym of the keyword. A hypernym is a more general term that can hold multiple hyponyms. The following figure presents an example of the relationship between hypernyms and hyponyms.



**Figure 3:** Example of the relationship between hypernyms and hyponyms (Patil, et al., 2022)

The second method which is Sense2Vec is very different as it is an NLP approach that represents every meaning of a word in form of a distinct vector of real numbers (Trask, et al.,

2016). Sense2Vec can capture the different meanings of words and also represent rare words because of its multi-sense architecture (Trask, et al., 2016). As a final step, the software filters the generated distractors to keep only the most efficient and relevant ones. To do this, I used the combination of two methods: mmr (maximal marginal relevance) and Levenshtein distance. Mmr is defined as a method that identifies documents that satisfy two conditions: they are related to the query and distinct from previously selected elements (Carbonell & Goldstein, 1998) In my program, this method was used to delete irrelevant or repeated distractors. Additionally, for the second method which is the Levenshtein distance, it computes the minimum number of operations to change a sequence to another one (Devopedia, 2019). The operations are insertion, substitution, and deletion (Devopedia, 2019). The following figure showcases some examples of Levenshtein distance operations.



**Figure 4:** Example of the Levenshtein Distance operations (Devopedia, 2019)

The program presents its own limitations and needs further improvements including more fine-tuning for the models used in summarization and question generation. However, it is still worth comparing it with the available tools on the market and assessing their efficacy in MCQ generation.

## **3 Methodology**

### **3.1 Dataset**

As this research is based on a comparative analysis between the different available tools. It is important to use a uniform and well-structured dataset that provides different complexity levels to ensure a comprehensive evaluation of each tool. Additionally, using a consistent dataset provides a fair assessment for the software while generating useful and accurate results.

Recognizing the importance of the dataset's role in this research, I started to meticulously search for the relevant data. I decided to create my own dataset to guarantee unbiased analysis. This dataset contains four different texts that I used to test the tools. The texts are of different CEFR (Common European Framework of Reference for Languages) levels ranging from the beginner stage A2 to the advanced one C2. I have chosen the articles from two main different sources: the British Council website and Wikipedia. The inclusion of materials from those two sources highlighted the significance of combining academically oriented texts with more broad, publicly sourced material. The combination of these two offered a complete depiction of various styles and complexities, which enhances the dataset considerably. As the primary goal of my research is to emphasize the relevance of adaptability in the tools assessed, the best software should be capable of producing MCQs coming from different sources whether they are academic or broad platforms. Additionally, I purposely chose different areas of texts such as: education, zoology, computer science, and astronomy. to further enhance my dataset and guarantee a high-quality evaluation of each tool.

The diversity of the complexity and topics discussed in the dataset's texts allowed me to conduct a robust examination of the tools' capabilities in generating MCQs depending on the level of difficulty. By using a high-quality dataset like this one, this research assesses not only the ability of the tools to generate MCQs, but also their capability of recognizing and adjusting to each text difficulty.

Furthermore, using my dataset offered me two major benefits. First, it established a uniform and consistent standard for measuring the effectiveness of all tools, including my software. Second, as the dataset is aligned with CEFR, the results of the assessment have wider relevance in the global educational system.

### **3.2 Evaluation Metrics**

Before starting the comparative analysis of the available tools and my software, it is important to define clear evaluation metrics to measure the efficiency of each application. These metrics

can also be considered as assessment guidelines in order to ensure that the evaluation is objective and free from any biases. In the case of an absence of well-defined evaluation criteria, the assessment can easily turn to a subjective and inconsistent one, which results in ambiguous outcomes.

First, the assessment of the quality of the questions generated represents the primary metric in my evaluation. This examination checks the clarity, relevance, and difficulty level of the questions generated by each tool. In order to have an efficient MCQ quiz, we should have cohesive and clear questions that match the complexity of the text. In that way, the students can be efficiently assessed about their learnings depending on their English level and the topic covered. This metric allows us to differentiate between the tools that can only generate basic questions and the ones that are more sophisticated. This is reflected through the quality of each question generated especially in the context of high English level texts discussing complex topics.

Second, the quality of the correct solution provided by the tools is also an important metric as it often represents the keyword extracted from the source sentence and encapsulates its central idea. A good quality keyword is one that is appropriate and relevant to the context which allows a solid basis for the MCQ. Additionally, for some tools, the right answer might be based on the understanding of the main idea of the sentence which can reflect if the program has had a correct idea about each part of the text or misunderstood it which led to a bad quality answer. Having a good quality answer is crucial in MCQ quizzes as they represent the basis of the students' assessment, for this reason, it is important to avoid any false, misleading, or confusing answers.

Furthermore, the quality of the distractors generated is another essential evaluation metric. Distractors' role is to raise the difficulty level for students taking the test to assess their knowledge and spot their learning deficiencies. For this reason, it is important to have reasonable yet inaccurate distractors, that force learners to distinguish between small discrepancies. In the case of weak distractors, it is possible to mistakenly reveal the right answer which defeats the MCQ's goal. So, the tool's ability to produce contextually relevant, difficult, and different distractors demonstrates its expertise and value.

Moreover, there are some secondary evaluation metrics which I think are important to include in my evaluation as they influence the user experience and might impact positively or negatively their overall opinion about every tool. First, the ease of usage metric, this allows us

to evaluate the practicality of the program by assessing its user-friendliness. This is very important for different users of these tools as they can be from different backgrounds concerning AI but would like to automatically generate MCQs. When users are faced with a software solution that is easy to use, their experience while using the application becomes more enjoyable and fruitful (Lewis, 2014). For this reason, it is important to include the ease-of-use metric in the evaluation. Second, as time is a critical factor in any sector, it is crucial to include the speed of generation of questions for each tool. The goal of these programs is to help educators in the creation of MCQs which includes reducing the time needed for content development especially when handling a significant number of inquiries. However, in the case of a slow tool, its efficiency and productivity decline rapidly resulting in users' dissatisfaction.

Finally, some tools present some limitations compared to others which can create problems in use. For instance, as test takers can range from any English level, it is important to keep the versatility and adaptability when taking an input text. This includes reducing the requirements of words or characters needed if not removing this need. The word number necessity can sometimes lead educators to not use the program due to an insufficiency or surplus. For this reason, I included the word or character limitation in my evaluation, in case of the presence of this requirement, it is important to assess its impact on the usage of the tool.

### **3.3 Collaborative Evaluation**

After defining the evaluation metrics, I proceeded with a collaborative evaluation to ensure the robustness of my assessment. I invited ten participants (professors, and university lecturers) to evaluate the different tools too based on the same metrics. The diverse group of reviewers provided a lot of expertise to the evaluation, providing a comprehensive review of the tools' performance.

Every participant was asked to evaluate the tools based on the primary metrics that included: the quality of every question, the quality of each set of distractors, and the quality of the right answer. As I wanted to have an objective evaluation of the tools, their names were not stated on the sheet to avoid any prejudices. The assessment was on a scale from 0 to 5, the following table represents the details of the scoring system used for the evaluation metrics previously mentioned.

| <b>Evaluation Metrics</b> | <b>Score 0</b> | <b>Score 1</b> | <b>Scores 2-3</b> | <b>Scores 4-5</b> |
|---------------------------|----------------|----------------|-------------------|-------------------|
|---------------------------|----------------|----------------|-------------------|-------------------|

|                                    |                                                 |                                                     |                                                     |                                                     |
|------------------------------------|-------------------------------------------------|-----------------------------------------------------|-----------------------------------------------------|-----------------------------------------------------|
| <b>Quality of the question</b>     | Not understandable or not connected to the text | Not well formulated or has major grammatical errors | Not too relevant to the topic or too obvious        | Relevant to the topic and can be answered logically |
| <b>Quality of the right answer</b> | Incorrect or not related to the question        | Partially incorrect                                 | Correct but not directly answering the question     | Correct and directly answering the question         |
| <b>Quality of the distractors</b>  | No distractors generated                        | Contain grammatical errors or does not make sense   | Not relevant, confusing, or reveal the right answer | Useful, relevant, seems plausible, and well-crafted |

**Table 1:** Evaluation Syllabus

As the table shows, each evaluation metric receives a score from 0 to 5 to reflect the tool's performance in some specific area. These scores are intended to offer a detailed evaluation of the programs' features. A score of 0 means that the tool fails to satisfy the expected requirements for that metric, whereas scores ranging from 1 to 3 reflect varied levels of performance that are still behind the optimal. For scores 4 and 5, they indicate that the program operates well, suggesting adequate or exceptional performance. Following this rating system guarantees that the assessment is thorough and objective including tools' strengths and weaknesses.

I used a systematic data collection technique to improve the collaborative evaluation approach. The participants were sent an Excel sheet that had been meticulously built up for this task. This document had four texts paired with the MCQs generated by each tool. The form contained the primary evaluation metrics organized to facilitate efficient assessment with a space for every tool-text combination. This helped the participants to go easily through the evaluation, rate the quality of each MCQ in a structured manner, and record their remarks if needed within the Excel sheet.

In addition to the assessment of the ten participants, I conducted individual testing to evaluate not only the primary metrics but also the secondary ones. I used each software system to generate the MCQs for the four texts, during this testing phase, I was able to analyze the user-

friendliness and limitations of each tool. Additionally, the individual testing allowed me to present to the participants identical sets of MCQs to ensure an equitable evaluation.

The collaborative evaluation not only gave a comprehensive picture of the tools' capacities, but it also added to the results' reliability. The involvement of experts in the field of education guaranteed that the evaluation was accurate, and its outcomes were founded on solid knowledge. The participants' assessment alongside my individual testing provided a solid basis for the comparative analysis reported in this research.

## 4 Comparative Analysis

In this section, I conduct a detailed comparative analysis of the different tools that can automatically generate MCQs, all of which are powered by NLP, in order to evaluate their effectiveness in this area. The comparison examines proprietary and available tools on the market such as: ChatGPT, Google Bard, Questgen, and QuizGecko. Using this comprehensive evaluation, I intend to present a complete overview of the tools' abilities, strengths, and weaknesses, providing an insight on their prospective achievements in the field of MCQ production. The section presents every evaluation metric paired with the results of the performance of each tool concerning that aspect. To better showcase the performance of the tools, I decided to create tables that summarize the evaluation results by presenting the average score for each tool depending on the metric analyzed. The texts are ordered as in the evaluation in an ascending order in terms of CEFR levels: A2, B2, C1, and C2. The average score is over 5, and the total average of all tools for every text is presented as the final row of the table. This total average allowed me to have a clearer idea about the performance of each tool compared to the others, as if the average score of some system is below the total average, it means that its performance was poor compared to other tools.

### 4.1 Quality of the questions

The table below presents the details of the results of the collaborative evaluation concerning the first primary metric. The participants rated every tool respecting the evaluation syllabus provided and completed this process for the four texts.

| <b>Tools</b>       | <b>Text 1</b> | <b>Text 2</b> | <b>Text 3</b> | <b>Text 4</b> |
|--------------------|---------------|---------------|---------------|---------------|
| <b>QuizGecko</b>   | 3.87          | 3.87          | 3.62          | 3.62          |
| <b>Questgen</b>    | 4.37          | 4.25          | 3.87          | 3.5           |
| <b>Google Bard</b> | 2.87          | 4.12          | 3.75          | 3.62          |

|                    |      |      |      |      |
|--------------------|------|------|------|------|
| <b>ChatGPT</b>     | 4    | 4.25 | 4.37 | 4.62 |
| <b>My software</b> | 2.25 | 2    | 3    | 2.5  |

|                      |      |      |      |      |
|----------------------|------|------|------|------|
| <b>Total average</b> | 3.46 | 3.70 | 3.71 | 3.56 |
|----------------------|------|------|------|------|

**Table 2:** results for “Quality of the questions” metric

The data presented in Table 2 represents the average performance of each tool concerning the quality of questions generated for every text. After analyzing the data gathered and the average scores present in Table 2, I was able to conduct a detailed comparative analysis of those tools in terms of the quality of questions generated.

#### **4.1.1 QuizGecko**

Going through Table 2, I noticed that this tool performed moderately throughout the four texts, with a total average score of roughly 3.68. In general, QuizGecko was able to have an average score greater than the total average except for the third text where it was slightly lower. Overall, its performance was above average, which reflects its steadiness. However, the questions created by the tool were still of lower quality compared to the top performers. The average scores of QuizGecko across the different texts suggest that it preserves a particular level of question quality, although it may fail to adapt to changes in text complexity as the results reflect for texts 3 and 4.

#### **4.1.2 Questgen**

From the results of Table 2, the performance of Questgen is strong in terms of question quality as it scored an average score of 4 over the four texts. Overall, its performance was always above the total average except for the fourth text where it was slightly lower. However, in general, Questgen presented a superior performance that is most likely because of its natural language comprehension capabilities, which allows it to construct concise and contextually appropriate questions. The tool’s adaptability to the nuances of each text is a remarkable asset, as seen by its consistent high scores. A possible assumption could be that Questgen’s success may be connected to its NLP-driven methodology, which indicates that its comprehension of the text’s content might play a major role in question quality.

#### **4.1.3 Google Bard**

The performance of this tool concerning the quality of questions generated varies between texts, with an average score of 3.34. Although it earned a high score in the second and third texts, it performed significantly worse in the first and last texts. The variances in its performance might be attributed to its architecture, which may struggle to handle differences in text complexity. It is important to keep in mind that the tool is still under development and improvement. So, the relationship between the performance of Bard and text complexity should be investigated further.

#### 4.1.4 ChatGPT

From Table 2, the high performance of ChatGPT is easily remarkable with an average score of 4.31 across the four texts. The tool's scores have always been above the total average for every text, which is exceptional as it reflects its high performance on any text complexity or topic. These results are because of the tool's strong NLP architecture and natural language comprehension abilities. The program constantly generates meaningful, structured questions that are suited to the particular context of each text. This excellent performance across texts demonstrates ChatGPT's versatility and capacity to preserve a high level of question quality. This led to the assumption that there may be a link between this tool's performance and its NLP-driven strategy.

#### 4.1.5 My software

Concerning my software, the results shown in Table 2 prove that it has performed poorly compared to the other tools as it had an average of 2.56 across the texts. This average is the lowest compared to the other tools, and my program had been constantly scoring lower than the total average for every text. This reflects its bad performance concerning the quality of the questions generated. This performance can be attributable to the question creation techniques and processes. The pre-trained T5 model used for question generation can be changed or improved by further fine-tuning to generate higher quality questions.

### 4.2 Quality of the right answer

The table below presents the details of the results of the collaborative evaluation concerning the second primary metric.

| Tools     | Text 1 | Text 2 | Text 3 | Text 4 |
|-----------|--------|--------|--------|--------|
| QuizGecko | 3.75   | 4.75   | 4.37   | 2.87   |
| Questgen  | 4.5    | 4.25   | 4.75   | 3.62   |

|                    |      |      |      |      |
|--------------------|------|------|------|------|
| <b>Google Bard</b> | 3.62 | 4.5  | 4.25 | 3.87 |
| <b>ChatGPT</b>     | 4    | 4    | 4    | 4.5  |
| <b>My software</b> | 3    | 2.12 | 2.25 | 2.75 |

|                             |      |      |      |      |
|-----------------------------|------|------|------|------|
| <b><i>Total average</i></b> | 3.76 | 3.92 | 3.92 | 3.52 |
|-----------------------------|------|------|------|------|

**Table 3:** results for “Quality of the right answer” metric

The data presented in Table 3 represents the average performance of each tool concerning the quality of the right answer generated. After analyzing the data gathered and the average scores present in Table 3, I was able to conduct a detailed comparative analysis of those tools in terms of the quality of the right answer generated.

#### **4.2.1 QuizGecko**

Based on the data presented in table 3, this tool reflected a variable performance with scores ranging from 2.87 to 4.75, indicating its inability to consistently generate high quality answers. QuizGecko scored the highest in text 2, and this may be due to its capability to answer efficiently factual questions. However, the tool’s performance was remarkably lower than the average for text 4, and this may be due to its difficulties in offering accurate responses to questions about complex topics. Another assumption is that QuizGecko might be relying on pre-defined templates that do not always fit individual text characteristics.

#### **4.2.2 Questgen**

This tool showed excellent performance across the four texts. Its score was always above the total average which means that it generated good answers for the questions compared to the other tools. Questgen’s good performance can be related to its NLP-driven strategy, which enables it to perceive the contextual nuances of each text and provide high quality correct answers. This led me to assume that there is a correlation between the NLP approach of Questgen and the high-quality answers it provided in the MCQs.

#### **4.2.3 Google Bard**

The data of Table 2 reflected Google Bard’s performance which was good and consistent through the texts ranging from 3.62 to 4.5. Although its score in the first text was lower than the total average, its overall performance in the subsequent texts was notably superior. Google

Bard's integration of NLP-driven techniques may be credited with this result. These techniques vary from tokenization to semantic analysis and others. The program is capable of producing high-quality answers that adhere to the set requirements for each text.

#### 4.2.4 ChatGPT

This tool's results showed that it also achieved high results consistently over the four texts by having an average of 4.12. Its average score was always higher than the total average for all texts, even though it was not the highest, but it kept the good performance overall. A possible assumption is that there is a significant relation between its strong language processing capabilities and its capacity to provide high quality answers. The fact that ChatGPT consistently performed well in all texts demonstrates its language understanding and versatility to provide contextually appropriate and correct answers.

#### 4.2.5 My software

From Table 2, my software did not show a good performance over the four texts, it consistently scored lower than the total average fluctuating from 2.12 to 3. The tool's reliance on certain patterns, as well as its restricted NLP skills, are likely to have affected the quality of the correct answers. The results indicate that more improvements are required to increase the quality of the answers generated.

### 4.3 Quality of the distractors

The table below presents the details of the results of the collaborative evaluation concerning the third primary metric.

| <b>Tools</b>       | <b>Text 1</b> | <b>Text 2</b> | <b>Text 3</b> | <b>Text 4</b> |
|--------------------|---------------|---------------|---------------|---------------|
| <b>QuizGecko</b>   | 4.5           | 4.12          | 4             | 3             |
| <b>Questgen</b>    | 4.37          | 4.25          | 4.25          | 3.5           |
| <b>Google Bard</b> | 4.12          | 4.37          | 3.75          | 3.75          |
| <b>ChatGPT</b>     | 4.25          | 4             | 4.25          | 4.62          |
| <b>My software</b> | 2.5           | 1.87          | 1.75          | 1.75          |

|                             |      |      |     |      |
|-----------------------------|------|------|-----|------|
| <b><i>Total average</i></b> | 3.95 | 3.71 | 3.6 | 3.31 |
|-----------------------------|------|------|-----|------|

**Table 4:** results for “Quality of the distractors” metric

The data presented in Table 4 represents the average performance of each tool concerning the quality of the distractors generated. After analyzing the data gathered and the average scores present in Table 4, I was able to conduct a detailed comparative analysis of those tools in terms of the quality of the distractors generated.

#### **4.3.1 QuizGecko**

Having an average score of 3.9, this tool performed well in this area. Across all four texts, it consistently created high-quality distractors. Overall, QuizGecko's scores were above the total average except for the last text, this might be because the text discusses a complex topic containing multiple technical terms. However, in general, the participants in the evaluation thought that the wrong answer selections were relevant, well-crafted, and appropriately difficult.

#### **4.3.2 Questgen**

This tool has also showed an excellent performance over the four texts concerning this metric with an average score of 4.09. It was capable of keeping a score greater than the total average for all texts which shows its good capacities regarding the generation of high-quality distractors compared to the other tools. However, I noticed that the tool generated lower quality distractors for the last text, and this may be because of the complexity of the texts and terms used. The produced distractors were also relevant to the questions and provided a reasonable amount of challenge.

#### **4.3.3 Google Bard**

From the data presented in Table 4, the tool has performed well while generating distractors with an average score of 3.99. Its different scores across the texts were always above the total average, which showcases its good performance compared to the other tools. There is a noticeable difference in the quality of distractors generated for the last two texts compared to the first ones, as the scores are lower, it may reflect the difficulty for Google Bard to keep up the same performance for more complex texts using technical terms for example.

#### **4.3.4 ChatGPT**

Concerning the metric of the quality of distractors generated, ChatGPT stands out compared to the other tools as it has an average score of 4.28 across the four texts. It has consistently

generated high-quality distractors that always scored above the total average which reflects the tool's good performance in terms of generating useful and relevant distractors.

#### **4.3.5 My software**

For my software, it is shown from the data in Table 4 that it has performed poorly in terms of generation of distractors as the average score of this program was 1.96 which is very low compared to the other tools. For the individual texts, my software scored below the total average, this reflects the need for improvement of the programs concerning distractors generation. The tools, techniques, and strategy used can be further refined or changed.

#### **4.4 Secondary metrics**

For the evaluation of the tools compared in terms of the secondary metrics, I have conducted an individual assessment by using each program several times on different texts. As mentioned previously, these metrics include the ease of usage of the tools, their speed of generation of the MCQs, and their individual limitations.

First, concerning the ease of use of the tools, I noticed while using QuizGecko and Questgen that they offered the ability to choose the number of questions to generate, the language, and the difficulty of the MCQ generated. This is an efficient way to help instructors to customize their quizzes depending on their needs, especially that the tools' interfaces were easy to navigate in and choose the needed options. To create MCQs using these tools, the user needs to enter the text, choose the desired options, and submit. Concerning Google Bard and ChatGPT, as they are both chatbots, the user only needs to enter an inquiry providing the details of the MCQ generation paired with the corresponding text. The advantage of these chatbots is that the user can customize as he wants the MCQs only using the simple sentences he enters. This means that there are no limitations or specific options of personalization of the MCQs generated, it all depends on the user's inquiries. As for my software, the user needs to enter the text then submit. For all these programs, I did not find any difficulty using any of them as they all had clear and easy-to-use interfaces providing the user with all the necessary options needed for the creation of the MCQs.

Second, for the speed of generation, most of the tools did not showcase any latency while generating the MCQs. However, for QuizGecko, it took a notably long time to analyze the texts and generate the MCQs which can negatively affect the user experience.

Finally, for the individual limitations of each tool. I noticed while using QuizGecko that it imposes multiple limitations for the user in case of using the free version of the program such as: not exceeding 1000 characters of text, not being able to generate more than 5 questions MCQs, not being able to use the program for more than 2 texts. These conditions can cause a problem when using the free version of the software as it does not give a large space to experience the platform. The user will most likely need to pay for the software if he wants to use it frequently. For Questgen, it also offers two plans, the free and the paid one. However, the user can use the free plan more freely than in the case of QuizGecko as he can enter up to 5000 characters and run the program 13 times before the need of payment. This allows the user to have a clearer idea about the performance of the tool before deciding to have the paid plan. Concerning ChatGPT, it presents a free and a paid version of the chatbot, however, there are no restrictions in usage for the free plan. The difference between the two versions is that the free one uses GPT-3.5 while the paid one uses GPT-4 model which is more advanced and presents better results. For Google Bard, it is free for public usage and does not limit the user to any requirements. However, it is important to keep in mind that this tool is still under development and can commit errors and have a fluctuant performance. Finally, my software does not limit the user in any aspect, however, it is still not available for public usage.

## 5 Conclusion

Based on the results presented in the comparative analysis section, the primary metrics yielded some noteworthy conclusions. ChatGPT outperformed in these primary criteria, owing largely to its good NLP-driven techniques and the huge number and diversity of the training data it uses. Questgen and QuizGecko were closely behind, providing medium to high performance, although this varied depending on the assessment metric and text utilized. Concerning Google Bard, its performance was not consistent and varied a lot, which is most likely because the tool is still under development and improvement. Finally, for my software, it fell behind all tools and performed poorly, which means that it still needs to be refined further. The future work on this software includes more training on larger and diversified datasets and model refinement through extensive iterations.

When the tools were evaluated using secondary metrics, a new set of observations appeared. In terms of usability, QuizGecko and Questgen provided user-friendly interfaces with the ability to alter quiz criteria, making them ideal for instructors looking for tailored quizzes. Both ChatGPT and Google Bard provided personalization depending on user inquiries, with no predefined constraints. My program also provided a straightforward process for creating MCQs. In terms of the speed of generation, most tools worked well, with the exception of QuizGecko, which had a noticeably longer processing time, potentially affecting the user experience. Finally, the limitations of each program were noted. Both QuizGecko and Questgen have free and premium plans, although the free versions include use restrictions such as character count and the amount of MCQs created. ChatGPT offers both free and premium versions, with no substantial use limits on the free plan. Google Bard is open to the public with no restrictions; however, it is still in development. My software has no limits on usage but is not yet publicly released.

These results add to an in-depth knowledge of the strengths and weaknesses of different MCQ generators, allowing instructors to make informed choices when selecting a tool that meets their individual requirements. Additionally, as technology advances and the demand for tailored evaluations grows, it is probable that MCQ generating technologies will improve significantly, providing more effective and accurate options for practitioners across several areas.

## 6 Summary

This paper begins with an interesting literature review that highlights the different available tools in the market for MCQ generation and shows the absence of comprehensive comparative analyses in the field of NLP-driven MCQ generators. This literature review establishes the groundwork for an in-depth examination of different tools including my program. Each of the five MCQ generating tools is presented in detail, demonstrating their unique architectures and features.

The methodology section outlines the key aspects of the evaluation, beginning with the dataset used, which includes a varied selection of texts to reflect practical applications. This dataset is supplemented by the addition of primary and secondary evaluation metrics, which are intended to ensure an objective and detailed assessment. The report goes on to describe the collaborative evaluation procedure, which included 10 experienced participants in the field of education to ensure a high-quality evaluation.

The comparative analysis reveals each tool's performance across the primary metrics, beginning with the quality of questions, then the quality of the right answer, and finally the quality of the distractors. Furthermore, an examination of secondary metrics is conducted through an individual evaluation, where each tool was assessed based on the simplicity of use, speed of MCQ generation, and particular restrictions. This helped to a thorough understanding of each tool's usefulness.

Finally, in the last section, the paper summarizes the findings and rates the tools, naming ChatGPT as the best capable MCQ generator. The study concludes with a look ahead, implying that the topic retains prospects for advancement, with possible improvements and advancements in MCQ generating techniques.

## 7 References

- Ahmed, I., Roy, A., Kajol, M. A. & Reza, M. R., 2023. *ChatGPT vs. Bard: A Comparative Study*, : Authorea.
- Azaria, et al., 2023. *ChatGPT is a Remarkable Tool—For Experts*, : arXiv preprint.
- Azzi, A., 2023. *Question Generation Agent*, Ifrane: AUI.
- Boudin & Florian, 2018. *Unsupervised Keyphrase Extraction with Multipartite Graphs*, Vienna: Proceedings of the 27th ACM International Conference on Information and Knowledge Management.
- Carbonell, J. & Goldstein, J., 1998. *The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries*.
- Chowdhery, A., Narang, S. & Devlin, J., 2022. *PaLM: Scaling Language Modeling with Pathways*, Long Beach: Proceedings of the 36th International Conference on Machine Learning.
- Devopedia, 2019. *Levenshtein Distance*. [Online]  
Available at: <https://devopedia.org/levenshtein-distance#qst-ans-0>  
[Accessed 03 November 2023].
- Floridi, L. & Chiriatti, M., 2020. *GPT-3: Its Nature, Scope, Limits, and Consequences*, : Philosophy and technology.
- Glossary, M., 2017. *Loss Functions*. [Online]  
Available at: [https://ml-cheatsheet.readthedocs.io/en/latest/loss\\_functions.html#:~:text=Cross%2Dentropy%20loss%2C%20or%20](https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html#:~:text=Cross%2Dentropy%20loss%2C%20or%20)  
[Accessed 03 November 2023].
- gouthamg, r. g., 2023. *Question generation using state-of-the-art Natural Language Processing algorithms*. [Online]  
Available at: <https://github.com/ramsrigouthamg/Questgen.ai#nlp-models-used>  
[Accessed 03 November 2023].
- Haladyna, T. M. & Downing, S. M., 2004. *Developing and validating test items*. Mahwah: Lawrence Erlbaum Associates Publishers.
- Kurdi, M., Haffari, G. & Rasooli, M. S., 2020. Automatic Multiple Choice Question Generation From Text: A Survey. In: *IEEE Transactions on Learning Technologies*. :IEEE, pp. 14-25.
- Lewis, J. R., 2014. Usability: Lessons Learned ... and Yet to Be Learned. *International Journal of Human-Computer Interaction*, 30:9(10), pp. 663-684.
- OpenAI, 2022. *OpenAI*. [Online]  
Available at: <https://openai.com/blog/chatgpt#OpenAI>  
[Accessed 02 November 2023].
- Patil, S. et al., 2022. *State-of-the-Art Approach to e-Learning with Cutting Edge NLP Transformers: Implementing Text Summarization, Question and Distractor Generation, Question Answering*, : arXiv preprint.

Questgen, 2023. *Questgen*. [Online]  
Available at: <https://app.questgen.ai/>  
[Accessed 03 November 2023].

QuizGecko, 2023. *QuizGecko*. [Online]  
Available at: <https://quizgecko.com/>  
[Accessed 03 November 2023].

Raffel, C. et al., 2020. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*, Vancouver: Proceedings of the 33rd International Conference on Neural Information Processing Systems.

Rajpurkar, P., Zhang, J., Lopyrev, K. & Liang, P., 2016. *SQuAD: 100,000+ Questions for Machine Comprehension of Text*, : arXiv preprint.

Rezwan & Rahat, 2023. *What is Google Bard?*

Rogers, A., Kovaleva, O. & Rumshisky, A., 2020. *A Primer in BERTology: What We Know About How BERT Works*, Online: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

Thorp & H.Holden, 2023. *ChatGPT is fun, but not an author*. *Science*, 379(6630), p. 313.

Trask, A., Michalak, P. & Liu, J., 2016. *Sense2Vec - A Fast and Accurate Method for Word Sense Disambiguation in Neural Word Embeddings*, San Diego: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

Vaswani, A. et al., 2017. *Attention Is All You Need*

Wolfe & H., J., 1976. *Automatic Question Generation from Text - An Aid to Independent Study*, San Diego: Navy Personnel Research and Development Center.